# A Survey on Video Indexing

**R. Brunelli and O. Mich and C. M. Modena**
Istituto per la Ricerca Scientifica e Tecnologica
I-38050 Povo, Trento, ITALY
*IRST-Technical Report 9612-06*

**Abstract**

Extracting information from the ever growing stream of multimedia data is becoming increasingly difficult. One of the main reasons lies within the unstructured way multimedia data are usually presented. Audio-visual material represents a large part of current multimedia material and can be structured in meaningful ways due to the nature of visual communication. This paper surveys several approaches and algorithms that have been recently developed to help in automatically structuring audio-visual data, both for annotation and access[1].

*Keywords:* audio-visual data, audio segmentation, image databases, image indexing, image retrieval by pictorial content, video annotation, video databases, video segmentation.

## 1. Introduction

The advent of digital technologies has enabled the emergence of a new type of database that supports an evolving collection of media elements [1, 7, 11, 23, 34, 38, 65, 67, 81, 82, 84, 97, 116, 143]. Multimedia database systems store large volumes of text, images, and other information. Efficient indexing methods are crucial if these information collections are to be useful. A growing body of research looks at the problem of how video should be indexed [2, 3, 30, 42]. Research areas include automatic parsing and matching, as well as human supported annotation activity. Once an author or logger has attached descriptors to a set of media elements, these materials can be retrieved according to simple or complex directed queries [8, 31, 44, 45, 55, 56, 64, 74, 89, 98, 106, 119, 129]. Indexing video data is the process of attaching content-based labels to video. This process is essential for providing content based access. One goal of content-based work is to provide algorithms that can automatically recognize the important features contained in an image without human intervention in the process [17]. The content-based work most notable for arts and humanities focuses on the recognition and description of color, texture, shape, spatial location, regions of interest, facial characteristics, and specifically for motion materials, video segmentation, extraction of representative key frames, scene change detection, extraction of specific objects and audio keywords. Speech-recognition techniques, music representation and recognition techniques can be applied in parallel with image understanding.

Facilitating content access is not just a problem of image processing. Audio processing and dozens of semantic and artificial intelligence issues (script and story knowledge, natural language, common sense understanding), interface issues (query input and presentation of results) as well as networking and service issues (delivery, copyrights, billing) are also involved. In [92] Picard overviews key image processing research problems which must be solved to give people access to the content of digital video and image libraries. Perhaps the two most challenging questions are: "What is a good set of models to represent images, to facilitate both compression and content-access?" and "What is a good measure of visual similarity?".

The acquisition and presentation of information are knowledge-intensive activities that have been performed in the past exclusively by humans. These activities in combination have come to be known as authoring. The task of an author is to collect a coherent body of information, structure it in meaningful and interesting ways, and present it in an appropriate fashion to a set of readers (or viewers) of the eventual work. This traditional notion of authoring commits the author to deciding the form as well as the contents of the work, well in advance of the actual time at which it is presented. The familiar book format conveys this point.

The limitations of the traditional approach to authoring are most obvious when these processes are applied into non-traditional media, such as in the video medium. The raw material must first be acquired, which involves filming or digitizing. From

[1]This work was supported by Esprit Project No. 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.

this raw source, video authors must assemble cuts into a cohesive presentation. The raw footage can be very voluminous, and very sparse. The process of identifying useful events and sequences has been called annotation, and a number of systems have been designed to expedite it. When the author has finally identified a useful set of cuts, the traditional notion of authoring would require assembling these into their final presentation order. This approach suffers from the aforementioned limitation, that such a presentation can not be tailored to the needs of individual viewers. In the case of video tape, it is a crippling limitation, due to the temporal linearity of the medium. It is here that video data diverges significantly from text, graphics and even animation. Video data is inherently uninterpreted information. There are currently no general computational mechanisms for content-searching video data with the precision and semantic exactness of generalized textual search. At present, the only practical way of accessing a video database is, for a human, to first annotate it so that the annotation can be used to guide the author and the viewer. Creating this annotation, at least with the current tools, is an inherently sequential operation (requiring the visual inspection of the complete video stream) and is a major bottleneck in the authoring of hypervideo documents. Even with annotation, we believe navigation through a hypervideo document is more difficult than with other types of hypermedia. In [22] Csinger et al. show how existing video transcription and presentation methods fail to make full use of the potential of video.

Concerning transcription, they identify the *semantic unpredictability problem* as that of not knowing a priori the intervals in the video record that will be required for subsequent presentation. The *syntactic ambiguity problem* stems from the inconsistent use that video annotators and automatic logging agents make of labeling. Concerning presentation, we recognize that traditional models do permit run-time generation of user-tailored output. Csinger et al. discuss a knowledge-based framework called intent-based authoring. This model addresses the above problems to provide savings in human effort in both the annotation and presentation phases of video authoring.

## 2. Computer Aided Video Indexing

One of the most popular approaches to representing video contents is the structured modeling approach: the video sequences are first divided into shots [2]. A shot usually represents a primitive concept, as in hypertext and is used as a basic unit of manipulation. The major draw-back of the structured modeling approach is that it imposes a specific intentionality of the author when creating the shots. Furthermore, a lack of contextual information between video shots derives from this annotation style. A first solution is to introduce a second, upper, descriptive level, the scene, to group together shots that share common attributes and provide the description of the context. This strategy is pursued in [18]: the shots provide the lower level, while the scenes, which are built from the shots or other scenes, provide the higher level. The system is supported by a concept-based search engine, based on a vector representation of textual annotations, that propagates relevance scores up the scene hierarchy.

Davenport et al. have proposed in [24] *Stratification*, a model for annotation that comes close to the way in which the editor builds an understanding of what happens in individual shots. In this approach, instead of representing video information as independent shots, the video sequence is divided into a set of overlapping *strata*. A *stratum* consists of the descriptive information such as title, keywords (describing perspective, camera and sound, content, and context), and the boundaries which it represents. The strata may be contained in other strata and may encompass a multitude of other descriptions. The content information is derived by examining the union of all the contextual descriptions that are associated to it. Within this approach, each stratum is logically independent and will make its contribution to the content when its descriptive information meets the user demands. Stratification can help support computer-generated links between similar pieces of footage. Stratified descriptions of content will therefore expedite on-the-fly creation of machine-generated, multi-threaded parallel narratives that users can adjust through linking. Such narratives are vital for effective multimedia environments. It is apparent that this approach requires a complex query management due to the temporal nesting and overlapping of the descriptions. A similar approach is the driving force of *Media Streams* proposed in [25].

Another system based on the the idea of *stratification* is proposed in [60]. Two different models are employed: a structural model for video, which facilitates browsing, and an unstructured model for thematic indexing. The model is refined in [61] where *VideoSTAR* is presented. In order to provide an efficient framework for the annotation, (re)use, and retrieval of video footage the notion of basic, primary, and secondary contexts are introduced. The basic context contains descriptions that are valid for the video material independently of the video production using it. The primary context relates to the use of the footage in a production. The same footage could be used in other productions, characterized by secondary contexts (with respect to a chosen primary context).

Indexing digital video based on its content can be carried out at several levels of abstraction beginning with indices like video program name and name of subject, to much lower level aspects of video like locations of edits and motion properties of video,

[2] Shot: an image sequence coming from a single operation of the camera, and presenting a continuous action.

[47, 49]. 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 indexes video based on the content and style of the shots used to compose the video. The automation of the indexing process becomes then essential when the granularity of the video access increases. It is in this context that the use of computer vision techniques, which provide low-level features describing the video sequences, can be of help. A first contribution is to automatically decompose the video stream into shots and possibly higher level structures. This provides useful indices to speed up the annotation process, suggesting temporal limits for the annotations and providing non-linear access to the video stream.

Domain constraints can be used to design a mapping from low level machine derived indices[3] to the desired video indices required for the specific application [49, 85].

While general object recognition can not be achieved without imposing heavy constraints on the domain, low level description of visual characteristics can be efficiently composed into index keys which allow the retrieval of images based on their visual structure. Once shots have been detected, representative frames can be selected automatically using image processing techniques. A video stream is then abstracted and transformed into a (very) reduced set of still images. The resulting database of stills can be accessed using novel techniques such as visual query by example [69, 70, 71] and sketch (i.e. searching for images that look similar to an available picture or to a sketch produced by the user) [33].

The next section will introduce computer vision algorithms that can support both the annotation of video data and its retrieval: shot detection, highlights extraction, key frame selection, motion analysis, acoustic indexing, video structure recovery, low level visual indices, tools, and interfaces. For each category of algorithms a brief description of their role within a computer aided video indexing system is given. Selected representative algorithms are then reviewed and further pointers to the current literature are provided.

## 3. 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 most basic unit, after the single frame is the *shot*, a contiguously recorded audio/image 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. Shots can also be concatenated by edited transitions such as dissolves, fade in/out, and wipes where images are superimposed at a shot boundary.

### 3.1. Cut Detection

The recent work on cut detection can 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. The methods in the first category are generally frame difference methods based on pixel-by-pixel comparison or on the distribution (histograms) of image values on the entire frames (or on a set of covering subregions). Many algorithms rely on the use of histogram comparisons (of intensity, color, etc.) 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 set of selected algorithms is the following:

- Three frames approach [105]: let $r, s, t$ be three consecutive frames and $D_{rs}, D_{st}$ the measure of frames dissimilarities (such as the absolute difference to sum ratio or the normalized difference energy computed on the frame pixels); 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 no change in the three frames under consideration, and a value close to zero otherwise.
- Pixel-wise comparison [139]: 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 above a given threshold.
- Likelihood ratio [139]: the comparison operates at the level of blocks. An image is subdivided into a set of blocks and a likelihood ratio is computed for each corresponding block, in $t$-th and $(t+1)$-th frames:
$$L_i = \frac{\left[ \frac{\sigma_t^2 + \sigma_{t+1}^2}{2} + \left( \frac{\mu_t - \mu_{t+1}}{2} \right)^2 \right]^2}{\sigma_t^2 \sigma_{t+1}^2} \tag{2}$$
where $\mu$ is the aritmetic average and $\sigma$ is the standard deviation of intensity histogram.
A camera break is assumed when a sufficient fraction of blocks satisfy $L_i > v$, where $v$ represents an appropriate threshold.

[3] These indices usally describe the color distribution and the textural properties of the images, considered as a whole or partitioned into a set of regions to account for the spatial distribution of the features

- Histogram comparison [139]: the area between the intensity (or color) distribution of two consecutive frames is computed. A camera break is assumed whenever the area gets greater than a given threhsold. Color distributions are computed by using a 6 bits code (two bits per component). A variant to using the area between the distributions is given by the $\chi^2$-test:

$$\sum_t \frac{(H_{t+1}(j) - H_t(j))^2}{H_{t+1}(j)} \tag{3}$$

  where $H(j)$ represents the $j$-th bin. A modified definition [105] can be used to overcome the singularities due to a single bin being zero:

$$\sum_t \frac{(H_{t+1}(j) - H_t(j))^2}{(H_{t+1}(j) + H_t(j))} \tag{4}$$

- Incremental ratio [20] of between frames differences:

$$\delta = \frac{D(f_{t+1}, f_t)}{D(f_t, f_{t-1})}. \tag{5}$$

  The difference between two succesive frames is obtained by dividing each frame into $n^2$ sub-regions each of them originating a difference value $D_i$ according to

$$D_i^{red}(f_t, f_{t+1}) = \sum_i a_i |m_i^{red}(t+1) - m_i^{red}(t)| \tag{6}$$

  and similarly for the other color components, where $m_i^{red}(t)$ represents the $i$-th moment of the histogram of the *red* component for the $t$-th frame, and $\boldsymbol{a}$ is a set of parameters experimentally tuned. The between frames difference is computed by adding the three color differences for each region, discarding the $k$ largest value, and averaging the remaining ones. Since there is no mathematical model underlying the production of a shot, the threshold value for $\delta$ must be determined empirically.

- Yakimovsky likelihood ratio test [105]. The test was originally proposed to detect the presence of an edge at the boundary of two regions. It is based on the computation of maximum likelihood estimates of the probabilities $P_0$ and $P_1$ that the hypothesis $H_0$ and $H_1$, respectively, are true. The estimated probabilities are computed under the assumption of Gaussian distribution. The expression of the Yakimovsky ratio, for frames of equal size, is:

$$y = \left(\frac{\sigma_0^2}{\sigma_{t-1}^2}\right)\left(\frac{\sigma_0^2}{\sigma_t^2}\right) \tag{7}$$

  where $\sigma_{t-1}^2, \sigma_t^2$ represent the variances of the past and current frames while $\sigma_0^2$ is the variance of the pooled data from both the histograms.

- Kolmogorov-Smirnov test is based on calculating the cumulative distributions $C_1(x), C_2(x)$ of two sets of data and measuring the maximum absolute difference between them. If the distributions are approximated by histograms, a defective estimate of the distance is obtained by

$$D = \max_j |C_{1j} - C_{2j}| \tag{8}$$

  where $j$ denotes the bins.

All of the above methods require a comparison with appropriate thresholds to detect the *camera breaks*. A way of choosing these threhsolds is presented in [139] where a Gaussian distribution is fitted to the histogram of frame to frame differences. The threshold is taken to be the average value plus six times the standard deviation. Multi-pass approaches are also introduced in [139] 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.

The motivation for the second category of methods using compressed video comes from increased efficiency as certain coefficients in encoded video, e.g. DCT coefficients in MPEG video [36], 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/P frames. The residual error after motion compensation is then DCT coded. When the residual error exceeds a given threhsold for certain blocks, 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. In order to detect a camera break it is sufficient to count the fraction of blocks for which no motion vectors have been computed. If this number exceeds a predefined threshold, a cut is assumed.

One method using compressed video has been proposed in [5] and is based on the correlation between DCT coefficients of consecutive frames of Motion-JPEG compressed video. 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 $\Psi$ 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 frames dissimilarity. A cut is detected whenever $D$ exceeds a predefined threshold. A variant is presented in [140]: frame dissimilarity is computed by taking a normalized $L_1$ distance of the vectors and is 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. In order to overcome this problem, a motion based segmentation has also been proposed in [140] for MPEG streams.
The same algorithm can be applied to H.263 streams [62] that also rely on motion compensation for coding.
Another algorithm working on compressed Motion-JPEG video is presented in [26]. The main idea is to use variations of the size of compressed frames to locate camera breaks which are characterized by a positive or negative step in the compressed frame size plot.

## 3.2. Gradual Effects Detection

Gradual transition is an important class of effects and includes dissolving, fading in, fading out, mattes, and translates. A *fade-in (out)* is an optical process, which allows the progressive darkening of a shot until the last frames becomes completely black (fade-out) or, on the opposite, allows the transition from black to light (fade-in). A *dissolve* is a superimposition of a fade-in and a fade-out: the first shot fades out while the following fades in to full light. Up to the 50's, dissolves have been considered very specific punctuations signs within the filmic speech: a dissolve for example used to indicate a place shift within the same action or a flashback. The presence of specific effects can then be used as a clue for detecting scene changes, as opposed to simple shot detection. A *matte* is a progressive obscuration of visual field, due to a mask that invades the screen. A *translate* is a spatial edit: the first shot translates out, uncovering the shot that follows the edit. A different kind of transitions are due to panning and zooming but these are not usually related to shot changes.

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

**3.2.1. Plateau Detection** Comparison based on successive frames alone is not adequate for the detection of gradual transitions because changes are small [131]. One alternative is to use every $k$-th frame instead, i.e. to perform uniform temporal subsampling. However the larger separation between two frames used for comparison implies larger difference statistic within a shot. Such an effect is especially pronounced 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 robust detector for gradual transitions can be obtained as follows. Let us compare the $i$-th frame to the following $i+k$-th frame. We obtain a sequence of delayed inter-frame 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 preceding or successive values:

$$D_i^k \geq l \times D_{i-\lfloor k/2 \rfloor -1}^k or D_i^k \geq l \times D_{i+\lfloor k/2 \rfloor +1}^k, \; l \gg 1 \tag{9}$$

The value of $s$ is proportional to the difference between $k$ and the transition length. This kind of detector would respond also to abrupt changes. It should then be applied within each shot surrounded by cuts. The method also applies to non-linear, gradual, transition: it is the shape of the rises and falls at the plateau boundaries that changes. False detections can be reduced by requiring that the plateau values be higher than the average frame differences at the hard cuts.

**3.2.2. Detection by Modeling** Another important approach to the detection of video effects has been proposed by [20, 50] and is based on *effect modeling.* Mathematical models of video based on video production techniques are formulated. The classes and models are used to systematically design the feature detectors for spotting edit effects in digital video. The difference of the two approaches resembles that of region segmentation using region-growing or edge detection techniques. 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 of video is ill defined while the space of edits is much smaller. It is then much easier to develop analytical model of edits.

A dissolve operation from scene X to scene Y is a sequence of frames represented by:

$$\begin{aligned} E(t) &= \left[\frac{t}{L_2} Y\right]_{t \in [T_2, T_2 + L2]} + \\ &+ \left[\left(1 - \frac{t}{L_1}\right) X\right]_{t \in [T_1, T_1 + L1]} \end{aligned} \tag{10}$$

where $T_1, T_2$ represent the times at which the scaling of $X, Y$ begin and $L_1, L_2$ are the duration 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, the following holds:

$$F(t) = \frac{\frac{\delta E}{\delta t}}{X(x, y, t)} = -\frac{1}{L} \tag{11}$$

where $L$ represents the length of the dissolve expressed in frames. The following indicator of constancy is suggested in [48, 50]:

$$\text{Constancy} = \frac{\mathcal{N}(X)}{\sigma(X)(1 + |C_x - c_x| + |C_y - c_y|)} \tag{12}$$

where $\mathcal{N}(X)$ is the number of non-zero 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 centroids of image $X$ and the geometric image center respectively. The different possible dissolves can be classified according to their start/end times. By looking at the resulting categories, it can be seen that using eq. 11 to detect them is appropriate except when very similar sequences are being dissolved with precisely equal fade rates over the dissolve. This particular situation is seldom encountered in most commercially produced videos. A similar approach can be used to detect spatial edits, such as translates. As shown in [50], assuming that there is no scene action in progress during the edit, the following holds when the edit is a pure translation in the $x$ direction:

$$T(x, y, t) = \frac{\frac{\delta E}{\delta t}}{\frac{\delta E}{\delta x}} = \alpha_x \qquad (13)$$

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 [20]. 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.

**3.2.3. Feature Based Detection** Another interesting approach to the detection of gradual effects as well as hard cuts has been presented in [138]. During a cut or a dissolve, new intensity edges appear far from location of old edges. Edge pixels that appear far from existing edge pixels are considered as *entering* edge pixels, while edge pixels that disappears far from an existing edge pixel as an *existing* edge pixel. Cuts, fades and dissolves can be detected by counting the entering and exiting edge pixels, while whipes 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 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 $\rho_{in}$ and *exiting* edge pixel $\rho_{out}$ are computed.

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

**3.2.4. Twin comparisons** A simple technique is introduced in [139] to detect gradual transitions: the *twin-comparison* approach. The basic idea is that the frames before and after gradual transition are usually markedly different. By inspection of some video material, it was noted that the 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.

# 4. Highlights Extraction

The 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 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 [114] or from the detection of well defined events (such as flashlights). The following sections survey algorithms which are capable of detecting such highlights.

## 4.1. Flashlights

Scene containing flashlight can indicate the happening of important events or appearances of important persons. Besides providing important visual highlights to events, a flashlight may cause problems in scene analysis algorithms. An algorithm to detect flashlights is reported in [131]. It is observed that flashlights manifest themselves as two sharp peaks of approximately the same value in the plot of the inter-frame differences. Let us consider a moving window of size $m + 1$ along the frames and let $\{d_{i+k}\}_{k=0,\ldots,m-1}$ the inter-frame differences. These differences can be ordered in a non-increasing sequence $d_{l_0} \geq \cdots d_{l_{m-1}}$. A flashlight is detected whenever the following inequalities hold:

$$\frac{d_{l_0} - d_{l_1}}{d_{l_0}} \leq \epsilon \qquad (14)$$

$$d_{l_0} \geq p \times \frac{1}{m-2} \sum_{i=2}^{m-1} d_{l_i} \qquad (15)$$

$$d_{l_1} \geq p \times \frac{1}{m-2} \sum_{i=2}^{m-1} d_{l_i} \qquad (16)$$

Good results are reported adjusting $m$ to correspond to $1/2$ second of video, $\epsilon \approx 0.1$, and $p \in [3, 5]$.

## 4.2. Captions

Captions in video programs such as news, documentaries and coverage of sport events provide important information to the underlying stories. Captions 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. There are several approaches to locating captions in a video stream [73, 112, 120, 131]. After a caption has been located, it could be isolated from the background and then read by a OCR program. The following sections review two different approaches: the first relies on inter-frame detection and intra-frame reading while the second uses intra-frame detection and inter-frame validation and reading. A simpler strategy can be found in [112] 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 clusterd sharp edges, primarily due to characters with high contrast intensity. The detection algorithm proceeds by application of horizontal differential filter, thresholding to extract edge features, smoothing, clustering, and boundary rectangle computation.

**4.2.1. Caption Events** The algorithm was presented in [131]. The basic observation 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, the two cases necessitating slightly different approaches. Abrupt captions are characterized by high inter-frame pixel differences which are often localized at the bottom of the image. The algorithm presented in [131] is articulated into the following steps:

1. Extract a DC (or DC + 2AC) image sequence from an MPEG stream. This results in an image size of $40 \times 30$ from an original stream of $320 \times 240$.
2. Determine the cumulative pixel difference $d_m(A)$ over a region $A$ of consecutive frames $f_m$ and $f_{m+1}$. In the reported experiments region $A$ corresponded to the lower fourth of the frame.
3. A frame $f_l$ is marked as a possible start/end of a caption event if
$$d_l(A) \geq 2\ d_j(A),\ \forall j : 1 \leq |j - l| \leq m \qquad (17)$$
and let $E = \{j_l\}$ be the resulting set of frames.
4. The set $E$ is purged from shot boundaries by checking the inter-frame differences over the complement of region $A$.
5. Appearance/disappearance events are differentiated by looking at the amount of gradient in the vertical direction in region $A$. 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 of step 2. The cumulative pixel difference is not computed on adjacent frames but on frames that are $k$ (typically 20) frames apart.

**4.2.2. Caption Stills** A different approach is pursued in [73]. Each frame is analyzed to find candidate caption regions. A grey level image is segmented using a traditional split-and-merge algorithm. Some of the regions are removed from further consideration on the basis of their dimensions. Once the candidate regions are detected, they are tracked across frames by a block-matching technique. Linearly moving text as well as non perfectly stationary benefits from this tracking. The size of the blocks used for tracking is derived by the bounding box of the candidate text regions. These regions are found by binarizing and dilating the segmented image. The resulting connected regions are decomposed into smaller blocks until a condition on the fill factor is satisfied. The final blocks are then used for matching across frames. An important characteristic of captions in videos as generated by video-title machine is there relatively high contrast over the background. This provides yet another way to filter the candidate text regions: the blocks detected so far should intersect the area of high contrast as determined by high values returned by the Canny edge detector. The output of the algorithm is a new video, where each frame only contains candidate captions. Each frame can then be fed to an OCR program and the responses from the different frames can be integrated using a majority rule. The reported performance (fraction of overall text spotted by the algorithm) is very good (around 99% on a set of 8 videos providing the various combinations of stationary/moving text and stationary/moving scene).

## 4.3. Faces

Locating and recognizing faces in video sequences can provide useful information for segmenting the video stream (let us think at 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.

The following section will briefly review two different algorithms for detecting faces: the first one is based on the notion of feature space [13, 80, 117] while the second is based on multi-layer perceptrons [14].

The simplest way to detect objects in image processing has been that of normalized correlation. This approach can be shown to be optimal only for the case of a deterministic signal embedded in additive white Gaussian noise. When the problem becomes that of detecting a target *class* it is necessary to incorporate the underlying probability distribution of the objects of the given class in the pattern space. A principal components analysis is of particular interest as it provides hints to the dimensionality of the manifold spanned by the class individuals. Furthermore, an eigenspace decomposition (as derived from projecting onto the principal components) immediately lends itself to an alternative object detection strategy. An input (unknown) pattern is projected onto the linear subspace spanned by the principal components: the difference of the projected pattern from the original pattern (*residual error*) can be used to detect the patterns belonging to the target class by comparing it to a suitable threshold. The residual error can be interpreted as an estimate of a marginal component of the probability density of the object. A complete estimate must also incorporate a second marginal density based on the complementary *distance in feature space*. Using both probability densities results in much more reliable object detection [13, 117]. A multi-resolution approach can increase the efficiency of these techniques by reducing the computational load.

An alternative approach is presented in [14]. A large set of face examples and backgrounds (*counter-examples*) is used to train a Kohonen network. This results in a compressed base due to the Vector Quantization properties of the Kohonen networks. A multi-layer perceptron (MLP) is extensively trained on this base to ouput a class (face/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 shortly 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.

## 5. Reduced Data Representations for Video Shots

Once shots have been detected, the problem arises of summarizing them by extracting some representative *key-frames*. Different algorithms have been proposed so far, relying on the visual differences between shots [134, 142] or on the motion patterns [128].

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 [128] 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 of the algorithm, 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 $t$:

$$M(t) = \sum_i \sum_j |o_x(i,j,t)| + |o_y(i,j,t)| \qquad (18)$$

where $o_x(i,j,t), o_y(i,j,t)$ are the horizontal and vertical components of the optical flow at pixel $(i,j)$. The second step identifies local minima of $M(t)$. The algorithm progressively moves across the $M(t)$ curve and partitions the curve into intervals surrounded by local maxima whose values differes by at least 30%. The key frames are selected at the minimum of $M(t)$ in each interval. In the same paper the idea of a hierachical key frame selection methodology is also proposed: shot should be first classified into categories and appropriate key frame selection algorithm should then be chosen.

The algorithm proposed in [142] is based on a variant of the algorithms used for shot detection. 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 $F_k$ is found, it is considered as a key frame if it is followed by a continuos sequence of frames differing by at least $T_S$ and providing a cumulative difference (from $F_k$) $T_B$ greater than $T_S$.

Another algorithm is proposed in [134] as an aid to efficiently compute the similarity of shots. The use of a single representative frame is considered to be overly reductive. 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, or camera trajectories.

**Definition 1** *Given two video shots* $S_i = \left\{f^i_k\right\}_{k=b_i}^{e_i}$ *and* $S_j = \left\{f^j_k\right\}_{k=b_j}^{e_j}$*, the dissimilarity index* $D$ *is given by:*

$$D(S_i, S_j) = \min_{l,k} d(f^i_l, f^j_k), \quad b_i \le l \le e_i, \; b_j \le k \le e_j \qquad (19)$$

*where* $d(\cdot,\cdot)$ *is a suitable dissimilarity measure between two images.*

The experiments reported by the paper rely on the following dissimilarity measure:

**Definition 2** *Dissimilarity by luminance projections. For a given image* $f(j,k), j = 1,2,\ldots,J$ *and* $k = 1,2,\ldots,K$ *the luminance projection for the n-th row is*

$$l^r_n = \sum_{j=1}^{J} Lum(f(j,n))) \qquad (20)$$

*and the luminance projection for the m-th column is*

$$l^c_m = \sum_{k=1}^{K} Lum(f(m,k))) \qquad (21)$$

*The normalized dissimilarity of two images $f_i, f_j$ is then computed as:*

$$d_{l_p}(f_i, f_j) = \frac{1}{255 \cdot (J+K)} \left( \frac{1}{J} \sum_{n=1}^{K} |l_n^r(f_i) - l_n^r(f_j)| + \frac{1}{K} \sum_{m=1}^{J} |l_m^c(f_i) - l_m^c(f_j)| \right) \quad (22)$$

The performance using the luminance projections is claimed to be comparable to that obtained using full correlation between two frames. Video shots are then clustered hierarchically by successive grouping of the most similar pair. 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 computationally 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 above or equal to a given threshold $\epsilon$. 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. Shot similarity is most accurate when all of the frames are considered in the computation. However, the threshold $\epsilon$ 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. 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 [16]. 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 for detecting objects in the first place, to recover the kind of camera operation (i.e. zoom, pan, tilt, etc...), and to create salient video stills by mosaicing several frames.

### 6.1. 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 [124, 125]. 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 identifying 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 even if they are not connected. Each region can then be tracked over subsequent frames by using its descriptive motion model. The algorithm can be described by the following steps:

1. compute the motion vectors; this can be done by optical flow techniques or, possibly, by using the motion estimates for image blocks as computed by compression algorithms such as MPEG or H.263;
2. subdivide the image into non-overlapping regions: for each region estimate the affine parameters using a standard linear regression technique;
3. identify the set of motion hypothesis by clustering the motion models in the affine (six dimensional) space; outliers should be rejected from the analysis and nearby clusters are merged.

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 frame). A recent formalization of the layered representation can be found in [6]. Ayer and Sawhney present a formulation based on maximum likelihood estimation of mixture models and the minimum description length principle to address the main issues in layered motion representations: how many models adequately describe image motion, what are the motion model parameters and what is the spatial support layer for each model.

### 6.2. Camera Work

Camera operation information is very significant for the analysis and classification of video shots [57], since camera operation often explicitly reflects the communication intentions of the director. There are six main different camera operations: 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 [140, 112].

In order to detect camera operation, the motion vectors can be obtained by optical flow techniques [112] or by coding algorithms such as MPEG [140] or H.263. The first step is discriminating between static/motion scene: 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 the camera movement. Most of the motion vectors will be parallel to this vector. This may be checked by looking at 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 [112] or by a small absolute deviation from the modal direction $\theta_M$ as suggested in [140]:

$$\sum_k |\theta_k - \theta_M| < T \tag{23}$$

Zooming is characterized by a flat appeareance of the direction distribution: this can be detected by a small standard deviation of the distribution values. An alternative is to note that 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. A zoom is detected if the number of couple motion vectors at the start/end of rows/columns exceed a predefined threshold. This simple approach can be fooled by motion of large objects. More generally, the problem of recovering camera motion can be seen as that of estimating a $2 \times 2$ affine transformation which accounts for the dominant global view transformation. A thorough treatment can be found in [102].

### 6.3. 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 [28, 63, 72]. The approach presented in [63] is based on a three steps 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 are estimated by block-matching. The resulting vectors are then 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 that 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 temporaly uncertainty that usually characterize the queries. The solution proposed in [63] makes use of elastic matching to overcome the problem.
A different approach is proposed in [72]. 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 function are used. These functions are used 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 next 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 [28].

### 6.4. Mosaicing Video

A compact video representation is an important step toward efficient video coding, indexing and annotation. It has been observed [122] 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 [121]. 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 non-linear 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 multi-resolution 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 camera operations and motion. Once camera work has been accurately recovered [101, 102, 121] 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 mosaiced frames leads to the deletions of the putliers (moving objects) if their locations are not highly correlated over time. An interesting application is found in [136] where mosaics are used to described the trajectories of soccer players.

## 7. Acoustic Indexing

So far, research on video indexing has emphasized the visual aspect of the medium. However, it is obvious that sound carries a lot of information which can be necessary or ancillary to the comprehension of the video message. The information carried by the audio channel is itself extremely varied, ranging from dialogues to music to *sounds* like footsteps, doors slamming, etc. There are at least two levels at which the sound analysis can be carried out usefully: a first broad categorization of sounds (distinguishing music, speech, silence, noise) and a second, higher, level which relates to *transcripts* of complex acoustic phenomena such as music and speech.

### 7.1. Sound Analsysis

Detecting the audio context corresponding to a given set of frames provides useful indices into the video stream. Sound categorization can be used to detect scene transitions (as opposed to shot detection) and can segment the audio stream to improve the performance of the algorithm working at generation of transcripts of music or speech. In the following sections we will refer to two papers which have tackled the problem of audio context analysis. Both approaches aim at distinguishing among silent (low noise), dialogue, or music segments. The strategy presented in [86] is based on the audio characterization of the MPEG audio stream. MPEG specifies that the audio signal must be divided into equal width 32 sub-bands. The bands have uneven perceptual importance and compression is achieved by reducing the number of bits allocated to the non-perceptual ones. A polyphase filter bank is used which provides a good resolution wide band filter with reasonable frequency resolution. The audio characterization proposed in [86] proceeds along binary classifications. The first step is to discriminate between silent/non silent segments. This is achieved by looking at the average magnitude of the audio signal, which can be approximated by the $i$-th sub band data $s_i(n)$:

$$M_m = \frac{1}{32\;(N-1)} \sum_{i=1}^{32} \sum_{n=1}^{N-1} |s_i(n)| * h(m-n) \quad (24)$$

where $h(\cdot)$ represents the impulse response of the processing filter (Hamming, Welch or Bartlett). The use of $M_m$ is to be preferred to the short time energy function (which is based on the squared coefficients) as it is less sensitive to noise and large signals. Whenever $M_m$ is smaller than a predefined threshold the segment is assumed to be *silent* and removed from further consideration. The following step attempts to discriminate between dialog and non-dialog segments. The ratio of low frequency energy to high frequency energy is used. For the voiced signal energy concentrates below 1.5kHz frequency while for unvoice signal it is distributed in many bands. In MPEG encoded signals the ratio of energy in the first two sub-band with the cumulative energy of the remaining bands provides the required band energy ratio. Whenever the resulting ratio is above a given threshold, the segment is considered to be *non-dialogue*.

Continuity of the signal is another major property found in the non-dialog audio signal. Computation of the pause rate (i.e. the number of silent segments per time unit) provide a good measure of continuity of the signal and can further discriminate between dialog and non-dialog. Another clue to identify a dialogue segment is the presence of a well defined peak, in the autocorrelation of the low-frequency signal (*pitch*). If such a peak can be found, the segment is marked as dialogue. The value of the pause rate, together with loudness can be used as a measure of emotion in the talk. Promising results are reported. Different algorithms have been proposed in [54] to achieve similar sound categorization. A lot of music is characterized by the presence of pitches, i.e. notes with frequencies that remain approximately fixed for a finite period of time. These results in a striated structure of the spectrum. One way to measure *striation* is to slide an analysis window over the spectrum, collecting the peaks in frequency magnitude, and keeping a running tally of the average duration of a stationary frequency peak. The resulting histogram has a clear bimodal structure, with speach on the short duration side. The location of voiced speech is based on the use of split-comb filter to sense harmonic energy. When speech is present we expect to see several harmonics pitched within a speech-like band so the detector stretches a comb of $n_h$ harmonics over a range of frequency samples. When each harmonic settles on or very near a magnitude peak the comb output will be maximized and the pitch of the speech is given by the frequency of the topmost harmonic in the comb divided by $n_h$. To prevent unwanted response from noise, a split-comb is used: the original comb shifted up in frequency by half the pitch is subtracted. The information on pitch can also be used to discriminate the speaker among a small set of persons, such as found in a movie. Another interesting application is reported in [39]. Humming is converted into a sequence of pitch transitions. A note in the input is classified as either the same as the previous note (S), higher (U) or lower (D) than the previous note. The resulting words, composed from the alphabet $\{S, U, D\}$, are used by an approximate pattern matching algorithm to find the tune in a database of melodies. Content based retrieval from audio databases is also explored in [10].

### 7.2. Speech Analysis

The literature on speech recognition is extremely rich and the available systems are mature for useful applications in limited domains. Current state-of-the art systems are based on Hidden Markov Models and are capable of good recognition of continuous speech over large dictionaries (up to 60000 words [52, 103]). Current speech recognizers are very sensitive to different microphones and different environmental conditions in which their acoustic models were trained. Moreover, their performance is degraded by the presence of non linguistic phenomena in the sound stream (e.g. music, overlaid speech with noise/music and even foreign languages). The following section briefly review two approaches to the use of speech for indexing/retrieval of audio material. The first approach, used within the Informedia Project [52, 53], attempts to generate transcripts of the audio track, by recognizing (key)words which can then be used to index and retrieve the associated audio/video material. The second approach, used within the Speech Retrieval Project at ETH [103], is based on the generation of phone transcripts as opposed to word transcripts.

**7.2.1. Indexing by Word Recognition** The speech recognition system employed by Informedia is based on Sphinx-II, which is a large-vocabulary, speaker independent, continuous speech recognizer created at Carnegie Mellon University. It is based on senonic hidden Markov models to model between-word context-dependent phones. The system uses both frequency cepstral coefficients and the power spectrum. The language model consists of words, bigrams/trigrams with their (conditional) probabilities. Preliminary results on the performance of the system applied to broadcasted video material are reported in [51].

| Type of Speech Data | Word Error Rate |
|---|---|
| Speech benchmarks | 8-12% |
| Lab recorded speech | 10-17% |
| Narrator in TV studio | 20% |
| Documentaries video | 50-60% |
| Evening news | 65% |
| Commercials | 85% |

Extensive experiments have been done to quantify the values of recall and precision using the keywords extracted automatically from the audio material. The results obtained show that there is a close relation with the transcript word error. Tests to determine what levels of recall and precision are satisfactory for the users have not been performed yet.

**7.2.2. Indexing by Phone Recognition** As reported in [103] speech retrieval is similar to text retrieval with the following, important, dissimilarities:

- word boundaries are difficult to detect;
- recognition errors affect the retrieval effectiveness;
- the size of the indexing vocabulary is limited by the number of different words that can be recognized.

The last difference is of particular relevance. In a text retrieval applications it is possible to use indexing vocabulary with more than 500000 terms which are far beyond the possibilities of available speech recognition systems. A solution is to use different indexing features than words. In particular, if the indexing features can be modeled as acoustic units, a tractable vocabulary can be obtained by selecting a subset of words or by using arbitrary phone sequences. The approach described in [103] uses a set of 32 phones. By applying the phone recognizer to the available audio material a transcript of phones is generated. The transcript is then analyzed to extract efficient indexing features. The algorithm is based on a digital tree structure and proceeds by extension of the high frequency sequences and removal of low frequency ones. A set of 5000 features is derived, where each feature consists of 3 to 6 phones. The retrieval method is based on the well known technique TF/IDF (term frequency/inverse document frequency) to identify critical keywords and their relative importance for the video document. No quantitative evaluation of the system is availabale yet.

## 8. Recovering Video Structure

Efficient and reliable methods for the segmentation of the image part of a video into shots have already been reviewd in the previous section. In actual motion picture or video documents there can be up to thousands of cuts per hour. In order to browse quickly through the video material, both for indexing and retrieval, it is necessary to find objects with which span longer time intervals then shots, possibly representing sequences or story units [77, 130, 132, 133, 134, 135, 136, 137].

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 the capabilities of computer vision techniques, some edit effects (such as fades and dissolves) can be considered as hint to changes of scene (not only of shot boundary). A single scene can 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 very simplified approach is proposed in [27, 37]. The approach is based on a hierachical segmentation of the video stream using a decreasing threshold for the detection of cuts using histogram differences. Several color spaces are considered in order to find the one whose segmenting results more closely correlates with human judgements. The best spaces are found to be YIQ, $L^*a^*b^*$, and Munsell. A video segmenting method, called hierachical histogram-based video indexing, is proposed that segments a video sequence into several levels of subsequences using different levels of thresholding. 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 hierachical structure depends on the types of video sequences. Some characteristics of the derived structure could be profitably used to categorize the sequences (see also [32]). The structure of a video can also be explored by looking at the distribution over time over several, simple shot descriptors. An interesting example is reported in [54] 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 clearly 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 sub-average 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 [20]. After subdividing a video into shots, the first and last frame 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:

$$G_{ij} = \int L_{ij}(x)\, dx. \tag{25}$$

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 belong all 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 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 this section we will review three different approaches: using a program specific model [144], using visual similarity [133] and using general media-knowledge-based rules [4].

### 8.1. Video Structure by Visual Similarity

The whole presentation is based on the concepts of time constrained clustering of video shots and on the construction of a scene transition graphs 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 based on their visual contents and temporal localities to further condense repetitive shots of similar contents. Clustering is based on the shot distance defined by eq. (19) with the addition of a temporal component:

$$\hat{D}(S_i, S_j) = \begin{cases} D(S_i, S_j), & d_t(S_i, S_j) \leq T \\ \infty, & d_t(S_i, S_j) > T \end{cases} \tag{26}$$

   where $d_t(S_i, S_j)$ is the temporal distance between two shots. The following constraint is imposed to the clusters: *The distance between a shot of cluster $C_i$ and a shot of any other cluster must be inferior to the diameter of $C_j$*. Hierarchical clustering using a complete-link method satisfies this constraint and is used to generate shot clusters. The distance between two clusters is defined as the maximum distance between any two shots: the clustering algorithm is stopped whenever the distance between any two clusters exceed a predefined threshold (or all shots have been collapsed into a single cluster).
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.

All of the analysis steps do not require *a priori* models and have been tested on a variety of programs: cartoons, movie, documentary, sitcoms. The use of STG permits rapid nonlinear browsing of long video programs.

### 8.2. Video Structure by Media-based Rules

A quite different approach is presented by [4]. The paper 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 change. In motion picture and video, there are a wide range of possible temporally local clues for macroscopic changes, from the use of special transition effects, modification of soundtrack, change of editing rhythm, and modifications of the type of images due to change 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 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 $i > 2, j = 2, k > 2$, $S_{m+1}$ 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 3 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 shotting settings. A possibility is to compare the mean and the standard deviation of the distribution of hue-saturation vectors of the 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 structuration of viewer perception. Rhythm changes can be detected by looking at the prediction error of an order 2 auto-regressive modelling of shot durations:

$$T'_n = aT_{n-1} + bT_n \tag{27}$$

**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 seconds and music appears in the soundtrack, then there is a sequence limit at the beginning of the music.

In order to obtain a macro-segmentation from this data is necessary to go to the following steps:

- merging of compatible starting and ending points;
- resolving conflicts by use of precedence rules;
- adding segments where none was detected;
- choose representative images using the information on the rules which originated the sequences.

All of the steps are described in the paper. The rules have been derived from intensive analysis of video documents, readings in film theory, and discussion with film and video directors, teachers, critics and analysts.

### 8.3. 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 the available *a-priori* knowledge to parse the relevant programs. A television news program is a good example of this kind of video. The video parsing process proposed in [144] consists of three steps: temporal segmentation to partition a video sequence into shots, shot classification based on the spatial structure model of an anchorperson shot model and a temporal structure model of an entire news program, and, finally, visual abstraction to extract a set of key frames from each shot to represent the visual content of each news item.

1. *Temporal segmentation.* In this study, both the *pair-wise pixel comparison* and *histogram comparison* algorithms are used.
2. *Shot classification.* The second step in parsing is to classify the shots into anchorperson shots (*A shots*) and news shots. Three models are constructed for an *A shot*: region (anchorperson, news icon, news program title bar, anchorperson 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 been developing 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 follows:

$$P_A(i,j) = 1/N \sum_{n=1}^{N} p_n(i,j) \tag{28}$$

where N is the number of frames inside the shot and $p_n(i,j)$ is the value of a pixel at frame n.

3. *Visual abstraction.* The authors have been developing a unique and robust content-based key frame extraction technique that uses the algorithms already summarized (US patent pending).

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.
The resulting system was tested on news programs aired by the Singapore Broadcasting Corporation (SBC).

# 9. Low-level Visual Indices

The detection of shots, the analysis of camera work, the tracking of objects across frames and the acoustic analysys of the soundtrack, are operations that relate to the dynamic nature of video as opposed to the static characteristics of stills. Nevertheless, once a set of descriptive key frames has been selected to create a video abstract, the visual features that are commonly used to characterize still images can be used. This way, the set of available video descriptors is enriched by the description of image colors, textures, contrast, brightness. The new *low-level visual indices* are of remarkable importance as they provide the basis for intuitive query by sketch and for an efficient *vector* description of images in appropriate *features spaces.* Useful techniques such as relevance feedback, can be easily introduced to improve the retrieval effectiveness. It is also important to note that this kind of descriptors are important as they can be applied not only to video frames, but also to large databases of stills (*image banks*) whose market is becoming increasingly important. Low level indices are typically represented by small feature vectors (such as color histograms, texture descriptors, etc.) describing the complete image or a small number of regions within it. The indices can be used with *traditional* databases such as Postgres, which provide support for new data types along with operators tailored to them. In particular Postgres, allows support for user-defined indices as secondary indices with user-specified access methods. A review of algorithms for similarity indexing and their performances can be found in [126].

Currently there are only two general purpose products in this area. They are the outcome of the QBIC Project research at IBM and of the extensive work on similarity matching [66, 99, 100, 127] by R. Jain and co-workers at the Visual Computing Lab, UCSD, and at Virage Inc. IBM provides visual search mechanisms in their Ultimedia Manager product, and Illustra has a Visual Intelligence DataBlade that uses Virage's search engine and GUI tools.

The following sections present some algorithms to find low-level visual indices.

## 9.1. Color information

Several papers present works to visual-based image retrieval using *color information* [29, 33, 35, 40, 41, 46, 83, 109, 110, 111, 118, 115].

Within the SWIM project [41], images are divided into nine, not overlapping regions. Each image is segmented into regions using hue and chroma information. Each region is then assigned to the area to which its gravity center belongs. The hue space is divided into eight regions (corresponding to a single achromaticity area, six not overlapping color regions and a special skin color range). The shape of each region is then described by circularity and orientation. The region size is also described by a short code reflecting the fraction of total area covered. A 20-bit index is computed: location (3 bits), color (8 bits), size (4 bits), and shape (5 bits). All features are coarsely quantized to reflect qualitative specifications.

Another approach to image retrieval based on color information has been presented in [41]. The whole image and its nine subregion are described by a 512-entries color histogram. In order to perform efficient indexing, each histogram is keyed by two numbers derived from the 20 most populated bins. The bins are sorted by increasing distance from the color space origin and the key is computed by the weighted (bin contents) perimenter and angle using 20 bits. Range searches can then be efficiently implemented. A completely different approach based on Self-Organizing-Maps (SOMs) is reported in [145]. A hierachical structure of SOMs is built which clusters input patterns and represents them in a two dimensional space preserving the topological relations among patterns.

A different method for retrieving images based on the similarity of their color distribution has been used in QBIC (Query By Image Content) project[4] [46, 83]. The choosen color space is supposed to be quantized to $N$ levels. A match measure between the query and the data images can be defined as a pseudometric between the two distributions. Let $\mathbf{x}$ and $\mathbf{y}$ be two $N$-dimensional color histogram in some choosen color space (usually $N$ is 256 or 64), normalized to 1. Then the distance between $\mathbf{x}$ and $\mathbf{y}$ is defined as follows:

$$d^2_{hist}(\mathbf{x},\mathbf{y}) = (\mathbf{x}-\mathbf{y})^t \mathbf{A}(\mathbf{x}-\mathbf{y}) \qquad (29)$$

[4] The QBIC project aims to develop a system for indexing and retrieval of images from a large database based on information that capture the visual properties of images [29, 33, 46, 83]. QBIC tecniques serve as information filters and reduce the search for the user to a set of images with the desired properties from which (s)he makes a final selection. The content of an image is defined in terms of color, texture, shape of objects and regions, user sketch.

where $\mathbf{A} = [a_{ij}]$ is a similarity matrix, or $a_{ij}$ denote the perceptual similarity between the colors represented by bins $i$ and $j$. It can be demonstrated that $d^2_{hist}$ is a pseudometric under certain conditions on the $a_{ij}$s. In particular this is true if

$$a_{ij} = 1 - d_{ij}/d_{max} \qquad (30)$$

where $d_{ij}$ is the Euclidean distance between colors $i$ and $j$ in the choosen color space, and $d_{max} = max_{ij}(d_{ij})$.
The histogram distance defined above is computationally intensive and it operates on high dimensional features. In the QBIC system is proposed the use of a low-dimensional distance between color distributions, that is simple to compute. It is proved that this match measure, called *average color distance*, is a lower bound on the histogram distance, so it can be used to filter the database without any misses. Therefore the expensive histogram distance $\mathbf{C}$ can be limited to a smaller set of images. Specifically, let $\mathbf{C}$ be the $3 \times N$ matrix whose $i$-th column are the three coordinates of each color in the choosen $N$ dimensional color space. If $\mathbf{x}$ is the $N$ dimensional color histogram of an image, its $3 \times 1$ average color vector is

$$\mathbf{x}_{ave} = \mathbf{Cx} \qquad (31)$$

and the squared average color distance is defined by:

$$d^2_{ave}(\mathbf{x}, \mathbf{y}) = (\mathbf{x} - \mathbf{y})^t \mathbf{C}^t \mathbf{C} (\mathbf{x} - \mathbf{y}). \qquad (32)$$

It is shown that for each pair of color distributions $\mathbf{x}$, $\mathbf{y}$ and with the $\mathbf{A}$ matrix defined as above:

$$d^2_{hist}(\mathbf{x}, \mathbf{y}) \geq \lambda_1 d^2_{ave}(\mathbf{x}, \mathbf{y}) \qquad (33)$$

where $\lambda_1$ is a positive constant that depends only on the similarity matrix $\mathbf{A}$ and on the definition of the colors in the matrix $\mathbf{C}$.
Experiments with a variety of color spaces showed that the best performance was obtained using a variant of the Munsell color space. This is a 3-dimensional space where the Euclidean distance between two colors correspond over a large part of the space to the perceptual difference between them.

### 9.2. Shape information

Retrieval of images from large databes based on shape content has been proposed in several papers [15, 33, 76, 91, 94].

In the QBIC project [33] similarity by shape works on objects of the database, or parts of a scene that were outlined in the so-called *database population step*. The considered shape features are computed on the bitmap that defines the shape and are: *area, circularity, eccentricity, major axis orientation* and a set of 18 *moments* invariant to the affine transform. The distance between shapes is defined as the weighted Euclidean distance, where the weights are the inverse variance for each feature. If $\mathbf{x}$, $\mathbf{y}$ are the feature vectors weighted by the variance, the distance is defined as

$$d(\mathbf{x}, \mathbf{y}) = (\mathbf{x} - \mathbf{y})^t (\mathbf{x} - \mathbf{y}) \qquad (34)$$

therefore if the data are mapped via an orthonormal transformation $\mathbf{A}$ their distance will be preserved:

$$(\mathbf{Ax} - \mathbf{Ay})^t (\mathbf{Ax} - \mathbf{Ay}) = (\mathbf{x} - \mathbf{y})^t \mathbf{A}^t \mathbf{A} (\mathbf{x} - \mathbf{y}) = d(\mathbf{x}, \mathbf{y}). \qquad (35)$$

$\mathbf{A}$ is for example the Karhunen Loeve transform, or the discrete Fourier transform. If $a_i$ is the $i$-th row of $\mathbf{A}$, the distance can be written in two parts, so that:

$$d(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{m} (a_i(x_i - y_i))^2 + \sum_{i=m+1}^{n} (a_i(x_i - y_i))^2 \qquad (36)$$

where $m$ are the first most important entries. The error introduced by computing the distance on a subset of the transformed features is given by

$$\delta = \sum_{i=m+1}^{n} (a_i(x_i - y_i))^2 \geq 0. \qquad (37)$$

That means that the truncated distance can be used as filter, because it will always under-estimate the weighted Euclidean distance.

Two different types of image descriptions based on shape information are available in Photobook[5]. The first type [91, 94] is based on a principal components representation of images. The underlying idea is that patterns of a same category, e.g. faces, are not evenly distributed over the image space. Rather they cluster in a relatively low dimensional space which can be well represented by a small number of directions: the eigenvectors of the autocorrelation matrix. Projections of the visual patterns onto the set of principal directions yield a compact representation which can be used both for compression and for browsing [12]. This kind of pattern description could be called semantics-preserving and, due to its specificity to the category for which is computed, can also be used effectively to discriminate objects of the given class from other patterns.
The second type is related to the previous type. It is based on the computation of the eigenmodes of free vibration of the Finite Element Model of the available shape. The eigenmodes provide an orthogonal, frequency-ordered description of the shape and its natural (i.e. physical) deformations. They represent an object-centered coordinate system for describing feature locations. They also provide a compact description of the shape's deformations.

### 9.3. Sketch information

The retrieval of images is based on a rough user sketch.
In QBIC [33] the retrieval is supported by a reduced

[5]The Photobook system, developed at MIT Media Lab, is a set of interactive tools for browsing and searching images and image sequences [43, 79, 87, 88, 90, 91, 92, 93, 94, 95, 96]. Direct search is made possible by the use of semantics-preserving image compression, which reduces images to a small set of perceptually significant coefficients.

resolution ($64 \times 64$) binary edge map of each image in the database. The search algorithm works through the following steps:

- reduce the user sketch to size $64 \times 64$ pixels;
- divide this into blocks of $8 \times 8$ pixels;
- correlate each block with a corresponding search area of size $16 \times 16$ in the database images (the correlation is done as a logical binary correlation);
- compute a score for each database image as the sum of the correlation scores of each block.

### 9.4. Texture information

An important characteristic of images is their textural aspect [33, 41, 43, 96, 107, 108].

In the SWIM project [41], several techniques based on textures have been compared [68] using a performance measure relevant to image retrieval applications. The performance is defined by the percentage of textures belonging to the same class of the target among the $n$ nearest patterns found. Four different statistical techniques have been compared:

- SAR (simultaneous auto-regressive) model;
- RISAR (rotationally invariant SAR) model;
- MRSAR (multiresolution SAR) model;
- Tamura features (contrast, coarseness, and directionality).

The best performance on the complete Brodatz database are obtained by MRSAR using 4 different resolution levels. Texture pickers can be used to supplement the other interactive tools used by the user to compose a visual query to access an image database using the image contents.

A different technique based on a texture description that mirrors similarity judgements of human observers has been used in the Photobook system [43, 96]. The image description is based on a Wold decomposition of regular stationary stochastic processes. In particular, a random field is considered as a superposition of three mutually orthogonal components:

1. An harmonic field, corresponding to discontinuities in both dimensions, is closely related to the periodicity of the texture. The main peaks of the energy spectrum are used to described the texture.
2. An evanescent field, corresponding to discontinuities in only one dimension, is related to the directionality of the texture. The evanescent field is estimated by fitting a line in the 2D spectral domain.
3. An indeterministic field, corresponding to bidimensional continuity, is related to randomness. The indeterministic field is described by a multiscale simultaneous auto-regressive model.

Other texture characterizations have been investigated in Photobook, among them the detection of the main perceptually relevant orientation. A useful application of texture characterization is that of annotation. Several object/stuff categories can be characterized by a set of positive/negative examples, described by their textural features using different models. The user then starts providing the system with labels for the stored patches. Patches are organized as a hierachical tree for each of the models and subsets of the trees describe a given category, propagating the label to all parented nodes.

Texture features are used in the QBIC project too [33]. They are computed on the gray level images. The selected features are a modified version of *coarseness (O)*, *contrast (C)*, and *directionality (D)*. The distance between two textures is defined as a weighted Euclidean distance between the corresponding vectors $\mathbf{x}$, $\mathbf{y}$ in the 3-dimensional texture space:

$$d_t(\mathbf{x}, \mathbf{y}) = \sum_{i=0}^{2} (x_i - y_i)^2 / \sigma_i^2 \tag{38}$$

where $\sigma$ is the standard deviation vector of the texture vectors in the database.

## 10. Tools and Interfaces

The segmentation of a video is only the first step toward building a video management system. After the proper basic unit are determined, the problem arises of interacting with the material: indexing, retrieval, and browsing should be supported. Browsing is a more casual application than information retrieval and is a highly interactive process. It is then necessary to support this kind of operation with friendly tools which enable the user to explore the video contents with ease [9, 19, 20, 21, 25, 58, 59, 75, 78, 104, 113, 122, 123, 141]. From this points of view, using simple VCR-like controls such as fast forward/backward is not appropriate, since it requires a high level of attention and generates strain.
The following sections present some tools for browsing video and annotating it.

### 10.1. Media Streams

Over the past two years, members of the MIT Media Laboratory's Learning and Common Sense Section (Marc Davis with the assistance of Brian Williams and Golan Levin under the direction of Prof. Kenneth Haase) have been building a prototype for the annotation and retrieval of video information. This system is called Media Streams [25].

In designing a visual language for video content we must think about the structure of what is being represented. A video camera produces a temporal stream of image and sound data represented as a sequence of frames played back at a certain rate. Traditionally, this stream of frames is segmented into units called *clips*. Current tools for annotating video content used in film production, television production, and multimedia, add descriptors (often keywords) to clips. There is a significant problem with this approach. By taking an incoming video stream, segmenting it into various clips, and then annotating the content of those clips, a fixed segmentation of the content of the video stream results.

In the representation proposed by *Media Streams*, the stream of frames is left intact and is annotated by multi-layered annotations with precise time indices (beginning and ending points in the video stream). Annotations could be made within any of the various categories for media annotation discussed below (e.g., characters, spatial location, camera motion, dialogue, etc.) or contain any data the user may wish. The result is that this representation makes annotation pay off - the richer the annotation, the more numerous the possible segmentations of the video stream. Clips change from being fixed segmentations of the video stream, to being the results of retrieval queries based on annotations of the video stream. In short, in addressing the challenges of representing video for large archives what we need are representations which make clips, not representations of clips.

### 10.2. VideoMAP and VideoSpaceIcon

A set of tools for anatomizing video content and interacting with stored video has been presented in [122]. The proposed system is based on four processing stages. The first stage is image processing for general feature extraction on each frame in the video sequence (average intensity, intensity distribution, intensity histogram,...). The second stage filters the output of the first stage to extract more specific features (correlation, inter-frame difference,...). The third stage processes the data streams to generate the basic information needed for characterizing the video (frame change rate, global change flows,...). In the fourth stage, cut points, camera operations, and other shot-identifying characteristics such as shot color are obtained. The following tools are provided to visualize the result of the processing stages and to view video material in new, compact forms:

- VideoMAP is the visual representation of the video indices held in the computer. The indices are displayed on a time line. The cut points are clearly shown in each feature as pattern discontinuities. The image flow patterns is obtained by filtering the edge images of spatio-temporal video frames. The authors call this the *video X-ray.* Camera operations and object motion within a scene can be discerned from the video X-ray.
- VideoSpaceIcon is a tool that displays spatial and temporal video features as icons. It utilizes the camera operation parameters that can be automatically detected from the video X-ray of VideoMAP. VideoSpaceIcon is space oriented. It is created by changing the image position and size for each frame according to the camera operation parameters. By analyzing the flow generated by the motion vectors, camera operation parameters can be determined. One interesting aspect of the VideoSpaceIcon is that object movements can be reproduced in a very simple manner. By overlapping sequential image frames in the video space, a moving object actually moves.

### 10.3. Hyper-linked Video

The automatic visualization of structure for video editing is considered in [123]. The video information has a hierarchical structure: scenario, scenes and shots. The scenario is an entire story composed of a number of scenes and each scene consists of one or more shots. In this structure, the upper parts have symbolic descriptions and the lower parts have more visual descriptions. The symbolic descriptions consist of verbal and semantic indications. The visual descriptions consist of geographical and temporal indications. Examples of these descriptions are the shapes and positions of certain objects, the tracks of moving objects, the times when objects appear or disappear, and the times when objects meet in the video. The authors have been developing a system which can analyze and visualize this hierarchical structure, including hyper-links between descriptions. Image-recognition technology is used to achieve automatic descriptions and hyper-links of video structure.

- *Video Analysis Functions.* The automatic cut-separation function divides a video strip into cuts. The cut-separation function evaluates the correlation between the color of two adjacent frames in a video. The motion analysis function looks at the motion characteristics of the filmed object or at the camera operation. To extract a moving picture, two differential images of three consecutive frames are binarized and the logical AND operation of these two is performed. To extract a still object, edge detection along with color and brightness comparison is used. A new function determines the presence of objects by their unique color combinations. The object presence judgment function enables an automatic description of every cut in which a specific object appears. In other words, the function builds up hyper-links between identical objects in different cuts. In this system, the node of the link is the filmed object itself which, users normally feel, carries semantic information. This difference in node has enabled great progress in visualization and in handling of the hierarchical video structure. Therefore, the authors call it an *Object-Link.*
- *Visualization.* In this project, the authors represent descriptions which are automatically provided by video analysis functions, as well as user annotations. In designing visualization as an interface of the intelligent video authoring system, there are two important considerations: visualization of the results of automatic analysis in the video structure and visualization of how the intelligent functions

work in the interactive session. To ensure usable visualization of a function, the algorithm of the function should be designed so that it can be understood and used intuitively. In this system, all the intermediate results in the processing of functions are made visible. Therefore it is easy even for non-experts in image processing to adjust the procedural parameters of functions.

The next step is to develop more meaningful visualizations, which may include some semantic recognitions in the hierarchical structure of information (including audio), in order to achieve a more friendly and cooperative environment for users of computing systems.

### 10.4. Hierarchical Video

The hierarchical structure of video has exploited in the SWIM project. This project relies [113, 141] on a combination of tools already presented in the multimedia literature:

- interactive video icons (or micons): each video shot is represented by a selected key-frame, can be played using a video streamer and has a volumetric representation obtained by stacking frames one over the other (starting from the last one) with a single pixel displacement in the horizontal and vertical direction; the top and side views of the resulting parallelogram can provide the user with visual cues to the structure of the video material;
- video streamers based on the micon structure: frames are stacked dynamically in reverse order with respect to the micons (the current frame is displayed on top of stack);
- hierarchical video magnifier: video is divided into (a small number of) segments of equal length, each of them represented by the central frame; each of them is then exploded into subsegments on request of the user: this allow for a compact representation of selected segments of video material which can be presented on the screen within their higher level contexts.

### 10.5. STL

An approach to video-indexing relying on a description language is proposed by Del Bimbo et al. [9, 19, 20, 21] at the Image Science and Technology Research Center. In order to avoid the direct access to raw image data they devised an appropriate description language supporting the representation of the contents of pictorial data. This language, named *Spatio Temporal Logic* (STL), comprises a framework for the qualitative representation of the contents of image sequences, which allows for treatment and operation of content structures at a higher level than pixels or image features. This language inherits the possibility of formulating qualitative and incomplete descriptions from Temporal Logic. Furthermore, the use of Symbolic Projections to describe spatial relationships of objects enables the description of contents at a level of granularity suitable for the decription of video sequence dynamics. A scene sequence is described by specifying the geometric relationships of couples of objects. Queries are expressed in terms of sample scenes that are created through the use of a visual scene editor, following the specification-by-example paradigm. The scene is then automatically translated into an STL description by a parser. The user has control on the level of detail that the parser uses in the description. Finally, the resulting description is matched to the available database. There is currently no fully automatic way by which an STL description of a new video sequence can be entered into the database. The user must reproduce the relevant sequence dynamics with the same interface used to state queries, with the support of a digital VCR interface to the digitized videos. Extensions of STL with metric qualifiers to account for distances, speed and the like, have also been proposed by the authors. The STL description language has been incorporated into a prototype system for navigation and visualization of movies content [21]. The system is aimed at providing a rich hypermedial framework to describe filmic information browsing taking into account both the *naive* (e.g. questions like 'What is in the scene?') and *expert* (regarding shooting and editing techniques) approach. An abstract model of a film is built using domain classes, attributes, and relationships. The resulting model resembles a grammar so that a film can be considered as a *grammatical sentence*. The syntactic element is the *shot*, the *scene* is the semantic unit, and the *edit effects* can be considered as a set of punctuation signs.
Queries are performed by selecting a *perspective* on the film model. Selecting a perspective amounts to focus the query only on the selected objects. The user can then create new perspectives or modify previous ones using a friendly interface.

## 11. Conclusions

Effective use of multimedia material requires efficient way 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 unit, enabling effective non-linear browsing through the material;
- automatic image and shot annotation, through computer derived visual indices and constrained object recognition (e.g. captions and faces).

The algorithms available in the literature provide a good coverage of the issues relevant to the development of a Computer Aided Video Indexing System (CAVIS). The development of open, commercial, database systems of the object-relational type (e.g. Ilustra) which provide support for user data types and user access functions, holds promise of seamless integration of the outputs of a CAVIS system with the more traditional kind of data managed by DBMS.

## References


[1] N. R. Adam and B. K. Bhargava. *Digital Libraries.* Springer, 1995.

[2] G. Ahanger, D. Benson, and T. D. C. Little. Video Query Formulation. Technical Report MCL 01-09-1995, Multimedia Communication Laboratory, Boston University, 1995.

[3] G. Ahanger and T. D. D. Little. A Survey of Technologies for Parsing and Indexing Digital Video. *Journal of Visual Communication and Image Representation*, 7(1):28–43, March 1996.

[4] P. Aigrain, P. Joly, and V. Longueville. Medium Knowledge-Based Macro-Segmentation of Video into Sequences. In Mark Maybury, editor, *Proceedings IJCAI '95 Workshop on Intelligent Multimedia Information Retrieval*, Montreal, August 1995.

[5] F. Arman, A. Hsu, and M. Y. Chiu. Feature Management for Large Video Databases. In *Proceedings SPIE Storage and Retrieval for Image and Video Databases*, pages 2–12, 1993.

[6] 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.

[7] F. Banfi. Image Databases: State of the Art and Future Directions. Internal Working Paper 96-10, University of Fribourg, Fribourg, Switzerland, September 1996.

[8] J. Barros, J. French, and W. Martin. System for Indexing Multi-Spectral Satellite Images for Efficient Content-Based Retrieval. Technical report, University of Virginia, 1995.

[9] A. Del Bimbo, E. Vicario, and D. Zingoni. Symbolic Description and Visual Querying of Image Sequences Using Spatio Temporal Logic. *IEEE Trans. on Knowledge and Data Engineering*, 1995.

[10] T. Blum, D. Keislar, J. Wheaton, and E. Wold. Audio Databases with Content-based Retrieval. In Mark Maybury, editor, *Proceedings IJCAI '95 Workshop on Intelligent Multimedia Information Retrieval*, Montreal, August 1995.

[11] A. Brink, S. Marcus, and V. S. Subrahmanian. Heterogeneous Multimedia Reasoning. *Computer*, pages 33–39, September 1995.

[12] R. Brunelli and O. Mich. SpotIt! An Interactive Identikit System. *Graphical Models and Image Processing*, 58(5):399–404, September 1996.

[13] R. Brunelli and T. Poggio. Template Matching: Matched Spatial Filters and Beyond. A.I. Memo No. 1549, Massachusetts Institute of

Technology, 1995. to appear on Pattern Recognition.

[14] G. Burel and D. Carel. Detection and Localization of Faces on Digital Images. *Pattern Recognition Letters*, 15:963–967, 1994.

[15] A. Califano and R. Mohan. Multidimensional Indexing for Recognizing Visual Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(4):373–392, April 1994.

[16] C. Cedras and M. Shah. Motion-Based Recognition: a Survey. *Image and Vision Computing*, 13(2):129–155, 1995.

[17] S.-F. Chang, J. R. Smith, and H. Wang. Automatic Feature Extraction and Indexing for Content-Based Visual Query. CU/CTR technical report, Columbia University, 1995.

[18] T.S. Chua and L.-Q. Ruan. A Video Retrieval and Sequencing System. *ACM Transactions on Information Systems*, 13(4):373–407, October 1995.

[19] G. Congiu, A. Del Bimbo, and E. Vicario. Iconic Retrieval by Contents from Databases of Cardiological Sequences. In *Int. Symp. on Visual Databases*, Lausanne, Switzerland, April 1995.

[20] 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.

[21] J. M. Corridoni, A. Del Bimbo, and D. Lucarella. Navigation and Visualization of Movies Content. In *Proceedings of International Conference on Visual Languages*, Darmstadt, Germany, September 1995.

[22] A. Csinger and K. S. Booth. Reasoning about Video: Knowledge-Based Transcription and Presentation. In J. F. Nunamaker and R. H. Sprague, editors, *27th annual Hawaii International Conference on System Sciences*, volume 3, pages 599–608, January 1994.

[23] G. Davenport and M. Murtaugh. ConText: Toward the Evolving Documentary. In *ACM Multimedia 95*, pages 5–8, San Francisco, California, November 1995.

[24] G. Davenport, T. A. Smith, and N. Pincever. Cinematic Primitives for Multimedia. *IEEE Computer Graphics & Applications*, pages 67–74, July 1991.

[25] M. Davis. Knowledge Representation for Video. In *Proceedings of the 12th National Conference on Artificial Intelligence*, volume 1, pages 120–127, Seattle, Washington, July 31 - August 4 1994. AAAI.

[26] 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)*, number 2187 in SPIE, pages 44–55, San Josè, CA, February 1994.

[27] 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*, volume 2420, 1995.

[28] N. Dimitrova and F. Golshani. Motion Recovery for Video Content Classification. *ACM Transactions on Information Systems*, 13(4):408–439, October 1995.

[29] C. Faloutsos, M. Flickner, W. Niblack, D. Petkovic, W. Equitz, and R. Barber. Efficient and Effective Querying by Image Content. Technical Report RJ 9453 (83074), IBM Research Division, Almaden Research Center, 1993.

[30] 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*, pages 163–174, 1995.

[31] C. Faloutsos and D. W. Oard. A Survey of Information Retrieval and Filtering Methods. Technical Report CS-TR-3514, University of Maryland, August 1995.

[32] S. Fischer, R. Lienhart, and W. Effelsberg. Automatic Recognition of Film Genres. Technical Report TR-95-036, Universitaet Mannheim, 1995.

[33] 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 System. *Computer*, pages 23–32, September 1995.

[34] E. A. Fox. Advances in Interactive Digital Multimedia Systems. *Computer*, pages 99–21, October 1991.

[35] B. V. Funt and G. D. Finlayson. Color Constant Color Indexing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(5):522–529, May 1995.

[36] D. Le Gall. MPEG: A Video Compression Standard for Multimedia Applications. *Communications of the ACM*, 34(4):46–58, April 1991.

[37] U. Gargi, S. Oswald, D. A. Kosiba, S. Devadiga, and R. Kasturi. Evaluation of Video

Sequence Indexing and Hierachical Video Indexing. In *Proceedings of SPIE Conference on Storage and Retrieval for Image and Video Databases III*, volume 2420, 1995.

[38] D. J. Gemmell, H. M. Vin, D. D. Kandlur, P. V. Rangan, and L. A. Rowe. Multimedia Storage Servers: a Tutorial. *Computer*, pages 40–49, May 1995.

[39] A. Ghias, J. Logan, D. Chamberlin, and B. C. Smith. Query By Humming – Musical Information Retrieval in an Audio Database. In *Proceedings of ACM Multimedia '95*, San Francisco, CA, November 5-9 1995.

[40] Y. Gong and M. Sakauchi. Detection of Regions Matching Specified Chromatic Features. *Computer Vision and Image Understanding*, 61(2):263–269, 1995.

[41] Y. Gong, H. Zhang, H. C. Chuan, and M. Sakauchi. An Image Database System with Content Capturing and Fast Image Indexing Abilities. In *Proceedings of IEEE International Conference on Multimedia Computing and Systems*, pages 121–130, Boston, May 1994.

[42] A. S. Gordon and E. A. Domeshek. Conceptual Indexing for Video Retrieval. In Mark Maybury, editor, *Proceedings IJCAI '95 Workshop on Intelligent Multimedia Information retrieval*, Montreal, August 1995.

[43] M. M. Gorkani and R. W. Picard. Texture Orientation for Sorting Photos "at a Glance". In *IEEE Conference on Pattern Recognition*, Jerusalem, October 1994.

[44] V. N. Gudivada and V. V. Raghavan. Content-Based Image Retrieval Systems. *Computer*, pages 18–22, September 1995.

[45] V. N. Gudivada and V. V. Raghavan. Design and Evaluation of Algorithms for Image Retrieval by Spatial Similarity. *ACM Transactions on Information Systems*, 13(2):115–144, April 1995.

[46] J. Hafner, H. S. Sawhney, W. Equitz, M. Flickner, and W. Niblack. Efficient Color Histogram Indexing for Quadratic Form Distance Functions. *IEEE Trans. on PAMI*, 17(7):729–735, July 1995.

[47] 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*. American Association of Artificial Intelligence, August 1994.

[48] A. Hampapur, R. Jain, and T. Weymouth. Digital Video Segmentation. In *Proceedings Second Annual ACM MultiMedia Conference and Exposition*. ACM, October 1994.

[49] 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.

[50] A. Hampapur, R. Jain, and T. Weymouth. Production Model based Digital Video Segmentation. *Journal of Multimedia Tools and Applications*, 1(1):9–46, 1995.

[51] A. G. Hauptmann. Speech for Multimedia Information Retrieval. In *Proceedings of the International Conference on Tools with Artificial Intelligence ICTAI '95*, Washington DC, October 1995.

[52] A. G. Hauptmann and M. Smith. Text, Speech, and Vision for Video Segmentation. In Mark Maybury, editor, *Proceedings IJCAI '95 Workshop on Intelligent Multimedia Information Retrieval*, Montreal, August 1995.

[53] A. G. Hauptmann and M. J. Witbrock. Informedia: News-on-Demand Multimedia Information Acquisition and Retrieval. In Mark Maybury, editor, *Proceedings IJCAI '95 Workshop on Intelligent Multimedia Information Retrieval*, Montreal, August 1995.

[54] M. K. Hawley. *Structure out of Sound.* PhD thesis, School of Architecture and Planning, Massachusetts Institute of Technology, September 1993.

[55] Th. Hermes, Ch. Klauck, J. Kreyss, and J. Zhang. Content-based Image Retrieval. In *Proceedings of CASCON '95*, Toronto, Ontario, Canada, November 1995.

[56] Th. Hermes, Ch. Klauck, J. Kreyss, and J. Zhang. Image Retrieval for Information Systems. In *Proceedings of IS&T/SPIÈS Symposium on Electronic Imaging: Science & Technology*, San Josè, CA, USA, 5-10 February 1995.

[57] N. Hirzalla and A. Karmouch. Detecting Cuts by Understanding Camera Operation for Video Indexing. *Journal of Visual Languages and Computing*, 6:385–404, 1995.

[58] R. Hjelsvold. Video Information Contents and Architecture. In *Proceedings of the 4th Intl. Conf. on Extending Database Technology*, Cambridge, UK, 28-31 March 1994.

[59] R. Hjelsvold, S. Langorgen, R. Midtstraum, and O. Sandsta. Integrated Video Archive Tools. In *Proceedings of Multimedia '95*, San Francisco, 1995. ACM.

[60] R. Hjelsvold and R. Midtstraum. Modelling and Querying Video Data. In *Proceedings of the 20th Very Large Databases Conference*, Santiago, Chile, 1994.

[61] R. Hjelsvold, R. Midtstraum, and O. Sandsta. Searching and Browsing a Shared Video Database. In *Proceedings of the International Workshop on Multimedia Database Management Systems*, Blue Mountain Lake, New York, August 28-30 1995.

[62] 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.

[63] M. Ioka and M. Kurokawa. Estimation of Motion Vectors and their Application to Scene Retrieval. *Machine Vision and Applications*, 7:199–208, 1994.

[64] C. E. Jacobs, A. Finkelstein, and D. H. Salesin. Fast Multiresolution Image Querying. In *Proceedings of SIGGRAPH*, Los Angeles, 1995.

[65] R. Jain. InfoScopes: Multimedia Information Systems. Technical Report VCL-95-107, Visual Computing Laboratory, University of California, San Diego, 1995.

[66] R. Jain, S. N. J. Murthy, P. L.-J. Chen, and S. Chatterjee. Similarity Measures for Image Databases. In *FUZZ-IEEE '95*, 1995.

[67] 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.

[68] A. Kankanhalli, H. J. Zhang, and C. Y. Low. Using Texture for Image Retrieval. In *Proceedings of ICARCV'94*, Singapore, 1994.

[69] 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*, pages 252–258, Charlottesville VA, September 1994.

[70] 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*, pages 64–75, Washingotn DC, October 12-14 1994.

[71] 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*, number 2420 in SPIE, pages 238–248, 1995.

[72] 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*, pages 25–36, 1993.

[73] R. Lienhart and F. Stuber. Automatic Text Recognition in Digital Videos. In *Proceedings of SPIE 2666-20: Image and Video Processing IV*, 1996.

[74] D. C. Little, G. Ahanger, R. J. Folz, J. F. Gibbon, F. W. Reeve, D. H. Schelleng, and D. Venkatesh. A Digital Video-on-Demand Service Supporting Content-Based Queries. In *Proceedings ACM Multimedia*, Anaheim CA, August 1993.

[75] W. E. Mackay and G. Davenport. Virtual Video Editing in Interactive Multimedia Applications. *Communications of the ACM*, 32(7):802–810, July 1989.

[76] R. Mehrotra and J. E. Gary. Similar-Shape Retrieval in Shape Data Management. *Computer*, pages 57–62, September 1995.

[77] 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.

[78] S. Messelodi and C.M. Modena. VCB: Video Clip Browser and more. Technical Report #9512-09, IRST, Trento, Italy, December 1995.

[79] T. P. Minka and R. W. Picard. Interactive Learning using a *Society of Models*. Technical Report No. 349, MIT Media Laboratory, 1995.

[80] B. Moghaddam and A. Pentland. Probabilistic Visual Learning for Object Detection. In *Proceedings of the International Conference on Computer Vision*, Cambridge, MA, 1995.

[81] S. Moni and R. L. Kashyap. A Multiresolution Representation Scheme for Multimedia Databases. *Multimedia Systems*, 3:228–237, 1995.

[82] A. D. Narasimhalu. Multimedia Databases. *Multimedia Systems*, 4(5):226–249, 1996.

[83] W. Niblack, R. Barber, W. Equitz, M. Flickner, E. Glasman, D. Petkovic, P. Yanker, C. Faloutsos, and G. Taubin. The QBIC Project: Querying Images by Content Using Color, Texture, and Shape. In *Proceedings SPIE Storage and Retrieval for Image and Video Databases*, pages 173–181, 1993.

[84] H. Nishiyama, S. Kin, T. Yokoyama, and Y. Matsushita. An Image Retrieval System Considering Subjective Perception. In *Proceedings of CHI '94*, volume 4, pages 30–36, Boston, 1994.

[85] V. E. Ogle and M. Stonebraker. Chabot: Retrieval from a Relational Database of Images. *Computer*, pages 40–48, September 1995.

[86] N. V. Patel and I. K. Sethi. Audio Characterization for Video Indexing. In *Proceedings*

*of IS&T/SPIE 2670: Storage & Retrieval for Image and Video Databases IV*, San Josè, CA, 1996.

[87] A. Pentland, R. W. Picard, G. Davenport, and K. Haase. Video And Image Semantics: Advanced Tools for Telecommunications. *IEEE Multimedia Magazine*, 1(2):73–75, 1994.

[88] A. Pentland, R. W. Picard, and S. Sclaroff. Photobook: Tools for content-based manipulation of image databases. In *SPIE, Storage and Retrieval Image and Video Databases II*, 1994.

[89] E. G. M. Petrakis and S. C. Orphanoudakis. Methodology for the Representation, Indexing and Retrieval of Images by Content. *Image and Vision Computing*, 11(8):504–521, October 1993.

[90] R. W. Picard. Content Access for Image/Video Coding: *The Fourth Criterion.* Technical Report 295, M.I.T. Media Laboratory, Perceptual Computing Section, 1994.

[91] R. W. Picard. Digital Libraries: Meeting Place for High-Level and Low-Level Vision. In *Proceedings of the Asian Conf. on Comp. Vis.*, Singapore, December 1995.

[92] R. W. Picard. Light-years from Lena: Video and Image Libraries of the Future. In *Proceedings of Int. Conference on Image Processing*, Washington DC, October 1995.

[93] R. W. Picard. Toward a Visual Thesaurus. In *MIRO 95*, Workshops in Computing. Springer Verlag, 1995.

[94] R. W. Picard and T. Kabir. Finding Similar Patterns in Large Image Databases. In *IEEE ICASSP, Vol. V*, pages 161–164, Minneapolis, April 1993.

[95] R. W. Picard and F. Liu. A new Wold Ordering for Image Similarity. In *IEEE ICASSP, Vol. V*, pages 129–132, Adelaide, April 1994.

[96] R. W. Picard and T. P. Minka. Vision Texture for Annotation. Technical Report 302, M.I.T. Media Laboratory Perceptual Computing Section, 1994.

[97] A. A. Rodriguez and L. A. Rowe. Multimedia Systems and Applications. *Computer*, pages 20–23, May 1995.

[98] L. A. Rowe, J. Boreczky, and C. Eads. Indexes for User Access to Large Video Databases. In *Proceedings of IS&T/SPIÈS Symposium on Electronic Imaging: Science & Technology*, San Josè, CA, USA, 1994.

[99] S. Santini and R. Jain. Similarity Matching. *submitted to IEEE Transactions on PAMI*, July 1995.

[100] S. Santini and R. Jain. Similarity Queries in Image Databases. In *Proceedings of CVPR '96*, San Francisco, CA, June 1996.

[101] H. S. Sawhney and S. Ayer. Compact Representations of Videos Through Dominant and Multiple Motion Estimation. *EEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8), August 1996.

[102] H. S. Sawhney, S. Ayer, and M. Gorkani. Model-based 2D&3D Dominant Motion Estimation for Mosaicing and Video Representation. In *Proceedings of ICCV 95*, Cambridge, MA, June 1995.

[103] P. Schauble and M. Wechsler. First Experiences with a System for Content Based Retrieval of Information from Speech Recordings. In Mark Maybury, editor, *Proceedings IJCAI '95 Workshop on Intelligent Multimedia Information Retrieval*, Montreal, August 1995.

[104] S. Sclaroff. World Wide Web Image Search Engines. Technical Report 95-016, Boston University, Computer Science Dept., 1995.

[105] 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*, number 2420 in SPIE, San Josè, CA, USA, 1995.

[106] M. Shneier and M. Abdel-Mottaleb. Exploiting the JPEG Compression Scheme for Image Retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8):849–853, August 1996.

[107] J. R. Smith and S. F. Chang. Quad-Tree Segmentation for Texture-Based Image Query. In *Proceedings of the ACM Multimedia 94 Conference*, October 1994.

[108] J. R. Smith and S. F. Chang. Transform Features for Texture Classification and Discrimination in Large Image Databses. In *Proceedings of the IEEE International Conference on Image Processing*, November 1994.

[109] J. R. Smith and S.-F. Chang. Automated Image Retrieval Using Color and Texture. *submitted to the IEEE Pattern Analysis and Machine Intelligence – Special issue on Digital Libraries: Representation and Retrieval*, Columbia University, July 1995.

[110] J. R. Smith and S.-F. Chang. Single Color Extraction and Image Query. In *Proceedings, IEEE International Conference on Image Processing (ICIP-95), Washington, DC* (to appear), October 1995.

[111] J. R. Smith and S.-F. Chang. Tools and Techniques for Color Image Retrieval. In *Storage & Retrieval for Image and Video Databases IV*, Proceedings of IS&T/SPIE Vol. 2670, 1996.

[112] M. A. Smith and T. Kanade. Video Skimming and Characterization through Language and Image Understanding Techniques. Technical Report CMU-CS-95-186, Carnegie Mellon University, 1995.

[113] S. W. Smoliar, H. J. Zhang, S. L. Koh, and G. J. Lu. Interacting with Digital Video. In *Proceedings of IEEE TENCON'94*, pages 952–956, Singapore, 1994.

[114] R. K. Srihari. Automatic Indexing and Content-Based Retrieval of Captioned Images. *Computer*, 28(9):49–56, September 1995.

[115] M. Stricker and M. Swain. The Capacity and the Sensitivity of Color Histogram Indexing. Technical Report 94-05, Communications Technology Lab, ETH-Zentrum, 1994.

[116] V. S. Subrahmanian and S. Jajodia. *Multimedia Database Systems.* Springer, 1996.

[117] K. K. Sung and T. Poggio. Example-based Learning for View-based Human Face Detection. In *Proceedings Image Understanding Workshop*, 1994.

[118] M. J. Swain and D. H. Ballard. Color Indexing. *International Journal of Computer Vision*, 7(1):11–32, 1991.

[119] D. L. Swets and J. J. Weng. Efficient Image Retrieval using a Network with Complex Neurons. In *Proceedings of the Internation Conference on Neural Networks*, Perth, Western Australia, November 1995.

[120] A. Takeshita, T. Inoue, and K. Tanaka. Extracting Text Skim Structures for Multimedia Browsing. In Mark Maybury, editor, *Proceedings IJCAI '95 Workshop on Intelligent Multimedia Information retrieval*, Montreal, August 1995.

[121] L. Teodosio and W. Bender. Salient Stills from Video. In *Proceedings of the ACM Multimedia '93 Conference*, Anheim, CA, August 1993.

[122] Y. Tonomura, A. Akutsu, K. Otsuji, and T. Sadakata. VideoMAP and Video SpaceIcon: Tools for Anatomizing Video Content. In *Proceedings of INTERCHI '93*, pages 131–136. ACM, 24-29 April 1993.

[123] H. Ueda, T. Miyatake, S. Sumino, and A. Nagasaka. Automatic Structure Visualization for Video Editing. In *Proceedings of INTERCHI '93*, pages 137–141. ACM, 24-29 April 1993.

[124] J. Y. A. Wang and E. H. Adelson. Spatio-Temporal Segmentation of Video Data. In *Image and Video Processing II*, volume 2182 of *Proceedings of the SPIE*, San Josè, CA, February 1994.

[125] 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*, volume 2187 of *Proceedings of the SPIE*, San Josè, CA, February 1994.

[126] D. A. White and R. Jain. Similarity Indexing: Algorithms and Performance. In *Proceedings of SPIE Vol. 2670: Storage and Retrieval for Image and Video Databases*, San Josè, CA, February 1996.

[127] D. A. White and R. Jain. Similarity Indexing with the SS-tree. In *Proceedings of the 12th IEEE International Conference on Data Engineering*, New Orleans, Lousiana, February 1996.

[128] W. Wolf. Key Frame Selection by Motion Analysis. In *Proceedings of ICASSP '96*, 1996.

[129] J. K. Wu, A. D. Narasimhalu, B. M. Mehtre, C. P. Lam, and Y. J. Gao. CORE: a COntent-based Retrieval Engine for Multimedia Information Systems. *Multimedia Systems*, 3:25–41, 1995.

[130] B. Yeo and B. Liu. Rapid Scene Analysis on Compressed Videos. *IEEE Transactions on Circuit and Systems for Video Technology*, 5(6):533–544, December 1995.

[131] B.-L. Yeo. *Efficient Processing of Compressed Images and Video.* PhD thesis, Dept. of Electrical Engineering, Princeton University, January 1996.

[132] 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.

[133] 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.

[134] M. M. Yeung and B. Liu. Efficient Matching and Clustering of Video Shots. Technical report, Princeton University, 1995.

[135] 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.

[136] 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.

[137] H.-H. Yu and W. Wolf. Scenic Classification Methods for Image and Video Databases. In C.-C. J. Kuo, editor, *Proceedings of the ACM*

*Multimedia '93 Conference*, number 2606 in SPIE Proceedings, pages 363–371, 1995.

[138] 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, November 1995.

[139] H. J. Zhang, A. Kankanhalli, and S. W. Smoliar. Automatic Partitioning of Full-motion Video. *Multimedia Systems*, 1:10–28, 1993.

[140] H. J. Zhang, C. Y. Low, and S. W. Smoliar. Video Parsing and Browsing Using Compressed Data. *Multimedia Tools and Applications*, 1:91–113, 1995.

[141] H. J. Zhang and S. W. Smoliar. Developing Power Tools for Video Indexing and Retrieval. In *Proceedings SPIE Storage and Retrieval for Image and Video Databases*, pages 140–149, 1994.

[142] 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.

[143] 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, 1995.

[144] H. J. Zhang, S. Y. Tan, S. W. Smoliar, and G. Yihong. Automatic Parsing and Indexing of News Video. *Multimedia Systems*, 2:256–266, 1995.

[145] H. J. Zhang and D. Zhong. A Scheme for Visual Feature based Image Indexing. In *Proceedings SPIE Storage and Retrieval for Image and Video Databases*, 1995.