REIHE INFORMATIK
5/97

VisualGREP:
A Systematic Method to Compare and Retrieve Video Sequences

Rainer Lienhart
Universität Mannheim
Praktische Informatik IV
L15,16
D-68131 Mannheim

VisualGREP: A Systematic Method to Compare and Retrieve Video Sequences

Rainer Lienhart[†‡], Wolfgang Effelsberg[†°] and Ramesh Jain[‡]

[†]University of Mannheim, Praktische Informatik IV, 68131 Mannheim, Germany
[‡]University of California at San Diego, Visual Computing Lab, La Jolla, CA 92093-0407, USA
[°]International Computer Science Institute, 1947 Center Street, Berkeley, CA 94704-1198, USA
{lienhart, effelsberg}@pi4.informatik.uni-mannheim.de, jain@ece.ucsd.edu

ABSTRACT

In this paper, we consider the problem of similarity between video sequences. Three basic questions are raised and (partially) answered. Firstly, at what temporal duration can video sequences be compared? The frame, shot, scene and video levels are identified. Secondly, given some image or video feature, what are the requirements on its distance measure and how can it be "easily" transformed into the visual similarity desired by the inquirer? Thirdly, how can video sequences be compared at different levels? A general approach based on either a set or sequence representation with variable degrees of aggregation is proposed and applied recursively over the different levels of temporal resolution. It allows the inquirer to fully control the importance of temporal ordering and duration. The general approach is illustrated by introducing and discussing some of the many possible image and video features. Promising experimental results are presented.

1 Introduction

The daily growing number of video databases and their sheer volume place content-based search tools in high demand. One proven search technique is query by video sample. The MoCA (Movie Content Analysis) project [7] at the University of Mannheim is currently working on a system called VisualGREP, whose query paradigm follows the well-known UNIX "grep" command for text files. The user specifies a video sample and the type of similarity he/she is interested in, and VisualGREP searches the video database for similar video sequences. Prerequisite to the construction of such a search tool is the systematic analysis both of the various methods to compare video sequences and of the distance measures between them. As for any kind of "grep", the query video sequence is much shorter than the video database which is searched.

Our paper presents a systematic method to compare and retrieve video sequences at the four levels of temporal resolution of videos: frame, shot, scene and video. At each level, features are employed to transform the video sequences into an appropriate representation. A normalized measure of distance between the representations of two video sequence is defined to capture their similarity. To the authors' knowledge, this is the first paper to present not only domain-independent methods to compare frames, shots or short sequences but also techniques to compare temporally large entities such as scenes and full-length feature films for general video. A domain-specific approach has been presented in [19].

The paper is structured as follows. Following a review of related work, Section 3 presents the types of similarities between video sequences in which we are interested. Section 4 discusses the requirements on the features' distance measures and how the distance measures can be "easily" transformed into the visual similarity judgement desired by the inquirer. It also introduces some "real" image and video features in order to make the subsequent discussion more concrete. These features are important components in Section 5, which goes through the four levels of temporal resolution and presents and analyses various techniques of comparison. Section 6 describes some aspects of how to combine the various comparison methods, and Section 7 shows experimental result. Section 8 concludes the paper with an outlook on future research.

2 Related Work

Image Databases
Many content-based image indexing and retrieval systems support querying the database by example. Recently they have extended their range to the video domain. To do so they have basically added cut detection and create for each shot one or several representative frames, either by some kind of image mosaics [4] or by reference frame selection. These images are then indexed as standard still-images. Often motion information is added on a per-reference-frame or shot basis. Well-known examples are QBIC [4], VisualSeek [18] and Virage [5]. However, since these systems originated from still-image systems they do not go beyond the shot level to scenes or full-size videos. Moreover, they map the video into the still-image domain, ignoring the temporal order. Besides these systems, White and Jain have developed a general framework for efficient global and local matching in image databases. They call their framework ImageGREP [20].

Video Abstracting
Yeo and Yeung have proposed a scheme to recover story units from video using time-constrained clustering [21][22]. The basic idea is to assign the same label to similar shots and analyze the patterns of the resulting strings. Three different kinds of temporal events are extracted from those label patterns: dialog, action, and story unit. Each event can be regarded as a scene of the video. Yeo and Yeung use the patterns only for abstracting purposes, not to compare of dialogs, actions, and story units. Such comparisons, however, are covered by our method. Another way to retrieve the scenes is described in [8]. Again, only for video abstracting purposes.

Similarity Measures
Any measure of video similarity should imitate the human visual judgement. A thorough investigation of the psychological findings regarding human judgement of similarity can be found in [17], whose basic statements are the basis for our design of distance measures.

Video Query Systems
One of the few architectures designed exclusively for video retrieval by example is described in [3]. Consequently, their proposed algorithms take the temporal order of the frames into account. In a first step, a signature is derived for each video sequence by using the DC coefficients of window pairs and their motion components. Then, the distance measure between a query video sequence and a database video sequence is defined by the average distances between corresponding frames in the signature representation.

3 Video Similarity

When are two videos similar? There is no straightforward answer. The aspects of video similarity are manifold and its definition at the semantic level vague. For example, let us assume that one video shows the US president at a gala dinner. Would a second video showing the president be similar, or a third video showing a family dinner? We conclude that there is no absolute measure of similarity, instead one has to let the user decide in the query what he/she is looking for. This is somewhat similar to regular expressions the user has to specify in the UNIX grep command.

In this paper, we consider three orthogonal aspects of video sequences:

- the levels of temporal resolution,
- the temporal order, and
- the temporal duration.

Levels of Temporal Resolution
Video sequences can vary considerably in length. They may range from several frames up to tens or hundreds of thousands of frames. Obviously, the type of similarity between two short video sequences (several seconds in length) and the assessment procedure ought to differ from that between two long video sequences (several minutes in length). Thus, video sequences have to be classified with respect to their temporal length before an appropriate comparison scheme can be applied. Following the standard hierachical video model, we classify video sequences into shots, scenes and video. A shot refers to a continuous camera recording. In a query it can also be a subsequence of that. A scene denotes a video sequence that is longer than one shot and shorter than a video, exhibiting some characteristic

shot feature pattern. The type of characteristic shot pattern is determined by the query video. This definition differs slightly from the definition in film art, where a scene is "a segment in a narrative film that takes place in one time and space or that uses crosscutting to show two or more simultaneous actions" [2]. Last but not least, a video denotes a full-length video production. At these four levels video similarity will be investigated using image and video (i.e. motion) features.

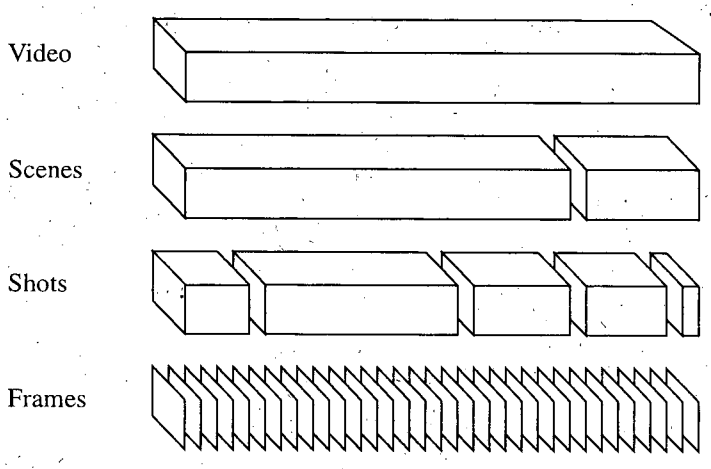

Figure 1: Standard video structuring model

Temporal Order
Video sequences consist of images. Hence, video sequences can be considered as a set of images and one can use the similarity between image features to judge video similarity. The similarity between two video sequences can then be measured by the number and amount of similar image features. We call such a view a *set representation* of a video sequence. Note, this view ignores any temporal ordering of the image features.
Often it is desirable to consider the temporal ordering of video sequences by imposing the ordering constraint on the still-image features. We call such a view a *sequence representation*.

Temporal Duration
The temporal duration in this context denotes to what extent the temporal duration of a feature is important. For instance, having a shot consisting of two parts, a 3-second near still-image of a human who suddenly runs out of the frame within 2 seconds raises the question of how important the temporal duration is. Usually we may represent such a shot by one reference frame on the still-images and five on the running human. Is it then acceptable to use these ordered reference sequences as a shot representation? Or should we also consider the time a reference frame covers? That decision is independent on the choice of representation.

It should be up to the user to set the importance of the temporal duration. In the case of the sequence representation, for instance, one user might demand that the precise temporal development be met; another user might be satisfied if the video sequences have the same relative pace (slow-motion / normal-motion); while a third user might only be interested in the fact that a video sequence is developing, ignoring any pace.

These three different orthogonal aspect will be discussed further in Section 5.

4 Features and their Distance Measures

Whenever comparing two things of the same type, one has to specify the criteria for the comparison. Usually the criteria are denoted as features in the field of image and video content analysis. The value of a feature can be any kind of data such as a scalar value, a vector value, another image or a text string. In this article, features are derived from a set or sequence of individual frames, shots, scenes and videos.

Since the features are employed to judge the visual similarity of video sequences, it is crucial that these distance measures resemble human similarity/distance perception to some extent. For some of the possible features such as color and lightness there exist well-accepted perceptually uniform distance measures. However, for all other features, measures which model human perception to some extent or - less pretentiously - may be useful, have to be defined. Note that each feature requires its own distance measure.

4.1 Some Image and Video Features

In order to make the subsequent discussion more concrete, we now introduce some low-level to high-level image and video features together with reasonable distance measures for them. They represent only a small set of all possible features.

4.1.1 Color Atmosphere

The color atmosphere is an important feature of a frame and a frame sequence. It is often viewed as a compact summary. In practice, it is usually measured by some sort of refined color histogram technique. The basic color histogram H_i^v of a frame sequence FS_i^v is defined as the vector $\langle (h_1), ..., (h_n) \rangle$, where h_j specifies the number of pixels of color j in frame sequence FS_i^v normalized by the total number of pixels. Typically, only a few of the most significant bits of each component of a color representation in some color space are used to calculate the color histogram. Since we are interested in measures which approximate human perception, the CIE L*a*b* color space is used. It was designed for perceptual uniformity [11].

A refined and thus better color histogram technique is the color coherence vector (CCV) [12]. It makes use of the spatial coherence and is thus much more discriminative. It outperforms the basic color histogram in similarity retrieval in large image database. Instead of counting only the number of pixels of a certain color, the CCV additionally distinguishes between coherent and incoherent pixels within each color class j depending on the size of the color region they belong to. If the region (i.e. the connected 8-neighbor component of that color) is larger than threshold t_{ccv}, a pixel is regarded as coherent, otherwise as incoherent. Thus, there are two values associated with each color j:

- α_j, the number of coherent pixels of color j and
- β_j, the number of incoherent pixels of color j.

Then, the color coherence vector CCV_i is defined as the vector $\langle (\alpha_1^i, \beta_1^i), ..., (\alpha_n^i, \beta_n^i) \rangle$ normalized by the number of pixels. Two CCVs CCV_1 and CCV_2 are compared by

$$\sum_{j=1}^{n} \left(\frac{|\alpha_j^1 - \alpha_j^2|}{\alpha_j^1 + \alpha_j^2 + 1} + \frac{|\beta_j^1 - \beta_j^2|}{\beta_j^1 + \beta_j^2 + 1} \right)$$

In experimental results this measure outperforms the Euclidean distance [12]. The distance values range from 0 to about $2n$.

4.1.2 Lightness

The luminance of mise-en-scenes plays an important role in filmcraft, usually varying significantly in different segments of the video: Actions, for instance, can be performed in the dark (e.g. at night, under water, or in the underground) or in the light (e.g. on a California beach).

Such differences can be captured by lightness. It is defined as the perceptual response to luminance and denoted by L^* [11]. Luminance, denoted Y, is defined as the radiant power weighted by a spectral sensitivity function that is characteristic of human vision [11]. CIE defines L^* as:

$$L^* = \begin{cases} 116 \cdot \left(\frac{Y}{Y_n}\right)^{\frac{1}{3}} - 16 & 0.008856 < \frac{Y}{Y_n} \\ 903.3 \cdot \frac{Y}{Y_n} & 0.008856 \geq \frac{Y}{Y_n} \end{cases}$$

This formula states that lightness perception is roughly logarithmic. L^* ranges from 0 to 100. Since L^* is perceptually uniform, two lightness values are compared by means of their difference. The *Lightness of a frame sequence f* is the average of the lightness of all pixels of the frame sequence.

4.1.3 Motion Intensity

Motion intensity is another important feature of a frame sequence. It describes whether or not there is much motion present: either object motion or camera motion. The shortest possible sequence for which motion intensity makes sense is a sequence of two contiguous frames. In this paper motion intensity is measured by means of the edge change ratio (ECR) [23]. In our experience, ECR seems to be a little more reliable than the block motion vectors and faster to calculate.

Let σ_n be the number of edge pixels in frames *n*, and X_n^{in} and X_{n-1}^{out} the number of the entering and exiting edge pixels in frames *n* and *n-1*, respectively. Then the edge change ratio ecr_n between frame n-1 and n is defined as

$$ecr_n = max\left(\frac{X_n^{in}}{\sigma_n}, \frac{X_{n-1}^{out}}{\sigma_{n-1}}\right)$$

and ranges from 0 to 1. In order to make the measure robust against small movements, pixels in one image which have pixels nearby in the other image (e.g. within 6 pixels' distance) do not count as entering or exiting pixels. Notice that unlike [23], the edge change ratio here is not used to detect scene breaks, and no global motion compensation is performed before the ECR's calculation.

The *ECR of a frame sequence f* is defined as the average of the ECR values over all frames in the sequence. The advantage of the ECR as a characteristic parameter is that it registers structural changes in the sequence such as entering, exiting and moving objects, as well as fast camera operations. However, it is somewhat independent of variations in color and intensity since it relies on sharp edges only.

4.1.4 Frontal Faces

In most video genres the people are an essential or even the most important part of a video. Thus, a face detector and a method of identifying faces of the same person within the same shot/scene/video and across videos is highly desirable. Such a feature is a rich resource of semantics as we will see in Section 5.

One of the most reliable *face detectors* in digital image research was developed by Rowley, Baluja, and Kanade [13]. Their system recognizes about 90% of all upright and frontal faces while only sporadically mistakenly identifying non-face areas as faces. Therefore, we have recreated their proposed neural network-based frontal face classification system for arbitrary images as a basis for our frontal face detector in video sequences. To widen the range of detectable faces, our detector also searches for slightly tilted/rotated faces (∓ 30 degree). This is necessary because the faces of the people in motion pictures are always moving, as opposed to the faces in typical still images such as portrait and sports team photographs, which are usually depicted upright. However, this more general face search increases by a factor of three the number of patterns which have to be tested in each image. To speed up processing, the number of candidate frontal face locations is drastically reduced by an extremely fast pre-filtering step: Only locations whose pixel colors approximate human skin colors [6] and which show some structure (such as nose, mouth and eyes) in their local neighborhood are passed to the face detector. This pre-filter reduces the number of candidate face locations by 80%. Moreover, only every third frame of the video sequence is investigated.

Each face detected is described by the vector $(t_j^i, x_{pos}, y_{pos}, s, \gamma)$. It specifies the frame t_j^i, in which a face of size s (in pixels) was detected, as well as the x- and y-coordinates (x_{pos}, y_{pos}) of its center and its angle of incline γ.

Two faces F_1 and F_2 are compared according to one of the following three features: size, position and visual similarity. The size difference measure is given as the ratio between the larger and the smaller face, i.e. as $max\{s_1/s_2, s_2/s_1\}$, the spatial distance is measured by means of the Euclidean distance between their centers, and the visual distance is measured by means of the Euclidean distance of their eigenface representation, i.e. of their projection into the face space [9][16]. Our face space was determined from a training set of 1247 frontal images of faces, the same set used to train the neural frontal face detector.

4.1.5 Type of Framing

There exist two different kinds of framing of frame sequences, especially shots: static framing and mobile framing.

STATIC FRAMING: Static framing of the objects in a video shot gives us a sense of the camera distance, i.e. of being far away from or close to the mise-en-scene. Based on the position and size of their frontal views of faces, shots can be classified as [2]

- a long shot: the whole human figure is visible
- an American shot: the human figure is framed from the knees up.
- a medium shot: human bodies are framed from the waist up, or as
- a close-up: just the head is visible.

This classification is commonly used in film art. For shots with frontal face appearances the camera distance can be estimated. The ranges of the face positions and sizes for the different classes may be determined empirically based on experimental inquires. However, for the query by video paradigm the closeness of two frame sequences with respect to their types of static framing is evaluated based on the similarity of the average size of the largest face in each frame of a frame sequence.

MOBILE FRAMING: Mobile framing usually denotes the dominant camera motion such as pan/dolly, tilt and zoom. Extraction of that motion is a complex task and algorithms have been proposed in [14], [1] and [24]. Once extracted, camera motions can be compared logically based on their classes and within the same classes, based on their distance measure. However, their integration into the VisualGREP is an open task for the future.

4.2 Normalization of Distance Measures

In [17] Santini and Jain presented a thorough investigation of the psychological findings regarding human similarity judgement. Although there still exists no generally accepted model of similarity perception and their proposed model requires the estimation of too many parameters in order to be practically usable, some basic rules nevertheless emerge. From a cognitive point of view, it is essential

(1) to find the range of distance values to which humans are sensitive (with respect to a query) and

(2) to achieve saturation of the distance values outside these regions.

Saturation is a vital point since it prevents a large dissimilarity with respect to one individual feature value from dominating the whole distance measure. It also suppresses small distances that are often the result of noise. Note that the degree of human sensitivity to dissimilarity is very adaptive: It differs greatly between a set of similar images on the one hand, and a set of in-homogeneous and diverse images on the other hand.

For a moment let us assume that the range $[a_1,a_2]$ represents the range of distance values of a feature to which an inquirer is sensitive with respect to his/her query. Then, his/her similarity judgments can be modeled by a simple fuzzy membership function as depicted in Figure 2. It normalizes the distance measures of the chosen feature. Note that such a function can always be approximated by an infinitely differentiable function, the so-called logistic function [17].

The next question is about how or by which means shall we let the inquirer specify his/her desired similarity judgements? Essentially there exist two kinds of features: The first kind comprises all features whose desired similarity

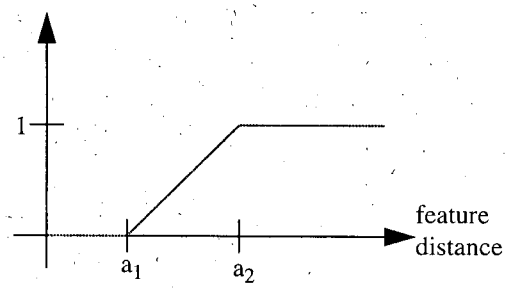

Figure 2: Prototype of the normalized feature distance function

judgements usually should not vary from query to query. Consequently, default values can be determined that are suitable for most queries. Examples are face similarity and CCV. Their default parameters are determined by presenting frames or frame sequences to users and letting them assess their similarity with respect to the feature. In our experiments users answered the questions by means of the position of a slider ranging from complete agreement (0 = yes, differences are not significant) to complete disagreement (1 = no, differences are too great; the objects have nothing in common). When users formulate a new query they can use these default values or modify them cautiously. A useful and intuitive tool would show some examples from the video database which lie at the distance values specified for a_1 and a_2.

In contrast to that, the desired similarity judgement of the second kind of features differs considerably from query to query. Default values can be specified, however, they are generally re-set, since their values are context dependent. Examples are the lightness and lightness difference, as well as face location and size. For each such feature a visually intuitive way of specifying one's similarity judgments should be found. For instance, one's spatial locations similarity judgement can easily be specified by showing two frames of the query frame sequence. The user then specifies his/her similarity preference by drawing the lower and upper distance boundaries upon these frames. Any distance less than or equal to the lower boundary (so-called "don't care" distance) will be regarded as the same position, while any distance equal to or larger than the upper boundary will be regarded as completely different. (see Figure 3).

Figure 3: An example of a visually intuitive way to specify one's similarity judgments with respect to tolerable spatial deviations in the location of objects.

5 Comparison of Motion Picture Sequences

In this section a new and general video comparison approach is proposed and applied recursively over the different levels of temporal resolution. It is based on either a set or sequence representation with a variable degree of aggrega-

tion. The comparison strategy at each level of temporal resolution works as follows: the features of levels of higher temporal resolution are employed to represent the video sequences appropriately; two video sequences are then compared by means of a normalized distance measure which in turn is computed from the distances of the employed features.

Without loss of generality, we will use only one feature in the following. Section 6 describes how the representations derived from different features can be combined.

5.1 At Frame Level

Frames may be compared by any image feature whose defined distance measure fulfills the requirements stated in Section 4. Such a distance measure should be constructible for most image feature. Applicable in addition are video features such as the optical flow which can be derived from a sequence of frames and assigned to an individual frame.

5.2 At Shot Level

Shots are represented by the features derived from their respective frames. A feature can be derived from one or several frames, and is assigned to that or those frames. If a feature is calculated for each frame in the shot, such a feature description is called *non-aggregated*. If a feature is derived from á set/sequence of frames, such a feature description is called *partially aggregated*. If at the very most, only one feature is derived for the whole shot, it is called *completely aggregated*.

Along with the degree of aggregation goes the importance of duration of the individual feature values. Non-aggregated representations capture the precise temporal duration of a feature. Thus, two non-aggregated representations are only judged as similar (besides other requirements) if the duration of the individual feature values are almost identical. For completely aggregated representations the duration is irrelevant. Again, partially aggregated representations allow to control the degree of importance of duration of individual feature values.

In addition, as mentioned above, we distinguish between a sequence and a set representation. In a sequence representation, the features are over time. If the feature values are computed for each frame, ordering is scalar. The ordering becomes increasingly ordinal with increasing aggregation since we allow the aggregation to be adaptive throughout the shot. Contrary thereto, set representation ignores the temporal order. It only considers the amount and degree of similarity between the feature values. At the highest level of aggregation, both sequence and set representation degrade to a completely aggregated representation: the shot is described by only one feature value.

The interdependencies among these three representations are summarized in Figure 4. Note that transitions are continuous. The meaning of the threshold value shown in that figure will be explained shortly.

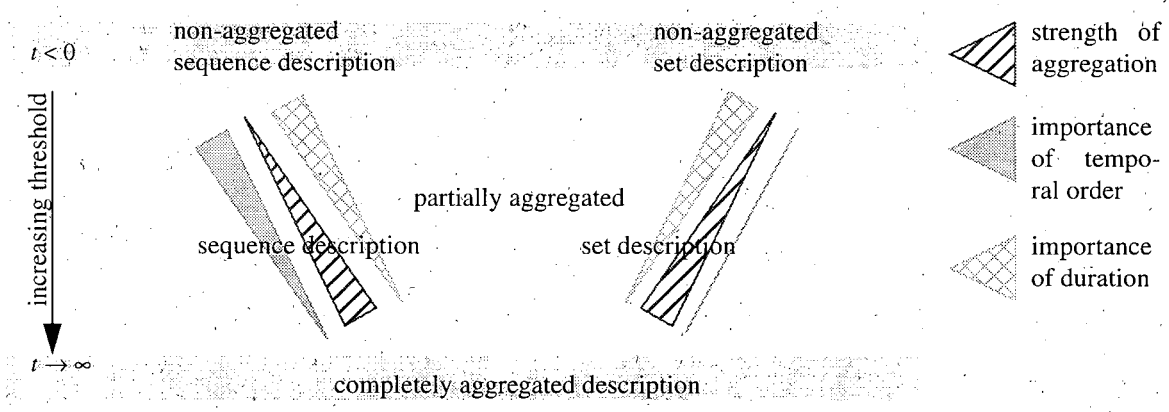

Figure 4: Interdependencies among the possible frame sequence representations. The width of the arrows indicates the strength/importance of its describing attribute.

5.2.1 Sequence Representation

Non-aggregated Sequence

The aim of this representation is to preserve the precise temporal development of a shot. Therefore, a shot is represented by the sequence of the feature values calculated per frame. Then, each value is regarded as a *character*, the domain of possible values as an *alphabet*, and the sequence of characters as a *string*. Two sequences can now be compared by applying well-known string searching algorithms such as the approximate substring matching and longest common subsequence search. They are defined as follows:

APPROXIMATE SUBSTRING MATCHING: Given a query string A of length P and a longer subject string B of length N, the approximate substring matching *asm(A,B)* finds the substring of B that aligns with A with minimal substitutions, deletions and insertions of characters [15]. The minimal number of substitutions, deletions and insertions transforming A into B is called the *minimal distance* D between A and B. A cost of 1 is assigned to deletions and insertions, while the cost of transforming a character from a to b is weighted by the normalized distance multiplied by 2. This metric is called the *edit distance*.

The usage of the similarity measure between two feature values as the variable cost function for substitutions allows the ranking of two sequences with respect to similarity. Two sequences A and B are regarded as identical if the minimal distance D between query string A and subject string B does not exceed the threshold t_{ASM1} and as completely different if it exceeds the threshold t_{ASM2}. These thresholds are used to construct the normalized feature distance function and are determined empirically. In our experiments they are set to 5% and 90%, respectively, of the length of the query string A.

LONGEST COMMON SUBSEQUENCE: Given two strings A and B of length M and N, respectively, their longest common subsequence *lcs(A,B)* of A and B is the longest subsequence which is common to both strings [15]. This algorithm is used to determine the parts two strings have in common, as well as to evaluate their similarity by the length of the *lcs(A,B)*, denoted $|lcs(A,B)|$. The maximum of $|lcs(A,B)|$ is the minimum of $|A|$ and $|B|$.Thus the distance measure is defined as

$$1 - \frac{|lcs(A,B)|}{min(|A|,|B|)}$$

It is 0 if the shorter sequence is a subsequence of the longer one and 1 if they have nothing in common. Since identical subsequences coming from different video sources are likely to differ in the precise feature values and, in addition, we also want to retrieve similar scenes, subsequences are allowed to differ from each other up to a threshold t_{LCS} which was empirically set to 5% of $min(|A|,|B|)$. Hence, small differences are tolerated within the *lcs*. Again, in our experiments the parameters of the normalized distance function are set to 5% and 90%.

Both normalized sequence measures, one based on the *asm* and the other on the *lcs*, have their own strengths and weaknesses. The *asm* measure judges similarity based on the whole shorter string, i.e. nothing is left out of the comparison, while the *lcs* measure uses only the longest common subsequence and sets its length into relation to the length of the shorter string. Therefore, the *asm* is usually more appropriate in query-by-video sample applications where the query video contains only and all aspects important to the user. Moreover, the computation of the *asm* is less expensive than that of the *lcs*. Both measures also determine the position of the subsequences which led to that distance judgement, an aspect important to the presentation of the result list in query-by-video applications.

Examples of generic questions which can be answered by the *asm* and the *lcs* with respect to some feature in the case of non-aggregated sequences are:

- Are two shots identical?
- Is one shot a subsequence of the other?
- Do two shots have a subsequence in common?

This information has many useful applications. For instance,

- the CCV feature can be used to automatically set up hyperlinks between identical shots such as

between original news footage from news agencies broadcast at different times on different channels during newscasts,

- the lightness or lightness difference feature used together with a fade sequence as the query sequence can be used to retrieve other fade sequences with similar absolute or relative temporal changes in lightness, and
- the similarity between faces and their positions can be used to determine all shots in a video database showing the same person(s) in a spatial and temporal setup similar to those in the query shot.

Non-aggregated sequences do not tolerate major deviations in the rate at which a feature develops. Therefore, in most cases partially aggregated frame sequences are more appropriate.

Partially Aggregated Frame Sequence

This representation aims to preserve the rough temporal development of a shot by means of a few representative frames, so-called *r-frames*. The level of "roughness" can be controlled. The less important the precise temporal succession of the frames is, the fewer r-frames are needed. In our work, temporal precision is controlled by the maximum allowed feature distance between two contiguous r-frames, specified by $threshold_{feature}$. Simultaneously the visual precision of the representation of the sequence decreases with the increase of the threshold since each r-frame covers a larger area of tolerated "visual" differences. The use of a negative threshold results in a non-aggregated sequence representation. Note also that a threshold of zero aggregates still-image sequences.

Given a shot S consisting of n frames $f_1, \ldots, f_n$ and the maximum allowed feature distance between two r-frames by $threshold_{feature}$, r-frames are generated as follows:

1. $i := 1$; $rNo := 1$; $rFrame_{rNo} := f_1$
2. while ((i <= n) && (distance($rFrame_{rNo}, f_i$) < (0.5 * $threshold_{feature}$))
   - **2.1.** i++
3. $rFrame_{rNo} := f_{i-1}$
4. while (i <= n)
   - **4.1.** if (distance($rFrame_{rNo}, f_i$) > (0.5 *$threshold_{feature}$))
     - rNo++
     - $rFrame_{rNo} := f_i$
   - **4.2.** i++

The algorithm is visualized in Figure 5.

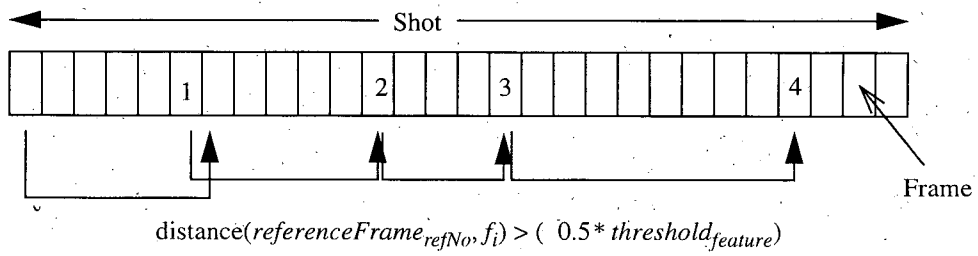

Figure 5: The r-frame generation process

Note that for each feature or feature combination we demand an individual reference frame selection process. Very often r-frames are selected as a visual abstract for users and then afterwards mis-used for feature extraction. From the features' point of view this two-step approach with different goals in different steps is likely to cause many errors since the r-frames selection process is neither tailored to nor suited for most features.

Employing approximate string matching the following questions can be answered

- Is one shot a slow motion of the other? (using CCV)
- If the query sequence is a fade-in sequence, find all fade-in sequences in the database, independent of

their precise temporal development (using lightness).

- Find all shots where persons moves similar to the one in the current shot (using the position of frontal faces).

5.2.2 Set Representation

The aim of this model is to preserve the *static* content of a shot by means of a few representative frames. Temporal development and temporal order are of no interest here. Obviously, if we are tolerant in classifying similar frames there is no need to use all frames. On the other hand, one r-frame per shot will not suffice. By employing the r-frame generation process introduced in Section 5.2.1 and depicted in Figure 5, the distance between some reference frame of S_1 and S_2 will be at most $2*threshold_{feature}$ if the distance between some frame in shot S_1 and some frame in S_2 is less than or equal to $threshold_{feature}$. As a result, we lose some precision by concentrating only on the reference frames, while reducing significantly the computational cost of comparison and storage.

Given two shots $S_1 = \{f_1^1, \ldots, f_n^1\}$ and $S_2 = \{f_1^2, \ldots, f_m^2\}$ and their respective r-frame sets $R_1 = \left\{f_{r_1^1}^1, \ldots, f_{r_l^1}^1\right\}$ and $R_2 = \left\{f_{r_1^2}^2, \ldots, f_{r_k^2}^2\right\}$, they are compared based on the following sets

$$R_1 \cap R_2 = \left\{ f_i^1 \middle| i \in \{r_1^1, \ldots, r_n^1\}, j \in \{r_1^2, \ldots, r_k^2\}, distance(f_i, f_j) \leq 2 \cdot threshold_{feature} \right\}$$

$$R_2 \cap R_1 = \left\{ f_j^2 \middle| i \in \{r_1^1, \ldots, r_n^1\}, j \in \{r_1^2, \ldots, r_k^2\}, distance(f_i, f_j) \leq 2 \cdot threshold_{feature} \right\}$$

Note that $R_1 \cap R_2$ does not specify the usual set intersection and is not commutative.

To derive a normalized distance measure four cases have to be investigated:

1. For every reference frame of shot *S1* we can find at least one similar reference frame of shot *S2* and vice versa, i.e. $(R_1 \cap R_2 = R_1) \wedge (R_2 \cap R_1 = R_2)$. Thus, the two shots are identical with respect to the static feature.
2. For every reference frame of shot *S1* we can find at least one similar reference frame of shot S2, but for some reference frames of *S2* we cannot find a similar one in *S1*, i.e $(R_1 \cap R_2 = R_1) \wedge (R_2 \cap R_1 \subset R_2)$. Thus, *S1* is a subset of S2 with respect to the static feature.
3. For every reference frame of shot *S2* we can find at least one similar reference frame of shot *S1*, but for some reference frames of *S1* we cannot find a similar one in *S2*, i.e. $(R_1 \cap R_2 \subset R_1) \wedge (R_2 \cap R_1 = R_2)$. Thus, *S2* is a subset of *S1* with respect to the static feature.
4. There exists at least one reference frame for *S1* and *S2* that is not similar to any reference frame of the other shot, i.e. $(R_1 \cap R_2 \subset R_1) \wedge (R_2 \cap R_1 \subset R_2)$.

Based on these four cases two normalized distance measures are defined: An asymmetric distance by

$$dist_{asym}(R_1, R_2) = 1 - \frac{|R_1 \cap R_2|}{|R_1|}$$

and a symmetric distance by

$$dist_{sym}(R_1, R_2) = 1 - \frac{|R_1 \cap R_2| + |R_2 \cap R_1|}{|R_1| + |R_2|}$$

They are 0 if R_1 and R_2 are identical and 1 if they have nothing in common. Note that the total temporal duration of the feature values influences the result to some extent. If the temporal duration should be ignored the normalized distances may be defined as the minimum of all pair-wise normalized distances between all elements of R_1 and R_2:

$$dist(R_1, R_2) = min\{d_{feature}(r_1, r_2) | r_1 \in R_1, r_2 \in R_2\}$$

5.2.3 Completely Aggregated Set or Sequence

This representation describes a shot as one completely aggregated unit. As mentioned above, there is no difference between the completely aggregated sequence and set representation. Characteristic features are only computed for the complete shot. They may result from the sequence or set representation by setting the feature threshold to infinite. Another possibility might be to use the shot labelling approach proposed in [21]. All information is contained in a sin-

gle feature value. The shots are compared directly by the feature's normalized distance function.

By means of the features described in Section 4.1 the following shot information can be extracted and compared:

- type of static framing (long shot, American shot, medium shot, close-up),
- type of mobile framing (i.e. dominant camera operation),
- amount of action (motion intensity),
- strength of average luminance (lightness),
- shot length, and
- number and list of faces.

5.3 At Scene Level

A scene can be represented as a sequence or set of features at some aggregation level, too. The basic units (or characters in string matching terminology), however, are the shots. They in turn are compared based on the concepts developed in Section 5.2, resulting in a recursive computation scheme.

Examples of queries that can easily be formulated are: Find all frame sequences which are similar to

- a certain spatial layout of frontal faces over the course of time (e.g. a dialog with speaker A to the left of the image center in shot 1 and 3 and speaker B to the right of the image center in shot 2 and 4),
- a given dialog with specific people,
- a typical shot length pattern (e.g. a scene with shots of decreasing length from Hitchcock's "The Birds"), or
- a typical action pattern (e.g. a scene of alternating calm and action-loaded shots).

Shot length and action patterns are often summarized under the term film rhythm.

5.4 At Video Level

At this level frame sequences consisting of several scenes up to full video productions such as feature films, documentaries, sitcoms, etc. are compared. Questions of the following type can be answered:

- If both input videos are different versions of the same feature film, what are the differences? This is a reasonable question since for many feature films one can find short and long versions as well as special versions edited for video cassettes, airlines and countries.
- If two videos have several shots or scenes in common, do these appear in the same temporal sequence or is the temporal structure completely different, perhaps indicating a re-purposing of existing material?
- With regard to the characteristic shot patterns found at the scene level, do two videos share a similar temporal structure?

We use two normalized measures: the *correspondence measure* and the *re-sequencing measure*. Consider two videos $video_1$ and $video_2$ given by their list of entities $E^1 = (E^1_1,\ldots, E^1_{N_1})$ and $E^2 = (E^2_1,\ldots, E^2_{N_2})$, respectively. The entities can either be the shots or the scenes composing the video. In principle, the entities could also represent the individual frames, though at that level their consideration does not seem to make much sense and is therefore omitted here.

In a first step, the entities are compared against each other in order to construct a graph where nodes represent entities and edges entities similar with respect to some feature. Two entities are considered to be similar if their feature distance is below a specified threshold (see sample graph in Figure 6). Then, the following useful measures can be computed:

CORRESPONDENCE MEASURE: The correspondence measure specifies the percentage of the entities of $video_1$ which are similar to entities in $video_2$. It is formally defined as

$$Correspondence(video1, video2) = \frac{|E^1 \cap E^2|}{|E^1|}$$

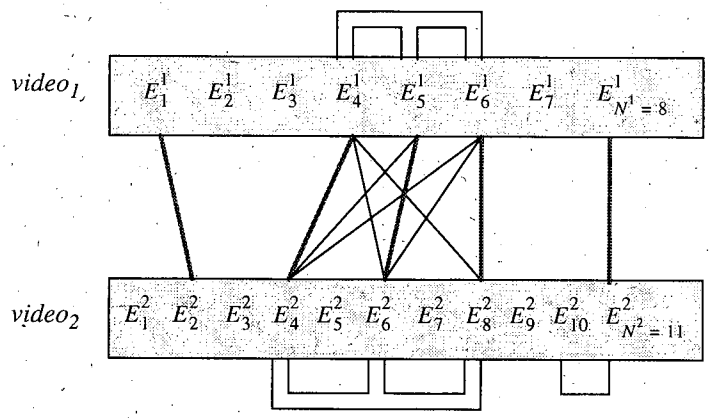

Figure 6: Similarity graph constructed to calculate the correspondence measure and the re-sequencing measure

with $|E^1 \cap E^2|$ denoting the cardinality of the set of entities of E^1 for which at least one similar entity could be found in E^2. Note again that the correspondence measure is not symmetrical, i.e. exchanging the roles of $video_1$ and $video_2$ usually affects its value. For example, if *Correspondence*($video_1$,$video_2$)=0.9 and *Correspondence*($video_2$,$video_1$)=0.5, the conclusion can be drawn that $video_1$ is a shortened version of $video_2$.

RE-SEQUENCING. The *re-sequencing measure* analyzes whether the entities two videos have in common appear in the same sequence or in a reordered sequence. Such a measure can be used, for instance, to judge whether some video source material has been compiled in a content-, context- and structure-preserving manner. A low re-sequencing value indicates that the material is mainly used in its original structure, thus preserving the content and context; a high re-sequencing value signals that the context of the shot has been changed by editing. Most probably this will result in a new content; thus, a high re-sequencing value can indicate a re-purposing of the source material. The re-sequencing measure is calculated as follows:

1. *countReOrderings* := 0 // count the number of re-orderings found
2. *countOrderAgreements* := 0 // count the number of order agreements
3. $i_1 := 1$; $i_2 = 1$ // indices into $video_1$ and $video_2$; pointing to the first entity
4. while ($i_2 <= N_2$)
   - **4.1.** i_2 := find index of next entity of $video_2$ linked to $video_1$ starting from i_2
   - **4.2.** i_{temp} := find index of earliest linked entity in $video_1$ to i_2 starting from i_1
   - **4.3.** if ($i_{temp} == \emptyset$)
     countReOrderings++;
     i_1 := find index of earliest connected entity in $video_1$ to i_2;
     else
     countOrderAgreements++;
   - **4.4.** $i_1 := i_{temp} + 1$
5. Resequencing($video_1$, $video_2$) = *countReOrderings* / (*countReOrderings* + *countOrderAgreements*)

The algorithm determines the number of minimum relative re-orderings necessary to transfer the sequence of cross-video-linked entities of $video_1$ into a sequence with the same ordering as the sequence of cross-video-linked entities of $video_2$. In the worst case, the ordering is reversed, resulting in N-1 re-orderings. The measure ranges from 0 (same ordering) to 1 (reversed ordering). An example is given in Figure 6: The thick lines show the edges selected by the algorithm. No re-ordering was necessary.

SET AND SEQUENCE REPRESENTATION: Obviously, the set and sequence algorithms are also applicable. In this case appropriate shot and scene representation are considered as the basic unit or characters and the cost of transforming these features is given by the normalized distance between them. For instance, given a trailer of a movie, the original feature film can be found by using the shot set representation and comparing shots based on a non-aggregated set representation of CCVs.

6 Combining Features into a Query

Each feature captures only a specific aspect of a video frame or video sequence, and is usually not sufficient to describe the similarity judgement desired by an inquirer. Therefore, several features are usually combined to a new feature in a query to capture the desired similarity judgement more accurately.

A new feature and its distance measure is defined either by a logical expression or by a weighted sum of exiting features and their normalized distance measures. The weighted sum allows to formulate queries such as: "Find all shots which are similar to the query video with respect to the ECR and CCV. The importance of ECR is 30% and of CCV, 70%. This method of combining feature is often used in image retrieval database systems (e.g. [4] and [5]). Logical expressions of features allow similar queries. Since the membership functions are inverted here, unlike usual membership functions (a value of 0 stands for fully belonging to the set instead of a value of 1), a logical AND is defined as the maximum over all distance values of the distances of the various features and a logical OR as its minimum. A logical NOT remains one minus the distance value.

Currently, the user has to specify the method of combination. In future, we will provide a set of pre-defined combinations for specific semantic domains.

7 Experiments

7.1 Methodology

The performance of any indexing and retrieval system is usually measured by its recall and precision values. In our case, *recall* specifies the ratio of the number of relevant video sequences found to the total number of relevant video sequences in the database. *Precision* specifies the ratio of the number of relevant video sequences to the total number of returned video sequences. Since in our experiments the search result list always consists of the ten most similar video sequences, the definition of precision here is changed in the case where all relevant video sequences in the database are retrieved. In that case precision is defined as the ratio of the number of all relevant video sequences to the rank of the least relevant video sequence in the result list. The ground truth, i.e. the decision whether a video sequence is relevant or not, has to be determined by humans. Naturally the measure for relevance is the human similarity judgement.

Analogous to the evaluation of performance of image or text databases tens of thousands of shots and scenes are needed to evaluate the performance of any video comparison algorithms reliably. Unfortunately, building up such a large video database and determining the ground truth presently exceeds the possibilities of our multimedia lab. Thus we had to restrict our experiments to a much smaller database.

7.2 Setup

The proposed recursive comparison scheme as well as all feature computations have been implemented in C++ using its template mechanism. Experiments were performed with the following setup:

DATABASE. The video database consisted of 4 hours of video from various sources. They were digitized from German TV as M-JPEG of size 360x270. In detail, the video database consisted of the feature films "Groundhog Day" and "True Lies", the newscast "Tagesschau", the series "Baywatch" as well as several TV commercials and live concert recordings. For each of them we calculated the features listed in Section 4.1 and stored them in a large database file. One of these videos, namely "Groundhog Day", is very peculiar and very well suited for evaluating our algorithms: It consists of many very similar but not identical scenes repeating throughout the whole movie.

QUERIES. In preparation for our experiments we watched all the videos several times and made a note whenever we thought we saw similar shots or scenes with respect to some feature. Based on these video sequences we constructed five queries at the shot level and one query at the scene level. They are listed in Table 1. All query sequences were taken from "Groundhog Day".

Query #	Content	Level	Frequency of Occurrence	Feature
Query 1	a putative old acquaintance meets the main actor on the street	shot	5	ccv
Query 2	main actor behind a shower curtain	shot	2	ccv
Query 3	building on the street with a red and white striped awning in the front	shot	2	ccv
Query 4	main actor looks down the street out of the window	shot	2	cvv
Query 5	main actor looks down the street out of the window	shot	2	ccv + motion
Query 6	Dialog between the main actors	scene	10	face position

Table 1: List of queries

7.3 Results

The experimental results with non-aggregated video sequence representations (i.e. with a feature threshold less than zero) are summarized in Table 2 and those with a feature threshold of 0.5% of the maximal possible feature distance in Table 3. They show two properties: Firstly, the sequence representation performs better on sequences with motion since it takes the temporal development into account. For calm scenes, however, the set and sequence representations perform similar on our queries. Secondly, there seem to be no difference in retrieval performance for non-aggregated and slightly aggregated sequences.

Query #	Set Representation		Sequence Representation	
	Recall	Precision	Recall	Precision
Query 1	0.8	0.44	0.8	0.4
Query 2	0.5	0.1	0.5	0.1
Query 3	0.5	0.1	0.5	0.1
Query 4	1	0.5	1	0.4
Query 5	1	0.66	1	1
Query 6	0.9	0.9	0.9	0.9

Table 2: Experimental results with $threshold_{feature} < 0$, $a_1 = 0$, $a_2 = 10\%$

Query #	Set Representation		Sequence Representation	
	Recall	Precision	Recall	Precision
Query 1	0.8	0.44	0.8	0.4
Query 2	0.5	0.1	0.5	0.1
Query 3	0.5	0.1	0.5	0.1
Query 4	1	0.5	1	0.4
Query 5	1	0.66	1	1
Query 6	0.9	0.9	0.9	0.9

Table 3: Experimental results with $threshold_{feature} = 0.5\%$ of maximal possible feature distance, $a_1 = 0$, $a_2 = 10\%$

During our experiments we observed that the results of the video comparison algorithms improved with the size of the database. This suggests that the proposed algorithms are appropriate for large video archives of thousand of hours.

8 Conclusion

We have proposed a general method for similarity matching of video sequences of different lengths and at different levels of temporal resolution. Four levels are considered: frame, shot, scene and video. At each level the features of the level of higher temporal resolution are employed, leading to a recursively defined video sequence similarity measure. The temporal ordering and duration can be fully controlled by the user via the type of representation (sequence or set) and the level of aggregation (non-aggregated, partially aggregated or completely aggregated). At the video level we have also introduced two new specialized comparison metrics: the correspondence and resequencing measure. Moreover, we showed how a feature's distance measure can be easily adjusted to the actual desired similarity judgements of a user.

The experimental results on our video database of four hours of video are very promising. The experiments suggest that the proposed algorithms scale with the size of a video database with respect to retrieval quality. In the next few years we plan to build up a large database of thousands of hours of video with some broadcasting stations. This will enable us to evaluate the proposed and other video comparison schemes more thoroughly.

Generally, videos are audio-visual streams. In this paper we have concentrated on the visual part only. Our approach is general enough to be used for an AudioGREP or an AudioVisualGREP. Assuming that audio features such as amplitude, frequency, pitch, onset and offset are available together with their distance measures the AudioGREP would work just like the VideoGREP. For instance, given an explosion as the query audio, other explosions can be retrieved from an audio database [10]. Therefore, we plan to add audio features to our VisualGREP resulting in an AudioVisualGREP.

References

[1] A. Akutsu and Y. Tonomura. Video Tomography: An Efficient Method for Camerawork Extraction and Motion Analysis. Proc. *ACM Multimedia 94*, San Francisco, CA, pp. 349-356, Oct. 1994.

[2] David Bordwell, Kristin Thompson. Film Art: An Introduction. *McGraw-Hill*, Inc., 4th ed., 1993.

[3] Nevenka Dimitrova and Mohamed Abdel-Mottaleb. Content-based Video Retrieval by Example Video Clip. Proc. SPIE Vol. 3022, *Storage and Retrieval for Image and Video Databases V*, Ishwar K. Sethi; Ramesh C. Jain; Eds., p. 59-70, Jan. 1997.

[4] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Qian Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, P. Yanker, P. Query by Image and Video Content: the QBIC System. *Computer*, vol.28, (no.9), p.23-32, Sept. 1995.

[5] Arun Hampapur, Amarnath Gupta, Bradley Horowitz, Chiao-Fe Shu, Charles Fuller, Jeffrey R. Bach, Monika Gorkani, Ramesh C. Jain. Virage Video Engine. Proc. SPIE Vol. 3022, *Storage and Retrieval for Image and Video Databases V*, Ishwar K. Sethi; Ramesh C. Jain; Eds., p. 188-198, Jan. 1997.

[6] H. Martin Hunke. Locating and Tracking of Human Faces with Neural Networks. Master's thesis. University of Karlsruhe. 1994. http://ernie.sfsu.edu/~hunke/

[7] Rainer Lienhart, Silvia Pfeiffer, and Wolfgang Effelsberg. The MoCA Workbench: Support for Creativity in Movie Content Analysis. *Proc. of the IEEE Conference on Multimedia Computing & Systems*, Hiroshima, Japan, pp. 314-321, June 1996.

[8] Rainer Lienhart, Silvia Pfeiffer, and Wolfgang Effelsberg. Video Abstracting. *Communications of the ACM*, to appear.

[9] A. Pentland, R.W. Picard, and S. Sclaroff. Photobook: Content-Based Manipulation on Image Databases. *International Journal of Computer Vision*, Vol. 18, No. 3, pp. 323-254, 1996.

[10] S. Pfeiffer, S. Fischer, and W. Effelsberg. Automatic audio content analysis. *Proc. ACM Multimedia 96*, Boston, MA, pp. 21-30, Nov. 1996.

[11] Charles A. Poynton. A Technical Introduction to Digital Video. *John Wiley & Sons*, Inc. 1996.

[12] Greg Pass, Ramin Zabih, Justin Miller. Comparing Images Using Color Coherence Vectors. Proc. *ACM Multimedia 96*, Boston, MA, pp. 65-73, Nov. 1996.

[13] Henry A. Rowley, Shumeet Baluja, and Takeo Kanade. Human Face Recognition in Visual Scenes. Technical Report CMU-CS-95-158R, School of Computer Science, Carnegie Mellon University, November 1995.

[14] Harpreet S. Sawhney and Serge Ayer. Compact Representations of Videos Through Dominant and Multiple Motion Estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 18, No. 18, pp. 814-830, Aug. 1996.

[15] Graham A. Stephen. String Searching Algorithms.World Scientific Publishing Co. Pte. Ltd., 1994.

[16] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, Vol. 3, No. 1, pp. 71-86, 1991.

[17] S. Santini, R. Jain. Similarity Matching. submitted to: IEEE Transactions on Pattern Analysis and Machine Intelligence, URL: "file://vision.ucsd.edu/pub/simone/vision/SimPaper.ps.Z".

[18] J. R. Smith and S.-F. Chang, "VisualSEEk: A Fully Automated Content-Based Image Query System," *ACM Multimedia Conference*, Boston, MA, Nov.1996.

[19] Deborah Swanberg, Chiao-Fe Shu, Ramesh C. Jain. Knowledge-guided parsing in video databases. In *SPIE Vol. 1908, Storage and Retrieval for Image and Video Databases*, Wayne Niblack; Ed., p. 13-24, Apr. 1993.

[20] David A. White and Ramesh Jain. ImageGREP: Fast Visual Pattern Matching in Image Databases. In *SPIE Vol. 3022, Storage and Retrieval for Image and Video Databases V*, Ishwar K. Sethi; Ramesh C. Jain; Eds., p. 96-107, Jan. 1997.

[21] Minerva Yeung, Boon-Lock Yeo, and Bede Liu. Extracting Story Units from Long Programs for Video Browsing and Navigation. *Proc. of the IEEE Conference on Multimedia Computing & Systems*, Hiroshima, Japan, pp. 296-305, June 1996.

[22] Minerva Yeung, Boon-Lock Yeo. Video Content Characterization and Compaction for Digital Library Applications. In *SPIE 3022, Storage and Retrieval of Image and Video Databases 1997*, pp. 45-58, Jan. 1997.

[23] Ramin Zabih, Justin Miller, and Kevin Mai. A Feature-Based Algorithm for Detecting and Classifying Scene Breaks. *Proc. ACM Multimedia 95*, San Francisco, CA, pp. 189-200, Nov. 1995.

[24] HongJiang Zhang and Stephen W. Smoliar. Developing Power Tools for Video Indexing and Retrieval. In *SPIE Vol. 2185, Storage and Retrieval for Video Databases II*, San Jose, CA, USA, pp. 140-149, Feb. 1994.