ELSEVIER

Information Sciences 118 (1999) 37–73

INFORMATION SCIENCES
AN INTERNATIONAL JOURNAL

www.elsevier.com/locate/ins

A new content-based access method for video databases ☆

P.J. Cheng, W.P. Yang *

Department of Computer and Information Science, National Chiao Tung University, Hsinchu, Taiwan, ROC

Received 1 August 1998; received in revised form 1 May 1999; accepted 29 June 1999

Abstract

This paper presents a novel video data model and a nested annotation language for describing complex information of video data. In contrast to conventional approaches, the proposed model classifies different video materials of interest to users into different representation frameworks according to their individual properties. It makes the model flexible and capable of sharing video materials. The nested annotation language effectively describes scenarios in video data and can be effectively analysed. With the assistance of domain knowledge and index organizations this investigaton also develops algorithms to effectively process five types of familiar video queries: semantic query, temporal quaery, similar query, fuzzy query and hybrid query. In addition, a SQL-like query language for video content retrival is provided. Experimental results indicate that combining the concepts of Bayesian networks and inheritance of attributes by context significantly improves the content based retrieval of video data. Moreover a prototype system based on proposed model has been implemented. © 1999 Published by Elsevier Science Inc. All rights reserved.

Keywords: Content-based retrieval; Video data model; Video indexing; Video database system; Knowledge representation; Information retrieval

☆ This research was partially supported by the National Science Council, Taiwan, ROC, under contract No. NSC87-2213-E009-001.

* Corresponding author.

E-mail address: wpyang@cis.nctu.edu.tw (W.P. Yang)

1. Introduction

Video has found extensive applications owing to its richness and intuitive ability to capture complex information in the real world. Advances in high-capacity storage and high-performance compression technologies have made large scale video archive an important source in modern databases. A video database is a software system capable of managing a video data collection and providing content-based access to users [13]. Recently, object storage systems [6,7] and video server systems [23,30] have received considerable attention. However, the lack of an efficient mechanism to present and annotate video content limits the ability of databases to retrieve video data with the same facility as conventional data types such as text, graphics and image. In this work, we adopt an example which is a video documentary entitled "*The University Campus*" containing information about the faculty, students, departments, Founder's Day celebration, commencements, intramural sports and other relevant topics. This example documentary will be used throughout the whole paper. Some queries that may be asked of the example are illustrated as follows:

Type 1. Semantic query. Semantic queries specify a predicate that involves the semantic content of video data. The semantic content is comprehensive knowledge that a video conveys to users. It is usually referred to as an event, object or relationship among events and objects in video clips. Consider the following queries: *find all video frames in which the president appears, find all professors whose research interests include video databases* and *find all video frames in which professor Yang is teaching graduate students databases.*

Type 2. Temporal query. Temporal query adopts temporal information as inputs. Temporal information contains either a range of video frames or an ordering relationship between two intervals. Consider the following queries: *find all events that occur in the interval from frame* 1000 *to* 1500 and *find all lectures that appear after the Founder's Day celebration.*

Type 3. Similar query. Similar queries find the audio-visual features of video data that resemble the given feature examples. While helping users understand the semantic content of video data, audio-visual features lack their meanings. Owing to the difficulty that users have in describing audio-visual features precisely, this kind of query needs a mechanism to support the similarity measurement. For example, users may attempt to search for swimming pools by sending the request: *find all objects whose color is close to blue.*

Type 4. Fuzzy query. Fuzzy queries view a vague description involving linguistic terms as inputs. Consider the following queries: *find all professors who are famous* and *find all basketball games with a large audience*, where *famous* and *large* are vague linguistic terms.

Type 5. Hybrid query. Hybrid queries are a Boolean combination of the queries described above. Consider the following queries: *find all events in which* *a professor, who is famous and noted for his teaching, is speaking from the middle frame to the last frame.*

Video data contents are classified by Hampaur [13] according to the following criteria: semantic, audio-visual and textual contents. Semantic and fuzzy queries are applied to find high-level semantics and textual information. Similar and temporal queries are applied to find low-level features such as color, texture, audio, camera operation and spatiotemporal relationship. In this paper, we present a novel generic video data model and an annotation language for providing great expressive power to describe complex content of video data. Based on this model, we consider several index schemes to evaluate queries with the assistance of domain knowledge. Also developed herein is a SQL-like query language and algorithms to present and process the above-mentioned queries, respectively. The primary contributions of this paper are summarized as follows:

1. Our schemaless data model, which provides the flexibility and sharability of annotation/audio-visual features, categories the entities of relevant interest in video data, respectively, according to their different characteristics;
2. the nested annotation structure of the proposed data model can accurately interpret complex scenarios in the real world and has a lower computational complexity than natural language;
3. the proposed data model offers a more feasible means of content-based retrieval by describing conditional dependence relationships among meaningful scenes and
4. the proposed system can efficiently handle different types of queries with the assistance of domain knowledge and index organizations. In addition, high-level semantics can be inferred from low-level features automatically and vice versa.

The rest of this paper is organized as follows. Section 2 describes the pertinent literature related to content-based accesses to video data. Section 3 introduces the basic ideas, underlying assumptions, and then, presents our video data model and annotation structures. Next, Section 4 defines the syntax of query language, classifies the queries into five categories, and describes how to process these queries with the assistance of indices and domain knowledge. Section 5 then describes the architecture and data structures of the prototype system. Conclusions are finally made in Section 6.

2. Related work

Having received considerable attention, content-based retrieval of video data can be categorized as follows based on the indexed and annotated information content: semantic-based approach [1,9,10,15,19,21,22,26,29,31], audiovisual feature-based approach [5,11,16,28,32], and hybrid approach [8,14,24,33].

Focusing mainly on annotating the semantic content in video data, the semantic-based approach is suitable for exploring the content description with meaning and intuition to users. However, this method fails to automate annotation processes exactly. Moreover, semantic annotation is generally ambiguous and application-dependent.

Adali et al. [1] proposed the AVIS system which uses a frame-segment tree and an object and an event array to derive an efficient structure for query processing. However, that system [1,29] did not consider the relationships among objects and events. Moreover, the nodes of a frame-segment tree can be replicated in many ordered-linked lists of object or event arrays, resulting in low storage utilization. Oomoto and Tanaka [26] proposed the OVID system based on a schemaless object-oriented data model. Their model provides interval-inclusion-based inheritance and operations to composite video objects. Later, Weiss et al. [31] proposed an algebraic video data model. The video algebra supports nested stratification by recursively employing a set of algebra operations, such as create, delay and union. However, [26] and [31] did not address query-processing algorithms. A. Dan et al. [9] proposed a hierarchical multimedia annotation structure to provide the multiplicity of views and the merits of sharing and reusing annotations among users.

Several annotation structures have been developed to represent semantic information, including keywords [14,29], attributes [1,9,15,21,24,26,31,33], natural languages [8,19,22], and iconic languages [10]. The keyword approach is relatively simple. The attribute approach takes the form of attribute-value pairs. By taking attributes into account, this method depicts the meanings of values more precisely than the keyword annotation. However, both of keyword and attribute approaches are limited in terms of satisfactorily describing complex scenarios due to their restricted structures. Conversely, while providing a powerful expressive capacity, the natural language approach is inhibited by the computational complexity of intelligent parsers. Kim [22] encoded the video content by natural language. Each sentence can be parsed into the form of subject-centered dependency structures, which are joined into a multi-path index tree to efficiently retrieve the description. Jiang [19] introduced a VideoText data model based on free text annotations, and in doing so, integrated information retrieval technologies to evaluate content-based queries and to rank query results.

The audiovisual feature-based approach derived from image information systems extracts low-level features from video data. While completely automating the indexing processes, this method lacks semantics associated with the extracted features. Despite the visual effectiveness of this method, users have difficulty in querying audio-visual features, such as color, texture, shape and object motion. The highly promising approach attempts to integrate semantic-based and feature-based contents. In a related work, Chua and Ruan [8] proposed a VRSS system based on a two-layered, concept-based data model. To evaluate a similar query, their system propagates similarity values proportionally up the model. Traversing a scene hierarchy limits its performance. Later, Lee et al. [24] proposed a VIMS system based on a dynamic data model. Their model consists of clustering trees and role trees, where cluster trees resemble composite hierarchies and role trees are similar to "is-a" hierarchies. While introducing the concept of domain knowledge, Yoshitaka et al. [33] proposed a system for evaluating indirect queries. In their system, the type of specified condition may differ from that of target data.

3. The video data model

3.1. Basic ideas and term definitions

In our video databases, raw video data consist of a sequence of continuous frames, where a set of objects interacts with each another. A *video frame*, commonly referred to as a static image, is the basic unit of video data. A *frame sequence* is defined as a set of frame intervals, where a *frame interval* $[i, j]$ is a sequence of video frames from frame i to j. A *video object* is a concrete or abstract entity appearing in video frames, such as a professor, a department, a course, and the notion of object-oriented approaches. This object contains its inherent attributes and composite relations to other objects. A situation in which the interplay among a collection of objects has important implications for us is referred to, herein, as an *event*, such as a lecture, a basketball game, and a commencement. An event can be expressed in terms of spatial-temporal relations among objects. In addition to including the proper attributes, an event also contains scenarios and represents semantic abstractions. Theoretically, the entire video stream and the single frame are two extreme abstraction levels. Other intermediate abstractions may include shots, scenes, sequences and compound units [13]. None of the video objects must describe scenarios and multi-level abstractions. Consequently, events and objects have different properties from the aspect of knowledge representation. Herein, these events and objects are modeled, respectively, according to their individual characteristics and the relationship between them is maintained.

From a querying point of view, this investigation assumes that objects, events and video frames are the entities of relevant interest in most applications. This assumption is reasonable and used in [1,14,17,24]. Consider the example queries mentioned in the Introduction. Type 1 involves objects/events and the semantic relationships among them. Type 2 deals with temporal relations of video data. Types 3 and 4 concern the vague properties of objects/events.

To classify the features of objects and events, we propose a video data model with two major components: (1) *the Object Net* – a net structure which describes primitive and composite references among objects and (2) *the Event Hierarchy* – a hierarchy structure which exhibits the multi-level abstraction of video data and has the ability to annotate complex scenarios. The relationship between an object and an event refers to a situation in which objects are role players appearing in an event or which events involve the objects.

Our data model can integrate semantics and features of video data. Arbitrary text-based or feature-based descriptions can be associated with an object or an event, such as keywords, free text, icons, key frames, video clips, colors, textures, spatial locations and temporal relations. An increasing number of successful applications involving image feature processing have been made [5,8,14,16,28,32]. However, the expressive capability and computational complexity are trade-offs in the environment of semantic annotations with keywords, attributes and natural language. This paper focuses mainly on developing a nested, unambiguous annotation language in the proposed data model with a great descriptive power and less computational complexity. Also presented herein are the corresponding query language and processing algorithms.

In the following, we present our main approach: the object net and the event hierarchy. The concept of domain knowledge, which favors the annotation of video data and the capacity of query processing, is then introduced. Assume that all objects and events are recognized only by their own identifiers.

3.2. The object net

The properties of each object can be divided into two categories: *static* and *dynamic*. A static property implies that its values are fixed and a dynamic property's values vary over time. In other words, an object's static attributes are shared in all events and an object's dynamic attributes are described in each event. An object net attempts to represent static parts of video objects, including static values and static relationships. These static properties are inherent and are independent of time.

Definition 1. A *video object* (*or object*) is a 4-tuple (*OID*, *DID*, *PT*, *I*), where
- *OID* is the unique object identifier.
- *DID* the domain identifier referring to the object's domain.
- *PT* denotes the stored data and is represented as a set of *static properties*. A *static property* is defined as ⟨*R*, *DC*⟩, where *R* is the property's name and *DC* denotes a set of descriptive components. A descriptive component is defined as ⟨*DID*, *V*⟩, where *DID* is the values' domain identifier and *V* a single or a set of values. A value is an *OID*, an *EID* (event identifier), a static property, or an atomic value, such as integer, real number and string.
- *I* denotes the *frame sequence* of video frames in which the object appears.

According to Definition. 1, each object belongs to a domain with a domain identifier. A static property's values support nested structures and set values. A value's domain serves as its data type. Most schemaless semantic-based systems [1,9,21,24,26,31] are limited in terms of the computational ability of values owing to the absence of definitions of operations. Different from conventional programming languages, each property may contain multiple domains. The frame sequence reveals that every object is temporary and the video program limits its lifetime. Regarding this annotation language, Appendix A(a) presents an unambiguous CFG to parse the static content of objects and to avoid the semantic confusion. An angle bracket refers to a non-terminal symbol.

An object net is formed when objects in a group are related. An object net is a directed graph in which each arrow denotes the direction of a reference. Composite references are not defined in Definition. 1 since we assume that the query processor is responsible for supporting integrity constraints, including dependent shareable reference, dependent exclusive reference, independent shareable reference and independent exclusive reference. Based on the CFG, Fig. 1(a) presents an example of the object Tom. Fig. 6 displays its domain knowledge. Intuitively, all the properties described in object *Tom* are static, such as name, birthday and sex. Similarly, according to Fig. 1(b), all the references revealed in the object net are also invariable throughout video programs. Each property's values may belong to different domains, such as the property *Picture*, which can be represented as pcx or bmp formats. Other useful visual properties of an object include the cinematic parameters and features of colors, shapes, sizes, positions and textures. These low-level image features are then mapped to the object's high-level semantics.

Object ID: Oid_20
Domain ID: student
{Name: (string (Tom)).
Birthday: (date (Year: (int (1972)).
Month: (int (2)).
Day: (int (10)).).
Sex: (string (Male)).
Height: (cm (185)).
Weight: (kgw (80)).
Hobby: (string (swimming, jogging)).
Picture: (pcx (Oid_801);
bmp (Oid_802)).
Voice: (mp3 (Oid_102)).
Major: (cs (Oid_32)).
Lab: (lab (Oid_54)).
Video Source: (video (Oid_2100)). }
Frame Sequence: [1,1200], [5600,8020], [10216,12180], [21088,23100]

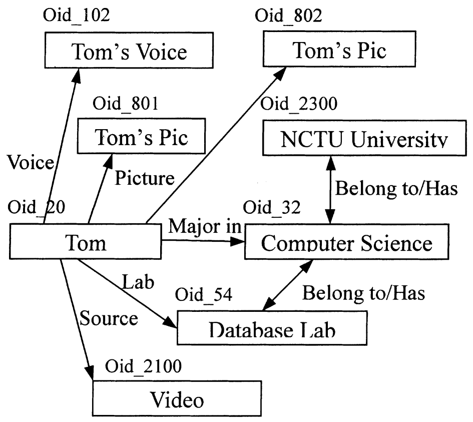


Fig. 1. An example of: (a) an object and (b) an object net.

3.3. The event hierarchy

An event hierarchy attempts to develop a mechanism capable of describing multi-level abstractions and representing complex scenarios. In particular, this hierarchy concerns itself with two aspects of a semantic relationship: inheritance of attributes by context and conditional dependence between two events. For any event which involves the interplay among objects, it not only contains the event's inherent attributes but also describes dynamic properties of the participant objects.

Definition 2. An *event* is a 5-tuple (*EID*, *DID*, *PT*, *CPT*, *I*), where
- *EID* is the unique event identifier.
- *DID* the domain identifier referring to the object's domain.
- *PT* denotes the stored data and is represented as a set of *properties*. A *property* is defined as ⟨*R*, *DC*⟩, where *R* is the property's name and *DC* denotes a set of descriptive components. A descriptive component is defined as ⟨*DID*, *V*⟩, where *DID* is the values' domain identifier and *V* a single or a set of value. A value is represented as ⟨(*VID*), *SV*⟩, where *VID* is the value's identifier (the bracket means it may be omitted), and *SV* a single value, including an *OID*, an *EID*, a *VID*, a 2-tuple ⟨*OID*, *DP*⟩, a *property* and an atomic value, such as an integer, real number, and string. For each 2-tuple ⟨*OID*, *DP*⟩, *DP* denotes a set of the dynamic properties of the object with the identifier *OID* and can be treated as a 2-tuple *property* ⟨*R*, *DC*⟩. *CPT* denotes a conditional probability table and is represented as ⟨*C*, *ET*⟩, where *C* denotes an ordered list of its children's *EIDs* and *ET* denotes a set of conditional probability. Each conditional probability is a function mapping from integer to [0, 1] and defines the probability distribution, as listed in Table 1.
- *I* denotes the frame sequence of video frames in which the event appears.

Table 1
An example of CPT of the event *Lecture*

No.	Introduction	Talk	Discussion	Probability (Lecture)
0	True	True	True	1.0
1	True	True	False	0.8
2	True	False	True	0.2
3	True	False	False	0.1
4	False	True	True	0.8
5	False	True	False	0.7
6	False	False	True	0.1
7	False	False	False	0.0

According to Definition. 2, an event clusters a collection of objects, which interact with each other. Each event only needs to describe dynamic properties of the participant objects. Static information of those objects is stored in an object net and is referred as OIDs. Appendix A(b) illustrates an unambiguous CFG for the annotation of an event's properties. From the event *Lecture* in Fig. 2, we can infer that *student Tom* (Oid_20) *and student Alan* (Oid_40) *presented ch.* 3 of the *book* "*Video Database Systems*" (Oid_51) in the classroom 130 (Oid_63) from 9:10 a.m. to 12:00 a.m. on 30 March 1998. That lecture topic was "*Video Databases*" *Tom presented* his work with *slices* and *demonstrated his program* (Oid_24) with a *computer*. *Alan presented* his work with *slices nervously*. Notably, some static information is derived from the corresponding object net, such as the book name, the classroom, and Tom's program. Similar to an object, low-level image features must also be attached to an event.

To capture the feature of multi-level abstractions of video data, we organize all events into an event hierarchy, as depicted in Fig. 3. An arbitrary video

```
Event ID: Eid_30
Domain ID: lecture
{Topic: ( string (Vid_0, Video Databases)).
 Speaker: ( student
                (Vid_1, Oid_20,                              // Tom
                    (Action: (present
                                    (Method: (string (Slice)).
                                     Content: (book (Vid_3)). ));
                               (demo
                                    (Method: (string (Computer)).
                                     Content: (program (Oid_24)). )).
                     State: (string (Normal)) )),
                (Vid_2, Oid_40,                              // Alan
                    (Action: (present
                                    (Method: (string (Slice)).
                                     Content: (book (Vid_3)). )).
                     State: (string (Nervous)) ))).
 Content: (book (Vid_3, Oid_51, (Chapter: (int (3))) )).
 Location: (location (Vid_4, Oid_63)).
 Time: (time_interval (From: (time (9:10 a.m.)).
                     To: (time (12:00 a.m.)) )).
 Date: (date (Year: (int (1998)).
             Month: (int (3)).
             Day: (int (30)). )).
 Video Source: (video (Oid_2100)). }
Conditional Probability Table: ( {Eid_31, Eid_32, Eid_33},
          {(0, 1.0), (1, 0.8), (2, 0.2), (3, 0.1), (4, 0.8), (5, 0.7), (6, 0.1), (7, 0.0) })
Frame Sequence: [10,2300], [4600,4800], [8016,10180], [13300,16020]
```

Fig. 2. An example of the even *Lecture*.

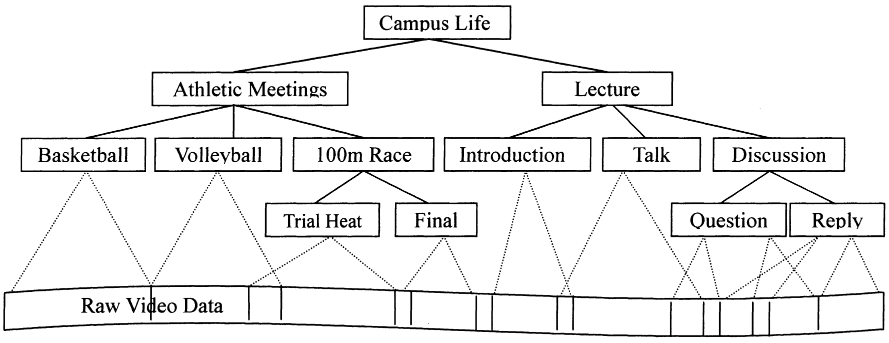


Fig. 3. An example of the event hierarchy about *Campus Life*.

segment can be mapped into a set of annotations according to different viewpoints. An event hierarchy is a graph in which the children of an event have links connected to it. The meaning of a link between a parent E1 and its child E2 involves the following: E2 directly influences E1; E1 has an independent and shareable composite reference to E2 implicitly; and E2 may inherit the properties of E1.

To describe the conditional dependence relationship among events, we maintain a *conditional probability table* (*CPT*) for each event. Table 1 presents the CPT of event *Lecture*. If the number of children of an event is n, then its CPT has 2^n entries. Consider the example in Fig. 3. Three events (*Introduction, Talk and Discussion*) constitute a composite event *Lecture*. The second row in Table 1 shows the following semantics:

$$Probability(Lecture|Introduction, Talk, \neg Discussion) = 0.8.$$

From the querying viewpoint, an event hierarchy can be treated as a *Bayesian network* defined in [27]. The following situation is examined in which an event hierarchy holds the rules of Bayesian networks:

1. the nodes of an event hierarchy can be viewed as a set of Boolean random variables with the domain ⟨True, False⟩;
2. as a link between two events can be treated as an arrow pointing to a parent from a child, an event hierarchy corresponds to a DAG, which denotes how children influence their parents and
3. each event contains a CPT, which stores the information of conditional probabilities.

To effectively display the multi-level abstractions of video data, an event hierarchy must provide the mechanism of inheritance of descriptions by context for sharing common information among meaningful scenes [24,26,31]. This mechanism reduces the amount of annotations for events with a low-level abstraction. According to Fig. 4, a plus-marked attribute can be inherited and vice versa. Inheritance by context is only adapted when the inheritable properties in an event and all its children have the following relationships: *generalization, composition or identification.* For example, consider the properties *Location and Topic* in event *Lecture* and all its children. *Modern Database, Video Database and Mobile Database* are kinds of *Database* (generalization) and *Room* 130 *and Room* 132 are parts of *CS Hall* (composition). Event *Introduction* omits the location *CS Hall* (identification). Consider a situation in which someone poses the following query: *find all the events, where Student Tom is in the location Main Campus*, the result is event *Talk* 1. Notably, no domain knowledge is required to inform a query processor that *Room* 130 is a part of *Main Campus*. In addition, it is inappropriate to inherit some properties. For example, the attribute *Chairman* in event *Lecture* only occurs in one of its children.

However, consider a situation in which someone wants to identify all the events where *Student Tom* or *Student Alan* is located in *CS Hall*. Under this circumstance, we obtain three events (*Introduction*, *Talks* 1 *and* 2) as a result. However, returning to event *Lecture* would appear to be more appropriate. The event *Lecture* involves high-level concepts (*CS Students*) but loses some important information (*Student Tom and Student Alan*) of its children. Therefore, the mechanism of inheritance by context is limited in terms of the capacity of content-based retrieval. Most related investigations only address how an event impacts its descendants without discussing the above situation. With respect to querying relative events, the conventional approach examines inheritance by context, thereby having two major disadvantages:

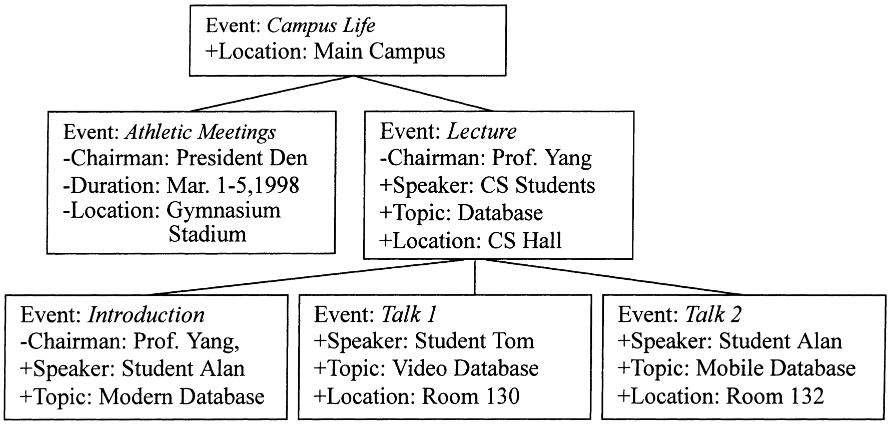


Fig. 4. A detailed example of the event hierarchy about *Campus life*.

(1) *User level*: Users who are querying relevant events can generally not accurately describe in detail the contents of the event they are interested in. Individuals may merely remember the important, apparent part of objects or motions in an event. That is, video data models are required to assemble several salient parts into a more complete concept. However, the conventional method lacks the ability to draw inferences from events. For vague or inexact queries, the conventional method also fails to yield more complete results.

(2) *System level*: To save storage space, video data models attempt to describe high-level semantics to represent larger events and vice versa. Restated, large events are only for a common description, and only small events are described in detail. Even if users can offer accurate queries, video data models still have to possess the ability of inference, which is facilitative in obtaining an acceptable outcome from simplified annotations. This ability is not only effective for semantic-based comments, but also helpful for audio-visual content of video data owing to limitations of image processing technologies. A complete annotation of video data is impossible. With the assistance of inference engines, a new result can be deduced from the audio-visual features already captured or from the text-based comments already generated.

The following considers two kinds of properties in an event: *inheritable* and *not inheritable*. All of its descendants inherit the inheritable properties in an event. To solve the above problem, we apply the data structure of Bayesian networks to event hierarchies. Bayesian networks based on Bayes' theorem compute the posterior probability distribution for a set of query variables, given some evidence variables. An evidence variable or a query variable corresponds to an event. When users pose a query, an event hierarchy propagates inheritable information top-down first and then evaluates each event's probability bottom-up. While considering the inference between an event and its children, this method significantly improves the approach of inheritance by context. This inference mechanism is also applied to audio-visual features. In addition, the determination of a CPT for each event is application-dependant and can be generated either automatically or manually. Later, two formulas are introduced to produce each event's conditional probabilities to experiment with our approach. Moreover, detailed algorithms and experiments will be stated in Sections 4.2 and 4.3

3.4. Domain knowledge hierarchy

Oomoto [26] introduced two difficulties encountered in defining attributes and querying. To facilitate query processing and annotation of video data, this work proposes the concept of a domain knowledge hierarchy as illustrated in Fig. 5.

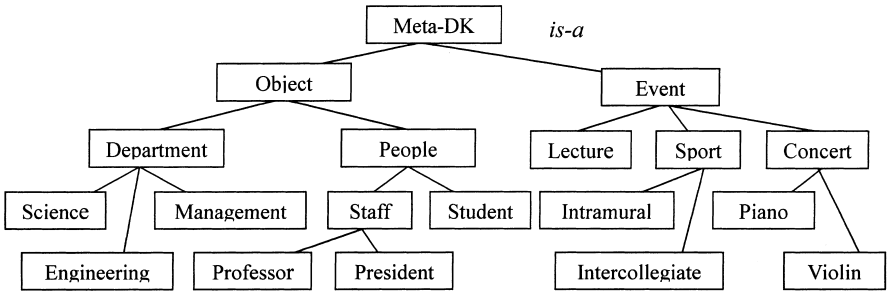


Fig. 5. An example of the DKH about *University Campus.*

Definition 3. A *domain knowledge hierarchy* (*DKH*) is a 3-tuple (*D*, *ISA*, *T*), where

- *D* is a countably infinite set of *domain knowledge* (*DK*). Each domain knowledge is represented as a 4-tuple ⟨*DID*, *PR*, *DT*, *RL*⟩, where *DID* is the unique domain identifier, *PR* denotes a set of properties, *DT* denotes a set of shared data and *RL* is a set of if-then rules. A property is defined as ⟨*P*, *CD*⟩, where *P* is the property name and *CD* a set of candidate domain knowledge ($CD \subseteq D$). Shared data is defined as ⟨*P*, *V*⟩, where *V* is a set of values. Operators can be treated as shared data.
- *ISA* denotes the generalization (*is-a*) relationship among *DK*s and is a function mapping *D–D* ($ISA \subseteq D \times D$). For each *s* and $t \in D$, if *s* is a child of *t*, we denote it as "s ISA t" and the properties of *t* may be inherited by *s*.
- *T* is a special *DK* ($T \in D$) and called *Meta-DK*. It denotes the root of a *DKH*.

According to Definition 3, a DK can represent the properties of a domain. Each DK inherits the properties of its parents by default and consists of properties, shared data, operators and if-then rules. A property describes certain aspects of a domain; its values must also be the instances of its candidate DKs. All instances of a DK share sharable data, such as linguistic values. Operators compute the domain's instances and sharable data. If-then rules are responsible for evaluating the type of fuzzy query and connecting high-level semantics to low-level audio-visual features. Detailed descriptions will be stated in Sections 4.4 and 5, respectively. Fig. 6 presents an illustrative example of the DKs *Student and Lecture.*

Domain knowledge focuses mainly on achieving the following goals: (1) to serve as the source of properties of objects/events, (2) provide the computing ability of values, (3) support taxonomies, (4) describe the shared data among common objects/events, (5) facilitate query processing and (6) bridge the gap between different abstraction levels. Notably, users can also attach arbitrary properties to objects/events if necessary. This is despite the fact that they can define their properties differently from those of its domain knowledge. Herein,

Domain Id: student
Property:

Static	Dynamic
Name: {String}	Action: {Action}
Birthday: {Date}	State: {State}
Picture: { Pcx, Bmp }	
Voice: { Mp3, Wav }	
....	

Operator:
int Age();
Rule:
If weight is heavy then shape is fat.
Sharable Data:
Weight: { Heavy, Light }
Shape: { Fat, Thin }
Heavy: { membership function of Heavy }
Fat: { membership function of Fat}

Domain Id: lecture
Property:
Speaker: { Professor, Student }
Content: { Book, Paper }
Location: { Classroom, Hall }
Time: { Time_Interval }
Date: { Date }
Operator:
Display();
Sharable Data:
Time: { Long, Short }
Long: { membership function of Long}
Short: { membership function of Short}

Fig. 6. An example of domain knowledge: *Student and Lecture.*

we assume that a DKH is given in advance by users who are familiar with the application domain, such as directors. The simplest structure of a DKH consists of three DKs: *Meta-DK*, *Object* and *Event*. Completely building a DKH at one time is unnecessary. Instead, arbitrary DKs can be added incrementally in a DKH. Notably, limiting the context or the application allows us to reduce the complexity of building a DKH.

3.5. Discussion

Although object recognition in an unconstrained domain is still limited by the lack of progress in image processing technologies, the advent of new video compression standards relying on content-based representation of visual data is helpful in extracting visual objects from the encoded video streams, such as MPEG-4 [34]. At present, most researches use visual features and motion information to detect and trace objects such as human faces [4]. Regarding scene change detection, several algorithms have been developed for compressed and uncompressed video [13,18]. Therefore, our data model is targeted at partial automation of text-based annotation and full automation of feature-based annotation.

4. Query language and processing

In this section, we develop a SQL-like query language for retrieving objects, events and video frames. All the content-based queries can be categorized as semantic query, temporal query, similar query, fuzzy query and hybrid query. For each query, this section provides some examples and corresponding query language, as well as describes how to process these queries with the assistance of indices and domain knowledge. The complexity of each query depends mainly on its index structures.

4.1. Query language

The query language for our data model consists of three clauses:

- *Select clause*: This clause specifies the properties of the entities whose values are to be retrieved and the maximum number or the probability threshold of returned results. The keyword *RELATIVE* is only appropriate for querying relevant events, where query processors will infer answers from the conditional dependence relationship among events. For example, the statement, *select RELATIVE lecture.topic* (5), is to output the topics of top five relative lectures resulting from the inference.
- *From clause*: This clause specifies domain names, domain variables and their frame sequences required to process the query. For example, the statement, *Video V*[1000,3000], reveals that domain variable *V* belongs to domain *video* and its frame scope is *from* 1000 *to* 3000. Notably, objects and events are also temporal.
- *Where clause*: This clause specifies conditional expressions, consisting of the set of domain variables defined in the From clause, a set of Boolean operators (AND, OR and NOT), and a set of temporal operators, including the 13 relations of two given intervals [25]. The simplified syntax of a conditional expression is defined in Appendix B.

This query language makes use of domain names as inputs because users are normally familiar with the domains of interesting things. A more natural approach would be for users to pose a query of *find out all the people named Tom* than to ask a question of *find out all the entities named Tom.* In addition, each type of content-based query referred to hereinafter can be identified by the format of its query language. This format helps query processors analyze given inputs. Table 2 lists the notations for facilitating the descriptions of query language and processing.

4.2. Semantic query

A semantic query inquires the information about objects, events, and their relationships. This query is further classified into several more detailed queries as follows:

Elementary domain query. This query finds all objects/events of the same domain.

Example: Find out all professors in the video *campus.*

Query lang:

Table 2
The notations defined in query language and processing

Notation	Description
RVD	The set of all the video sources, which describe bibliographic data, such as video title/name, video file name, total frame number, and so on.
$v.EVT$	The set of all the events in the video v, $v \in RVD$
$\prec_{v.E}$	A function mapping v.EVT to v.EVT, $\prec_{v.E} \subseteq v.EVT \times v.EVT$. e1 $\prec_{v.E}$ e2 means e1 is a child of e2
$v.OBJ$	The set of all the objects in the video v, $v \in RVD$
$v.DM$	The set of all the domains in the video v, $v \in RVD$
$\prec_{v.D}$	A function mapping v.DM–v.DM, $\prec_{v.D} \subseteq v.DM \times v.DM$. d1 $\prec_{v.D}$ d2 means d1 is a child of d2
$X.p$	The values of *X*s property p, where $X \in v.EVT \cup v.OBJ \cup v.DM \cup RVD$. For example: $X.i$: *X*'s identifier, $X \in v.EVT \cup v.OBJ \cup v.DM$. $X.d$: *X's* domain, $X \in v.EVT \cup v.OBJ$. $X.f$: *X*'s frame sequence, $X \in v.EVT \cup v.OBJ$. $X.s$: the set of instances contained in the X, $X \in v.DM$

```
Select O.name
From Video V, Professor O
Where V CONTAIN O
    AND V.name = "campus"
```

Algorithm *DomainQuery*(*RVD, Vname, Dname, Pname*)
Input: video source *RVD*, video name *Vname*, domain name *Dname*, property name *Pname*.
Output: the values of the property *Pname*

```
Step 1. Locate video $v$, $v \in RVD$ and $v$.name = $Vname$
Step 2. Locate domain $d$, $d \in v.DM$ and $d$.name = $Dname$
Step 3. Set $R$ to be an empty set, $R = R \cup \{d\}$
Step 4. For each $r \in R$
    If $\exists d', d' \in v.DM,\ d' \notin R,$ and $d' \prec_{v.D}\ r$
    then $R = R \cup \{d'\},$ repeat Step 4
Step 5. For each $r \in R$
    For each $t \in r.s$
        return $t.Pname$
```

This algorithm must search for the domain identifiers of the given domain and all its descendants in the DKH for the links among DKs represent *is-a* relationships. By doing so, the bodies of objects/events can be acquired by an access method, which translates meaningless identifiers into their disk addresses. Several conventional index structures facilitate the Steps 1 and 2, such as B+ tree and hashing. We recommend that each DK keep all of its instances.

Elementary content query. An elementary content query finds all objects/events whose properties satisfy given conditions independent of object/event identifiers.

Example: Find all video frames where student *Tom* appears and find all lectures that are held in the location *130* in the video *campus*.

Query lang:

```
Select O.f                          Select E.name
From Video V, Student O             From Video V, Lecture E
Where V CONTAIN O                   Where V CONTAIN E
        AND V.name = "campus"               AND V.name = "campus"
        AND O.name = "Tom"                  AND E.location = 130
```

Algorithm *ContentQuery*(*RVD, Vname, Dname, ,Clist, Pname*)

Input: video source *RVD*, video name *Vname*, domain name *Dname*, conditional list *Clist*, property name *Pname*

Output: the values of the property *Pname*

```
Step 1. Call QueryDomain($RVD$, $Vname$, $Dname$, Identifier) to get a set R
Step 2. For each $r \in R$,
   Compute $prob(r)$ based on Clist, where $prob()$ can be treated as the similarity function defined in Section 4.3
Step 3. Rank $prob(r)$ for all $r \in R$. If $prob(r) > 0$ then return $r.Pname$
```

Similar to conventional databases, indexing on the properties involved in the given conditions is able to improve query processing. However, if a condition contains complex scenarios in an event, we need to deal with the issue of matching annotation structures with query forms.

Object-event relation query. This query corresponds to elementary content queries except that their conditions must be associated with object/event identifiers. These queries attempt to find the objects in a given event or to find the events in which a given object appears.

Example: Find the students who are playing basketball and find all the events in which student *Tom* appears in the video *campus*.

Query lang:

```
Select O.name                       Select E.name
From Video V, Student O, Sport E    From Video V, Student O, Event E
Where V CONTAIN E                   Where V CONTAIN E
        AND E CONTAIN O                     AND E CONTAIN O
        AND V.name = "campus"               AND V.name "campus"
        AND E.name = "basket-               AND O.name = "Tom"
ball"
```

Algorithm ObjEvtRelationQuery(*RVD, Vname, Ename, Oname, Eclist, Oclist, Pname, Flag*)

```
Input: video source $RVD$, video name $Vname$, event 's domain name $Ename$, object's domain name $Oname$, event's conditional list $Eclist$, object's conditional list $Oclist$, property name $Pname$, choice $Flag$
Output: the values of the property $Pname$
Step 1. Locate video $v$, $v \in RVD$ and $v$.name = $Vname$
Step 2. If $Flag$ = True then goto Step 8 (find the events in which a given object appears)
Step 3. Call QueryContent($RVD$, $Vname$, $Ename$, $Eclist$, Identifier) to get a set R
Step 4. For each $r \in R$
    If $\exists d, d \in v.EVT,\ d \notin R,$ and $d \prec_{v.E} r$
    then $R = R \cup \{d\}$, repeat Step 4
Step 5. Set $S$ to be an empty set
    For each $r \in R$,
    Locate the set of objects $O \in r$, $S = S \cup O$
Step 6. For each $s \in S$
        If $s.d = Oname$ then compute $prob(s)$ based on $Oclist$
Step 7. Rank $prob(s)$ for all $s \in S$. If $prob(s) > 0$, then return $s.Pname$
Step 8. Call QueryContent($RVD$, $Vname$, $Oname$, $Oclist$, Identifier) to get a set $R$
Step 9. Set $S$ to be an empty set
    For each $r \in R$
      Locate the set of events $E$ in $r$, $S = S \cup E$
Step 10. For each $s \in S$
    If $\exists\ d, d \in v.EVT,\ d \notin S,$ and $s \prec_{v.E} d$
    then $S = S \cup \{d\}$, repeat Step 10
Step 11. For each $s \in S$
        If $s.d = Ename$ then compute $prob(s)$ based on $Eclist$
Step 12. Rank $prob(s)$ for all $s \in S$. If $prob(s) > 0$, then return $s.Pname$
```

Based on users' annotation, this algorithm takes into account the composition relationship among events. Applying temporal queries can get a more precise result. In other words, the above example can be represented as:

```
Select O.name
From Video V, Student O, Sport E
Where V CONTAIN E AND V CONTAIN O
    AND V.name = "campus" AND E.name "basketball"
    AND E INTERSECT O.
```

Object path query. This query finds all objects with a particular relation to a given object.

Example: Find all students who are members of the *database* lab which belongs to the *CS* department in the video *campus*.

Query lang:

```
Select O.name
From Video V, Student O
Where V CONTAIN O
    AND V.name = "campus" AND O.lab.belong_to = "CS"
```

Algorithm *ObjectPathQuery(RVD, Vname, Dlist, Opath, Clist, Pname)*

Input: video source *RVD*, video name *Vname*, domain set *Dlist*, path expression *Opath*, conditional list *Clist*, property name *Pname*

Output: the values of the property *Pname*

```
Step 1. For each domain $d_i \in Dlist$
            Call DomainQuery($RVD$, $Vname$, $d_i$, Identifier) to get a set $R_i$
Step 2. Join all the $R_i$ into a set $R$ according to $Opath$
Step 3. For each $r \in R$
            Compute $prob(r)$ based on $Clist$
Step 4. Rank $prob(r)$ for all $r \in R$. If $prob(r) > 0$, then return $r.Pname$
```

Fig. 1(b) illustrates the path expression of above example. An object path query has difficulty in evaluating join operations (Step 2) efficiently if a query path has no indices. Several index organizations proposed to support object-oriented databases are also appropriate for processing this query, such as multi-index, join index, nested index and path index.

Event inference query. This query finds all events that are relative to given predicates.

Example: Find all the sports where student *Tom* is in the location *gym* in the video *campus*.

Query lang:

```
Select RELATIVE E.name
From Video V, Sport E, Object O
Where V CONTAIN E AND E CONTAIN O
    AND V.name = "campus" AND O.name = "Tom" AND E.location = "gym"
```

Algorithm *EventInferenceQuery(RVD, Vname, Ename, Oname, Eclist, Oclist, Pname)*

Input: video source *RVD*, video name *Vname*, event 's domain name *Ename*, object's domain name *Oname*, event's conditional list *Eclist*, object's conditional list *Oclist*, property name *Pname*

Output: the values of the property *Pname*

```
Step 1. Call ContentQuery($RVD$,$Vname$,$Ename$,$Eclist$,Identifier) or call ObjectEventRelationQuery($RVD$,$Vname$,$Ename$,$Oname$,$Eclist$,$Oclist$,Identifier,True) to locate a set of events $R$ and corresponding probabilities.
Step 2. For each $r \in R$
            If $\exists\ d,\ d \in v.EVT, d \notin R$, and $(\ r \prec_{v.E} d\ or\ d \prec_{v.E} r\ )$
            Then $R = R \cup \{d\}$ and set $prob(d) = 0$
Step 3. Locate a topological order $\langle r_1\ ,\ r_2\ ,\ \ldots\ ,\ r_q \rangle$ of all the events in $R$, where a link between two events is treated as an arrow pointing to a parent from a child.
Step 4. For $i = 1$ to $q$
        If $r_i$ is not a lowest-level event in $v$.E
        Then prob$(r_i)$ = max$(prob(r_i), \sum_u prob(r_i|u) * \prod_j prob(u_j))$,
            where $u$ is the vector of children of $r_i$, $\langle u_1, u_2, \ldots \rangle$
Step 5. Rank $prob(r)$ for all $r \in R$, If $prob(r) > 0$, then return $r.Pname$.
```

For each event inference query, a set of evidence events can be found by making use of conventional index structures, and then, applying the causal inference algorithm [27] of Bayesian networks to evaluate the probabilities of the other events in an event hierarchy. Other distinct kinds of inferences in [27] are not adopted herein because Bayesian networks are only used for determining how children influence their parents.

4.3. The experiment

To examine the impact of the event inference approach on content-based video retrieval, we experiment with our method (integration of conditional probability dependence and inheritance by context) and the conventional method (only inheritance by context). Consider a video database with a set of events, denoted as $EVT = \{e_1, e_2, \ldots, e_N\}$, where $N \geqq 1$. These events are formed as a multi-level event hierarchy. Each event e_i has a term vector, $t_i = \langle\ tf_{i1}, tf_{i2}, \ldots, tf_{iM} \rangle$, and a weight vector, $w_i = \langle w_{i1}, w_{i2}, \ldots, w_{iM} \rangle$, where M is the total number of terms. tf_{ij} and w_{ij} represent the frequency and the weight of term j in event e_i, respectively. A term can be viewed as a keyword or an audio-visual feature. The derived weight vector of a high-level event can be defined as:

$$w_i = T(\{w_j | e_j \in S_i\}),$$

where S_i the set of descendants of event e_i and T a transfer function mapping the set of weight vectors of e_i's descendants to a weight vector. For audio-visual features, T may be a mechanism of key-frame selection [12,24] or a computation of arithmetic mean [8]. For text-based annotation, T represents the relationships of generalization, composition and identification discussed in Section 3.3. This transfer function helps derived weights capture high-level semantics but results in the loss of detailed information. Herein, information retrieval technique is applied as the basis of judging the retrieval performance. For the composition relationships of multi-level abstraction, the derived term frequency of event e_i satisfies the following condition:

$$tf_{ij} = \sum_{e_k \in S_i} tf_{kj}.$$

The weight value of weight vector of event e_i is defined as [6]:

$$\frac{W_{ij} = tf_{ij} * idf_j}{\sqrt{\sum_{k=1}^{M} (tf_{ik} * idf_k)^2}} \quad \text{and} \quad idf_j = log\left(\frac{N}{n_j}\right),$$

where idf_j denotes the inverse document frequency of term j and n_j the number of events in which term j appears. In an ideal situation, above formula evaluates weight vectors of all the events. However, in a real situation, a transfer function T computes the weights of non-lowest-level events. Given a query vector $Q = \langle q_1, q_2, \ldots, q_M \rangle$, the similarity between w_i and Q is defined as:

$$\mathrm{Sim}(w_i, Q) = \sum_{k=1}^{M} w_{ik} * Q_k \Bigg/ \sum_{h=1}^{M} Q_h,$$

where the denominator generates normalized term similarities between 0 and 1 so as to satisfy the condition of probabilities. The conditional probability dependence among event e_i and all its children C_i is computed as:

$$\mathrm{Prob}(e_i | T_i, \neg(C_i - T_i)) = \sum_{e_j \in T_i} \sum_{k=1}^{M} tf_{jk} * idf_k \Bigg/ \sum_{e_j \in C_i} \sum_{k=1}^{M} tf_{jk} * idf_k \tag{1}$$

or

$$\mathrm{Prob}(e_i | T_i, \neg(C_i - T_i)) = \min\left(1, \sum_{j=1}^{M} \frac{idf_j * \sum_{e_k \in T_i} tf_{kj}}{\sqrt{\sum_{n=1}^{M} \left(idf_n * \sum_{e_k \in T_i} tf_{kn} \right)^2}} * W_{ij} \right), \tag{2}$$

where T_i is a subset of C_i denoting all the true evidence events of C_i.

Consider an event hierarchy with five levels. The degree of fan out of a node is three. The total numbers of terms (M) and events (N) are, respectively, 250 and 121. Each lowest-level event contains 10% of M terms uniformly distributed in its term vector and each term frequency is in uniform distribution varied between 1*base_freq and 10*base_freq. The base_freq denoting the minimum number of continuous frames in which a term appears is set to be 15.

We evaluate all the similarity values for four approaches: the ideal approach, the integrated approach (combination of conditional probability dependence and inheritance by context), using Eq. (1), the integrated approach, using Eq. (2), and the conventional approach (only inheritance by context). Suppose that the ranked results of the four approaches are, respectively, F_{ideal}, $F_{inference1}$, $F_{inference2}$, and $F_{conventional}$. Each result in a decreasing order is a permutation of events whose similarity values are not equal to zero. We examine the retrieval effectiveness according to two parameters, recall (R_k) and precision (P_k), where k is a threshold varying between 1 and N. R_k and P_k are defined as:

$$\begin{aligned} R_k(X) = |\{&\text{the top } k \text{ events in } F_x\} \\ &\cap \{\text{the top } k \text{ events in } F_{ideal}\}|/|\{\text{the top } k \text{ events in } F_{ideal}\}| \end{aligned}$$

and

$$\begin{aligned} P_k(X) = |\{&\text{the top } k \text{ events in } F_x\} \\ &\cap \{\text{the top } k \text{ events } F_{ideal}\}|/|\{\text{the top } k \text{ events in } F_x\}|, \end{aligned}$$

where X can be denoted as inference 1, inference 2 or conventional. We ran the simulation with 1000 iterations randomly. Fig. 7 how the recall and precision results. The experimental results show that the proposed integrated policies outperform the existing ones substantially in maximizing the criteria of recall and precision. The performance orders in both criteria are as follows: *Inference 2 > Inference 1 > Conventional approach.* According to Fig. 7, the recall of the conventional policy is bounded by a lower value even when k is close to N. The main reason for this phenomenon is that high-level events lack of the detailed information. Query processor cannot retrieve them by searching their partial features arbitrarily. In addition, Fig. 7 shows that not only the proposed policies get a better overall recall but the permutations of their results are similar to that of the ideal case. That is, the better an event satisfies query

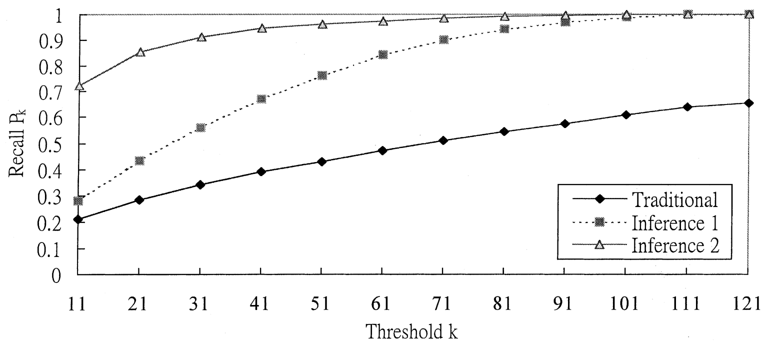


Fig. 7. Recall for traditional and interference approaches.

conditions, the larger its similarity value. Consider the approaches of *Inference 1* and *Inference 2*. *Inference 1* adopts Eq. (1), which takes into account the importance of an event compared with its siblings. *Inference 2* applies Eq. (2), which takes the similarity between an event and its parent. The results show that Eq. (2) is more appropriate for presenting conditional probabilities in practice. Fig. 8 indicates that the integrated approaches have better precision in average. Hence, from the experimental results, combining the concept of Bayesian networks and inheritance by context significantly improve the capability of content-based retrieval.

4.4. The other video queries

In addition to semantic queries, the proposed model and language support other four video queries, including temporal query, similar query, fuzzy query and hybrid query. This subsection introduces these queries by providing some examples and corresponding query language. Different from semantic queries, the conditional expressions of these queries contain particular information such as temporal information, low-level visual features and vague linguistic terms. Query processing of temporal, similarity and fuzzy queries is similar to that of elementary content query. The major difference is that they require different similarity measures to evaluate the probability values of the ContentQuery algorithm mentioned in Section 4.3. Consequently, we simply bring out the similarity functions instead of the complete algorithms.

Temporal query. *Temporal queries* view temporal information as inputs. Temporal information is either a range of video frames or one of the 13 relations of two intervals [25].

Example: Find out all the speeches in the commencement in the frame interval [3000,9000] of the video *campus*.

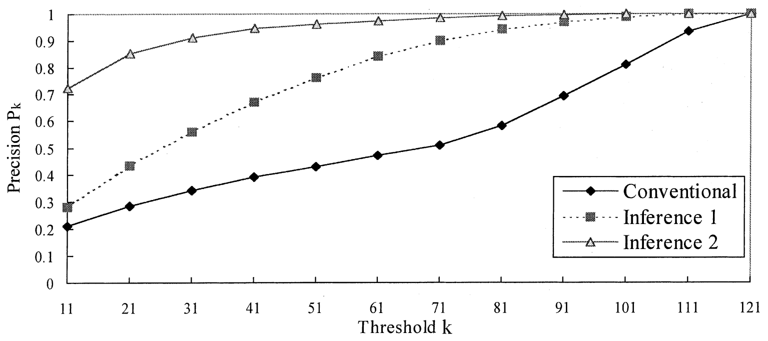


Fig. 8. Precision for conventional and inference approaches.

Query lang:
Select E.name
From Video V[3000,9000], Event E_1, E_2
Where V CONTAIN E_1 AND V CONTAIN E_2 AND V.name = “campus”
AND E_1.name = “speech” AND E_2.name = “commencement”
AND E_1 **DURING** E_2

In this work, a two-dimensional index structure is used to evaluate temporal queries, such as R*-tree, SS-tree and SR-tree [2,20]. For each frame interval [*i*, *j*] of a given object/event, the coordinate (*i*, *j*) is a key inserted into a two-dimensional index structure. Hence, a temporal query can be viewed as a range query in a two-dimensional space.

Similar query. *Similar queries* find the audio-visual features of video data that resemble the given examples. The low-level features are generally queried by examples.

Example: Find out the name of the student in a given picture in the video *campus*.

Query lang:
Select O.name
From Video V, Student O
Where V CONTAIN O AND V.name = ”*campus*” AND O.picture ≈

The similarity between two images can be evaluated according to several visual features such as color histogram, texture, and sketch. The operators of a DK define similarity measures. Consider the DK *Student* in Fig. 6 whose property *Picture* belongs to the DK *Pcx* or *Bmp*. Assume that *Pcx* and *Bmp* are image formats. The DK *Image* can define a similarity operator (≈) to compare the color histograms of two images I_1 and I_2 as follows:

$$\mathrm{Sim}(I_1,I_2) = \sum_{i=1}^{N}\sum_{j=1}^{N} a_{ij}(H(I_1,i) - H(I_2,i))(H(I_1,j) - H(I_2,j)),$$

where N is the number of colors and $H(I,i)$ the number of pixels of color i in the image I and a_{ij} the similarity coefficient between color i and j.

Similar to temporal queries, the feature vector of an object/event can be indexed by a multi-dimensional tree, which has been extensively applied to retrieve an image content [14,32]. Similar queries can be treated as the nearest neighbor queries in a multi-dimensional space. In addition, we combine graphical user interfaces with SQL-like query language, thereby allowing users to simultaneously query semantic content of video data and indicate visual features and spatial relations of video objects at the same time.

Fuzzy query. *Fuzzy queries* correspond to elementary content queries except that their predicate contains vague linguistic terms.
Example: Find out all the students who are fat in the video *campus*.
Query lang:
Select O.name
From Video V, Student O
Where V CONTAIN O AND V.name = "*campus*" AND O.shape $\approx$ fat.

Query processor evaluates fuzzy queries by applying elementary domain queries with the assistance of if-then rules defined in domain knowledge. Consider the rule of if X' is A then Y' is B. Let sets of values of X' and Y' be X and Y, respectively, where Y varies between 0 and 1. Assume that A and B are fuzzy sets on the universal sets X and Y, respectively. $A = \sum_{x\in X} \mu_A(x)/x$ and $B = \sum_{y\in Y} \mu_B(y)/y$, where μ is a membership function mapping from X or Y to [0, 1]. A fuzzy relation R on $X \times Y$ can be defined as follows: $R(x,y) = \sum_{x\in X,\ y\in Y} \min(\mu_A(x), \mu_B(y))/(x,y)$. Then, given the proposition of if X' is A then Y' is B, two fuzzy sets A and B and a fact expressed by $X' = x$, a new fuzzy set $C = \sum_{y\in Y} \mu_c(y)/y$ can be derived from the equation of $\mu_c(y) = \sup\ \min(\mu_A(x), R(x,y))$. To select one representative value of the set Y, defuzzification methods are used to determine the similarity value such as the center of gravity and the mean of maximum.

In this example, consider the fuzzy rule of *if weight is heavy then shape is fat* in Fig. 6. The two linguistic terms *heavy* and *fat* represent fuzzy sets defined in the DK *Student*. If a student's weight is w, the probability of his/her being fat can be the defuzzification of $\mu_{shape\_is\_fat}(y)$, where $\mu_{shape\_is\_fat}(y) = sup\ \min(\mu_{Heavy}(w), R_{Heavy,Fat}(x,y))$.

Hybrid query. Hybrid queries involve a Boolean combination of the queries described above.
Example: Find all the events where the student *Alan* appears and the student *Tom* is presenting the book *Video Databases* with *slices* in the video *campus*.
Query lang:
Select RELATIVE E.name
From Video V, Event E, Student O1, O2, Present P
Where V CONTAIN E AND E CONTAIN O1 AND E CONTAIN O2
AND V.name = "*Campus*" AND O1.name = "Tom" AND O2.name-
= "Alan"
AND O1.action = P AND P.method = "slice" AND P.Content = "Video Databases".

5. Implementation

We have implemented a prototype system based on the proposed model under a Microsoft Windows environment. The system manages a collection of raw video clips, objects and events that can be queried and browsed. The overall architecture illustrated in Fig. 9 consists of three major modules: *interface*, *content management* and *storage modules.*

In the interface module, *author tool* extracts objects and events from video data, and allows users to annotate their content according to the CFGs defined in Appendix A. Notably, users do not need to remember the exact syntax of annotation language since the authoring interfaces will direct users to process it, as illustrated in Figs. 10–15. The *playback area* has the functions of video clip playback and static frame display. The *edit area* guides users in video content annotations. The *display area* shows the extracted content. When someone needs to describe dynamic properties of the participant objects in an event or the relevant objects of an object, *browse area* helps him/her look for a domain knowledge, object or event. At present, domain and event hierarchies are only tree structures, namely sub-domains and sub-events are of a single-inheritance relationship. A table-of-content (ToC) represents a DKH or an annotation language, as illustrated in browse and display area. Expanding a condensed ToC can obtain a detailed ToC. The *selection area* allows users to view a single object or a collection of objects. The *query tool* shown as Fig. 16 is responsible for the validity of SQL-like queries by parsing users' inputs. The *presentation tool* shown in Fig. 17 allows users to arrange an output format and to browse the video content by traveling the links among objects and events.

In the content management module, *annotation manager* maintains object nets and event hierarchies consistently. To store individual instances in the storage system, the data structures of V-Node, D-Node and I-Node are

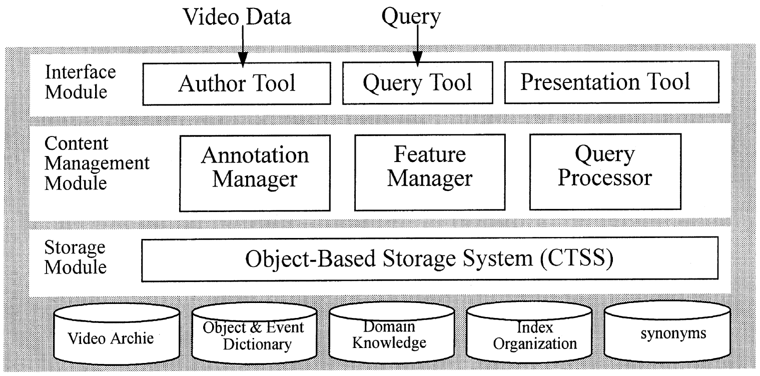


Fig. 9. The architecture of the protype system.

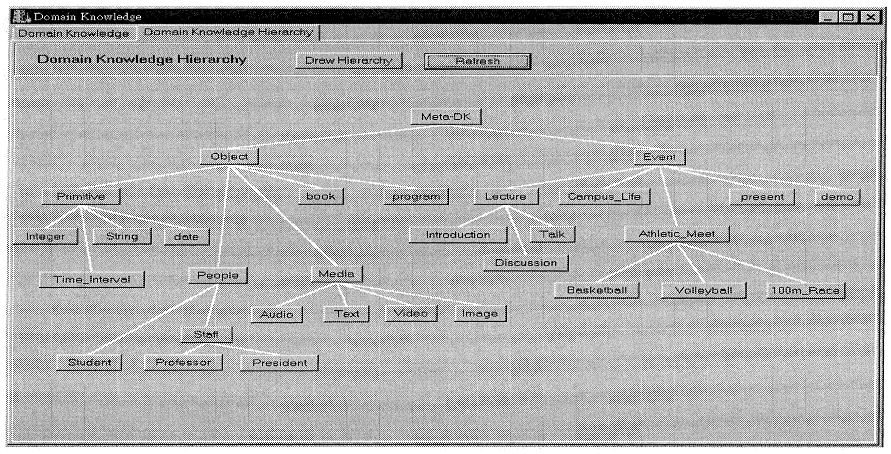
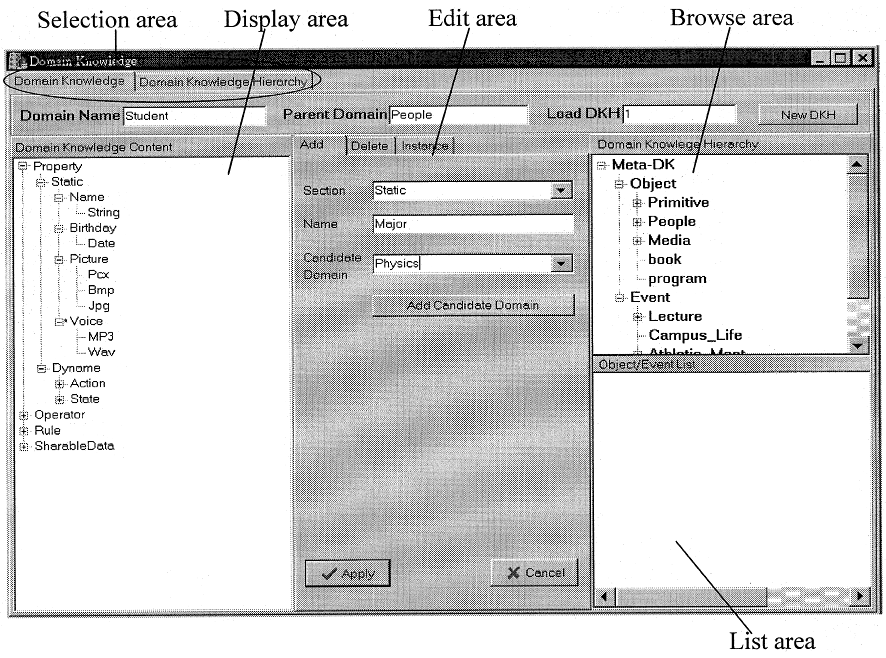


Fig. 10. The interface of domain knowledge hierarchies.



Domain Name: Student
Parent Domain: People
Load DKH: 1

Section: Static
Name: Major
Candidate Domain: Physics

- Property
  - Static
    - Name
      - String
    - Birthday
      - Date
    - Picture
      - Pcx
      - Bmp
      - Jpg
    - Voice
      - MP3
      - Wav
  - Dyname
    - Action
    - State
- Operator
- Rule
- SharableData

- Meta-DK
  - Object
    - Primitive
    - People
    - Media
    - book
    - program
  - Event
    - Lecture
    - Campus_Life
    - Athletic_Meet

Fig. 11. An example of domain knowledge *People* annotation

considered, as illustrated in Fig. 18(a). V-Node is responsible for storing a DKH, object net or event hierarchy. D-Node and I-Node describe an annotation language. Each node has a globally unique identifier, which is generated

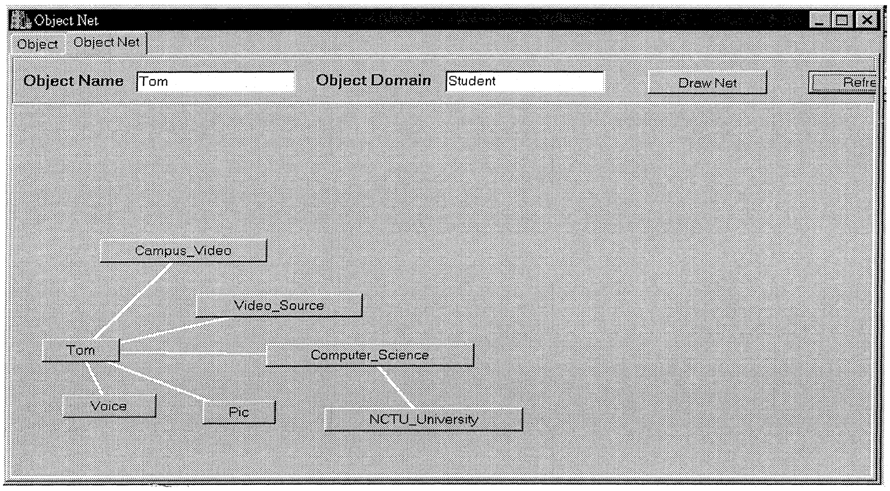


Fig. 12. The interface of object nets.

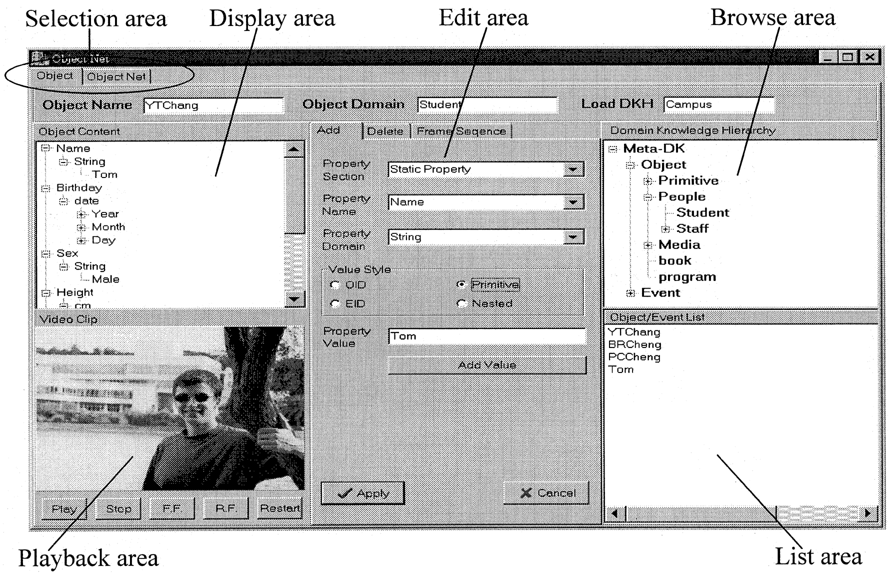


Fig. 13. An example of object *Tom* annotation.

by the storage system. Node-Type identifies a domain (D), object (O) or event (E). Node-Length denotes the total length of the node. Link-Count and Data-Count record the numbers of links and data, respectively. Link-Vector consists of the identifiers pointing to other nodes such as a link from a parent to a child

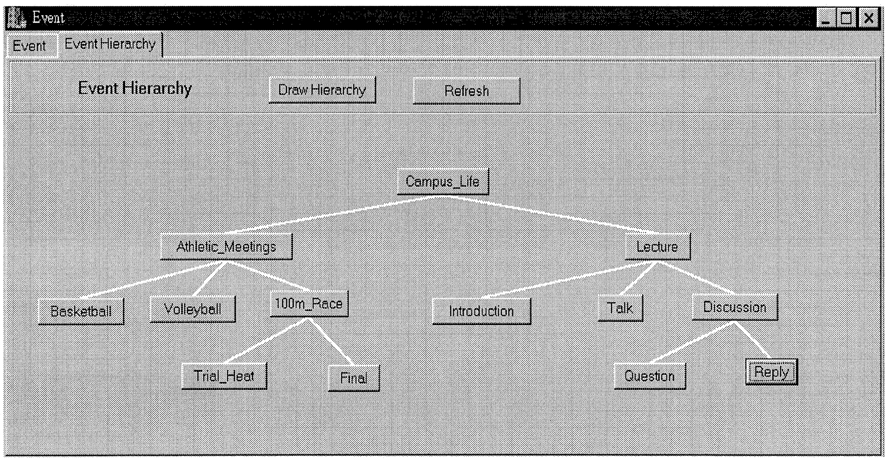
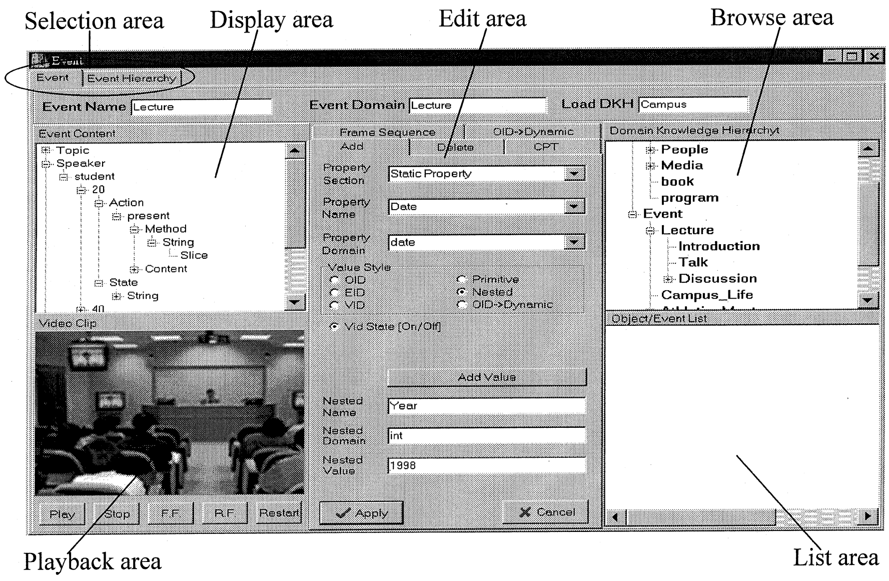


Fig. 14. The interface of event hierarchies.



Event Name Lecture
Event Domain Lecture
Load DKH Campus

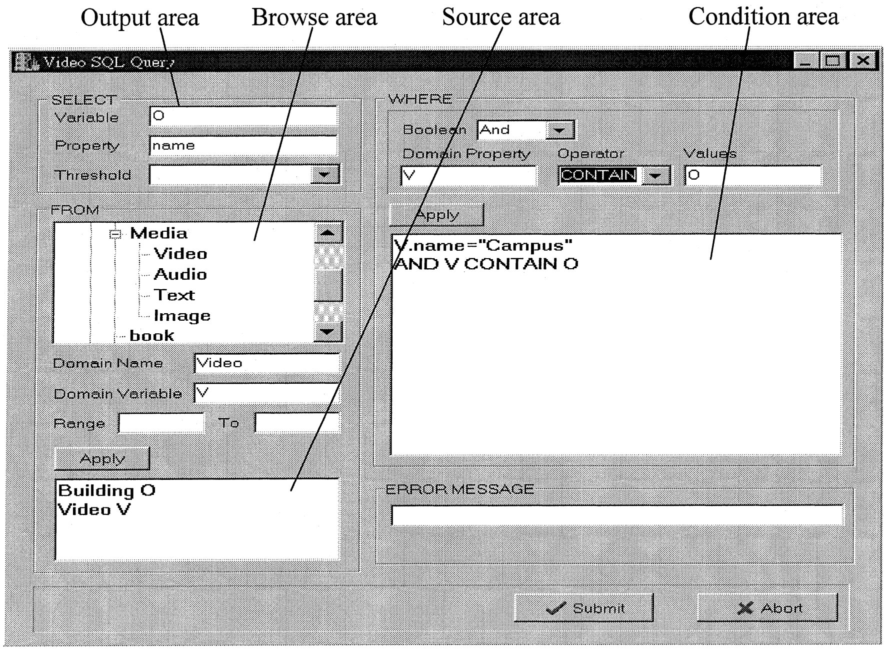
Event Content:
- Topic
- Speaker
  - student
    - 20
      - Action
        - present
          - Method
            - String
              - Slice
          - Content
      - State
        - String
    - 40

Property Section: Static Property
Property Name: Date
Property Domain: date
Value Style: ( ) OID ( ) EID ( ) VID ( ) Primitive (•) Nested ( ) OID->Dynamic
(•) Vid State [On/Off]
Nested Name: Year
Nested Domain: int
Nested Value: 1998

Domain Knowledge Hierarchy:
- People
- Media
- book
- program
- Event
  - Lecture
    - Introduction
    - Talk
    - Discussion
  - Campus_Life

Fig. 15. An example of event *Lecture* annotation.

in a domain or event hierarchy. For each object, Link-Count is always set to be zero. Data-Offset Vector consists of the offsets of data in the Data-Vector part of the V-Node storage format. Data-Vector is a data array. Each D-Node represents data and comprises of an item array. Each item is represented as an

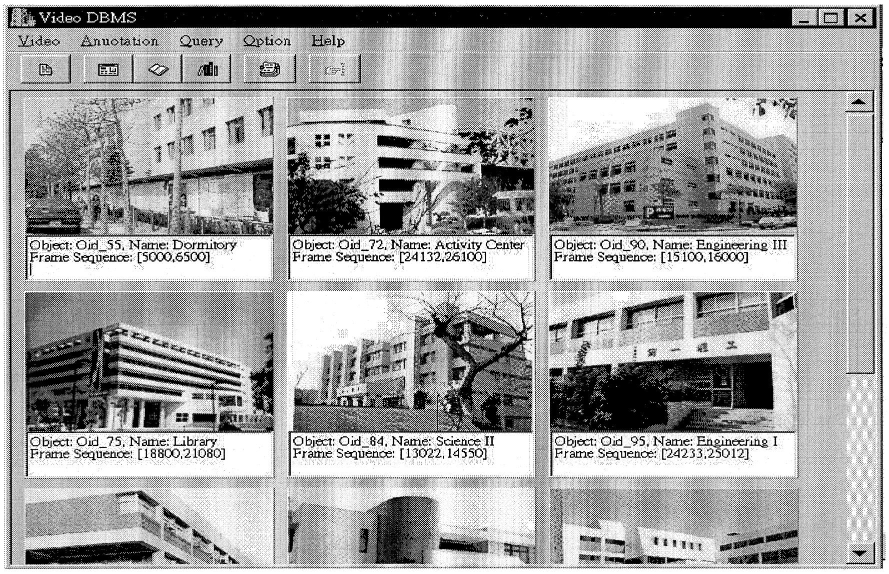


Variable: O
Property: name
Domain Name: Video
Domain Variable: V
Boolean: And
Domain Property: V
Operator: CONTAIN
Values: O

Building O
Video V

V.name="Campus"
AND V CONTAIN O

Fig. 16. An example query of finding out all buildings in the video *Campus*.



Object: Oid_55, Name: Dormitory
Frame Sequence: [5000,6500]

Object: Oid_72, Name: Activity Center
Frame Sequence: [24132,26100]

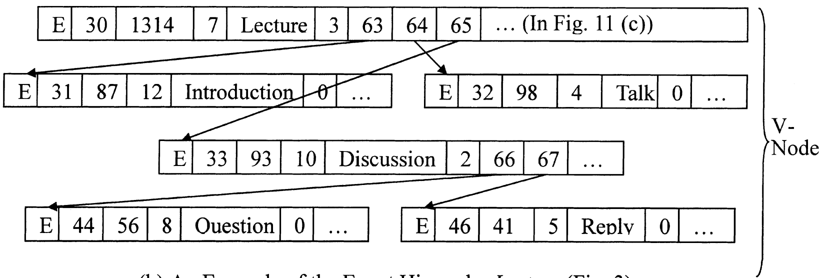
Object: Oid_90, Name: Engineering III
Frame Sequence: [15100,16000]

Object: Oid_75, Name: Library
Frame Sequence: [18800,21080]

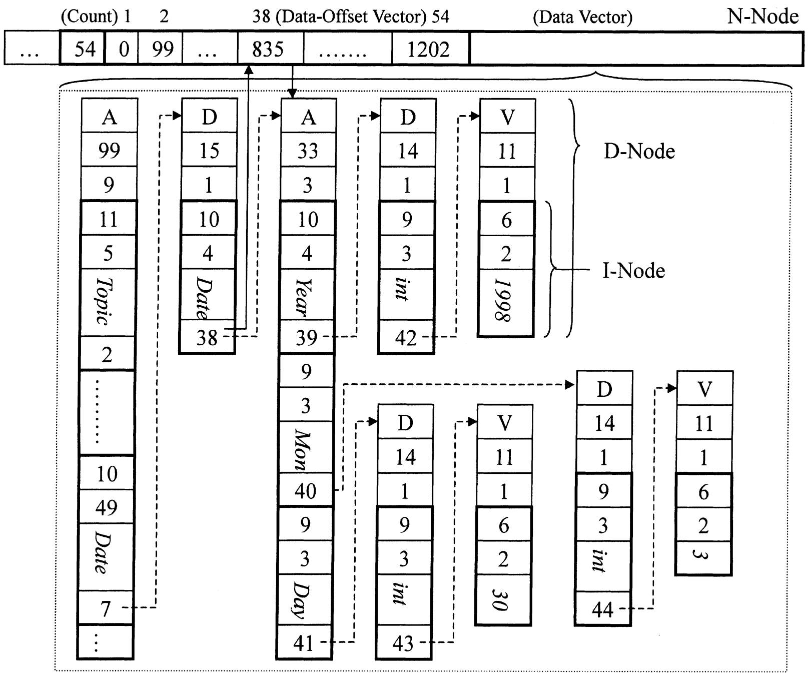
Object: Oid_84, Name: Science II
Frame Sequence: [13022,14550]

Object: Oid_95, Name: Engineering I
Frame Sequence: [24233,25012]

Fig. 17. The Query results of searching the object *Building*.


E 30 1314 7 Lecture 3 63 64 65 ... (In Fig. 11 (c))
E 31 87 12 Introduction 0 ...
E 32 98 4 Talk 0 ...
E 33 93 10 Discussion 2 66 67 ...
E 44 56 8 Question 0 ...
E 46 41 5 Reply 0 ...
V-Node

(b) An Example of the Event Hierarchy *Lecture* (Fig. 3).



(c) An Example of the Event *Lecture* (Fig. 2).

Fig. 18. The Data Structures of the Data Model.

I-Node. Data-Type identifies an attribute (A), domain (D) or value (V). Data-Pointer records a D-Node's identifier. To reduce the computational cost of query processing, each annotation should be parsed in advance and transformed into a data array. Figs. 18(b) and (c) show the examples of the event hierarchy and the event *Lecture*, respectively. In Fig. 18(c), a D-Node stands for a ToC and involves an item vector, which consists of items pointing to (detailed) ToCs. Each item records the index number of an element of

Data-Offset Vector. Following the offset of data in the Data-Vector part can locate the detailed ToC.

The *feature manager* analyzes the audio-visual features of extracted events and objects, and then inserts these feature values into the structures of objects and events, and finally updates the index organizations automatically. The prototype system focuses mainly on processing semantic queries and temporal queries. Color playing a significant role in image retrieval is supported in current version. However, retrieving video content either manually or automatically is easy to miss some important features possibly. To avoid the situation in which users cannot find video content that is not completely extracted by systems, we introduce two possible solutions. One is the Bayesian network as stated in Section 3.3 and the other is to generalize each DK to prescribe feature extraction from raw data, content matching, query analysis and semantic translation. If-then rules can be applied to automatically connect high-level semantics to low-level audio-visual features and vice versa. Consider a situation in which the DK *people* contains the rule of *if age is old then hair is white*, where *age* is a linguistic term and *white* is a visual concept. If someone wants to search for the persons with white hair (similar query), the system may attempt to automatically transform it into a new request: *find all people whose hair is white (similar query) or whose age is old* (*fuzzy query*). One of fuzzy reasoning algorithms has been illustrated in Section 4.4.

The *query processor* primarily implements the algorithms described in Section 4. To reduce the gap between annotator's and users' comprehension of the video contents, synonym specifications are adopted herein. Each such synonym is then automatically substituted for the original query term to retrieve video data with respect to the original and the substituted terms, which can be vague or nonvague. In the storage module, we adopt CTSS [6] developed in our laboratory as the object-based storage system, which provides the capabilities of BLOB, clustering, indices (B+-tree, extendible hash, and SR-tree [20]), video data compression/decompression, and recovery.

6. Conclusion

This paper has presented a novel video data model for content-based accesses to video databases. An example video documentary is also adopted to show how this model represents it. The proposed data model comprises of an object net and an event hierarchy. This structure consists of the trend of video compression standards and is targeted at a partial automation of text-based annotation and full automation of feature-based annotation. Consider the twin criteria [3], image preprocessing granularity and the level of abstraction, for classifying existing approaches of modeling video data. For the first criterion, the proposed model involves identification of objects within a video frame (fine granularity), scene change detection of consecutive frames (coarse granularity). Notably, a scene can be viewed as an event. For the second one, this model supports both audio-visual features (low-level abstraction) and semantics (high-level abstraction). Multi-level semantics can be associated with an object or event such as keywords, key frames, icons and video. Consequently, the proposed model is capable of capturing complex information in video data for each dimension.

Regarding video contents, this paper focuses mainly on semantic information annotation because conventional approaches either cannot describe complex scenarios [1,9,15,21,26,31] or are inhibited by computational complexity [8,19,22]. Audio-visual features are discussed as well. In contrast to previous methods [1,8,19,26,29,31], the proposed model clearly describes the structures of objects and events in video data and the relationships among them. Primary features of the proposed video data model can be summarized as follows:

- The characteristics of objects are divided into two classes: static properties and dynamic properties. Static properties are usually shared among events over video programs. This provides the merits of sharability and reusablility of annotation.
- Events are organized as an event hierarchy to capture the feature of multi-level abstractions. According to this hierarchy, common properties among events are inherited and shared; conditional dependence relationships among events are depicted as well. To confirm the merits of proposed policy, a simulation experiment is performed to compare its performance with those of existing policies. Experimental results indicate that the integrated method improves the capacity of content-based retrieval significantly, thereby maximizing the criteria of recall and precision.
- The annotation structure is organized recursively. Similar to Entity-Relationship model, this scheme provides a great expressive power to represent complex scenarios in the real world. Each entity in ER model corresponds to an object in the proposed model and each relationship corresponds to a dynamic property or a static reference of an object. In addition, the nested structure does not need to analyze the free text statistically [8,19] or parse natural language smartly [22]. It can be decomposed and processed efficiently.
- The division of objects and events helps users directly understand video content, abstract important features early, and query/browse video information in detail. This model is properly applied into the applications, where a set of objects interact with one another, such as a movie with many actors and a soccer game with many players. Moreover, an event hierarchy can be extended to represent multi-level abstractions of multi-dimensional data like images. The integration of inheritance by context and conditional dependence relationship can be applied as well.
- This schemaless data model allows users to incrementally define their own properties for each object/event and provides the advantage of flexibil-

ity. With the assistance of DKHs, different types of queries are processed efficiently and can be represented effectively by a SQL-like query language.
- According to fuzzy inference (if-then) rules defined in domain knowledge, high-level semantics can be inferred from low-level audio-visual features to associate what users desire with what systems can deliver and vice versa.

Further work should focus on capturing spatial–temporal information of objects in different events. We are planning to extend this model to the network environment such as digital libraries. Interoperability will be the key challenge in digital video library systems, which necessarily address a broad range of issues such as metadata management, handle generation and interconnectivity protocol. We also intend to investigate how to optimize video query processing and apply our system to more real world applications.

Appendix A

(a) The unambiguous CFG for the static information of an object.
G = (***V***, ***T***, ***S P***), where
V = {⟨***Static Description***⟩, ⟨***Property***⟩, ⟨***S1***⟩, ⟨***DC***⟩, ⟨***S2***⟩, ⟨***Value***⟩, ⟨***Property Name***⟩, ⟨***Domain ID***⟩, ⟨***V***⟩}
T = { . , "{", "}", : , (,) , String, Real, Integer, DID, EID, OID }
S = ⟨***Static Description***⟩
P = {⟨***Static Description***⟩:: = {⟨***Property***⟩}
⟨***Property***⟩:: = ε | ⟨***S1***⟩ | ⟨***S1***⟩.⟨***Property***⟩
⟨***S1***⟩:: = ⟨***Property Name***⟩: (⟨***DC***⟩)
⟨***DC***⟩:: = ⟨***S2***⟩ | ⟨***S2***⟩; ⟨***DC***⟩
⟨***S2***⟩:: = ⟨***Domain ID***⟩(⟨***Value***⟩)
⟨***Value***⟩:: = ⟨***V***⟩ | ⟨***V***⟩, ⟨***Value***⟩
⟨***Property Name***⟩:: = *String*
⟨***Domain ID***⟩:: = *DID*
⟨***V***⟩:: = *OID* | *EID* | ⟨***Property***⟩ | *String* | *Real* | *Integer*}

(b) The unambiguous CFG for the annotation of an event.
G = (***V***, ***T S***, ***P***), where
V = {⟨***Description***⟩, ⟨***Property***⟩, ⟨***S1***⟩, ⟨***DC***⟩, ⟨***S2***⟩, ⟨***Value***⟩, ⟨***Property Name***⟩, ⟨***Domain ID***⟩, ⟨***V1***⟩, ⟨***V2***⟩, ⟨***DP***⟩}
T = { . , "{", "}", : , (,) , String, Real, Integer, DID, EID, OID, VID, }
S = ⟨***Description***⟩
P = {⟨***Description***⟩:: = {⟨***Property***⟩}
⟨**Property**⟩:: = ε | ⟨***S1***⟩ | ⟨***S1***⟩.⟨***Property***⟩
⟨***S1***⟩:: = ⟨***Property Name***⟩: (⟨***DC***⟩)
⟨***DC***⟩:: = ⟨***S2***⟩ | ⟨***S2***⟩; ⟨***DC***⟩

⟨S2⟩:: = ⟨Domain ID⟩(⟨Value⟩)
⟨Value⟩:: = ⟨V1⟩ | ⟨V1⟩, ⟨Value⟩
***⟨Property Name⟩:: =** String*
***⟨Domain ID⟩:: =** DID*
⟨V1⟩:: = VID, ⟨V2⟩ | ⟨V2⟩
***⟨V2⟩:: =** OID | EID | VID | OID, **⟨DP⟩** | **⟨Property⟩** | String | Real | Integer*
⟨DP⟩:: = ⟨Property⟩}

Appendix B

The simplified syntax of a conditional expression in the *Where clause* is defined as follows:
⟨expression⟩:: = ⟨expression⟩⟨Boolean op⟩⟨query term⟩ | ⟨query term⟩
⟨query term⟩:: = NOT (⟨query unit⟩) | ⟨query unit⟩
⟨query unit⟩:: = ⟨V⟩CONTAIN⟨E⟩ | ⟨V⟩CONTAIN⟨O⟩ | ⟨E⟩CONTAIN⟨O⟩ |
 ⟨D⟩⟨temporal op⟩⟨D⟩ | ⟨D⟩.⟨property⟩⟨compare op⟩⟨value⟩
⟨Boolean op⟩:: = AND | OR
⟨temporal op⟩:: = START, FINISH, BEFORE, MEET, OVERLAP, DURING, EQUAL, INTERSECT
⟨compare op⟩:: = $\approx$ | = | < | > | $\leqq$ | $\geqq$ | $\subset$ | $\supset$ | $\subseteq$ | $\supseteq$
⟨D⟩:: = ⟨V⟩ | ⟨E⟩ | ⟨O⟩
⟨V⟩:: = String
⟨E⟩:: = String
⟨O⟩:: = String
⟨property⟩:: = String
⟨value⟩:: = {⟨set value⟩} | ⟨single value⟩
⟨set value⟩:: = ⟨single value⟩,⟨set value⟩
⟨single value⟩:: = String | Real | Integer | ⟨D⟩,

where ⟨V⟩, ⟨E⟩ and ⟨O⟩ are, respectively, the domain variables of video, events, and objects defined in the *From clause*. The keyword *CONTAIN* denotes the relations among domain variables. The query statement, *A CONTAIN B*, implies that *A* is the entity where *B* appears. The temporal operator *INTERSECT* suggests that the intervals of two domain variables have intersection. The comparison operator ($\approx$) is used in the fuzzy query and the similar query. The set operators ($\subset, \supset, \subseteq$ and $\supseteq$) are used to evaluate set values.

References

[1] S. Adali, K.S. Candan, S.S. Chen, K. Erol, V.S. Subrahmanian, The advanced video information system: data structure and query processing, Multimedia Systems 4 (1996) 172–186.

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

[3] W.A. Khatib, Y.F. Day, A. Ghafoor, P.B. Berra, Semantic Modeling and Knowledge Representation in Multimedia Databases, IEEE Transactions on Knowledge and Data Engineering 11 (1) January 1999 64–80.

[4] Y.A. Aslandogan, C.T. Yu, Techniques and Systems for Image and Video Retrieval, IEEE Transactions on Knowledge and Data Engineering 11 (1) January 1999 56–63.

[5] M.L. Cascia, E. Ardizzone, JACOB: Just A Content-Based Query System for Video Databases. in: Proceedings of ICASSP-96, May 1996, pp. 7–10.

[6] P.J. Cheng, W.P. Yang, The Design and Implementation of Modern Object-Based Storage System, CTSS, Technical Report, National Chiao Tung University, Computer and Information Science Department, 1996.

[7] H. Chou, D. DeWitt, R. Datz, A. Klug, Design and Implementation of the Wisconsin Storage Systems, Software Practice and Experience 15 (10) October 1985.

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

[9] A. Dan, D. Sitaram, J. Song, BRAHMA: Browsing and Retrieval Architecture for Hierarchical Multimedia Annotation, Multimedia Tools and Applications 7 (1) July 1998 83–101.

[10] M. Davis, Media Streams: An Iconic Visual Language for Video Annotation, IEEE Symposium on Visual Languages, 1993.

[11] Y.F. Day, S. Dagtas, M. Iino, A. Khokhar, A. Ghafoor, Object-Oriented Conceptual Modeling of Video Data, in: Proceedings of the Eleventh International Conference on Data Engineering, 1995, pp. 401–408.

[12] N. Dimitrova, T. McGee, H. Elenbaas, Video Keyframe Extraction and Filtering: A Keyframe is not a Keyframe to Everyone, in: Proceedings of the Sixth International Conference on Information and Knowledge Management, 1997, pp. 113–120.

[13] A.K. Elmagarmid, H. Jiang, A. Helal, A. Joshi, M. Ahmed, Video Database Systems, Kluwer Academic Publishers, Dordrecht, 1997.

[14] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, P. Yanker, Query by Image and Video Content: The QBIC System, IEEE Computer September 1995 23–32.

[15] R. Hjelsvold, R. Midtstraum, Databases for Video Information Sharing, in: Proceedings of SPIE – The International Society for Optical Engineering, 1995, 268–279.

[16] T.S. Huang, S. Mehrotra, K. Ramchandran, Multimedia Analysis and Retrieval System Project, in: Proceedings of the 33rd Annual Clinic on Library Application of Data Processing – Digital Image Access and Retrieval, 1996.

[17] E. Hwang, V.S. Subrahmanian, Query Video Libraries, Journal of Visual Communication and Image Representation 7 (1) March 1996 44–60.

[18] H. Jiang, A. Helal, A.K. Elmagarmid, A. Joshi, Scene change detection techniques for video database systems, Multimedia Systems 6 (1998) 186–195.

[19] H. Jiang, D. Montesi, A.K. Elmagarmid, VideoText Database Systems, in: Proceedings of the 1997 IEEE International Conference on Multimedia Computing and Systems, 1997, pp. 344–351.

[20] N. Katayama, S. Satoh, The SR-tree: An Index Structure for High-Dimensional Nearest Neighbor Queries, in: Proceedings of ACM SIGMOD vol. 26 (2), June 1997, pp. 369–380.

[21] K.W. Kim, K.B. Kim, H.J. Kim, VIRON: An Annotation-Based Video Information Retrieval System, in Proceedings of the IEEE Computer Society's International Computer Software and Applications Conference, 1996, pp. 298–303.

[22] Y.B. Kim, M. Shibata, Content-Based Video Indexing and Retrieval – A Natural Language Approach, IEICE Transactions on Information and Systems E79-D (6) June 1996 695–705.

[23] C.I. Lee, Y.I. Chang, W.P. Yang, An efficient conflict-resolution approach to support read/write operations in a video server, International Journal of Software Engineering and knowledge Engineering 7 (3) (1997) 1–29.
[24] J.C.M. Lee, Q. Li, W. Xiong, VIMS: A video information management system, Multimedia Tools and Applications 4 (1997) 7–28.
[25] T.D.C. Little, A. Ghafoor, Interval-Based Conceptual Models for Time-Dependent Multimedia Data, IEEE Transactions on Knowledge and Data Engineering 5 (4) August 1993 551–563.
[26] E. Oomoto, K. Tanaka, OVID: Design and Implementation of a Video-Object Database System, IEEE Transaction on Knowledge and Data Engineering 5 (4) August 1993 629–642.
[27] S.J. Russell, R. Norvig, Artificial Intelligence, Prentice-Hall, Englewood Cliffs, 1995.
[28] J.R. Smith, S.F. Chang, VisualSEEk: A Fully Automated Content-Based Image Query System, ACM Multimedia Boston, November 1996.
[29] T.G.A. Smith, G. Davenport, The Stratification System: A Design Environment for Random Access Video, in: Workshop on Networking and Operating System Support for Digital Audio and Video, 1992.
[30] F.A. Tobagi, J. Pang, R. Baird, M. Gang. Streaming RAID – A Disk Array Management System for Video Files, in: Proceedings of the 1st ACM International Conference on Multimedia, 1993, pp. 393–400.
[31] R. Weiss, A. Duda, D.K. Gifford. Content-Based Access to Algebraic Video, IEEE International Conference Multimedia Computing and Systems, 1994, 140–151.
[32] J.K. Wu, A.D. Narasimhalu, B.M. Mehtre, C.P. Lam, Y.J. Gao, CORE: a content-based retrieval engine for multimedia information systems, Multimedia Systems 3 (1995) 25–41.
[33] A. Yoshitaka, S. Kishida, M. Hirakawa, T. Ichikawa, Knowledge-assisted content-based retrieval for multimedia databases, IEEE Multimedia 1 (4) (1994) 12–21.
[34] Description of MPEG-4, ISO/IEC JTC1/SC29/WG11 N1410, October 1996.