JOURNAL OF VISUAL COMMUNICATION AND IMAGE REPRESENTATION
Vol. 8, No. 2, June, pp. 107–120, 1997
ARTICLE NO. VC960346

Video Content Representation, Indexing, and Matching in Video Information Systems

Chueh-Wei Chang and Suh-Yin Lee

Institute of Computer Science and Information Engineering, National Chiao Tung University, Hsinchu, Taiwan, Republic of China

Received April 18, 1996; accepted December 12, 1996

Changing between frames is one of the most obvious information in video data. This frame-by-frame time series data is essential to many application areas. However, how to extract and compare video contents in a video information system is still an important problem to be solved. In this paper, we focus on the problem of design a fast searching method in a video information system to locate video segments that match a content-based query, approximately by time series feature values. The idea is to extract video contents via low-level feature extraction and/or high level semantic retrieval mechanisms according to a specific point of view, then segment video contents into bounding boxes via a box segmentation mechanism by their time series feature values. Video content indexing is constructed by the characteristics of prominent points that accompany bounding boxes. We also propose an efficient and effective video content matching algorithm to find similar sequences. With the help of the video indexing and matching mechanisms, several high level box-to-box and low level point-to-point query types can be requested. The implementation and performance evaluation of our video information prototype system is described. © 1997 Academic Press

1. INTRODUCTION

Due to advances in data acquisition and computer technologies, many new applications involving the video information retrieval system are emerging. Video is a medium with high complexity. It has temporal and spatial characteristics. Information related to position, timing, distance, temporal and spatial relationships are included in video data implicitly. Also, a variety of statistical features is contained in a video frame, such as object color, shape, and location.

In order to manage information in video data, a video information system must be provided. A number of special requirements distinguish the video information system design approach from traditional databases. A video information system needs complex structural representation of its multilevel contents. Video content in a video information system can be represented as a text-type keyword, a paragraph of words, a related image. It allows a user to generate queries containing both temporal and spatial concepts, and also provides content-based searching. However, how to extract and compare video contents in a video information system is still an important problem to be solved.

Therefore, the problem we deal with is the design of a video information system with an efficient video content representation, an effective multilevel query processing capability, and a fast searching method. According our researches, frame-to-frame object changing is one of the most obvious information in video data. With temporal extension, frame-to-frame object changing cause a series of frame-by-frame data. This frame-by-frame time series data is essential to many areas, such as gesture recognition in human-centered information systems, dynamic industrial processed monitoring, scene segmentation [1, 2], automatic object tracking, and dynamic scene understanding. That is, the searching method should include an indexing and a matching mechanism that can search a video information system by time-series feature values or even by multilevel semantic meanings, in order to locate video subsequences that match a query sequence approximately. Furthermore, time-series data indexing and matching mechanism can also be applied to many other applications, such as banking, policy decisions, inventory control, and scientific databases, where the history and prediction are important.

In current video database systems, only fundamental techniques, such as keyword-based searching [3], hierarchical video icon browsing and indexing [4], are provided. Most of the previous researches in video data are focused on motion and scene analysis. Very little work has been done on the design of index structures that combine spatial and temporal attributes for video databases.

In this paper, we provide several algorithms to solve these indexing and content-based matching problems. In Section 2, we describe a generic architecture of a content-based video information system, and survey some of the research projects relevant to our work. In Section 3, we define the video representation and evaluation model for

1047-3203/97 $25.00

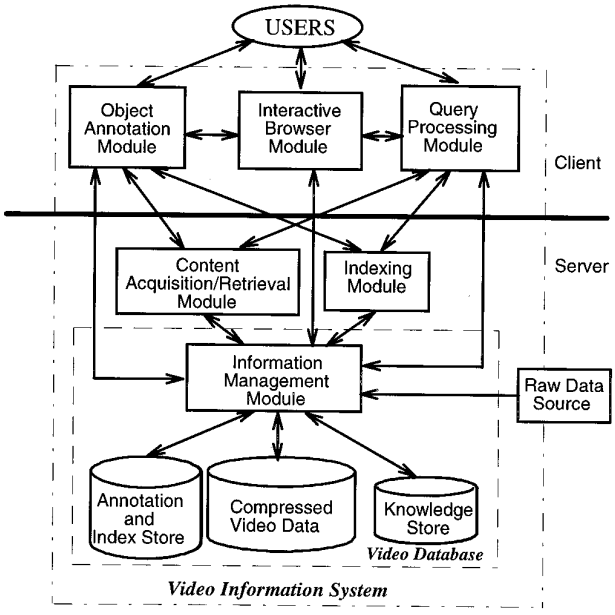

FIG. 1. Generic video information system architecture.

a video sequence. In Section 4, we show the bounding box concepts, box segmentation and indexing mechanism. In Section 5, we solve the video content approximation matching problem starting from the definition of a similarity measure. After that, a time-series video content query processing mechanism is proposed. Section 6 is the description of the implementation details of our video information prototype system, including the performance evaluation of our point-to-point approximate matching method. Section 7 includes concluding remarks.

2. OVERVIEW

We first present an overview of a video information system on which our proposed methods are based. Next, we introduce several related topics in order to make the scope of our research on time-series video content matching more clear.

2.1. Video Information System

A generic video information system architecture, as illustrated in Fig. 1, contains six modules: (i) Content Acquisition/Retrieval Module, (ii) Object Annotation Module, (iii) Video Indexing Module, (iv) Interactive Browser Module, (v) Query Processing Module, and (vi) Video Information Management Module.

An Object Annotation Module provides users an interface to build annotations of each video sequence. A video may include a variety of information that is interrelated in many ways. So that, an object annotation module supports the capability to keep a variety of complex relationships among the data, to present all the annotation descriptions, and to retrieve related data in an easy manner. It provides users many kinds of data types and the ways to establish textual and/or content-based annotations. It invokes the content extraction function to calculate the specified video features as a portion of annotation information. It can also read video files through the video management module and display on the interactive browser screen.

Next, users can make content-based retrievals via the Query Processing Module and watch the video segments of query results using the Interactive Browser Module. There are several approaches for content-based retrieval of video data. One is to attach textual and/or numerical information describing the contents and/or features to the video data. Another type of approach is to evaluate video data directly by an evaluation formula. These content-based retrieval techniques need multiple useful query types, similarity measure criteria, and user interfaces that let users pose and refine queries visually and navigate their way through the database visually. The types of queries may include free-text query, object class hierarchy selection and image/video feature-based retrieval queries.

In a video information system, a special type of video content always needs special access and indexing method. Video Indexing Module [5, 6] provides the annotation and other information with indices to speed up retrieval of the desired video segments.

Every video segment in this system can be edited with some other video segments and composed with computer graphics special effects by using the digital video editing functions in the Interactive Browser Module. Then, the editing results can be stored in the video storage and treated as another video segments.

Currently, raw video data is always very large. It must be compressed to reduce the storage space and speed up network transfer time. A Video Management Module has the capability to compress video data from video sources to save storage space and to decompress it for playing. It also provides internal level physical storage structure and an access path for the database, including an assisted knowledge store [10], video indices, and raw video data.

2.2. Related Work

Basically, we can classify content-based video queries into four categories as follows: Type-1 query by alphanumeric data and answer by alphanumeric data; Type-2 query by alphanumeric data and answer by video data; Type-3 query by video data and answer by alphanumeric data; Type-4 query by video data and answer by video data. In Type-3 query, there should contain many kinds of video computing and representations, with high or low level temporal data and spatial data interpretations [7, 8]. For

Type-1 and Type-3 queries, even though these queries involve accessing video data, the answer is just a list of text strings. Type-2 and Type-4 queries ask for relevant video footage. With formal definition, *content-based retrieval* of video data is a retrieval process based on the understanding of the semantics of the objects in a collection [9]. Content-based video query allows incompletely specified queries, which are processed through a knowledge module [10]. Most early video content retrieval systems are text-based, where relevant text keywords and/or annotations are attached to each video sequence as the basis for retrieval [3, 4, 11]. Unfortunately, the users of such systems sometimes need to provide a long list of textual query constraints to locate the desired video sequences in the video database. Several researches have been done in content-based retrieval for image and video data [12, 13], but they do not provide the capability for content matching in the temporal extension.

Further researches by Arman [14] and Hampapur [1] work in the detection of boundaries and transitions between camera shots and the classification of different types of camera operations. As stated in their paper, a camera shot, consisting of one or more frames recorded contiguously and representing a continuous action in time and space, is treated as the smallest unit for video indexing. Breaking a video into its components allows a video to be indexed and retrieved. It is the first step toward structured video [8, 15]. One of a similar model about time-series subsequence indexing and matching is discussed by Faloutsos *et al*. [16]. They use R*-tree as their basic storage structure and access method. Their subsequence matching method does not provide the multilevel searching capabilities for a variety of query types we propose in this paper.

Therefore, we need to design a new mechanism for this content-based video indexing and multilevel query matching problem.

3. VIDEO CONTENT REPRESENTATION

The video information system allows a complex structural representation of its contents. The design of an appropriate video content representation model will ensure precise association between the descriptive annotations and the objects in the video.

3.1. Video Segment Description Model

A video segment is a sequence of video shots concatenated by scene transitions (e.g., fade in/out, cross dissolve, . . . etc.). A meaningful scene is a video segment with the result of continuity in perceived, temporal, or spatial dimensions from the view points of the users. These temporal and/or spatial meanings in a video segment change frame-by-frame. By using this frame changing information contained in video frames, we can overcome many difficulties, e.g., measuring the speed of a car, encountered in interpreting a single video frame.

In our definition, a video segment[1] is a meaningful scene, $V = \{v_i, v_{i+1}, \ldots, v_{i+r-1}\}$, where v_i is the starting frame of a video sequence with frame number (or time code) i, and r is the duration of this segment. A video segment consists of several meaningful objects, such as a dog, a color, or even a thought, appearing in this video segment. That is, each video segment has content attributes and associated attribute values to describe the contents. Prior to storing video content into the video information system, the video content annotation module must first identify the relevant objects automatically or manually, then give descriptive representations of objects. Therefore, we designed a *Video Segment Description Model* (VSDM) with the annotation structures and related operations [17]. Annotations of a video segment can be described by attributes with several different data types. They can be a text type keyword, a paragraph of words, a related spatial position in one video frame, a series of specific video features in this video sequence (time-series data), or even another content related video segment. As depicted in Fig. 2, a video segment can also be represented by a video icon with image data type, a salient clip with video segment data type recursively, and secondary information, such as video ID, video segment ID. In this paper, we only address the indexing and matching problem on time-series data of video segments.

3.2. Time-Series Data and Point of View

In this subsection, we explore the relationships between the time-series data and the point of view. From a video segment, the frame changing can occur in combination of primitive feature(s), such as color, size, shape, and/or high level feature(s), such as action and timing, used to describe objects or behavior of objects in the video frames. After the image processing, annotation, or media conversion processes [10], a sequence of raw video data can be transformed into a variety of attribute values of text, or numerical data types with temporal extension. A specific point of view, addreviated as a *view*, in a video sequence can be represented by a special projection of these features, as illustrated in Fig. 3. The evaluation value of a specific view generated by domain knowledge can be a single real number obtained from the combinations of relevant features in a single video frame and this evaluation value is application dependent. For example, the evaluation value can be a weighted sum of relevant features, or some other formula specified by users and/or domain knowledge. In different video applications, different viewpoints and different similarity criteria may be required. Those relevant

[1] We use video segment and video subsequence interchangeably in this paper.

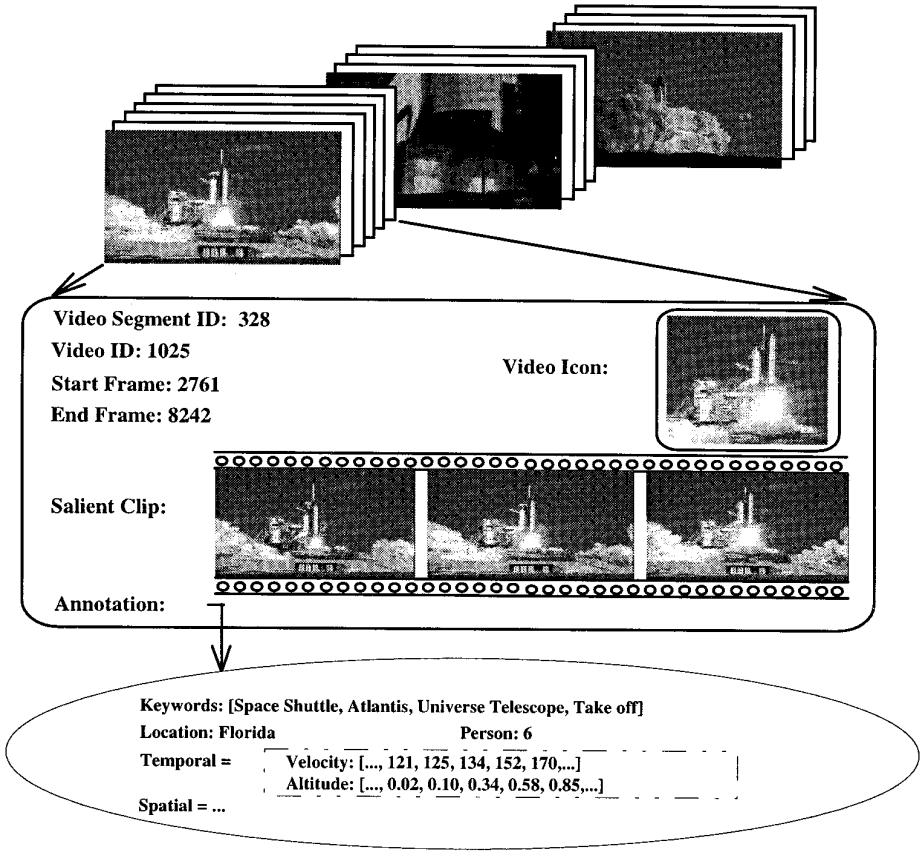

FIG. 2. An example of video segmentation description model.

feature values for the evaluation value can be calculated by image processing or feature extraction routines. Using the notions proposed in [12] and [18], similarity measure of evaluation value can be specified according to different application domains.

3.3. Evaluation Function

For a specific view in a video sequence, we call this frame-by-frame time-series evaluation values the evaluation function of this view, as defined in Definition 1. Each evaluation function can be treated as a function of time, and also be called a *curve* in this paper. Notice that an evaluation function can be a mapping from multidimension relevant feature space to one dimension evaluation value, the design of an evaluation function should take care of the problem of similarity ranking.

If we use the surveillance of road traffic as our example of application [5], the average car velocity can be an evalua-

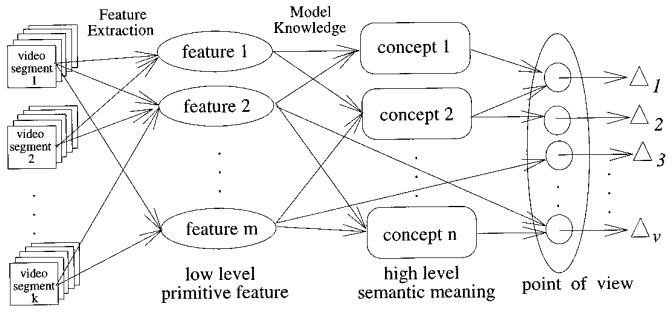

FIG. 3. Viewpoint hierarchy.

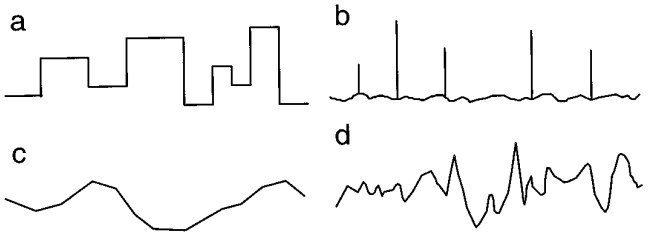

FIG. 4. Typical curve distribution in sequence evaluation function. (a) Stepwise; (b) peak; (c) smooth change; (d) irregular.

tion function of this application. The overall approach of this application is based on a moving object recognition procedure. A moving object in one video frame is searched for in a succeeding video frame. If the corresponding moving object is found, the velocity is calculated from the positional shift and perspective transformation. That is, the average car velocity is the specific view of this application, and the car velocity is obtained by calculating a combination of several position features extracted from the video sequences as well as the help of specific model knowledge.

DEFINITION 1. An evaluation function of a video segment V according to a specific view with q features is defined as

$$E(V, A, T_s, T_e) = \Delta(Ti), \tag{1}$$

where Δ is the formula of relevant feature vector combination; Ti is the time interval from starting frame T_s to ending frame T_e; $A = [f_1, f_2, \ldots, f_q]$ is a set of features for the specific video view. We use $E(t)$ that stands for the single evaluation value at time t for a specific video segment and point of view.

4. VIDEO CONTENT SEGMENTATION AND INDEXING

In this section, we attempt to divide those evaluation functions into manageable units by finding the prominent feature points. We classify curve distribution and bounding box pattern matching constraints into several categories for making our discussion clear. After that, the bounding box concept, box segmentation strategy, and indexing method are described.

4.1. Curve Distributions of Evaluation Functions

Before we provide a segmentation strategy, we first examine several typical curve distributions which occur in time-series video contents. Figure 4a shows the changing of semantic meaning in the video segment, or the variation over time in the number of a certain object (e.g., cars on a street). We say that this curve distribution has the property of being stepwise constant during each interval. In Fig. 4b, several large peaks appear in this curve distribution (e.g., the frame difference in a cut detection process). This case is very important since it represents the suddenly happened events. Figure 4c shows a situation of smooth changing (e.g., slow motion object or slow color intensity change). Figure 4d represents the randomly distributed irregular curve (e.g., fast action). According to our census, the evaluation functions of video sequences are irregularly distributed in most of the video applications. To overcome this irregular distributed time-series data indexing and matching case, we need to design a feature point finding and segmentation mechanism.

4.2. Bounding Box Principle

In this paper, we propose a *bounding box principle* as the basis of curve segmentation mechanism. Because contents in video segments can be represented as streams of symbols, the bounding box concept is motivated by the problems which arise in fields of pattern matching and similarity measure. As stated in Section 3.3, we can define an evaluation function that analyzes a video segment by using its low level features, such as representative color, and/or high level semantic meaning, such as the running and jumping of a person. Therefore, each video segment can be segmented into several structured units by a set of special evaluation values or semantic meanings. This mechanism of video segmentation using bounding box principle is so called video structuring, and the result is a structured video.

In consequence of segmentation, the video subsequence between two special successive feature points can be separated and bounded by a bounding box. That is, the evaluation function of a video segment can be divided into a series of bounding boxes by special feature points, as shown in Fig. 5. We call these special feature points the prominent index points (or *prominent point*, for short). Each segmented subsequence is represented as a rectangle box with prominent point value and related information. Except the prominent point value, the following box features can also be included in the related information if necessary: sequence and box ID, minimum and maximum values in this box, offset of duration (box length), interbox connection type, starting frame/time number, density information of box, previous and next subsequence linkages, and high level semantic meaning. An example of related information in bounding boxes is shown in Table 1.

4.3. Box Segmentation and Prominent Points

According to the curve distributions, several kinds of curve features can be found. We classify the curve features into four categories. They are suddenly up edge, suddenly down edge, increase out of range, and decrease out of

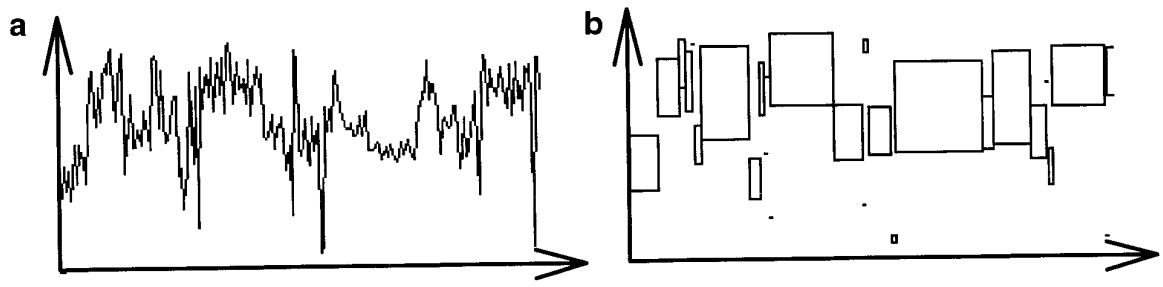

FIG. 5. Transformation between evaluation function and bounding boxes. (a) Original sequence; (b) divided bounding boxes.

range in a curve. We can derive seven connection types from these four categories. Notice that connection type 0 is used for the unstable area, such as a starting and an ending box.

• *Connection Type 1.* Large pulse when edge up and down happens in a short time period, e.g., 1/30 sec, as shown in Fig. 6.

• *Connection Type 2, 3.* Edge up/down, as shown in Fig. 7.

$$\text{(a) } E(t) - E(t-1) > \rho \text{ for edge up}$$

$$\text{(b) } E(t) - E(t-1) < -\rho \text{ for edge down,}$$

where $E(t-1)$ and $E(t)$ are the evaluation function values at time $t-1$ and t, respectively.

• *Connection Type 4, 5.* Increase/decrease, as shown in Fig. 8.

$$\text{(a) } E(t) - P(c) > \sigma \text{ for increase}$$

$$\text{(b) } E(t) - P(c) < -\sigma \text{ for decrease,}$$

where $E(t)$ is the evaluation function value at time t and $P(c)$ is the value of current prominent point.

• *Connection Type 6.* Long duration λ of steady situation, as shown in Fig. 9.

Therefore, a prominent point can be defined, as shown in Definition 2, by these seven connection types. The parameters ρ, σ, and λ are used to justify whether two sequences are similar. They could be either user-defined or determined automatically by the distribution of time-series data. The method that we use is to find the prominent points with a large peak of value change (Connection Type 1, 2, 3) or with local increase/decrease in evaluation value in the data stream (Connection Type 4, 5). No matter how the data stream is shifted, the edge-type prominent point of this evaluation function is unique. Same or similar curves will get same or similar prominent points if they follow the same prominent point definitions. That is, if we find two curves with same or similar sequence of prominent points and similar related information, we can say that they are approximate. We can take advantage of this observation of prominent points for efficient indexing in a large database.

TABLE 1
An Example of Prominent Points and Related Information

Box ID	Prominent point	Minimum value	Maximum value	Box offset	Connection type	Starting position	Average value	Accumulated difference
1	158	158	283	13	0	0	215	51
2	324	323	451	10	4	13	393	58
3	484	387	491	3	4	23	454	55
4	331	331	467	3	5	26	406	68
5	481	481	481	1	4	29	481	0
6	220	220	304	3	3	30	272	47
7	390	270	475	22	4	33	347	51
8	214	137	228	5	5	55	197	43
9	436	325	436	2	4	60	380	111
10	241	241	241	1	5	62	241	0
11	409	409	409	1	4	63	409	0
...								

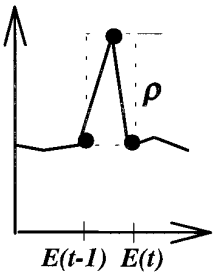

FIG. 6. Large pulse case.

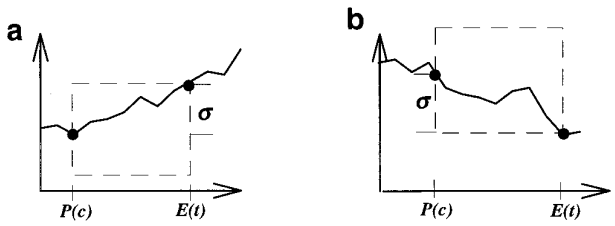

FIG. 8. Increase/decrease cases. (a) $E(t) - P(c) > \sigma$ for increase; (b) $E(t) - P(c) < -\sigma$ for decrease.

DEFINITION 2. A prominent index point of evaluation function at time t is the point P that satisfies at least one of the following conditions:

(i) An evaluation value at time/frame t has a change from previous point $t - 1$

$$|E_t - E_{t-1}| > \rho, \tag{2}$$

where E_t is the current evaluation value at time/frame t, and ρ is the threshold value.

(ii) The evaluation value difference between the previous prominent point and the current evaluation point is greater than a threshold

$$|E_t - P_c| > \sigma, \tag{3}$$

where E_t is the current evaluation value at time/frame t, and P_c is the current prominent point at time/frame c. σ is the threshold value.

(iii) The time/frame difference between the previous prominent point and the current evaluation point is greater than a threshold λ.

4.4. Video Indexing

We use B-tree [19] as our index structure with prominent points as the keys because B-tree has the efficient storage structure and is a robust access method for data points. For each new bounding box, inserting a new prominent point in the index tree is done by a searching index tree and adding the prominent point in a node. The related information about this bounding box is stored in the storage space of a corresponding link list structure and can be accessed through a link list pointer accompanied with the prominent point in the leaf node. Overflowing nodes are split and splits are propagated to parent nodes. If the prominent point in the index tree has already existed, the related information of this new bounding box will be attached at the front of this corresponding link list. That is, the linked list refers to index structure in which an index point may be associated with a list of reference fields pointing to video sequences that contain the same or similar prominent points. By using the linked list, we can easily find the bounding boxes with similar prominent points and similar box shapes. Figure 10 is an example of index structure with an index tree of order 4 (nine branch pointers for each node) with integer search value and a box link list.

To avoid too many index points in the index tree, we can concatenate several prominent points with similar values in the same child node by using prominent point quantization mechanism, or by proper selection of the threshold values ρ, σ, and λ. The best choices of threshold values are dependent on the distribution characteristics of evaluation values and the query behaviors. We can see these variations in the performance evaluation of Section 6.

4.5. Prominent Point Quantization

The purpose of quantization is to reduce the number of keys and storage space for an index structure. But it will

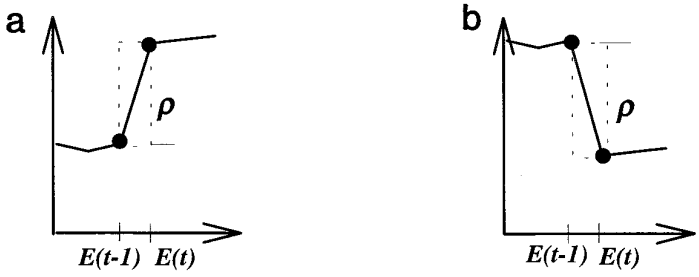

FIG. 7. Edge up/down cases. (a) $E(t) - E(t-1) > \rho$ for edge up; (b) $E(t) - E(t-1) < -\rho$ for edge down.

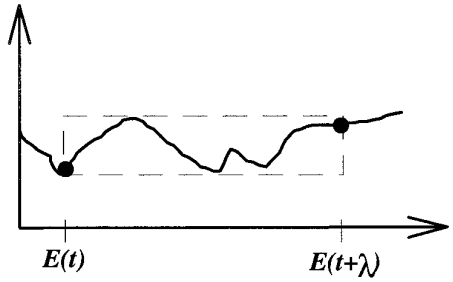

FIG. 9. Long duration case.

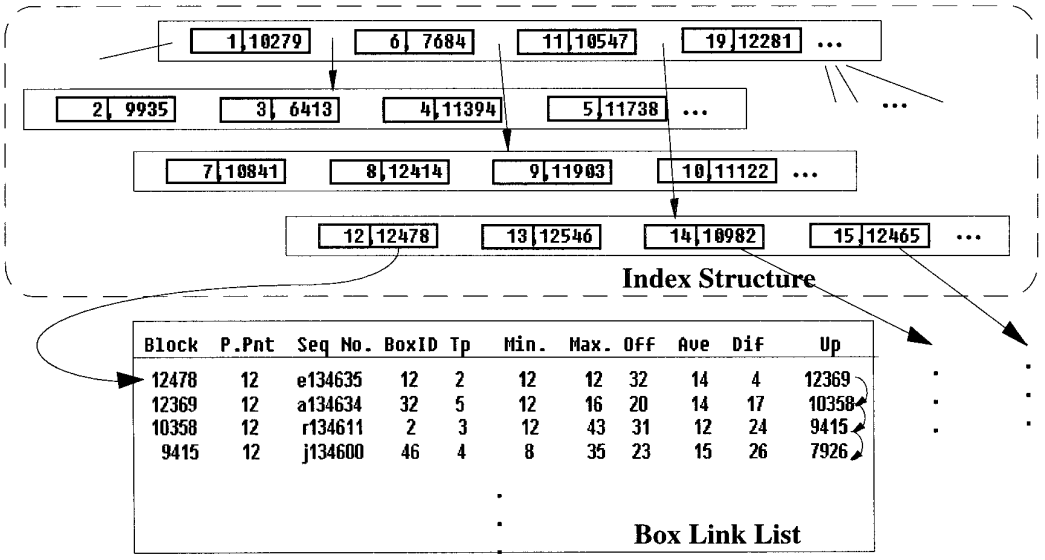

FIG. 10. An example of video indexing structure.

increase the index structure searching time for the existing of potential search values. Quantization is defined as division of each prominent point by its corresponding quantizer step size, followed by rounding to the nearest integer:

$$P^q = IntegerRound\left(\frac{p}{Step}\right). \tag{4}$$

5. VIDEO SEQUENCE QUERY PROCESSING

Generally speaking, the time-series video content query is the problem of pattern matching with the help of indexing. Two steps must have been done before proceeding to the matching process. One is the choice of video content query type. Another is the choice of query constraints.

5.1. Multilevel Video Content Query Types

In a video information system, it is necessary to be able to locate some or all occurrences of similar box patterns quickly. We know that the contents of a video segment should be expressed in terms of a set of low level primitive features, and/or combine low level features to form more complex high level semantics. For example, we can specify the query ''A person walks on the sidewalk, then suddenly runs across the street and sits on a street chair'' by a high level query pattern ''(walk)(run)(sit).'' Another example is the sequence of video shot types for parsing of news episode in [2]. From the bounding box principle, no matter what kind of video content expressions, the query patterns are considered to be a sequence of values provided by query bounding boxes.

The demand of finding an exact match between two video segments of specific view might be too strict since the real numbers may vary widely. In most of the video applications, users often require finding close or similar but not necessarily exact occurrences. Alternatively, an approximate matching is to find all subsequences in sample video sequences that are close to a query video segment according to some similarity criteria. Therefore, multilevel approximate queries of video segments can be diversified into several categories:

(i) Box-to-Box Matching

• *Existence matching*. Find those shortest sample box sequences that for each box in the query box sequence, there exists at least one box, which has the same box type, in the matched sample sequence. The order of box types in the matched sample sequences can be neglected. An example is shown in Fig. 11a. Notice that sample and query sequences do not need to have the same box length. The letters in each box stand for the semantic meaning or the prominent points in each bounding box.

• *Sequence matching*. Find those shortest sample box sequences that for each box in the query box sequence, the corresponding box in the sample box sequence has the same box type and also has the same order.

case 1. Exact Sequence Matching—exact one-to-one mapping, as shown in Fig. 11b.

case 2. Partial Ordering Matching—can have a redundant pattern within sample sequence, as shown in Fig. 11c.

(ii) Point-to-Point Matching

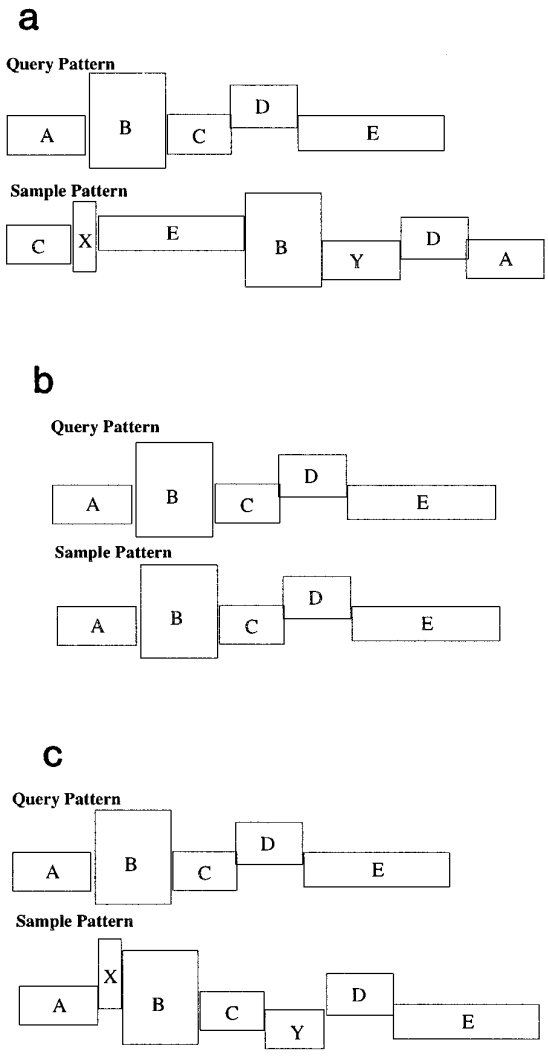

FIG. 11. Query types. (a) Existence matching; (b) exact sequence matching; (c) partial ordering matching.

• Exact Curve Matching. Find those sample sequences that the corresponding values are exactly the same as query values.

• Approximate Curve Matching with Error Tolerance. Find those sample sequences that the distance between query and sample sequences are within the tolerance of similarity threshold. In other words, those candidates should have a similarity relation for each corresponding value.

Both types of point-to-point curve matching are based on a definition of the *good-match* [20] criterion, as defined in Definition 3. The good-match retrieval is to find the sequence of patterns or evaluation values that are sufficiently similar within some distance ($\tau \sim 0$). With the definition of the good-match, the matching approach should find those patterns or evaluation values close to the search pattern in the video information system within a similarity threshold.

DEFINITION 3. Given two sequences of patterns, $X = x_1x_2 \cdots x_n$ (sample pattern) and $Y = y_1y_2 \cdots y_m$ (query pattern), over an infinite alphabet of real numbers, where n and m are respective length of sequence X and Y, if there exists a position (alignment) k in X such that for each pair of corresponding alphabet in these two sequences the similarity measure is smaller than the similarity threshold τ, then subsequence $X' = x_kx_{k+1} \cdots x_{k+m-1}$ is a good-match with Y.

5.2. *Query Constraints and Similarity Measure of Query Types*

The query result for each query type by some specific query constraint is a set of qualified candidates and the candidates' similarity factor. The result returning from the filtering process of query constraint is a list of qualified candidates that have passed all the checking of the selection conditions. The *similarity factor* (or accumulated penalty) is the summation of similarity measures between the query pattern and the qualified sample patterns for the selected query types.

Except the prominent point, a query constraint can be composed by the selection of the box connection type, min/max range, box density, box aspect ratio, semantic meaning, etc. For example, the box density is defined as the average value of the accumulated difference between two consecutive evaluation values within the same bounding box,

$$D(B_s, B_e) = \sum_{t=B_s+1}^{B_e} \frac{E(t) - E(t-1)}{(B_e - B_s - 1)}, \tag{5}$$

where $E(t)$ is the evaluation value at time t, and B_s and B_e are the starting and ending frames of a bounding box, respectively.

The point-to-point similarity measure between two single video frames is defined in Definition 4. The box-to-box similarity measure between two bounding boxes is defined in Definition 5. The *penalty function* determines the difference between two bounding boxes. This function value is dependent on how dissimilar these two boxes are and also what kind of query constraints they select. The value of *similarity threshold*, as defined in Definition 6, is application dependent and can be specified by users. This similarity threshold value is the error tolerance for an approximate matching and can heavily affect the performance of searching. If the similarity threshold increases, the number of qualified subsequences would increase. If the similarity threshold is equal to zero, this process becomes an exact match.

DEFINITION 4. The point-to-point similarity measure (point-to-point distance) of a specific view between two video frames at time t_i and t_j is defined as

$$Sp\mathrm{V}_1, V_2, A, t_i, t_j) = |E(V_1, A, t_i, t_i) - E(V_2, A, t_j, t_j)|, \quad (6)$$

where t_i determines the frame in sample video sequence V_1, and t_j determines the frame in query video sequence V_2.

DEFINITION 5. The box-to-box similarity measure (box-to-box distance) of a specific view between two bounding boxes b_i and b_j is defined as

$$S_b(V_1, V_2, A, C, b_i, b_j) = \sum_{k=1}^{n} Penalty(C_k, b_i, b_j), \quad (7)$$

where b_i and b_j are bounding boxes of sample video sequence V_1 and query video sequence V_2, respectively. C is the set of n query constraints for this query. $Penalty(\cdot)$ is the penalty function for each specified type of constraint C_k.

DEFINITION 6. The evaluation function $E(V_1, A, t_i, t_j)$ and $E(V_2, A, t_i, t_j)$ or bounding box b_i and b_j have the similarity relation $\sim$, if and only if

$$S_p(V_1, V_2, A, t_i, t_j) < \tau p, \quad \text{for point-to-point matching or} \quad (8)$$

$$S_b(V_1, V_2, A, C, b_i, b_j) < \tau b, \quad \text{for box-to-box matching,} \quad (9)$$

where τp and τb are the similarity threshold of point-to-point matching and box-to-box matching, respectively.

5.3. Matching Strategies and Box-to-Box Approximate Matching

Because existence matching is easy to handle, and also partial ordering matching can be considered as the longest common subsequence problem [21], their matching algorithms will not be discussed in this paper. Therefore, we focus our matching problems only on exact sequence matching and point-to-point matching.

In the matching processes, we first divide the query sequence into its bounding box representation form. Then, these bounding boxes compare with the sample sequences in the video information system with the help of an index structure. No matter what kind of query type it is, either box-to-box or point-to-point basis, we always start our matching process from an approximate box searching approach, as shown in Algorithm 1.

ALGORITHM 1. Approximate Box Searching

Input. A sequence of bounding boxes with related box information corresponding to query sequence, an index structure, and similarity threshold.

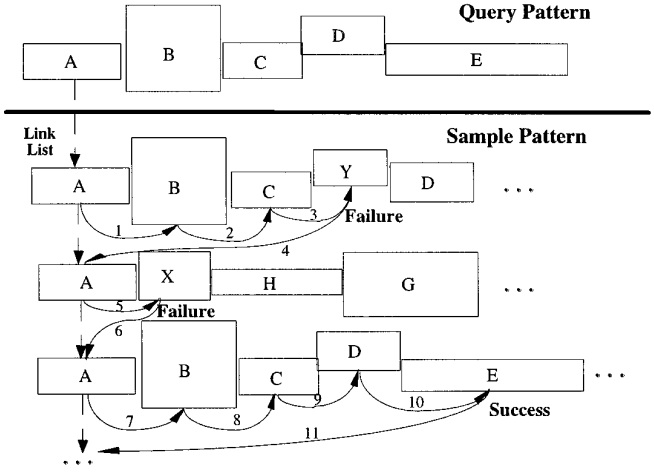

FIG. 12. An example of box-to-box exact sequence matching.

Output. A list of similar subsequences with good-match criterion.

Method.

find first bounding box with connection type 1, 2, or 3;
search index structure by prominent point value of first bounding box to find the link list of the starting position of candidate boxes;
for *each of the candidate boxes*
if *a consecutive sequence of boxes related to the candidate starting box satisfy the query constraints* then
print out the sequence ID, starting position and similarity factor;
end-of-if
end-of-for

End-of-Algorithm Approximate Box Searching.

In our approximate box searching approach, we use the first box with pulse or edge connection types (type 1, 2, or 3) in the query sequence as the alignment box. After a searching in index structure by the prominent point of alignment box, several candidates with the same or approximate prominent point value will be found. According to the specified query constraints, we can discard or prune some of the cases which do not satisfy the box-to-box similarity threshold when searching for the candidates in the index structure. An example of query constraint checking procedure for box-to-box exact sequence matching is depicted in Fig. 12.

This approximate box checking algorithm acts as a filter to quickly reduce the number of possible candidates and generates a candidate set as the result of query constraint checking. Notice that false alarms are possible in these steps, but no false dismissal will occur. Further processing of the candidate set is necessary to avoid mismatching.

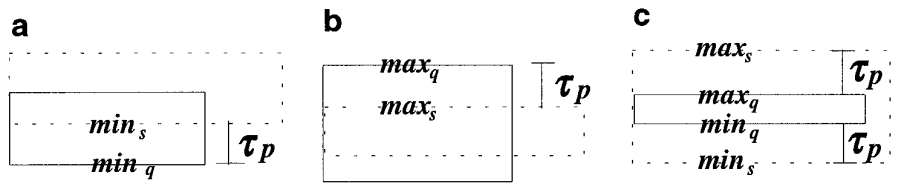

FIG. 13. Similarity relation checking between bounding boxes. (a) $|min_s - min_q| > \tau p$; (b) $|max_s - max_q| > \tau p$; (c) both of them.

5.4. *Point-to-Point Matching Algorithm*

For a point-to-point matching, two necessary query constraints should be checked. They are the point-to-point similarity measure and the box-offset checking. After the box alignment step and the index structure searching, the next step is the query constraint checking. If we check the similarity relation for the corresponding evaluation values in the candidate box point-to-point, it will be very time-consuming. Therefore, we provide a min/max bound similarity relation checking mechanism.

As stated in Theorem 1, we stop the searching when the difference of minimum and/or maximum values of bounding boxes of two corresponding sequences exceeds point-to-point similarity threshold τp, as shown in Fig. 13. We can declare these two corresponding sequences to be dissimilar and prune this sequence from the candidate set. If all of the segmented subsequences satisfy the similar relation, we are sure that the whole sequence satisfies the similar relation.

THEOREM 1. *There exists at least one evaluation value in the query box that can not satisfy the similarity relation, if*

$$|min_s - min_q| > \tau p, \text{ or} \tag{10}$$

$$|max_s - max_q| > \tau p, \tag{11}$$

where min_s and max_s are the minimum value and maximum value of a sample box, respectively; min_q and max_q are the minimum value and maximum value of a query box, respectively. τp is the point-to-point similarity threshold.

6. IMPLEMENTATION DETAILS

6.1. *A Video Information System Prototype*

We have implemented a prototype system on an IBM-PC compatible computer with MS-Windows 95 by using C++ language to test our indexing and approximate matching mechanism. In the experimental stages of our design, we do not concern ourselves with finding an appropriate video file format, and are using default size 160 × 120 AVI file. Another video file format such as MPEG II, is under consideration for fast playback speed and good compression rate.

6.2. *User Interface*

In this prototype system, we separate the user interface into nine different function areas, as shown in Fig. 14. In the Thumbnail Area, it can act as a display of six-divided frames, or a set of still salient images of active video sequences, or an A-to-F roll editing monitor each with a different video file, or the video icons of query results. In the Playback Area, it has the functions of mark in/out logging, nonlinear editing preview window, active video file playback, current processing frame display. In the Image Processing Area, it shows the results of video/image processing functions (for example, color key, caption, special effects, etc.), temporary duplicated still frame, region-based color feature extraction. In Single Frame Data Area, it shows the feature values in a current processing frame (for example, histogram, region size, etc.). In the Curve Data Area, it shows the evaluation curve, bounding boxes, query and matched curve. In the Video Parameter Area, it shows the related parameters of a current active video file. In the Annotation Area, it shows the default annotation about video contents. In the Status Bar Area, it shows the process percentage, current cursor coordinate and corresponding R, G, B color intensity, evaluation function value of curves, the video editing mark in/out cue points. In the Menu Bar Area, it provides the annotation, feature extraction, indexing, query processing and database management functions, as stated in Section 2.1, for this video information system.

6.3. *Performance Evaluation*

We have built up a video database of more than 500 video segments and time-series sample curves. We examined this indexing and matching mechanism by several feature extraction methods: the most significant color, the histogram difference, and the motion tracks of a specific object in video.

In order to see the performance and behaviors of our point-to-point approximate matching method, we started from several experiments. We ran these experiments on our video database of approximately 75,000 sample points. Each point was an integral number. We divided these sample points into three different sequence sets, Set 1, 2, and 3, according to their behavior of distributions. Sequence

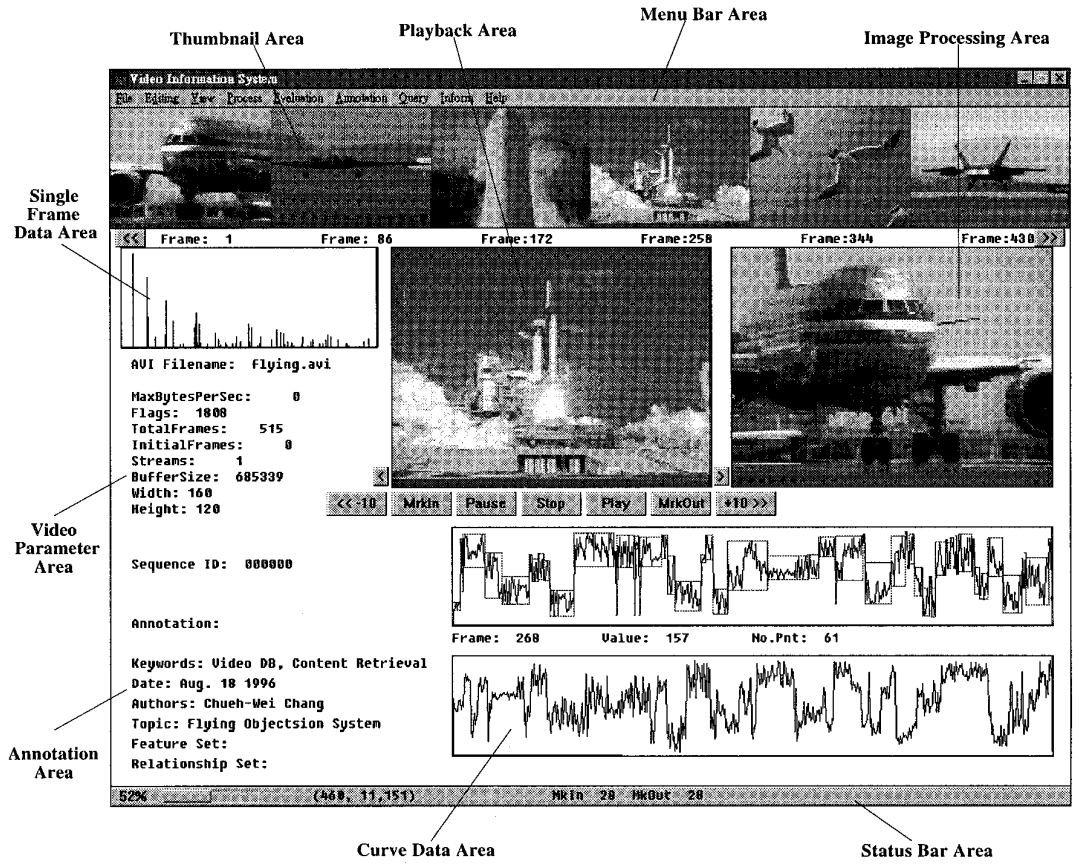

FIG. 14. The video information system prototype.

Set 1 has the largest data variation and Set 3 has the smallest one. We also set out two different segmentation groups, Group A and Group B, of experiments to compare the query response time between our matching method and the single subsequence approximate matching [5] (approximate KMP algorithm). Each segmentation group has different threshold parameters σ, ρ, and λ. The query sequences were generated randomly. A small similarity threshold τp (<1%) was specified.

Figures 15a and 15b give the relative query response time of the approximate KMP method versus our index-assisted matching method. In these experiments, the response time of our matching method is five to seven times faster than approximate KMP method when the length of query sequence is short (<50 integer values), and at least has two times faster when query sequence is long (>100). However, it is true that the number of bounding boxes is proportional to the length of query sequences. The increase in the number of bounding boxes leads to an increased box-to-box sequence matching process time. We can see these phenomena when query sequence is long (>300). Furthermore, Fig. 16 illustrates the ratio of response time under different threshold parameters, Group A versus Group B, in those three sequence sets. The graph shows that Group A always has a faster response time than Group B (time ratio Group B/Group A > 1). That is, appropriate selection of threshold parameters according to the curve distributions can provide a better matching performance.

7. CONCLUSIONS

By providing multilevel content-based retrieval, applications of digital video are broad in many aspects. Video records the changes of scenes according to time. Related change of objects between different frames provides much information about the behavior of these objects in the video. These time-series changes of video objects are useful for dynamic scene and motion analysis.

In this paper, we have presented the design of an approximate video content matching algorithm. The idea is to extract video contents via low level feature extraction and/or high level semantic retrieval mechanisms according to a specific point of view, and then segment video contents into bounding boxes via a box segmentation mechanism by their time series feature values. With the help of indexing

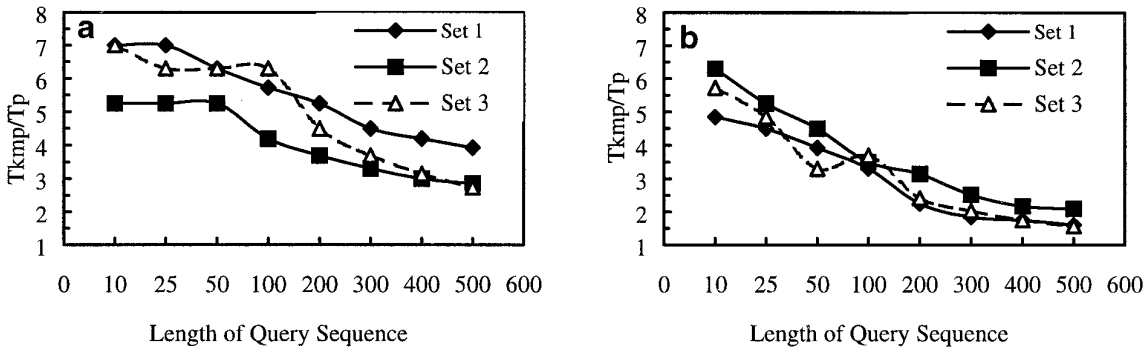

FIG. 15. Relative response time vs. query length. (a) Group A ($\rho = 240$, $\sigma = 180$, and $\lambda = 80$); (b) Group B ($\rho = 160$, $\sigma = 120$ and $\lambda = 80$).

mechanism using prominent index points, the searching speed is faster than the sequential scanning method. A video information prototype system example and several experimental results show how these mechanisms work. Notice that time-series data indexing and matching mechanism can also be applied to many other applications, such as banking, policy decisions, inventory control, and scientific databases, where the history and prediction are important.

APPENDIX: LIST OF SYMBOLS

E	Evaluation function
Δ	Formula of feature vector combination
A	A set of features for the specific video view
τ	Similarity threshold
$E(t)$	Evaluation function at time t
P	Prominent index point at time t
P_c	Current prominent index point
ρ	Edge threshold
σ	Increase/decrease threshold
λ	Offset threshold
P^q	Quantized prominent index point at time t
D	Bounding box density
S	Similarity measure
$\sim$	Similarity relation
B_s, B_e	Start frame and end frame of a bounding box

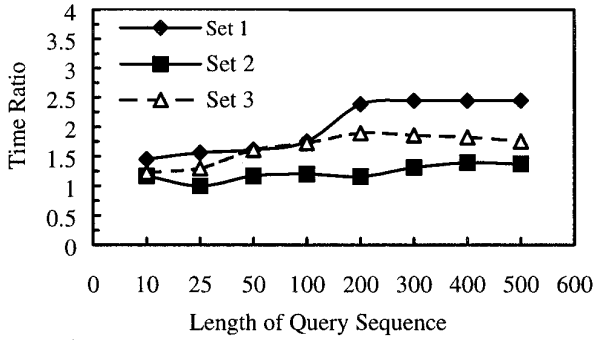

FIG. 16. Ratio of response time (Group B/Group A).

REFERENCES

1. A. Hampapur, R. Jain, and T. Weymouth, Digital video segmentation, in *Proceedings, ACM Multimedia, San Francisco, CA, 1994*, pp. 357–364.
2. H. J. Zhang, S. Y. Tan, S. W. Smoliar, and G. Yihong, Automatic parsing and indexing of news video, *Multimedia Systems,* **2,** 1995, 256–266.
3. E. Oomoto and K. Tanaka, OVID: Design and implementation of a video–object database system, *IEEE Trans. Knowledge Data Eng.*, **5**(4), 1993, 629–643.
4. S. W. Smoliar and H. Zhang, Content-based video indexing and retrieval, *IEEE Multimedia*, Summer 1994, 62–72.
5. C. W. Chang and S. Y. Lee, Indexing and approximate matching for Content-based time-series data in video database, in *Proceedings, First International Conference on Visual Information Systems, Melbourne, Australia, 1996*, pp. 567–576.
6. 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, Vol. 1908, 1993,* pp. 25–36.
7. Y. F. Day, S. Dagtas, M. I. A. Khokhar, and A. Ghafoor, Object-oriented conceptual modeling of video data, in *Proceedings, IEEE 11th International Conference on Data Engineering, Taipei, Taiwan, 1995*, pp. 401–408.
8. A. Nagasaka and Y. Tanaka, Automatic video indexing and full-video search for object appearances, in *Visual Database Systems II* (E. Knuth and L. M. Wegner Eds.), pp. 113–127, Elsevier, Amsterdam/New York, 1992.
9. A. D. Narasimhalu, Special section on content-based retrieval, *Multimedia Systems*, **3,** Feb. 1995.
10. A. Yoshitaka, S. Kishida, M. Hirakawa, and T. Ichikawa, Knowledge-assisted content-based retrieval for multimedia databases, in *Proceedings, IEEE International Conference on Multimedia Computing and Systems, Boston, MA, 1994*, pp. 131–139.
11. T. D. C. Little *et al.*, A digital on-demand video service supporting content-based queries, in *Proceedings, ACM First International Conference on Multimedia, Anaheim, CA*, 1993, pp. 427–436.

12. R. Bach, S. Paul, and R. Jain, A visual information management system for the interactive retrieval of faces, *IEEE Trans. Knowledge Data Eng.* **4**(4), 1993, 619–628.

13. M. Flickner *et al*., Query by image and video content: The QBIC system, *IEEE Comput*., 1992, 23–32.

14. F. Arman, R. Depommier, A. Hsu, and M. Y. Chiu, Content-based browsing of video sequences, in *Proceedings, ACM International Conference on Multimedia, San Francisco, CA, 1994*, pp. 97–103.

15. Y. Tonomura, A. Akutsu, Y. Taniguchi, and G. Suzuki, Structured video computing, *IEEE Multimedia*, Fall 1994, 34–43.

16. C. Faloutsos, M. Ranganathan, and Y. Manolopoulos, Fast subsequence matching in time-series databases, in *Proceedings, ACM SIGMOD, Minneapolis, MN, 1994*, pp. 419–429.

17. C. W. Chang, K. F. Lin, and S. Y. Lee, The characteristics of digital video and considerations of designing video databases, in *Proceedings, ACM Fourth International Conference on Information and Knowledge Management CIKM'95, Baltimore, MD, 1995*, pp. 370–377.

18. K. Wakimoto, M. Shima, S. Tanaka, and A. Maeda, Content-based retrieval applied to drawing image database, in *Proceedings, SPIE, Vol. 1908, 1993*, pp. 74–84.

19. D. Comer, The ubiquitous B-tree, *ACM Comput. Surv.* **11** (2), 1979, 121–137.

20. T. Ito and M. Kizawa, Hierarchical file organization and its application to similar string matching, *ACM Trans. Database Systems* **8**(3), 1983, 410–433.

21. Y. P. Wang and T. Pavlidis, Optimal correspondence of string subsequences, *IEEE Trans. Patt. Anal. Mach. Intell.* **12**(11), 1990, 1080–1087.

CHUEH-WEI CHANG received his B.S. and M.S. degree in computer engineering in 1984 and 1986 from Chiao Tung University. His research interests include multimedia information systems, data models, image processing, and computer graphics applications. He is a doctoral student in Institute of Computer Science and Information Engineering at Chiao Tung University.

SUH-YIN LEE received the B.S.E.E. degree from Chiao Tung University in 1972 and the M.S. degree in computer science from the University of Washington, Seattle, in 1975. She received the Ph.D. degree in electronic engineering in 1982. She is now a professor in the Department of Computer Science and Information Engineering at Chiao Tung University. Her current research interests include image database, multimedia information system, and computer networks.