Journal of Visual Communication and Image Representation **10**, 78–112 (1999)
Article ID jvci.1997.0404, available online at http://www.idealibrary.com on IDEAL®

A Survey on the Automatic Indexing of Video Data*,†

R. Brunelli, O. Mich, and C. M. Modena

ITC-irst, I-38050 Povo, Trento, Italy
E-mail: mich@irst.itc.it

Received December 30, 1997; accepted November 5, 1998

Today a considerable amount of video data in multimedia databases requires sophisticated indices for its effective use. Manual indexing is the most effective method to do this, but it is also the slowest and the most expensive. Automated methods have then to be developed. This paper surveys several approaches and algorithms that have been recently proposed to automatically structure audio–visual data, both for annotation and access. © 1999 Academic Press

Key Words: visual data; image databases; image indexing; image retrieval by pictorial content; video annotation; video databases; video indexing; video segmentation.

1. INTRODUCTION

Digital video is becoming an increasingly common data type in the new generation of multimedia databases [1–14]. Many broadcasters are switching to digital formats for broadcasting, and some of them already have a significant amount of video material available in digital format for previewing. Improved compression technologies and increased Internet bandwidth have made webcasting a real possibility. The production of multimedia material distributed on CD-roms has been increasing dramatically during the last few years and the introduction of the new DVD technology is now opening new possibilities. The ever growing amount of digital video poses new challenges, both of storage and access, as vast repositories are being built at an increasing pace. As shown by the Web explosion, the usefulness of vast repositories of digital information is limited by the effectiveness of the access methods. The key issues are those of *contents description* and of *information space navigation*. While textual documents in digital form are somehow self-describing (i.e., they provide explicit indices, such as words and sentences that can be directly used to categorize and access them), digital video documents do not provide

* This work was supported by Esprit Project 20636: EUROMEDIA and is made public with the permission of the Consortium members: TecMath GmbH, Digital Equipment GmbH, SWF, SDR, SVT, BBC, ORF, ITC-Irst, Emptic, Helix 5.

† The images used herein were obtained from IMSI's MasterClips® and MasterPhotos™ Premium Image Collection, 1895 Francisco Blvd. East, San Rafael, CA 94901-5506, U.S.A.

1047-3203/99 $30.00

such an explicit content description. In order to access video material in an effective way, without looking at the material in its entirety, it is therefore necessary to annotate the video sequences, providing an explicit content description targeted to the user needs. Digital video is a very *rich* medium, and the characteristics into which users may be interested are quite diversified, ranging from the *structure* of the video, i.e. its decomposition into shots and scenes, to the most representative frames or sequences, the identity of the people that appear in it, their movements and dialogues, and the accompanying music and audio effects.

Indexing digital video, based on its content, can be carried out at several levels of abstraction, beginning with indices like the video program name and name of subject, to much lower level aspects of video like the locations of edits and motion properties of video [15, 16]. The cost of manually generating such indices is inversely related to the level of abstraction of the index. For example, the indexing effort for video library applications which model video by title is much less than the indexing effort for a multimedia authoring application which indices video based on the content and style of the shots used to compose the video. Manual video indexing requires the sequential examination of the entire video clip in order to annotate it. This is a time-consuming, subjective, and expensive process; an hour long footage can take up to 10 h (depending on its contents) to be completely described and archived. The automation of the indexing process becomes then essential when the granularity of the video access increases.

A growing body of research studies the problem of how video indexing could benefit from the use of automated procedures [17–20] to make the process faster and cheaper. It is in this context that the use of computer vision techniques can help. A diagram showing the video analysis flow is reported in Fig. 1, where the contributions from *image processing* techniques are put into the wider context of video annotation and retrieval.

The most basic contribution is to automatically decompose the video stream into shots. This decomposition suggests temporal limits for the annotations to the professional archivists, and eases nonlinear access to the video stream. Motion analysis techniques can be used to classify camera work (e.g. panning) and to detect and track moving objects. Besides providing useful indices for subsequent retrieval, this analysis is useful for the new compression standard MPEG-4 [21], which is able to code separately multiple video objects. These objects could then be accessed individually to assist the presentation and manipulation of the original scene in a flexible way. General object recognition is still outside the reach of computer vision techniques, but, within limited domains, object detection and recognition are now possible. Two important examples are faces and captions, which often have the role of *highlights* (e.g. within newscasts). Once shots have been detected, representative frames (*key frames*) can be selected automatically, using image processing techniques. A video stream can be then abstracted and transformed into a (very) reduced set of still images. Audio analysis can also provide dialogue transcripts with important hints to video content. The information available at this stage can be used to structure the video into scenes and story units, which are at a semantic level appropriate for video browsing and retrieval. The usefulness of key frames is not limited to the compact presentation of video sequences or to the computation of the abstract. They can be described by their color distribution or texture quality and can be accessed using novel techniques such as visual query by example [22–25]. The importance of annotation of multimedia material is so high that a new member of the MPEG family, MPEG-7: *Multimedia Content Description Interface*, has been created to unify the efforts in the field.

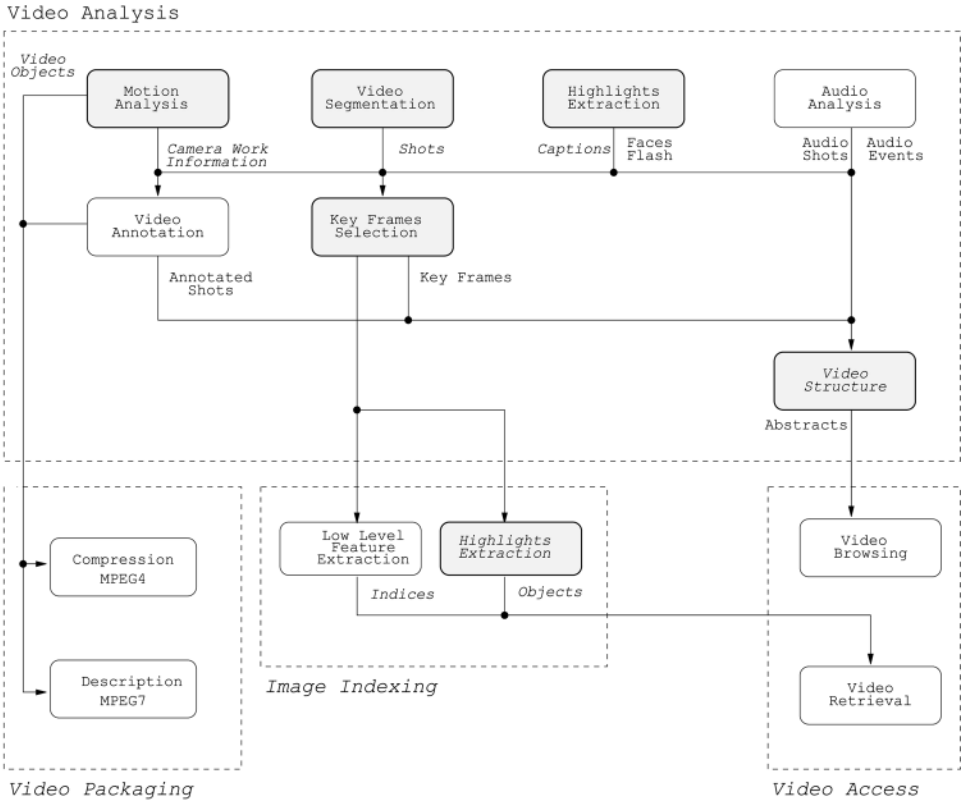


FIG. 1. The structure of a computer-assisted video indexing system. The shaded blocks are analyzed in the present paper.

Research in (semi)automated video indexing will then be useful in the following video management tasks:

- support to the activity of professional video archivists;
- packaging of annotated and compressed audio visual material (MPEG-4,7)
- development of video browsing/retrieval tools and interfaces for edutainment and video production.

The structure of this paper is based on the diagram of Fig. 1. Section 2 presents techniques for the segmentation of a video stream into shots and Section 3 considers the extraction of highlights. Motion analysis is considered in Section 4, while key framing and video structure recovery are covered in Sections 5 and 6, respectively. Each section presents a set of algorithms and is ended by some remarks which provide an assessment of current technology and discuss open issues. The state-of-the-art is then summarized and put into future perspective in the final section. A short glossary ends the paper.

2. SHOT DETECTION

Indexing text requires the use of words and phrases as pointers to sentences, paragraphs, pages or documents. In a similar way, indexing video material requires the selection of key frames and sequences as pointers to higher level units, such as scenes or stories. The

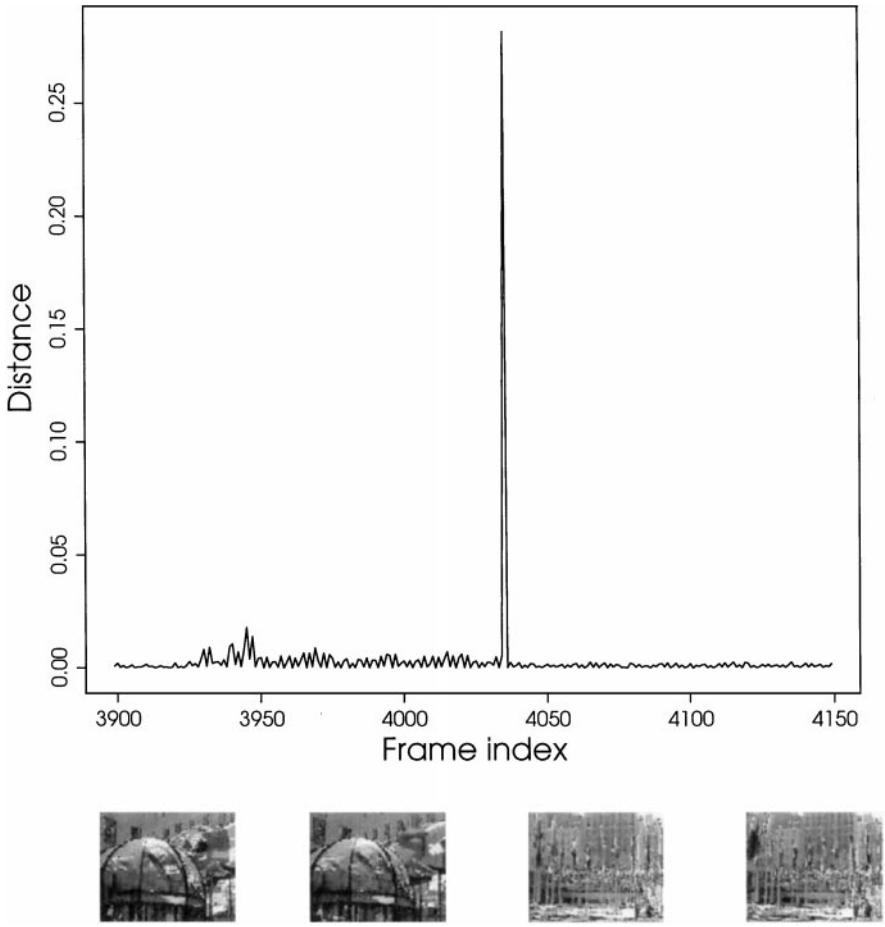


FIG. 2. An example of cut detection using luminance histogram differences.

most basic video unit, after the single frame, is the *shot*, a contiguously recorded sequence. Different shots are concatenated in a variety of ways, the simplest being a *camera break* or *cut*, characterized by an abrupt change from one frame to the next (see Fig. 2). Shots can also be concatenated by edited transitions such as fades in/out, dissolves, and wipes (see Fig. 3). Several algorithms for the detection of shot boundaries are presented in this section.

2.1. Cut Detection

Recent research on cut detection may be grouped in two categories. The first category consists of cut detection methods that use video in uncompressed form. The second category consists of methods that directly detect cuts in compressed video.

2.1.1. *Algorithms on Uncompressed Video*

The methods in this category are generally frame difference methods based on pixel-by-pixel comparison or on the distribution (histograms) of image values (color or luminance) on the entire frame (or on a set of covering subregions). Many algorithms rely on the use of histogram comparisons, because the global nature of histograms makes them less sensitive

to the typical changes within a shot, thereby reducing the number of false positive with respect to pixel-by-pixel comparisons.

A compilation of several algorithms is presented in the following paragraphs.

- Three frames approach [26]. Let r, s, t be three consecutive frames and D_{rs}, D_{st} the measure of frame dissimilarities; the observer motion coherence OMC, defined by

$$\mathrm{OMC}(r,s,t) = \left| \frac{D_{rs} - D_{st}}{D_{rs} + D_{st}} \right| \tag{1}$$

returns a value close to one if there is a change in the three frames under consideration, and a value close to zero otherwise. The authors proposed the two dissimilarity measures,

$$D'_{ij} = \frac{\sum_{x,y}[F_i(x,y) - F_j(x,y)]^2}{\sqrt{\left[\sum_{x,y} F_i^2(x,y)\right]\left[\sum_{x,y} F_j^2(x,y)\right]}} \tag{2}$$

$$D''_{ij} = \frac{\sum_{x,y} |F_i(x,y) - F_j(x,y)|}{\sum_{x,y} |F_i(x,y) + F_j(x,y)|}, \tag{3}$$

where $F(x,y)$ represents the image luminance at pixel of coordinates (x,y).

- Pixel-by-pixel comparison [27]. The absolute intensity difference between corresponding pixels of consecutive frames is computed and a camera break is assumed whenever the percentage of pixels whose difference is above a given value is greater than a given threshold. The same method can be applied to color pixels, e.g. by taking the average RGB differences. A much more complex algorithm based on a statistical model of the *range of pixel value changes* is given in [28].
- Likelihood ratio [27]. The comparison operates at the level of blocks. An image is subdivided into a set of blocks and a likelihood ratio L_i is computed for each corresponding block, in tth and $(t+1)$th frames,

$$L_i = \frac{\left[\frac{\sigma_t^2 + \sigma_{t+1}^2}{2} + \left(\frac{\mu_t - \mu_{t+1}}{2}\right)^2\right]^2}{\sigma_t^2 \sigma_{t+1}^2}, \tag{4}$$

where μ is the arithmetic average and σ is the standard deviation of the intensity histograms. A camera break is assumed when a sufficient fraction of blocks satisfy $L_i > v$, where v represents an appropriate threshold.

- Histogram comparison [27]. The area between the intensity (or color) distribution of two consecutive frames is computed:

$$\sum_j |H_{t+1}(j) - H_t(j)|. \tag{5}$$

A camera break is assumed whenever the area gets greater than a given threshold. Color distributions (using the RGB space) are computed by using a six-bit code (two bits per component). A variant to using the area between the distributions is given by the χ^2-test to compare two binned distributions,

$$\sum_j \frac{(H_{t+1}(j) - H_t(j))^2}{H_{t+1}(j) + H_t(j)}, \tag{6}$$

where $H(j)$ represents the jth bin. Another commonly used formula is the so-called *histogram intersection*,

$$\sum_j \frac{\min(H_{t+1}(j), H_t(j))}{\max(H_{t+1}(j), H_t(j))}. \tag{7}$$

Other histogram differencing formulas are used and some of them are compared in [29]. Differences among the separate histograms of multiple color components can be combined through a weighted average. The performance of the algorithm depends on the chosen color space; several color spaces (RGB, HSV, YIQ, L*a*b*, L*u*v*, Munsell) are compared in [29] and the Munsell space is found to be the best performer. Equations (5), (6), (7) can be used for multidimensional histograms (see also [30]) with no modifications.

- Incremental ratio [31] of between frames differences:

$$\delta = \frac{D(f_{t+1}, f_t)}{D(f_t, f_{t-1})}. \tag{8}$$

The difference between two successive frames is obtained by dividing each frame into a grid of n^2 subregions, each of them originating a difference value D_j according to

$$D_j^{\text{red}}(f_t, f_{t+1}) = \sum_i a_i \left| m_i^{\text{red}}(t+1) - m_i^{\text{red}}(t) \right| \tag{9}$$

and similarly for the other color components, where $m_i^{\text{red}}(t)$ represents the ith moment of the histogram of the *red* component for the tth frame, and $\boldsymbol{a}$ is a vector of parameters experimentally tuned. The between frames difference is computed by adding the three color differences for each region, discarding the k largest values, and averaging the remaining ones. If δ is above a given value, a camera break is assumed.

- Yakimovsky likelihood ratio test [26].

$$y = \left(\frac{\sigma_0^2}{\sigma_{t-1}^2}\right)\left(\frac{\sigma_0^2}{\sigma_t^2}\right), \tag{10}$$

where σ_{t-1}^2, σ_t^2 represent the variances of the pixel luminance values of the past and current frames while σ_0^2 is the variance of the pooled data from both the histograms. A camera break is said to occur between the previous and the current frames if the value of y exceeds a certain threshold.

- Kolmogorov–Smirnov test [26]. It is based on calculating the cumulative distributions $C_1(x)$, $C_2(x)$ of the pixel luminances in two consecutive frames and measuring the maximum absolute difference between them. If the distributions are approximated by histograms, a defective estimate of the distance is obtained by

$$D = \max_j |C_1(j) - C_2(j)|, \tag{11}$$

where j denotes the bins.

- Net comparison algorithm [32]. A set of nonoverlapping image regions is considered. The regions, which do not need to cover the whole image, are compared at successive frames by computing the difference of the average luminance values. Whenever the number of regions exhibiting a difference above a given threshold exceeds a predefined level, a camera break is assumed.

Multipass approaches are also introduced in [27] to reduce the computation time. A first scan of video material is done using a reduced threshold for camera breaks and undersampling in time the available frames. The video stream is then examined in correspondence of the candidate breaks at full temporal resolution.

2.1.2. *Algorithms on Compressed Video*

Increased efficiency is the major motivation for the second category of methods, using compressed video. Certain coefficients in encoded video, e.g. DCT coefficients in MPEG or Motion-JPEG video [33], carry information that can be directly used to detect cuts, saving unnecessary decompression. The MPEG coding standard relies on three different kinds of frames: intrapictures (I), predicted pictures (P), and interpolated pictures (B–for bidirectional prediction). A frame of type P is predicted from the previous I frame, while a B frame is predicted and interpolated from its preceding and succeeding I and P frames. The residual error after motion compensation is then DCT coded. When a block-residual error exceeds a given threshold, motion compensation prediction is abandoned and straight DCT coding is used. High residual error values are likely to occur in nearly all blocks across a camera shot boundary, making the fraction of blocks for which no motion prediction is available a good indicator of the presence of a camera break [34]. The same algorithm can be applied to H.263 streams [35] that also rely on motion compensation for coding. Similar algorithms are presented in [36, 37].

A method using a compressed video has been proposed in [38] and is based on the correlation between DCT coefficients of consecutive frames of Motion-JPEG compressed videos. A subset of DCT blocks corresponding to n connected regions are chosen for the comparison. For each block, a random subset of the AC coefficients is chosen and used to derive a vector representation for each frame in the sequence. Successive frames are compared using the inner product Ψ of the corresponding vectors. When the frames are similar, $|\Psi| \approx 1$. The value of $D = 1 - |\Psi|$ can be used as a measure of frame dissimilarity; a cut is detected whenever D exceeds a predefined threshold. A variant is presented in [34]; frame dissimilarity is computed by taking a normalized L_1 distance of the vectors and applied to the I frames of an MPEG stream. It should be noted that the low temporal resolution inherent to the use of I frames may introduce false positives that must be removed by subsequent processing.

Another algorithm working on a compressed Motion-JPEG video is presented in [39]; camera breaks are characterized by steps in the size of the compressed frames.

Patel ad Sethi [40] use an MPEG-1 stream to detect video shots. Besides comparing the global luminance histograms obtained from I frames, two additional comparisons are performed using both the row and the column intensity histograms. The authors use only I frames because they are the only accessible frames in MPEG video that can be processed independently of other frames. The histograms are computed using the first coefficient of each 8×8 DCT encoded block, which represents the average block intensity. A Gaussian averaging is performed on the histograms to reduce the sensitivity to intensity changes due to illumination and small motion. The comparison is done by applying the χ^2-test on the three pairs of histograms. The significance probability that each histogram pair is drawn from the same distribution is computed. The three significance probability values are then used to generate two comparison decisions for cut and gradual transitions.

A different approach that relies on an implicit static model is proposed by Ardizzone *et al*. [41]. They use a three-layer perceptron, whose input patterns are subsampled luminance images derived from an MPEG video stream.

The main advantage of methods working on compressed videos is that they do not require a complete reconstruction of the image. Some of the above algorithms only work on DC coefficients. This is equivalent to working on a subsampled version of the original image: the complexity is then 64 times smaller (as approximations to the DC coefficients for P and B MPEG streams are cheap to compute).

2.2. Gradual Effects Detection

Gradual transitions (edits) are an important class of effects that includes fades (in/out), dissolves (cross-fades), mattes, and wipes (see *Glossary*). A different kind of transitions is due to panning and zooming, but these are not usually related to shot changes. Up to the 1950s, dissolves were considered very specific punctuation signs within the filming language; a dissolve, for example, used to indicate a place shift within the same action or a flashback. The presence of specific effects could then be used as a clue for detecting scene changes, as opposed to simple shot detection.

The development of modern video editing technology has increased the use of edited transitions in video production. While the above-mentioned four classes of editing effects do not cover the full space of video editing effects (e.g. Adobe Premiere, a software tool for video editing, provides you with more than 75 different transition effects), they represent the most widely used effects. Furthermore, gradual transitions occur less frequently than cuts, reducing the impact of gradual effect detection on the overall performance of a video analysis system.

Several algorithms have been proposed so far to detect gradual transitions. Some of them are reviewed in the following sections.

2.2.1. *Plateau Detection*

Comparison based on sucessive frames alone is not adequate for the detection of gradual transitions because changes are small when an edit is present. An alternative is to use every kth frame instead, i.e. to perform uniform temporal subsampling. However, the larger separation between two frames, used for comparison, implies larger difference statistics within a shot, especially in the case of camera/object motion. The phenomenon of a very sharp peak in the middle of very small variations no longer holds. A detector for gradual transitions can be obtained as follows [42]. Let us compare the ith frame to the following $(i+k)$th frame. We obtain a sequence of delayed interframe distances $\{D_i^k = d(X_i, X_{i+k})\}$. If we choose k greater than the length of the gradual transition, measured in frames, the sequence $\{D_i^k\}$ exhibits a plateau of maximal width. A significant plateau at location i is characterized by a sequence of similar values D_j^k, $j = i-s, \ldots, i+s$, which are consistently higher than the preceding or successive values (see Fig. 3):

$$D_i^k \geq l \times D_{i-\lfloor k/2 \rfloor - 1}^k \quad \text{or} \quad D_i^k \geq l \times D_{i+\lfloor k/2 \rfloor + 1}^k, \quad l \gg 1. \tag{12}$$

The value of s is proportional to the difference between k and the transition length. The method applies to linear and nonlinear gradual transitions; it is the shape of the rises and falls at the plateau boundaries that changes. False detections can be reduced by requiring

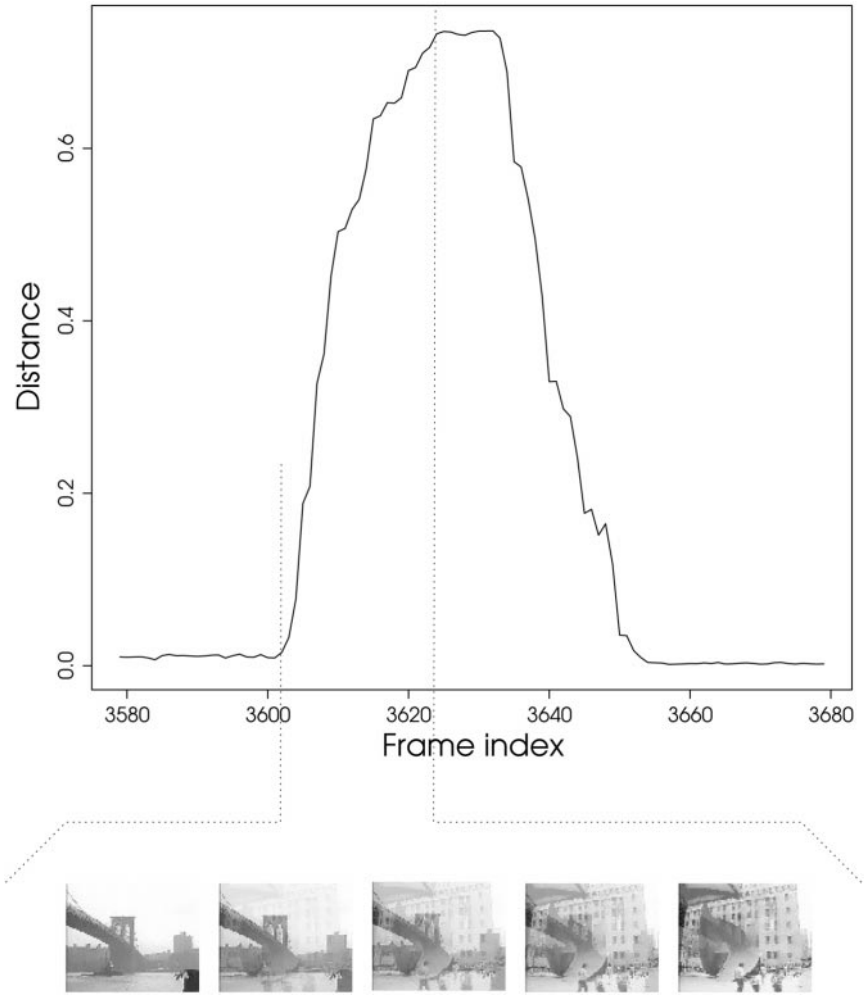


FIG. 3. An example of dissolve detection using the plateau algorithm.

that the plateau values be higher than the average frame differences at the hard cuts. A similar approach is reported in [43].

2.2.2. *Detection by Modeling*

Another important approach to the detection of video effects has been proposed in [31, 44] and is based on *effect modeling*. Mathematical models of edited transitions based on video production techniques are formulated and used to systematically design the feature detectors for spotting them in digital video. The shot boundary detection approach requires analytical (or at least statistical) models of the shots while the edit detection approach requires models of the edits. However, the space of all possible shots is ill defined while the space of edits is much smaller. A dissolve operation from scene X to scene Y is a sequence of frames represented by

$$E(t) = \left[\frac{(t-T_2)}{L_2}Y\right]_{t\in[T_2,T_2+L_2]} + \left[\left(1-\frac{(t-T_1)}{L_1}\right)X\right]_{t\in[T_1,T_1+L_1]}, \qquad (13)$$

where T_1, T_2 represent the times at which the (intensity) scalings of X, Y begin and L_1, L_2 are the durations for which the scalings last. A fade-in is a special case with $X = 0$ while a fade-out is characterized by $Y = 0$. If we look at a fade-out, we can see that, under the assumption that the only source of variation is the scaling

$$F(t) = \frac{\frac{\delta E}{\delta t}}{X(x, y, t)} = -\frac{1}{L} \tag{14}$$

holds, where L represents the length of the dissolve, expressed in frames. An indicator of constancy is suggested in [44, 45]

$$\text{Constancy} = \frac{\mathcal{N}(X)}{\sigma(X)(1 + |C_x - c_x| + |C_y - c_y|)}, \tag{15}$$

where $\mathcal{N}(X)$ is the number of nonzero pixels in image X, $\sigma(X)$ is the standard deviation of the pixel values, and (C_x, C_y), (c_x, c_y) represent the centroid of image X and the geometric image center, respectively. The different possible dissolves can be classified according to their start/end times. The method is effective, except when very similar sequences are being dissolved with precisely equal fade rates over the dissolve. A similar approach can be used to detect spatial edits, such as translates. As shown in [44], assuming that there is no scene action in progress during the edit

$$T(x, y, t) = \frac{\frac{\delta E}{\delta t}}{\frac{\delta E}{\delta x}} = \alpha_x \tag{16}$$

holds when the edit is a pure translation in the x direction, where $E(t)$ represents the sequence of edited frames. Again, the effect can be detected by looking for a constant image $T(x, y, t)$.

A method for detecting *mattes* is proposed in [31]. They are similar to fades, apart from the fact that luminance varies over the frames following a geometrical law. Once a fade has been detected, the central frames are considered; if the luminance has a big discontinuity, due to the presence of a black mask that partially covers the frame, a matte is detected.

2.2.3. *Feature-Based Detection*

Another interesting approach to the detection of gradual effects, as well as hard cuts, has been presented in [46]. During a cut or a dissolve, new intensity edges appear far from location of old edges. Edge pixels that appear/disappear far from existing edge pixels are considered as *entering*/*exiting* edge pixels. Cuts, fades and dissolves can be detected by counting the entering and exiting edge pixels, while wipes can be detected by looking at their spatial distribution. The algorithm is based on the following steps:

- frames F_t and F_{t+1} are aligned using a global motion compensation algorithm;
- edges are computed by applying the Canny algorithm to a smoothed version of the frames;
- the binary edge maps are dilated by radius r, so that the condition on the mutual distance of edge pixels can be easily verified by set intersection;
- the fraction of *entering* edge pixels ρ_{in} and *exiting* edge pixels ρ_{out} are computed.

Breaks are detected by looking at the edge change fraction $\rho = \max(\rho_{\text{in}}, \rho_{\text{out}})$. A cut leads to a single isolated high value of ρ while the other scene breaks lead to an interval where ρ's value is high. During a fade-in the value ρ_{in} is much higher than ρ_{out}. The reverse happens for fade-outs. A dissolve is characterized by a predominance of ρ_{in} during the first phase and of ρ_{out} during the second phase. The technique works properly also on heavily compressed image sequences.

2.2.4. *Twin Comparisons*

A simple technique is introduced in [27] to detect gradual transitions: the *twin-comparison* approach. The basic idea is that the frames before and after a gradual transition are usually markedly different. Histogram differences between successive frames in an edited transition are smaller than those across a camera break, but higher than those within a single shot. A reduced threshold can then be used to detect potential edited transitions. For each of the resulting candidates a progressive difference is computed over the subsequent frames; if the cumulative difference exceeds the threshold used for camera breaks, an edited transition is assumed.

2.2.5. *Step-Variable Method*

Xiong *et al.* [47] propose a novel algorithm to detect gradual transitions that extends the idea behind the Net comparison method [32]. The new method, called the Step-Variable method, subsamples video data, both in space and in time, and can detect both camera breaks and edits. The difference D_{ij} between frame i and frame $j = i + S$ is computed, where S is larger than the longest expected gradual transition. If D_{ij} is lower than a predefined threshold, frames i and j are moved forward by $S/2$, and the comparison is done again. Otherwise, frames i and j are gradually collapsed until $D_{(i+1)j}$ and $D_{i(j-1)}$ are not significant any more. If $i - j = 1$, a camera break is declared between i and j. Otherwise, a gradual transition is declared.

2.3. Remarks

The methodology for measuring the effectiveness of shot detection algorithms relies on the comparison with a ground truth derived from human annotation. If an algorithm requires parameter tuning (e.g. selection of an optimal threshold), the material should be divided into three disjoint sets: a training set, a validation set, and a test set. Parameters are chosen, optimizing the performance measure on the training/validation sets, and the performance of the tuned system on the test set is finally reported as an estimate of the algorithm performance. A commonly used parametric performance measure is

$$\mathcal{C}(\alpha, \boldsymbol{v}) = \alpha S_m(\boldsymbol{v}) + (1 - \alpha)\, S_f(\boldsymbol{v}), \tag{17}$$

where $S_m(\boldsymbol{v})$, $S_f(\boldsymbol{v})$ represent missed/false shot boundaries with respect to the available ground truth, $\boldsymbol{v}$ the tunable algorithm parameters, and $\alpha \in [0, 1]$ represents the trade-off between precision and recall. When $\alpha = 1$, maximal importance is given to missed transitions and over-segmentation will be favoured. A low value of α will under-segment the video, while $\alpha = 0.5$ provides a balanced result. Optimal parameters are then obtained by solving $\min_{\boldsymbol{v}} \mathcal{C}(\alpha, \boldsymbol{v})$. A complete characterization of each algorithm is given by its ROC (receiver operating curve), providing the number of missed boundaries as a function of false

TABLE 1
Average Values of Weighted False Alarm/ Missed Detection Errors for Cut Detection using Histogram Comparison on Different Color Spaces Using Eq. (17) with $\alpha = 0.25$

Cut detection	
Color space	Performance
LUV	0.343
MTM	0.343
OPP	0.352
HSV	0.374
LAB	0.380
YIQ	0.390
XYZ	0.430
RGB	0.455
YYY	0.466

Note. The performance is averaged over the bin-to-bin difference and histogram intersection methods which were found to be the best performing ones. The testing material contained only cuts. As most of the color space denominations are standard, we just note that MTM stands for *Mathematical Transform to Munsell*, YYY uses only luminance information, while OPP stands for opponent color axis space and is defined by $R-G$, $2B-R-G$, $R+G+B$.

boundaries. Extensive tests of color space influence on the performance of histogram-based methods are described in [30] and reported in Table 1. Extensive tests of algorithms have been reported in [48, 49] and the results are summarized in Tables 2 and 3. Let us note that the algorithms for cut detection are usually able to detect gradual transitions when they are applied to a temporally subsampled video stream or when a very low triggering threshold is

TABLE 2
The Performance of Some Commonly Used Algorithms for Video Segmentation

Shot detection			
Algorithm	Ref.	Correct (%)	False (%)
Absolute frame difference	[48]	73	56
Red histogram difference	[48]	94	175
Weighted histogram difference	[48]	94	135
χ^2 red histogram diff.	[48]	95	137
Red histogram intersection	[48]	94	174
Pure moment invariants	[48]	54	105
Range of pixel value changes	[48]	95	68
Feature based	[48]	92	59
Luminance histogram	[49]	99	521
Twin comparison	[49]	98	151
16 region histograms	[49]	96	205
Motion compensated differences	[49]	95	541

Note. The percentage of correct and false detections are given with respect to the true number of shot boundaries. For each algorithm the maximum performance in terms of found shot boundaries is reported. The testing material contained both cuts and edited transitions. The reference column reports the article describing the test.

TABLE 3
The Performance of Several Algorithms Working on Compressed Video Streams

Shot detection: Compressed video		
Algorithm	Correct (%)	False (%)
Armand [38], Zhang [34]	95	2490
Meng *et al.* [36]	65	338
Liu and Zich [37]	30	42
Yeo and Liu [95]	69	4
Sethi and Patel [26]	45	274
Shen and Delp [43]	68	69

Note. The percentage of correct and false detections are given with respect to the true number of shot boundaries (only cuts are considered).

used. The results in Table 2 show that the corresponding recall is quite high, but precision is adversely affected.

The performances of available algorithms for cut detection (with the exception of the methods based on pixel-by-pixel differences) are high. Detection of edited transitions, in particular dissolves, is proving more difficult and current algorithms need to be improved. The main reason for the unsatisfactory performance stems from the assumptions made in algorithm design. In particular, the assumption of stationary scenes at cross-fades is often violated. However, edited transitions account for a smaller percentage of shot boundaries than cuts, thereby reducing the impact of the corresponding algorithm performance on the final quality of video segmentation. This is true in particular for some categories of video, most notably raw footage to which video producers refer for the creation of new programs.

Overall, histogram-based algorithms provide good performance and speed and can be used to detect both cuts and dissolves. Furthermore, histograms provide a compact description of images that can be reused in other processing steps of a visual indexing system: image description for retrieval purposes, selection of representative frames (see Section 5), and video abstracting (see Section 6), by characterizing the *color mood* of shots and scenes. Algorithms relying on motion vectors provide high speed (if the motion vectors are available as in MPEG-1,2 or H.263) and motion information can be used in other important tasks, namely the classification of camera work, motion indexing, and detection of action scenes. Algorithms working only on the I frames of the MPEG sequences suffer from a higher number of false detections and do not provide a frame-accurate detection of shot boundaries, due to the lag between I frames.

An important open issue in the field of video segmentation is the synergetic integration of multiple shot detection algorithms. Work along this direction should provide effective strategies for combining computationally cheap detectors (missing only a few shot boundaries, at the cost of producing many false positives) acting as preprocessing filters, with more sophisticated (model-based) detectors reducing the number of false positives to an acceptable level.

3. HIGHLIGHTS EXTRACTION

Detection of shot boundaries is an important step towards the automated extraction of meaningful video segments. A further step is to characterize the resulting shots by extracting

TABLE 4
The Performance of Face Detection Algorithms on a Common Database

Face detection		
Algorithm	Correct (%)	False (%)
Osuna *et al.* [54]	74.2	13
Sung *et al.* [51]	79.9	3
Lew and Huijsmans [52]	94.0	43
Rowley *et al.* [57]	92.9	43

Note. The percentage of correct and false detections are given with respect to the true number of faces.

some high-level information from the corresponding frames. This information could derive from the recognition of well-defined objects within the frames (such as frontal views of faces), from the reading of captions, or from the detection of well-defined events (such as flashlights). The following sections survey algorithms which are capable of detecting such highlights.

3.1. Faces

Locating and recognizing faces in video sequences can provide useful information for segmenting the video stream (let us consider the typical structure of a news program, when the different segments are introduced by the anchor person) and for indexing it (e.g., by recognizing the actor or politician present in a given video segment). Also, knowing the image size of the detected faces can help in classifying the shot as close, medium, or far, an important information for video producers. Several successful algorithms for face detection have been reported in the literature and are presented in the following sections (see also Table 4).

Approaches based on principal component analysis (to focus on the relevant dimensions in pattern space) and probability density estimations, e.g. using mixture models, have proven to be particularly effective for this task [50, 51]. An information-theoretic approach to the selection of an optimal face template is presented in [52].

Moghaddam *et al.* [53] apply an unsupervised technique for visual learning to the probabilistic visual modeling, detection, recognition, and coding of human faces. This technique is based on density estimation in high-dimensional spaces using an eigenspace decomposition. These probability densities are then used to formulate a *maximum-likelihood* estimation framework for visual search and target detection for automatic object recognition and coding.

A rather novel approach, using *support vector machines*, a new method for training polynomial and neural network classifiers, is reported in [54].

An efficient system for detecting faces in MPEG video streams has been proposed in [55]. The algorithm consists of three main steps:

1. selection of possible face locations using skin-tone statistics;
2. detection of rectangular image regions satisfying appropriate shape constraints (size and aspect ratio);
3. verification of face detection by analysis of the energy distribution of DCT coefficients.

The first two steps are carried out using only the DC coefficients of the MPEG macroblocks; there is no need to completely decode the frames. The third step requires the complete set of DCT coefficients and can be carried out efficiently only for I frames.

An alternative approach to detect faces is presented in [56]. A large set of face examples and backgrounds (*counterexamples*) is used to train a Kohonen network. This results in a compressed base due to the vector quantization properties of the Kohonen networks. A multilayer perceptron (MLP) is extensively trained on this base to output a class (face or background) and a confidence value for each input pattern. The compressed base is then tuned to a specific environment by adding relevant faces and backgrounds. The MLP is then briefly trained (*tuned*) to the environment. The detection phase consists in scanning each image of a sequence at various resolutions. For each location and size of the scanning window, the window content is normalized to a standard size and then normalized in mean and variance to reduce the sensibility to lighting conditions. All local decisions are then fused spatially, taking into account the relative position of the localized faces.

Another neural network-based face detection system is presented in [57]. Rowley *et al.* use a retinally connected neural network examining small windows (20×20 pixels) of a gray-scale image and deciding whether each window contains frontal views of faces. They use a bootstrap algorithm for training the net to eliminate the task of manually selecting nonface training examples, which must be chosen to span the entire space of nonface images.

3.2. Captions

Captions in video programs such as news, documentaries and coverage of sport events provide important information on the underlying stories. They can be used to reinforce spoken words, to highlight the time stamps of the events or to present additional information. In sports, they are used to inform on the current score and timings of the games and offer a possibility to log automatically salient events. Sato *et al.* [58] claim that only one half of the content of superimposed captions is present in the audio or closed caption information and, therefore, textual information can be used to improve video indexing. After a caption has been located, it could be isolated from the background and then read by an OCR program, or the frames with text could be eliminated from the video stream, e.g. for the insertion into new productions, depending on the application (Table 5).

There are several approaches to locating captions in a video stream [42, 59, 60–62, 58, 63–65]. Some of them focus on the detection of caption events. Others aim at detecting text within a single frame (for example, key frames or captioned frames previously detected)

TABLE 5
The Performance of Some Caption Event Detection Algorithms

	Caption events	
Algorithm	Correct (%)	False (%)
Yeo [42]	95	3
Ariki [59]	92	0
Sato [58]	92	?

Note. The percentage of correct and false detections are given with respect to the true number of captions.

and use the temporal dimension for validating the text extraction and reading, tracking it frame by frame.

The basic observation of the algorithm presented in [42] is that a caption most often appears or disappears in the middle of a video shot. The extraction of an embedded caption is then achieved by the detection of its appearance/disappearance: the *caption event*. The appearance of the caption can be abrupt or gradual. Abrupt captions are characterized by high interframe pixel differences which are often localized at the bottom of the image. The algorithm presented in [42] uses the low-resolution DC images of an MPEG stream. Image differences over the lower fourth of the frame are considered within a shot span. Caption appearance/disappearance events are characterized by local maxima of image differences and are differentiated among themselves by the amount of gradient in the vertical direction. The frame is selected as a start caption event if this value is higher than that in the previous frame. The detection of gradual caption proceeds along similar steps with the exception that the cumulative pixel difference is not computed on adjacent frames but on frames that are k (typically 20) frames apart.

Another work following the approach based on interframe detection and within-frame reading is presented in [59], where TV news articles are analyzed in order to extract frames containing superimposed captions for indexing purposes. The main assumptions on text are: uniform color and brightness, clear character edge, disjoint characters, stationary and horizontally aligned text. Captioned frames are located by the following steps: (1) for each frame compute the number of bright pixels having approximately the same gray-level in the subsequent frame; (2) mark the frames in which this number falls in a certain range; (3) search for a sequence of at least 21 marked frames. An average frame is computed by averaging the frames in the found interval. Horizontal edges are then computed and rectangular regions corresponding to text lines are located by analyzing the Y- and X-*signature* of the edge map. Small regions are filtered away. After the binarization and noise removal of each text region, characters are recognized. After a morphological analysis, noun words are extracted and used as indices of the TV articles. Articles are then classified by topics into 10 classes by using an index–topic table that includes about 12,000 indices.

A straightforward strategy can be found in [58], where captions are detected to help in the generation of a short synopsis of the original video sequences. A caption region is taken to be a horizontal rectangular structure of clustered sharp edges, primarily due to characters with high contrast intensity. The detection algorithm proceeds by application of a horizontal differential filter, thresholding to extract edge features, smoothing, clustering, and boundary rectangle computation. The image quality is improved by (1) a subpixel interpolation in order to increase the resolution of each caption and (2) multiframe integration in order to reduce the variability in the background and to increase the contrast. Characters are extracted by means of the integration of the results of four directional filters applied to the preprocessed image. Experiments were performed on seven 30-min programs with 256 captioned frames, 392 text lines, 961 words. The detection rates are 91.8%, 89.6%, and 76.2%, respectively. False detections have not been reported. The word recognition rate is 70.1%.

A different approach that is not limited to static captions but is suitable also for moving text is presented in [62]. Starting from features of artificial text and text in commercials, each frame is analyzed in order to produce a bitmap containing only the textual information. Each frame is segmented using a split-and-merge algorithm. Regions with low contrast or unacceptable geometry are removed from further consideration. Once candidate characters

are detected, they are tracked across frames by means of feature comparison, building chains of characters from their appearance to their disappearance in the video sequence. Linear moving text, as well as text over moving background, benefits from this tracking. Characters are then clustered into text objects and fed to an OCR program. The responses from multiple instances of each character can be integrated using a majority rule. Experiments were performed on 10 videos for each of three different classes (feature films, commercials, newscasts) for a total of 22 min of video. The total number of characters was 3059, thereof correctly segmented 2848; segmentation rates within each class were 96%, 66%, 99%, respectively. The character recognition rate ranged from 41% (newscasts) to 76% (title or credit sequences in films).

In [61] a method for the automatic extraction of text lines from color video frames is presented. Target frames are selected from shots detected by a scene-change detection method. The input image is segmented by histogram color quantization in the RGB space with the Euclidean distance. A threshold controls the number of resulting regions. Each color plane gives rise to a binary image whose components are independently analyzed. After an heuristic filtering step an *Y-signature* is computed for each bitmap; for each row an index, 0 or 1, is computed which depends on the number of runs and on the number of foreground pixels in the row. Long sequences of consecutive 1's in the signature, along with some geometric considerations, identify line candidates. A second heuristic filter is applied to remove components which contain long runs or wide filled boxes. For each line candidate the vertical projection profile is computed. *Text segment* candidates are identified by the intervals in the profile with consecutive positive values. A test depending on several geometric characteristics is defined in order to establish if a text segment candidate is an actual text segment. Finally a third heuristic filtering is applied to the text segments which pass the previous test. The experiments were performed on 50 images with 124 text lines giving 86% detection rate. False detections were not reported.

In [60] an analogue method for the identification of caption and noncaption text in color images (and therefore in video frames) is proposed. The reported accuracy is 72% for 6952 video frames, selected from eight different videos. It is subjectively computed by considering the correctly located important text in the images.

3.3. Flashlights

In news video, it often happens that important events, such as the appearance of an important person, are marked by the light of the photographers' flash units (e.g. [42] reports 49 events in a 5-min long sequence depicting a Democratic Convention in the U.S.). There can be multiple flashlights over a few frames originating a burst of false cut detections due to the large differences among the frame descriptors. An algorithm to detect flashlights is reported in [42], where it is observed that flashlights manifest themselves as two sharp peaks of approximately the same value in the plot of the interframe differences over a window of approximately half a second. As reported in [66] the recall and precision of the algorithm are 40% and 100%, respectively; as the algorithm tends to merge nearby flash events, the *practical* performance is actually higher.

3.4. Remarks

Detection of *generic* objects in images is still out of the capability of current algorithms. Effective algorithms have been developed for at least two classes of objects which are

relevant to video indexing systems: text and faces. Frontal appearances of faces can be detected efficiently and with good effectiveness (see Table 4). Detection of nonfrontal views must be improved, but this may not be particularly significant as far as video indexing is concerned. Face detection algorithms can be used to spot specific person appearances in video by exploiting the availability of transcripts (see [67], where a system is able to infer the name of a given unknown face by using face similarity measures and co-occurence of face pattern and nouns in the transcripts). Current face detection algorithms are not optimized for video streams; tracking of candidate faces across multiple frames could improve the effectiveness of the algorithms, as already has been done for text recognition.

Text can be located (see Table 5) and read within frames providing useful indices even if the word recognition rate is not very high (from 70% to 80%). Accurate detection of gradual caption events remains a difficult task and problems arise when there are rapidly moving objects in the captioned frames [42]. While the working hypothesis of the reported algorithms, such as text brightness and horizontal alignment, are adequate for many practical applications, their relaxation is one of the central problems in text lines location and extraction.

4. MOTION ANALYSIS

The most discernible difference between still images and moving pictures stems from movements and variations. In order to obtain a more precise and complete semantic information from video, we need the ability to classify objects appearing in a video sequence based on features such as shape or color, as well as their movements [68]. By describing the movements derived from motion analysis, a dual hierarchy is introduced, consisting of spatial and temporal parts for video sequence representation. Besides providing information on objects trajectories, analysis of motion is useful to detect objects, to recover the kind of camera operation (e.g. zoom, pan, tilt), and to create salient video stills by mosaicking several frames.

4.1. Spatio-Temporal Segmentation

The time-varying nature of video data can be used to facilitate automatic segmentation of independently moving objects and significant structures, providing a *layered* representation of video. The different layers are used to identify significant objects in the scene for feature computation and querying.

An algorithm for the computation of such representations is presented in [69, 70]. Coherent motion regions are identified iteratively by generating hypotheses of motion and classifying each location of the image to one of the hypotheses. Each hypothesis is defined by a set of six parameters describing an affine motion model. This model can describe motions typically encountered in video sequences, such as translation, rotation, zoom, shear, and any linear combination thereof. Image segmentation, based on the affine motion model results in the identification of piecewise linear motion regions. As a scene usually consists of many moving objects, it is necessary to use multiple motion models that compete for region support at each iteration. Regions characterized by the same motion model are grouped together into a *layer*, even if they are not connected. Each region can then be tracked over subsequent frames by using its descriptive motion model. The resulting set of layers can be used as the target for manual annotation and for tracking related objects (and eventually propagating their labeling across frames). Another important application is

in the new compression standard MPEG-4, where this information can be used to improve compression.

Ayer and Sawhney [71] describe a formalization of the layered representation, based on maximum likelihood estimation of mixture models and the minimum description length principle (MDL). These techniques are used to select the appropriate number of motion models, to compute the corresponding parameters, and to decide the spatial support layer for each model.

Another type of spatio-temporal analysis is introduced by Chang *et al.* [72]. They present VideoQ, a web-based video search system, where the user queries the system using animated sketches which assign motion and other attributes (spatio-temporal ordering, shape, and the more familiar attributes of color and texture) to any part of the scene. A collection of regions exhibiting consistency across several frames in at least one feature becomes a *video object*. The features of each video object are then matched against the features of the objects in the databases. In the database population phase, the individual videos are decomposed into separate shots and then, within each shot, video objects are tracked across frames. For each shot the global motion (i.e. background motion) is automatically estimated using a six-parameter affine model [73] and used to compensate the global motion component of all objects in the scene. Segmentation and tracking of image regions is based on the fusion of color, edge, and motion information computed using a hierarchical block-matching method [74]. Each object is characterized by the motion of its centroid and by its color, texture, and shape features. The VideoQ system proved successful in retrieving video clips such as soccer players, high jumpers, and skiers.

Another algorithm using spatio-temporal information is described in [75]. Sahoria has developed a system to analyze and index surveillance videos based on the motion of objects in the scene. A segmentation and tracking module extracts trajectories from compressed video using the motion vectors from the MPEG-1 compressed video. Median filtering is first performed to remove noise from the vector field which is then clustered using the L_1 norm on the space of motion vectors to extract regions with uniform motion. The tracking process attempts to match objects between two motion fields based on the amount of overlap of their segmentation maps. The resulting trajectories are used in a video retrieval system, where the search is based on hand-drawn queries. Sahoria's algorithm is more successful in tracking vehicles (93% correct detections) than people (78% correct detections). This is due to the complex motion patterns of people in the scene which are not accurately described by the MPEG motion vectors.

4.2. Camera Work

Camera operation information is very significant for the analysis and classification of video shots [76], since it often explicitly reflects the communication intentions of the film director. There are six main different camera works: panning, tilting, zooming, tracking, booming, and dollying. Each of these operations induces a specific pattern in the field of motion vectors from a frame to the next. Simple methods for detecting panning (tilting) and zoom operations have been proposed in [77, 34]. In order to detect camera operation, the motion vectors can be obtained by optical flow techniques [77] or by coding algorithms such as MPEG [34] or H.263. The first step aims at discriminating between static/motion scenes; this can be done simply by looking at the average size of the motion vectors. The motion vector field for any combination of panning and tilting will exhibit a single strong modal vector value which corresponds to the direction of camera movement. Most

of the motion vectors will be parallel to this vector. This may be checked by analyzing the distribution of the direction of the motion vectors; a pan/tilt is characterized by a small standard deviation of the distribution as noted in [77] or by a small absolute deviation from the modal direction as suggested in [34]. Zooming is characterized by a flat appearance of the direction distribution. Alternatively zooming operations are characterized by vectors of opposite sign at the frame edges. This means that the magnitude of the difference between vertical (or horizontal) components exceed the magnitude of both components. This simple approach can be fooled by the motion of large objects. More generally, the problem of recovering camera motion can be seen as that of estimating an affine transformation which accounts for the dominant global view transformation. A thorough treatment can be found in [78, 79, 73].

4.3. Motion Indices

In image processing, it is generally hard to extract moving objects from motion pictures and even harder to recognize the objects. However, it is possible to estimate the motion of objects without exactly detecting the objects. The use of object motion to index video material to subsequently retrieve scenes has been investigated in the literature [80–83].

Ardizzone *et al.* [84] present a motion-based video indexing technique related to the optical flow field. This method is based on *a priori* video segmentation to extract shots [41]. Once the shots are extracted, key frames are detected using heuristic techniques and characterized via their optical flow field. To compute this instantaneous velocity field a few frames adjacent to the key frame are needed. Once the optical flow field is computed, its content has to be coded to allow content-based queries. To this aim, the flow field is spatially split into four regions of equal size to preserve spatially related information. For each region, the flow field mean and the histogram of directions are computed.

The approach presented in [81] is based on a three-step procedure: the automatic estimation of motion vectors in the frame sequence, the description of motions in spatio-temporal space, and the retrieval of sequences. The motion vectors estimated by block-matching are mapped in (blocked) spatio-temporal space. The vectors are then aggregated into several representative vectors by hierarchical clustering. The representative vector of a cluster is defined as the vector which is closest to the mean vector of the cluster and has the longest life in the cluster. Adequate specification of the motion condition is a difficult problem for the user. A retrieval system should then take into proper account the scale, directional, and temporal uncertainty that usually characterize the queries. The solution proposed in [81] makes use of elastic matching to overcome the problem.

A different approach is proposed in [82]. The main idea is to describe motion using a restricted alphabet of 16 characters, each of which denotes a particular kind of motion (e.g. north, north-east, etc.). For each character, two 16-bit hash functions are used. These functions are employed to build the *appearance* and the *order signatures* by *or*-ing the keys. The *appearance signature* is computed using the motion classification in each frame, while the *order signature* is built using the motion classification of each successive frame. The resulting signatures are then used as motion indices on which the retrieval can be based. A framework which uses MPEG macro-block tracking is presented in [80].

4.4. Mosaicking Video

A compact video representation is an important step toward efficient video coding, indexing, and annotation. It has been observed [85] that representing a complete video shot

with a single image is a significant forward step for the video representation problem. A new class of images, *salient stills*, is introduced in [86]. These images do not represent one discrete moment of time as a single frame; one image reflects the aggregate of temporal changes that occur in a moving image sequence with the salient features preserved. By the application of an affine transform and nonlinear temporal processing, multiple frames of an image sequence, which may include variations in focal length or field of view, are combined to create a single still image. The still image may have multiresolution patches, a larger field of view, or higher overall resolution than any individual frame in the original sequence. It may also contain selected salient objects from any one of the sequence frames.

A common scenario in video sequences is that of a mostly fixed background being imaged with or without independently moving objects. The dominant background changes in the image plane are mostly due to motion and camera operations. Once camera work has been accurately recovered [87, 73, 86] for a set of frames, a reference frame can be chosen and all subsequent frames are *warped* into the coordinate system of the reference frame. The resulting image, the *mosaic*, is usually bigger than the constituent frames. Temporal median filtering of the mosaicked frames leads to the deletions of the outliers (moving objects) if their locations are not highly correlated over time. An interesting application is found in [88], where mosaics are used to describe the trajectories of soccer players.

4.5. Remarks

Very sophisticated algorithms have been proposed so far for camera work estimation and segmentation of multiple moving objects. Reliable estimation of motion between subsequent frames is a computationally intensive task. Good algorithms exist for the estimation of dense optical flow fields [89], but they are currently too complex to be applied to large video databases. Many motion/camera work algorithms rely on sparse motion fields obtained by block matching techniques similar to those employed by the MPEG encoders. The major drawback of these algorithms is that they do not account for possible affine transformation of the blocks being matched. This limitation is restrictive as, with the exception of pan/tilt camera operations and object motion parallel to the focal camera plane, image blocks can present severe affine deformations. The effect is more marked if the video material is digitized at a low frame rate.

5. KEY FRAMES EXTRACTION

Different algorithms have been proposed for extracting some representative *key frames* from shots, relying on the visual differences between shots [90, 91], or on the motion patterns [92]. The observation that directors frequently use camera motion to build complex messages into a single shot, and actors pause to emphasize gestures, leads to the idea of motion-based key frame detection. The algorithm presented in [92] uses optical flow analysis to measure the motion in a shot and selects key frames at the local minima of motion. In the first step the optical flow is computed using Horn and Schunk's algorithm. The sum of magnitudes $M(t)$ of the components of optical flow at each pixel are computed for each frame t,

$$M(t) = \sum_{i,j} |o_x(i, j, t)| + |o_y(i, j, t)|, \tag{18}$$

where $o_x(i, j, t)$, $o_y(i, j, t)$ are the horizontal and vertical components of the optical flow at

location (i, j). The second step identifies local minima of $M(t)$; the key frames are selected at the minimum of $M(t)$ in each interval. In the same paper the idea of a hierarchical key frame selection methodology is also proposed; the shot should be first classified into categories and an appropriate key frame selection algorithm should then be chosen.

Related to the idea of selecting key frames at local minima of motion activity is the proposal by Liu *et al.* [93] of detecting significant pauses in a video stream as useful indices. Their approach is based on a χ^2-test between the intensity distribution of the frame representing the possible start of a pause and the subsequent frames. Good results have been obtained even in the presence of slow camera motion.

The algorithm for key frame extraction, proposed in [91], is based on a variant of the algorithm used for shot detection in the same paper. The first frame of each shot is selected first as a representative frame and a suitable distance between images is chosen. Subsequent frames are then compared to the first frame, looking for a frame whose difference is above a given threshold T_S. If such a frame is found, it is considered as a key frame if it is followed by a continuous sequence of frames differing by at least T_S from the previous key frame and providing a cumulative difference $T_B > T_S$ from itself.

Another algorithm is proposed in [90] as an aid to compute efficiently the similarity of shots. To match the contents of video shots in a way that agrees with human perception of similar contents, similarity is more suitably measured in terms of the intersection of the flow of imagery paths. In particular the distance between two shots is defined as the minimum distance among the constituent frames. The proposed dissimilarity measure between images is based on the normalized difference of luminance projections,

$$d_{l_p}(f_i, f_j) = \frac{1}{255 \cdot (J+K)} \left(\frac{1}{J} \sum_{n=1}^{K} \left| l_n^r(f_i) - l_n^r(f_j) \right| + \frac{1}{K} \sum_{m=1}^{J} \left| l_m^c(f_i) - l_m^c(f_j) \right| \right), \quad (19)$$

where $l_k^r(f_i)$, $l_k^c(f_i)$ represent the luminance projection for the kth row and the kth column, respectively. Video shots are then clustered hierarchically by successive grouping of the most similar pairs. The resulting dendrogram provides a visual representation of the hierarchical cluster. A sudden jump in the proximity level of the dendrogram is used to select automatically the proper partition of the video shots. Computation of shot similarity as outlined above is expensive. The paper proposes a combined spatial–temporal subsampling of the video stream. The spatial subsampling is obtained by considering only the DC (or DC + 2AC) image components, as derived by an MPEG or Motion-JPEG stream. Temporal subsampling, i.e. selection of representative frames, is obtained by adding a frame to the shot representatives whenever its dissimilarity from the last selected representative is greater than or equal to a given threshold ϵ. As the dissimilarities used are normalized to the interval [0, 1], when $\epsilon = 0$ all frames are selected as representatives, while when $\epsilon = 1$ only the first frame is considered. Threshold ϵ can be chosen so that a very high percentage (approximately 95%) of the shot dissimilarities computed using only the representative frames is in good accordance (within 20%) with the values derived from the complete set of frames.

6. RECOVERING VIDEO STRUCTURE

Efficient and reliable methods for the segmentation of the visual part of a video into shots have already been reviewed in Section 2. In actual motion picture or video documents there can be up to 1000 cuts per hour. In order to browse quickly through the video material, both

FIG. 4. An automatically generated abstract (TOP), compared to a lightable presentation of a 5-min long video clip showing a single frame per shot. The abstract is obtained by clustering the key frames of the shots.

for indexing and retrieval, it is necessary to find objects which span longer time intervals than shots, possibly representing scenes, story units, or abstracts [36, 94–96, 90, 97, 88, 98] (Fig. 4). Capturing higher level video structure is the basis for effective video abstracting and is mainly useful in the following tasks:

- *fact finding*, where subjects try to locate video segments covering a specific event;
- *gisting*, where subjects want to get the essence of a video without looking at it in its entirety.

The following sections present some algorithms to recover video structure.

6.1. Video Abstracting

Pfeiffer *et al*. [99] give the following definition of reduced video representation or video abstracts:

> A *video abstract* is a sequence of still or moving images (with or without audio) presenting the content of a video in such a way that the respective target group is rapidly provided with concise information about the content while the essential message of the original is preserved.

A still-image abstract is a collection of extracted salient images (*key frames*) from shots (see Fig. 4). A moving-image abstract is a collection of sequences of images from the original movie and is a movie itself. Pfeiffer *et al*. [99] present techniques for automatically producing moving-image abstracts. The basis of their approach to video abstracting for feature films is based on the following observations:

- action scenes are important and should be included in the video abstract;
- processing of visual stimuli within the human visual system is mainly based on perception of contrast: high contrast scenes should be presented in the abstract;
- colors are very important to transfer emotions: scenes characteristic of the color mood should be included;
- the sequence of scenes must be preserved to provide consistent context;
- presented scenes must be long enough (at least 3 seconds) to provide the viewer with enough time to completely analyze them.

A set of consecutive shots is grouped into a single scene whenever the Aristotelian properties of unity of space, place, time, and action hold. While automated scene detection is currently in its infancy, some edit effects (such as fades and dissolves) can be considered an hints to changes of scene (not only of shot boundary). A single scene can also be inferred when a sequence of shots can be broken into two (or more) sequences, each of them built from similar shots (*shot reverse-shot* technique).

A simplified approach is proposed in [100, 29], where a hierarchical segmentation of the video stream is obtained by using a decreasing threshold for the detection of cuts using histogram differences. The first frame of each subsequence is taken to be the representative one. In order to avoid an excessive number of subsequences from fast-paced episodes, a minimum subsequence length is imposed. The ability of this method to obtain a hierarchical structure depends on the types of video sequences. Some characteristics of the derived structure could be profitably used to categorize the sequences (see also [101]).

The structure of a video can also be explored by looking at the distribution over time of several, simple shot descriptors. An interesting example is reported in [102], where the feature film *Indiana Jones and the Last Crusade* is analyzed. The plots of the activity of a shot (i.e. the average amount of net pixels changing during the shot), the duration of the shot, and the brightness of the shot point to the overall structure of the film. The plot of the activity highlights the four main action sequences of the film. These also correspond to subaverage duration of the shots; shots tend to get shorter when the action picks up. Another clue is provided by the average brightness of the shots; action-packed scenes tend to be brighter.

Another simple approach to detecting scenes has been proposed in [31]. After subdividing a video into shots, the first and last frames of each shot are selected as its representatives. A semantic analysis is performed on them, sequentially looking for *similar* frames, which reveal unity of action. The feature used to highlight natural color image similarity is the shape of their color histograms. Similarity G_{ij} of shots i and j is computed by integrating a local measure of similarity L_{ij} (such as cross-correlation) on the cumulative histograms of frames of the compared shots:

$$G_{ij} = \int L_{ij}(x)\,dx. \tag{20}$$

To make the measure more reliable, histograms are smoothed before computing the correlation. The similarity measure is applied to the three components of color. In order to infer the presence of a scene, the representative frames of one shot are compared with all the preceding frames that are supposed to belong to the same scene. If no match is detected, then a *confidence counter* associated with those frames is incremented by one; otherwise if a similar frame i is found, the counters of all the frames following i are reset, the current frame replaces frame i, and the inference is made that those shots all belong to the same scene. Whenever a frame confidence counter overcomes a threshold represented by the maximum allowed number of different shots in the same scene, then a scene end is detected if a unitary scene had been inferred or an absence of judgement about the semantics is concluded.

More sophisticated models are needed to capture a structuring of the video stream that closely correlates with human perception. In the following subsection we review three different approaches relying on visual similarity [96], general media-knowledge-based rules [103], and a program-specific model [104].

6.2. Video Structure by Visual Similarity

The whole presentation [96] is based on the concepts of time constrained clustering of video shots and on the construction of a scene transition graph to automatically parse a video program, extract story structures, and identify story units. The proposed framework for the video analysis can be decomposed into the following functional blocks:

1. *Shot segmentation*. The boundaries of shots are determined by detecting changes in the visual characteristics of frames.

2. *Time-constrained clustering of video shots*. The shots are matched and clustered on the basis of their visual contents and temporal localities to further condense repetitive shots of similar contents. Clustering is based on the shot distance,

$$\hat{D}(S_i, S_j) = \begin{cases} D(S_i, S_j), & d_t(S_i, S_j) \leq T, \\ \infty, & d_t(S_i, S_j) > T, \end{cases} \tag{21}$$

where $d_t(S_i, S_j)$ is the temporal distance between two shots and D is the shot distance defined as the minimum distance of constituent frames. The following constraint is imposed to the clusters: *The distance between a shot of cluster C_i and a shot of any other cluster C_j must be higher than the diameter of C_i*. Hierarchical clustering using a complete-link method satisfies this constraint and is used to generate shot clusters with a cluster-to-cluster distance defined as the maximum distance between any two shots.

3. *Building of scene transition graph (STG)*. The clusters from the previous step are structured as a directed graph: each cluster represents a node and a (directed) link (edge) from cluster A to cluster B is inserted whenever a shot in B is immediately subsequent to a shot in A. The STG representation allows some form of analysis of video through the analysis of the graph. One important feature in a graph is a *cut edge*, i.e. an edge whose removal disconnects the graph.

4. *Scene segmentation*. Story units are extracted by finding the *cut edges* of the STG. Each unit is a connected subgraph by itself.

The algorithm does not require *a priori* models and has been tested on a variety of programs: cartoons, movies, documentaries, and sitcoms.

6.3. Video Structure by Media-Based Rules

A quite different approach is presented in [103]. The algorithm is based on the observation that in a temporal medium, local clues need to be given to the viewer or listener in order to help him or her identify macroscopic changes. In motion pictures and videos, there are a wide range of local clues for macroscopic changes, from the use of special transition effects, modifications of soundtrack, changes of editing rhythm, and modifications of the type of images due to changes in settings or to special effects.

Let a shot S_i be characterized by its duration T_i, representative image R_i, introductory transition effect E_i (which can be a cut, C, or a gradual transition, G), and by its second and penultimate image SEC_i and PEN_i, respectively. The following rules, derived from intensive analysis of video documents, readings in film theory, and discussions with film and video directors, teachers, critics, and analysts, are proposed:

- *Transition effect rules*. The complete list of transition effects can be seen as a word on the alphabet $\{C, G\}$. Transition effect rules are based on the recognition of subwords which are of the form $C^i G^j C^k$:

R1. If $i > 2$, $j = 1$, $k > 2$, then there is a sequence limit at the beginning of the G transition effect.

R2. If $i > 2$, $j > 2$, $k > 2$, and S_m is the shot introduced by the first gradual transition effect, and S_{m+j} is the shot introduced by the last gradual transition effect, then there is a sequence which begins at most 2 shots before S_m and which ends at the end of S_{m+j-1}.

R3. If $i > 2$, $j = 2$, $k > 2$, and S_{m+1} is the shot surrounded by gradual transition effects, then it is likely that S_{m+1} is an important shot in a sequence.

- *Shot repetition rule*:

R4. If similar shots are found at a distance of no more than three shots, then there has been no sequence break.

- *Contiguous shot setting similarity rule*. The idea is to introduce a distance between shots which is related to the difference in the shot settings. A possibility is to compare the mean and the standard deviation of the distribution of hue-saturation vectors in the reference images:

R5. If the reference images $R_i, \ldots, R_k$ of a succession of shots $S_i, \ldots, S_k$ differ (according to an appropriate setting-distance) less than a predefined threshold T and are preceded and followed by shots whose reference images R_{i-1}, R_{k+1} differ from R_i, R_k, respectively, more than T, the shots $S_i, \ldots, S_k$ constitute a sequence.

- *Editing rhythm*. Montage rhythm is a powerful tool for the structuring of viewer perception. Rhythm changes can be detected by looking at the prediction error of an order-2 autoregressive modelling of shot durations:

$$T'_n = aT_{n-1} + bT_n. \tag{22}$$

R6. If $T_n > 2T'_n$ or $T_n < 0.5T'_n$, then S_n is likely to be the first shot of a new sequence.

- *Soundtrack rule*:

R7. If there is no music during 30 s and music appears in the soundtrack, then there is a sequence limit at the beginning of the music.

In order to obtain a macrosegmentation from these data, it is necessary to

- merge compatible starting and ending points;

- resolve conflicts by use of precedence rules (e.g., R1 and R7 are considered more important than R4 and R5);
- add segments where none was detected;
- choose representative images using the information on the rules which originated the sequences.

The algorithm does not associate any confidence with the recognition of shots and transitions and rule importance is not characterized in a quantitative way but only through precedence relations.

6.4. Video Structure by Program Modeling

Some video material presents a constrained structure that lends itself to simple yet accurate modeling. In such cases, it is possible to exploit *a priori* knowledge to parse the relevant programs. A television news program is a good example of this kind of video. The algorithm proposed in [104] consists of three steps:

1. *Temporal segmentation*. The video sequence is divided into shots. In this study, both the *pixel-by-pixel comparison* and *histogram comparison* algorithms are used.
2. *Shot classification*. Shots are classified into anchor person shots (*A shots*) and news shots. Three models are constructed for an *A shot*: region (anchor person, news icon, news program title bar, anchor person name bar, and background), frame (as a spatial arrangement of region models), shot (as a sequence of frame models). To identify *A shots*, instead of using predefined model images, the authors have developed an approach that first locates *potential A shots* by using temporal features. They then acquire model images from these candidates for model matching. After the shot is confirmed to be an *A shot*, a model image is obtained as the arithmetic average of the shot frames.
3. *Visual abstraction*. Key frames are extracted using the algorithms reported in [91].

The index information for each news item includes the number of shots in the news item, the starting time, the duration, and, most importantly, a set of key frames that represent the visual content of each shot. Text descriptions about the news content can be input by an operator.

6.5. Remarks

It is important to note that the first two categories of analyzed algorithms (video abstracting and video structure by visual similarity) can be applied to nearly all video material, as they make no use of video specific attributes, relying only on very general assumptions. Recovering video structure by media-based rule is more appropriate for feature movies, where the *film* language is more amenable to categorizations. The last approach, *video structure by program modeling*, which heavily relies on a detailed model, can only be applied to the video productions corresponding to the employed model. While restricted, it is of great interest because newscasts are among the most frequently accessed video items in digital libraries and webcasts.

Evaluation of algorithms that try to recover higher level video structure is complex and costly. As far as we know, no publicly available databases exist so far for evaluation and few experimental studies have been reported. One of them [105] compares the effectiveness of several video abstracting techniques for *fact finding* and *gisting*. The main results of this

study is that human–computer interface issues are extremely critical and must be carefully considered in order to exploit the results of automatic video analysis.

Automatic video abstracting techniques, while providing useful results for casual browsing, do not yet provide high quality results, due to the difficulty of capturing the semantic of the video message.

Automatic recovery of video structure is still in its infancy and good results have been obtained so far only for highly structured videos, such as news. Recovering video structure is one of the most open research areas in the field, and probably the one where image processing techniques alone seem unable to provide quality results. More integrated approaches to the solution of this problem, relying on the synergic use of speech recognition and image understanding, are now appearing in the literature. A notable example is provided by Informedia [106], where the technique of *spotting by association* has been introduced in the analysis of newscasts. The basic assumption of this technique is that an important video segment must have mutually consistent image and language data. Video transcripts (obtained from closed-captions or by speech recognition techniques) are parsed to detect key sentences (clues to situations such as speech/opinion, meeting/conference, visit/travel, location) which are then associated to image clues (e.g. face close-ups, people, outdoor scenes). Topics are then identified by consistent image/language clues.

Another important aspect of video analysis, related to the recovery of video structure, which is only now beginning to be considered, is the characterization of video clips as *units* that can be compared as a whole by video retrieval systems [107]. Another attempt to capture the semantic of the video message is reported in [108], where low-level visual features, such as color characteristics, editing effects, and rhythm, are linked to the information being transmitted by commercial videos. The resulting system is able to categorize a commercial video into one of four available semiotic categories and can be used by marketing professionals to check the consistency of an advertising campaign, making sure that the transmitted message fits within the required category.

7. FINAL CONSIDERATIONS

Effective use of multimedia material requires efficient ways to describe it in order to browse and retrieve it. In this review several algorithms have been analyzed that provide support for:

- user annotation, by segmenting the video stream into meaningful units, enabling effective nonlinear browsing through the material;
- automatic image and shot annotation, through computer-derived visual indices and constrained object recognition (e.g. captions and faces);
- automatic video abstracting and structuring, providing compact representations for browsing large video archives.

The available algorithms provide good coverage of the issues relevant to the development of a computer-aided video indexing system (CAVIS). While the field is already mature enough to provide tools which are effective in helping users access multimedia data, many issues remain open:

- automatic video segmentation performance, while acceptable in some applications, should be further improved. Little work has been done so far to integrate in a synergetic way several segmentation algorithms; development in this area is expected to increase the overall performance, in particular by reducing the number of detected false shot boundaries;

• system evaluation needs large annotated databases which can be used for testing the algorithms; no such databases are currently available;

• there is no widely accepted methodology to assess the performance of CAVIS systems in high level tasks such as *facting* and *gisting*; without such a methodology it is extremely difficult to quantize the progress in the field;

• tackling high level information retrieval tasks requires an increasingly multidisciplinary approach; integration of the contribution from the different research fields (vision, speech recognition, natural language understanding, human-computer interfaces) will pose new challenges;

• bottom-up work, linking currently detectable features (from color characteristics to montage rhythm) to more semantically rich concepts needs further investigation and holds promise of increasing the usefulness of currently available systems;

• working on videos as *units* is still in its infancy; the possibility of characterizing video material (a video clip, a feature movie) as a whole and comparing it to other items will be increasingly important as the databases grow larger and larger.

• human–computer interfaces have a crucial role in accessing multimedia data; the work done so far in structuring document access for the Web needs to be further extended for multimedia browsers working on large collections of items (see [109–111] for some recent proposals).

The importance of automatic video indexing has not been diminished by recent technological innovations in professional cameras which permit the addition of meta-data at the shooting stage. While some of the major camera manifacturers have promoted new video formats supporting this added functionality, these efforts have not resulted in an accepted standard. In the mean time the MPEG team started MPEG-7, a new work item to provide a solution to the problem of tagging multimedia information. Current research in the computer vision community is already aware of the importance of the algorithms developed for automatic video segmentation and indexing in the new standards MPEG-4 and MPEG-7, respectively, for compression and description. Integration of the results of automatic video analysis in the definition of the new standards and in MPEG compliant systems is one of the trends in this field of applied research.

While automatic video analysis has already proven useful in the management of multimedia data, much work remains to be done in nearly all of the related research topics, from segmentation to description, to provide general, reliable, tools for accessing multimedia information with the same ease usually associated to the management of text documents.

APPENDIX: GLOSSARY

abstracting video, a video stream is transformed into a reduced set of key frames or sequences.

annotation, the process of identifying and describing useful events and sequences in a video stream. Structure, form, or content are added to information already acquired (e.g., adding a narration track to a video tape).

authoring, the process of assembling available information (e.g. raw footage) into a new cohesive presentation.

cut, an abrupt change from one frame to the next.

dissolve, a superposition of a fade-in and a fade-out; the first shot fades out while the following fades in to full light.

edit, a gradual transition from a shot to another shot. The video editing process involves two phases: ordering of shots and assembly, the physical process in which the editing phases result is converted into frames on the final cut. It is during the latter that new frames (called *edit frames*) are added to create a transition between two consecutive shots.

fade-in, the progressive transition from black to light.

fade-out, the progressive darkening of a shot until the last frame becomes completely black.

frame, a video image that is the combination of two fields and includes the odd and evenly scanned lines.

frame rate, the rate at which frames are output from a video decoding device or stored in memory.

video indexing, the process of attaching labels (indices) to video. There are three basic types of indices: bibliographic, structural, and content. Bibliographic indices resemble the sorts of indices we normally associate with library catalogs. Structural indices are based on identifying units of both temporal and spatial structure. The content indices describe the objects in the video stream, including the appearance and location of persons and physical objects.

key frame, a frame representative of a shot.

matte, a progressive obscuration of visual field, due to a mask that invades the screen.

MPEG, short for Moving Picture Experts Group, a working group of ISO. The term also refers to the family of digital video compression standards developed by the group. The newest members of the MPEG family are MPEG-4, for very low-bandwidth video, and MPEG-7, specifying a standard set of descriptors to be attached to multimedia information.

pan, the action of a camera when following a person or object.

scene, a group of shots that share common attributes and provide the description of the context.

video segmentation, decomposition of a video clip into its component units (shot, scene, ...).

shot, an image sequence (one or more contiguous frames) coming from a single operation of the camera and presenting a continuous action in time and space.

tilt, a camera "pan" in a vertical direction, down or up, from a stationary position.

wipe, picture transition from one scene to another, wherein the new scene is revealed by a moving line or pattern. In simplest form, it simulates a window shade being drawn.

zoom, the camera closes up on a subject without losing focus.

REFERENCES

1. N. R. Adam and B. K. Bhargava, *Digital Libraries*, Springer-Verlag, New York/Berlin, 1995.
2. F. Banfi, *Image Databases: State of the Art and Future Directions*, Internal Working Paper 96-10, University of Fribourg, Fribourg, Switzerland, September 1996.
3. A. Brink, S. Marcus, and V. S. Subrahmanian, Heterogeneous multimedia reasoning, *Computer*, September 1995, 33–39.
4. G. Davenport and M. Murtaugh, ConText: Toward the evolving documentary, in *ACM Multimedia 95, San Francisco, CA, November 1995*, pp. 5–8.
5. E. A. Fox, Advances in interactive digital multimedia systems, *Computer*, October 1991, 9–21.
6. D. J. Gemmell, H. M. Vin, D. D. Kandlur, P. V. Rangan, and L. A. Rowe, Multimedia storage servers: A tutorial, *Computer*, May 1995, 40–49.
7. R. Jain, *InfoScopes: Multimedia Information Systems*, Technical Report VCL-95-107, Visual Computing Laboratory, University of California, San Diego, 1995.

8. R. Jain, A. Pentland, and D. Petkovic, Workshop Report: NFS-ARPA workshop on visual information management systems, in *Proceedings of the 5th ICCV, Cambridge, MA, 1995*.
9. S. Moni and R. L. Kashyap, A multiresolution representation scheme for multimedia databases, *Multimedia Systems* **3**, 1995, 228–237.
10. A. D. Narasimhalu, Multimedia databases, *Multimedia Systems* **4**(5), 1996, 226–249.
11. H. Nishiyama, S. Kin, T. Yokoyama, and Y. Matsushita, An image retrieval system considering subjective perception, in *Proceedings of CHI '94, Boston, 1994*, Vol. 4, pp. 30–36.
12. A. A. Rodriguez and L. A. Rowe, Multimedia systems and applications, *Computer*, May 1995, 20–23.
13. V. S. Subrahmanian and S. Jajodia, *Multimedia Database Systems*, Springer-Verlag, New York/Berlin, 1996.
14. H. J. Zhang, S. W. Smoliar, J. H. Wu, C. Y. Low, and A. Kankanhalli, A video database system for digital libraries, in *Advances in Digital Libraries, Lecture Notes in Computer Science*, Springer-Verlag, New York/Berlin, 1995.
15. A. Hampapur, R. Jain, and T. Weymouth, Digital video indexing in multimedia systems, in *Proceedings of the Workshop on Indexing and Reuse in Multimedia Systems, August 1994*, AAAI Press, Menlo Park, CA, 1994.
16. A. Hampapur, R. Jain, and T. Weymouth, Feature based digital video indexing, in *Proceedings of IFIP 2.6 Third Working Conference on Visual Database Systems VDB.3, Lausanne, Switzerland, March 29–31, 1995*.
17. G. Ahanger, D. Benson, and T. D. C. Little, *Video Query Formulation*, Technical Report MCL 01-09-1995, Multimedia Communication Laboratory, Boston University, 1995.
18. G. Ahanger and T. D. C. Little, A survey of technologies for parsing and indexing digital video, *J. Visual Commun. Image Rep.* **7**(1), 1996, 28–43.
19. C. Faloutsos and K. I. Lin, FastMap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets, in *Proceedings of SIGMOD '95, 1995*, pp. 163–174.
20. A. S. Gordon and E. A. Domeshek, Conceptual indexing for video retrieval, in *Proceedings IJCAI '95 Workshop on Intelligent Multimedia Information Retrieval, Montreal, August 1995* (M. Maybury, Ed.).
21. T. Sikora, The MPEG-4 video standard and its potential for future multimedia applications, in *Proc. IEEE ISCAS Conference, Hong Kong, June 1997*.
22. M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker, Query by image and video content: The QBIC systems, *Computer*, September 1995, 23–32.
23. P. Kelly and M. Cannon, CANDID: Comparison algorithm for navigating digital image databases, in *Proceedings of the Seventh International Working Conference on Scientific and Statistical Database Management, Charlottesville, VA, September 1994*, pp. 252–258.
24. P. Kelly and M. Cannon, Experience with CANDID: Comparison algorithm for navigating digital image databases, in *Proceedings SPIE of the 23rd AIPR Workshop On Image and Information Systems: Applications and Opportunities, Washington DC, October 12–14, 1994*, pp. 64–75.
25. P. Kelly and M. Cannon, Query by image example: The CANDID approach, in *Proceedings of SPIE Storage and Retrieval for Image and Video Databases III, 1995*, Vol. 2420, pp. 238–248.
26. I. K. Sethi and N. Patel, A statistical approach to scene change detection, in *Proceedings of SPIE Storage and Retrieval for Image and Video Databases III, San Josè, CA, 1995*, Vol. 2420.
27. H. J. Zhang, A. Kankanhalli, and S. W. Smoliar, Automatic partitioning of full-motion video, *Multimedia Systems* **1**, 1993, 10–28.
28. P. Aigrain and O. Joly, The automatic real-time analysis of film editing and transition effects and its applications, *Comput. & Graphics* **18**(1), 1994, 93–103.
29. U. Gargi, S. Oswald, D. A. Kosiba, S. Devadiga, and R. Kasturi, Evaluation of video sequence indexing and hierarchical video indexing, in *Proceedings of SPIE Conference on Storage and Retrieval for Image and Video Databases III, 1995*, Vol. 2420.
30. U. Gargi, R. Kasturi, S. Strayer, and S. Antani, *An Evaluation of Color Histogram Based Methods in Video Indexing*. Technical Report CSE-96-053, Penn State University, Department of Computer Science and Engineering, 1996.
31. J. M. Corridoni and A. Del Bimbo, Film semantic analysis, in *Proceedings of the International Conference on Analysis of Image Patterns, Prague, Czech Republic, September 1995*.
32. W. Xiong, J. Chung-Mong, and R. H. Ma, Automatic video data structuring through shot partitioning and key-frame computing, *Mach. Vision Appl.* **10**, 1997, 51–65.
33. D. Le Gall, MPEG: A video compression standard for multimedia applications, *Commun. ACM* **34**(4), 1991, 46–58.

34. H. J. Zhang, C. Y. Low, and S. W. Smoliar, Video parsing and browsing using compressed data, *Multimedia Tools Appl*. **1**, 1995, 91–113.

35. International Telecommunication Union, ITU Place des Nations CH-1211, Geneva 2, *H.263: Video coding for low bit rate communication*, E 7244 edition, March 1996.

36. J. Meng, Y. Juan, and S.-F. Chang, Scene change detection in a MPEG compressed video sequence, in *Proceedings of the IS&T/SPIE Symposium on Electronic Imaging: Science & Technology, San Josè, CA, February 1995*.

37. H. C. Liu and G. L. Zick, Automatic determination of scene changes in MPEG compressed video, in *Proc. of ISCAS-IEEE International Symposium on Circuits and Systems, 1995*.

38. F. Arman, A. Hsu, and M. Y. Chiu, Feature management for large video databases, in *Proceedings of SPIE Storage and Retrieval for Image and Video Databases, 1993*, pp. 2–12.

39. E. Deardorff, T. D. C. Little, J. D. Marshall, D. Venkatesh, and R. Walzer, Video scene decomposition with the motion picture parser, in *Proceedings of the IS&T/SPIE Symposium on Electronic Imaging Science and Technology (Digital Video Compression and Processing on Personal Computers: Algorithms and Technologies), San Josè, CA, February 1994*, Vol. 2187, pp. 44–55.

40. N. V. Patel and I. K. Sethi, Video shot detection and characterization for video databases, *Pattern Recognition—Special Issue on Multimedia* **30**(4), 1997, 583–592.

41. E. Ardizzone, G. A. M. Gioiello, M. La Cascia, and D. Molinelli, A real-time neural approach to scene cut detection, in *IS&T/SPIE—Storage & Retrieval for Image and Video Databases IV, San Josè, CA, January 28–February 2, 1996*.

42. B.-L. Yeo, *Efficient Processing of Compressed Images and Video*, Ph.D. thesis, Dept. of Electrical Engineering, Princeton University, January 1996.

43. K. Shen and E. J. Delp, A fast algorithm for video parsing using MPEG compressed sequences, in *Proc. of IEEE International Conference on Image Processing, October 1995*, pp. 252–255.

44. A. Hampapur, R. Jain, and T. Weymouth, Production model based digital video segmentation, *J. Multimedia Tools Appl.* **1**(1), 1995, 9–46.

45. A. Hampapur, R. Jain, and T. Weymouth, Digital video segmentation, in *Proceedings Second Annual ACM MultiMedia Conference and Exposition, October 1994*.

46. R. Zabih, J. Miller, and K. Mai, A feature-based algorithm for detecting and classifying scene breaks, in *Proceedings of the Fourth ACM Conference on Multimedia, San Francisco, CA, November 1995*.

47. W. Xiong and J. C.-M. Lee, Efficient scene change detection and camera motion annotation for video classification, *Comput. Vision Image Understanding* **71**(2), 1998, 166–181.

48. A. Dailianas, R. B. Allen, and P. England, Comparisons of automatic video segmentation algorithms, in *Proceedings SPIE Photonics East'95: Integration Issues in Large Commercial Media Delivery System, Philadelphia, October 1995*.

49. J. S. Boreczky and L. A. Rowe, A comparison of video shot boundary detection techniques, in *Storage & Retrieval for Image and Video Databases IV* (I. K. Sethi and R. C. Jain, Eds.), *SPIE Proceedings Series*, Vol. 2670, pp. 170–179.

50. R. Brunelli and T. Poggio, Template matching: Matched spatial filters and beyond, *Pattern Recognition* **30**(5), 1997, 751–768.

51. K. K. Sung and T. Poggio, Example-based learning for view-based human face detection, in *Proceedings Image Understanding Workshop, 1994*.

52. M. S. Lew and N. Huijsmans, Information theory and face detection, in *Proceedings of ICPR, 1996*, pp. 601–605.

53. B. Moghaddam and A. Pentland, Probabilistic visual learning for object representation, *IEEE Trans. Pattern Anal. Mach. Intell.* **19**(7), 1997, 696–710.

54. E. Osuna, R. Freund, and F. Girosi, Training support vector machines: An application to face detection, in *Proc. of CVPR'97, Puerto Rico, June 1997*.

55. H. Wang and S. Fu Chang, A highly efficient system for automatic face region detection in MPEG video, *IEEE Trans. Circuits Systems Video Technol.* **7**(4), 1997, 615–628.

56. G. Burel and D. Carel, Detection and localization of faces on digital images, *Pattern Recognit. Lett.* **15**, 1994, 963–967.

57. H. A. Rowley, S. Baluja, and T. Kanade, Neural network-based face detection, in *Image Understanding Workshop, February 1996*, pp. 725–735.

58. T. Sato, T. Kanade, E. K. Hughes, M. A. Smith, and S. Satoh, Video OCR: Indexing digital news libraries by recognition of superimposed caption, *ACM Multimedia Systems, Special Issue on Video Libraries*, submitted.

59. Y. Ariki and T. Teranishi, Indexing and classification of TV-news articles based on telop recognition, in *Fourth Int. Conf. on Document Analysis and Recognition, Ulm, Germany, August 1997*, pp. 422–427.
60. A. K. Jain and B. Yu, Automatic text location in images and video frames, in *14th Int. Conf. on Pattern Recognition, Brisbane, Australia, August 16–20, 1998*.
61. H.-K. Kim, Efficient automatic text location method and content-based indexing and structuring of video database, *J. Visual Commun. Image Representation* **7**(4), 1996, 336–344.
62. R. Lienhart and W. Effelsberg, *Automatic Text Segmentation and Text Recognition for Video Indexing*, Technical Report TR-98-009, Universität Mannheim, Praktische Informatik IV, Mannheim, Germany, 1998.
63. A. Takeshita, T. Inoue, and K. Tanaka, Extracting text skim structures for multimedia browsing, in *Proceedings IJCAI '95 Workshop on Intelligent Multimedia Information Retrieval, Montreal, August 1995* (M. Maybury, Ed.).
64. V. Wu, R. Manmatha, and E. M. Riseman, Finding text in images, in *2nd ACM Int. Conf. on Digital Libraries, Philadelphia, PA, July 1997*, pp. 23–26.
65. S. Messelodi and C. M. Modena, Automatic identification and skew estimation of text lines in real scene images, *Pattern Recognit.* **32**(5), 1999, 789–808.
66. U. Gargi, R. Kasturi, and S. Antani, Performance characterization and comparison of video indexing algorithms, in *Proc. of Computer Society Conference on Computer Vision and Pattern Recognition, 1998*.
67. S. Sato, Y. Nakamura, and T. Kanade, Name-it: Naming and detecting faces in video by the integration of image and natural language processing, in *Proc. of IJCAI-97, 1997*.
68. C. Cedras and M. Shah, Motion-based recognition: A survey, *Image Vision Comput.* **13**(2), 1995, 129–155.
69. J. Y. A. Wang and E. H. Adelson, Spatio-temporal segmentation of video data, in *Image and Video Processing II, Proceedings of the SPIE, San Josè, CA, February 1994*, Vol. 2182.
70. J. Y. A. Wang, E. H. Adelson, and U. Desai, Applying mid-level vision techniques for video data compression and manipulation, in *Digital Video Compression on Personal Computers: Algorithms and Technologies, Proceedings of the SPIE, San Josè, CA, February 1994*, Vol. 2187.
71. S. Ayer and H. S. Sawhney, *Layered Representation of Motion Video using Robust Maximum-Likelihood Estimation of Mixture Models and MDL Encoding*, Technical report, IBM Almaden Research Center, 650 Harry Road, San Josè, CA 95120, December 1994.
72. S.-F. Chang, W. Chen, H. J. Meng, H. Sundaram, and D. Zhong, VideoQ: An automated content based video search system using visual cues, in *Proceedings of the Fifth ACM Multimedia Conference, Seattle, November 1997*.
73. H. S. Sawhney, S. Ayer, and M. Gorkani, Model-based 2D & 3D dominant motion estimation for mosaicking and video representation, in *Proceedings of ICCV 95, Cambridge, MA, June 1995*.
74. M. Bierling, Displacement estimation by hierarchical blockmatching, in *Proc. SPIE, Visual Communications and Image Processing '88, January 1988*. Vol. 1001, pp. 942–951.
75. E. Sahouria, Video indexing based on object motion, Master's thesis, Dept. of Electrical Engineering and Computer Science, University of California at Berkeley, 1998.
76. N. Hirzalla and A. Karmouch, Detecting cuts by understanding camera operation for video indexing, *J. Visual Lang. Comput.* **6**, 1995, 385–404.
77. M. A. Smith and T. Kanade, *Video Skimming and Characterization through the Combination of Image and Language Understanding Techniques*, Technical Report CMU-CS-97-111, School of Computer Science, Carnegie Mellon University, Pittsburg, PA 15213, February 1997.
78. P. Joly and H.-K. Kim, Efficient automatic analysis of camera work and microsegmentation of video using spatiotemporal images, *Signal Process. Image Commun.* **8**, 1996, 295–307.
79. J. Park and C. W. Lee, Robust estimation of camera parameters from image sequence for video composition, *Signal Process. Image Commun.* **9**, 1996, 43–53.
80. N. Dimitrova and F. Golshani, Motion recovery for video content classification, *ACM Trans. Inform. Systems* **13**(4), 1995, 408–439.
81. M. Ioka and M. Kurokawa, Estimation of motion vectors and their application to scene retrieval, *Mach. Vision Appl.* **7**, 1994, 199–208.
82. S.-Y. Lee and H.-M. Kao, Video indexing—An approach based on moving object and track, in *Proceedings SPIE Storage and Retrieval for Image and Video Databases, 1993*, pp. 25–36.
83. D. Zhong and S.-F. Chang, Video object model and segmentation for content-based video indexing, in *IEEE Intern. Conf. on Circuits and Systems, Hong Kong, June 1997*. [Special session on networked multimedia technology & application]

84. E. Ardizzone and M. La Cascia, Video indexing using optical flow field, in *Int. Conf. on Image Processing, ICIP-96, Lausanne, Switzerland, September 16–19, 1996*.

85. Y. Tonomura, A. Akutsu, K. Otsuji, and T. Sadakata, VideoMAP and video spaceIcon: Tools for anatomizing video content, in *Proceedings of INTERCHI '93, ACM, April 24–29, 1993*, pp. 131–136.

86. L. Teodosio and W. Bender, Salient stills from video, in *Proceedings of the ACM Multimedia '93 Conference, Anheim, CA, August 1993*.

87. H. S. Sawhney and S. Ayer, Compact representations of video through dominant and multiple motion estimation, *IEEE Trans. Pattern Anal. Mach. Intell.* **18**(8), 1996.

88. D. Yow, B.-L. Yeo, M. Yeung, and B. Liu, Analysis and presentation of soccer highlights from digital video, in *Proceedings of the Second Asian Conference on Computer Vision, December 1995*.

89. J. L. Barron, D. J. Fleet, and S. S. Beauchemin, Performance of optical flow techniques, *Int. J. Comput. Vision* **12**(1), 1994, 43–77.

90. M. M. Yeung and B. Liu, *Efficient Matching and Clustering of Video Shots*, technical report, Princeton University, 1995.

91. H. J. Zhang, S. W. Smoliar, and J. H. Wu, Content-based video browsing tools, in *Proceedings SPIE Conference on Multimedia Computing and Networking, San Josè, CA, February 1995*.

92. W. Wolf, Key frame selection by motion analysis, in *Proceedings of ICASSP '96*.

93. X. Liu, C. B. Owen, and F. Makedon, *Automatic Video Pause Detection Filter*, Technical Report PCS-TR97-307, Dartmouth College, Computer Science, Hanover, NH, February 1997.

94. B. Yeo and B. Liu, Rapid scene analysis on compressed videos, *IEEE Trans. Circuit Systems Video Technol.* **5**(6), 1995, 533–544.

95. B.-L. Yeo and B. Liu, A unified approach to temporal segmentation of motion JPEG and MPEG compressed video, in *Proceedings of the Second Internation Conference on Multimedia Computing and Systems, May 1995*.

96. M. Yeung, B.-L. Yeo, and B. Liu, Extracting story units from long programs, in *Proceedings of the International Conference on Multimedia Computing and Systems, June 1996*.

97. M. M. Yeung, B.-L. Yeo, W. Wolf, and B. Liu, Video browsing using clustering and scene transitions on compressed sequences, in *Proceedings of the IS&T/SPIE Multimedia Computing and Networking, San Josè, CA, February 1995*.

98. H.-H. Yu and W. Wolf, Scenic classification methods for image and video databases, in *Proceedings of the ACM Multimedia '93 Conference* (C.-C. J. Kuo, Ed.), Vol. 2606, 1995, pp. 363–371.

99. S. Pfeiffer, R. Lienhart, S. Fisher, and W. Effelsberg, Abstracting digital movies automatically, *J. Visual Commun. Image Representation* **7**(4), 1996, 345–353.

100. S. Devadiga, D. A. Kosiba, U. Gargi, S. Oswald, and R. Kasturi, A semiautomatic video database system, in *Proceedings of SPIE Conference on Storage and Retrieval for Image and Video Databases III*, Vol. 2420, 1995.

101. S. Fischer, R. Lienhart, and W. Effelsberg, Automatic recognition of film genres, in *Proc. ACM Multimedia 95, San Francisco, CA, November 1995*, pp. 295–304.

102. M. K. Hawley, *Structure out of Sound*, Ph.D. thesis, School of Architecture and Planning, Massachusetts Institute of Technology, September 1993.

103. P. Aigrain, P. Joly, and V. Longueville, Medium knowledge-based macrosegmentation of video into sequences, in *Proceedings IJCAI '95 Workshop on Intelligent Multimedia Information Retrieval, Montreal, Canada, August 1995* (M. Maybury, Ed.).

104. H. J. Zhang, S. Y. Tan, S. W. Smoliar, and G. Yihong, Automatic parsing and indexing of news video, *Multimedia Systems* **2**, 1995, 256–266.

105. M. G. Christel, M. A. Smith, C. R. Taylor, and D. B. Winkler, Evolving video skims into useful multimedia abstractions, in *Proc. of ACM CHI'98, Conference on Human Factors in Computing Systems, Los Angeles, CA, April 1998*.

106. Y. Nakamura and T. Kanade, Semantic analysis for video contents extraction—Spotting by association in news video, in *Proc. of the Fifth ACM International Multimedia Conference, November 1997*.

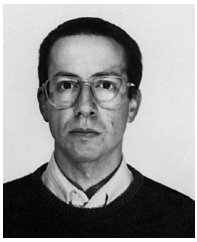
107. R. Lienhart, W. Effelsberg, and R. Jain, VisualGREP: A systematic method to compare and retrieve video sequences, in *SPIE, Storage and Retrieval for Image and Video Databases VI, January 1998*, Vol. 3312.

108. M. Caliani, C. Colombo, A. Del Bimbo, and P. Pala, Commercial video retrieval by induced semantics, in *Proc. of IEEE Internation Workshop on Content-Based Access of Image and Video Databases (CAIVD8), Bombay, India, 1998*.

109. M. Christel and D. Martin, Information visualization within a digital video library, *J. Intell. Inform. Systems* **11** (3), 1998. [Special issue on Information Visualization]

110. D. DeMenthon, V. Kobla, and D. Doermann, Video summarization by curve simplification, in *Proceedings of the Sixth ACM Multimedia Conference, Bristol, UK, September 1998*.

111. K. Manske, M. Muhlhauser, S. Vogl, and M. Goldberg, OBVI: Hierarchical 3D video-browsing, in *Proceedings of the Sixth ACM Multimedia Conference, Bristol, UK, September 1998*.



ROBERTO BRUNELLI was born in Trento, Italy in 1961. He received his degree (summa cum laude) in physics from the University of Trento in 1986. He joined ITC-irst in 1987, where he works in the Computer Vision Group of the Interactive Sensory Systems Division. In the past he was involved in research on computer vision tools, analysis of aerial images, development of algorithms working on compressed description of binary images, optimization, neural networks, and face analysis. His current major involvement is in OPAL (ESPRIT Project 25389), on-line programmes, digital archives, and distributed editorial collaboration. His current professional interests include optimization, robust statistics, object recognition, and multimedia databases. Personal interests include comics, inline-skating, science-fiction, motorbiking, and photography.

ORNELLA MICH was born in Tesero, Trento, Italy in 1959. She received her degree in electronic engineering from the University of Padova, Italy in 1987. Then she worked on radar electronics for jet-fighters. After teaching electrotecnics in a high school, she joined IRST in 1989, collaborating on VLSI research. From 1990 she is working in the field of computer vision in the Interactive Sensory Systems Division.

CARLA MARIA MODENA was born in Rovereto, Trento, Italy in 1961. She received her degree in mathematics from the University of Trento in 1985. In 1986 she worked at the Institut fuer Volkswirtschaftslehre, Universitaet Regensburg, Germany. She joined ITC-IRST in 1987, where she works in the Interactive Sensory Systems Divison. Her current interests in computer vision include text localization and recognition and document archives indexing.