1998 International Computer Symposium
Workshop on Software Engineering and Database Systems
December 17-19, 1998, N.C.K.U., Tainan, Taiwan, R.O.C.

A New Content-Based Video Retrieval System - Data Model and Query Processing

Pu-Jien Cheng and Wei-Pang Yang

Department of Computer and Information Science,
National Chiao Tung University, Hsinchu, Taiwan, R.O.C.
Email: pjcheng@dbsun1.cis.nctu.edu.tw, wpyang@cis.nctu.edu.tw.

Abstract

A video retrieval system manages large scale of video archive and provides content-based accesses to users. In this paper, we propose a video data model with the advantages of flexibility and sharability by grouping different video materials of interest to users into different representation frameworks according to their individual characteristics. Unlike pervious approaches of keywords or free text, we provide a nested annotation language for describing complex scenarios in video data effectively. A SQL-like query language based on the proposed model is presented. With the assistance of domain knowledge and index organizations, we develop algorithms to efficiently process six types of queries about semantic information of video content, including elementary domain query, elementary content query, object-event relation query, object path query, event inference query, and temporal query. Finally, a prototype content-based video retrieval system is implemented.

1. Introduction

With the progress of high-capacity storage and high-performance compression technologies, video has become an important source stored in modern databases. Suppose a video documentary entitled "*The University Campus*" contains information about commencements, departments, faculty, and so on. This example documentary will be used throughout the whole paper. Some semantic information that may be asked of the example is illustrated as follows: "find all video frames in which professor Yang is teaching graduate students databases," and "find all lectures that appear after the Founder's Day celebration."

Recently, numerous studies on content-based retrieval of video data have been carried out. They can be classified into three categories based on the indexed information content: the semantic-based approach [1,5,9,12,14,17,20,21], the audiovisual feature-based approach [2,10,19,22], and the hybrid approach [4,8,15,24].

The semantic-based approach focuses on annotating the semantic content contained in video data. It is suitable for exploring comprehensive knowledge that video conveys to users but fails to provide the automation of annotation perfectly. The semantic annotation is usually ambiguous and application-dependent. So far, major annotation structures include keywords [8,20], attributes [1,9,15,17,21,24], natural language [4,12,14], and iconic language [5]. The keyword approach has the advantages of simplicity and easiness. The attribute approach is as the form of attribute/value pairs. It depicts the meanings of values more clearly than the keyword annotation. Unfortunately, both of them have limitations on the ability to satisfactorily describe complex scenarios due to their restricted structures. On the contrary, the natural language approach provides a powerful expressive capacity but suffers from the computational complexity of intelligent parsers [14] or statistical analyses [4,12].

The audiovisual feature-based approach derived from image information systems focuses on low-level feature extraction from video data. This method supports complete automation of indexing processes but lacks semantics attached to the extracted features. The promising approach focuses on exploring the integration of semantic-based and feature-based content. By now, a number of researches [2,4,8,10,19,22] about image feature processing have been investigated for various applications successfully. However, the expressive capability and the computational complexity are trade-offs in the environment of semantic annotation with keywords, attributes, or natural language.

In this paper, we propose a generic video data model and primarily focus on developing a nested, unambiguous annotation language with great descriptive power and less computational complexity. In addition to inheritance by context, we take into account conditional dependence relationships between meaningful scenes to improve retrieval capacity. Based on this model, we apply several index schemes to facilitate the process of evaluating queries with the assistance of domain knowledge. We also develop a SQL-like query language and algorithms to respectively present and process these kinds of queries mentioned above.

The rest of this paper is organized as follows. In

This research was partially supported by the National Science Council, Taiwan, R.O.C., under contract no. NSC88-2213-E-009-016.

Section 2, we first introduce the basic ideas and then present our video data model and annotation structures. In Section 3, we define the syntax of query language, classify semantic information queries into six categories, and describe how to process these queries with the assistance of indices and domain knowledge. Section 4 describes the architecture of the prototype system and Section 5 concludes this paper.

2. The Video Data Model

2.1 Basic Ideas and Term Definitions

In our video databases, raw video data consists of a sequence of continuous frames, where a set of objects interact with one another. A *video frame* referring to 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 frame *j*. A *video object* is a concrete or abstract entity appearing in video frames, such as a professor, a department, and a course. It involves inherent attributes and composite relations to other objects. If the interaction among a collection of objects causes meanings to us, we call it an *event*, such as a lecture, and a basketball game. In addition to its proper attributes, an event also contains scenarios and represents semantic abstractions. In principle, the entire video stream and the single frame are two extreme levels of abstractions. Other intermediate abstractions may include shots, scenes, sequences, and compound units [7]. Consequently, events and objects have different properties in the aspect of knowledge representation. We will models them respectively according to their individual characteristics and keep the relationship between them.

In order to classify the features of objects and events, we propose a new 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 a multi-level abstraction of video data and has the capacity of annotating complex scenarios. The relationship between an object and an event is to denote which objects are role players appearing in an event or which events involve the objects. As semantic annotation is application-dependant, we assume a domain hierarchy is given in advance [17] where the links among domains represent *is-a* relationships.

Our data model is adequate to associate arbitrary descriptions with an object or an event, including text-based and feature-based properties, such as keywords, free text, icons, key frames, colors, textures, spatial locations, temporal relations, and so on. Here, we primarily focus on developing an annotation language to capture complex semantics of video data and presenting the corresponding query language and processing algorithms.

2.2 The Object Net

The properties of each object can be divided into two categories: *static* and *dynamic*. A static property means its values are fixed and a dynamic property's values are variable with the passage of 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.

Definition 1. An *video object (or object)* is a 4-tuple (*OID, DID, PT, I*), where

- ***OID*** is the unique object identifier.
- ***DID*** is 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* is a single or a set of values. A value is an *OID*, an *EID* (event identifier), a static property, or a atomic value, such as integer, real number, and string.
- ***I*** denotes the *frame sequence* of video frames in which the object appears.

In Def. 1, 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 [1,15,17,21] have limitations on the computing ability of values because of the absence of operation/type definitions. Different from conventional programming languages, each property in Def. 1 may contain multiple domains, such as property *Picture* can be represented as pcx or bmp formats in Fig. 1, which shows the static attributes of object *Tom*. When a collection of objects have references to each other, an object net is formed. Composite references are not defined here as we consider it is the query processor's responsibility to support integrity constraints (dependent/ independent and shareable/exclusive).

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]

Fig. 1. An example of the object *Tom*.

2.3 The Event Hierarchy

An event hierarchy is aimed at developing a mechanism of describing multi-level abstractions and representing complex scenarios. It is concerned with two aspects of semantic relationships: inheritance of attributes by context and conditional dependence between two events. Moreover, an event not only contains its inherent attributes but also describes the participant objects' dynamic properties.

```
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 (20)). )).
 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]
```

Fig. 2: An example of the event *Lecture*.

Definition 2. An *event* is a 5-tuple (*EID, DID, PT, CPT, I*), where

- –*EID* is the unique event identifier.
- –*DID* is the domain identifier referring to the event'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* is 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* is a single value, including an *OID*, an *EID*, a *VID*, a 2-tuple <*OID*, *DP*>, a *property* and an atomic value, such as 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.
- –*I* denotes the frame sequence of video frames in which the event appears.

In Def. 2, each event only needs to describe the participant objects' dynamic properties. Static information of those objects is kept in an object net, referred as OIDs, and shared among all events. From the description of event *Lecture* in Fig. 2, *student Tom* (Oid_20) and *student Alan* (Oid_40) *presented chapter 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 *March 20,1998*. The topic of that lecture was "*Video Databases.*" *Tom presented* his work with *slices* and *demonstrated his program* (Oid_24) with a *computer*. *Alan presented* it with *slices nervously*.

To capture the feature of multi-level abstractions of video data, we organize all events into an event hierarchy, as shown in Fig. 3. An arbitrary video 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 connecting to it. The meaning of a link between a parent E1 and its child E2 involves: (1) E2 has a direct influence on E1, (2) E1 has a independent and shareable composite reference to E2 implicitly, and (3) E2 may inherit the properties of E1. To describe the conditional dependence relationship among events, we maintain a *conditional probability table* (*CPT*) defined in *Bayesian networks* [18] for each event. From the querying point of view, an event hierarchy can be treated as a Bayesian network.

To exhibit multi-level abstractions of video data effectively, an event hierarchy must provide the mechanism of inheritance of descriptions by context for sharing common information among meaningful scenes [15,17,21]. This reduces the amount of annotations of the events with low-level abstraction. In Fig. 3, a plus-marked attribute is inheritable, and vice versa. Inheritance by context is only adapted when the inheritable properties of an event and all its children have the relationships of (1) *generalization* (2) *composition or* (3) *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). If someone poses a query: "find all the events where *Student Tom* is in the location *Main Campus,*" the result is event *Talk 1.* Note that no domain knowledge is needed to tell a query processor that *Room 130* is a part of *Main Campus.*

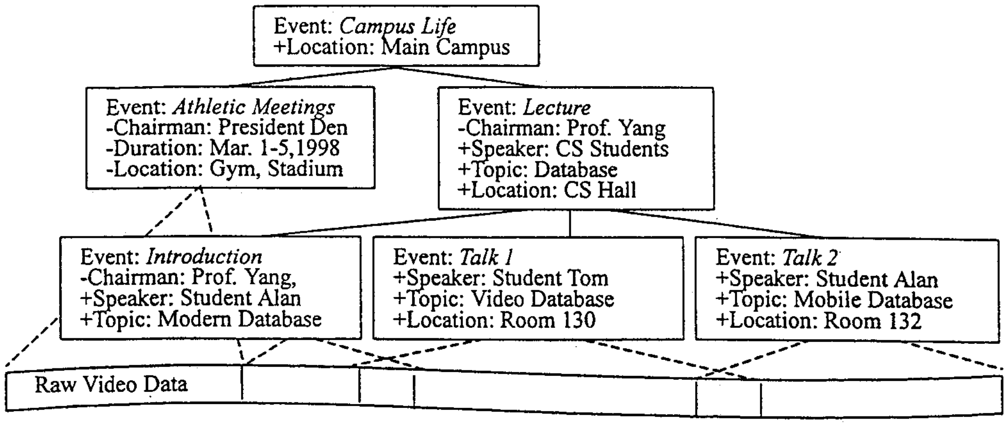


Fig. 4. A detailed example of the event hierarchy about *Campus Life*.

However, if someone wants to find all the events where *Student Tom and Student Alan* are in the location *CS Hall*, we will get three events *(Introduction, Talk 1 and Talk 2)* as a result. But it seems to be more reasonable to return event *Lecture*. The event *Lecture* involves high-level concepts *(CS Students)* but loses some important information *(Student Tom and Student Alan)* of its children. Most researches only concern how an event influences its descendants without discussing the situation mentioned above. In the respect of querying relative events, the traditional approach only paying attention to inheritance by context has two major disadvantages: (1) people may just remember the important, apparent part of objects or motions in an event and usually cannot describe it correctly and detailedly, and (2) video data models apply the description of high-level semantics to represent larger events to save storage space and vice versa. Therefore, a system will suffer from the inability to assemble several salient parts into a more complete concept.

Here, we consider two kinds of properties in an event. One is *inheritable*; the other is *not inheritable*. The inheritable properties in an event will be inherited by all its descendants. To solve the problem discussed above, we apply the data structure of Bayesian networks to event hierarchies. Bayesian networks based on Bayes' theorem are to 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. To verify the merit of proposed policy, a simulation experiment [23] was conducted to compare its performance with those of existing policies. From the experimental results, the integrated method improves the capacity of content-based retrieval significantly in maximizing the criteria of recall and precision. This method is also applied to audiovisual features. In addition, the determination of a CPT is application-dependant. It can be generated automatically or manually.

3. Query Language and Processing

3.1 Query Language

For facilitating the descriptions of query language and processing we introduce the notations as follows.

RVD:	the set of all the video sources
v.EVT:	the set of all the events in video *v*, $v \in RVD$.
v.OBJ:	the set of all the objects in video *v*, $v \in RVD$.
$\prec_{v.E}$:	a function mapping *v.EVT* to *v.EVT*. $e_1 \prec_{v.E} e_2$ means e_1 is a child of e_2.
v.DM:	the set of all the domains in video *v*, $v \in RVD$.
$\prec_{v.D}$:	a function mapping *v.DM* to *v.DM*. $d_1 \prec_{v.D} d_2$ means d_1 is a child of d_2.
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 in *X*, $X \in v.DM$.

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

- ***Select*** clause: This clause is to specify the properties of the entities whose values are to be retrieved and the maximum number or the probability threshold of returned results.
- ***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*. Notice that objects and events are also temporal.
- ***Where*** clause: This clause is to specify 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 thirteen relations of two given intervals [16]. The simplified syntax of a conditional expression is defined as:

```
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, dentifier)
        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
            Else $S = S - \{s\}$
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
            Else $S = S - \{s\}$
Step 12. Rank prob(s) for all $s \in S$.
         If prob(s)>0, then return s.Pname
```

- ***Object path query***

This kind of query is to find all objects with certain kind of relation to a given object.

Example: find all students who are the members of *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
```

Several index organizations proposed to support object-oriented databases are also suitable for speeding join operations, such as multiindex, join index, nested index, and path index.

- ***Event inference query***

This kind of query is to find 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 $(r \prec_{v.E} d \text{ or } d \prec_{v.E} r)$
            Then $R = R \cup \{d\}$ and set prob(d)=0
Step 3. Locate a topological order $\langle r_1, r_2, \dots, r_q \rangle$ 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 $\text{prob}(r_i) = \max(prob(r_i), \sum_u prob(r_i \mid u) * \prod_j prob(u_j))$,
            where $u$ is the vector of children of $r_i$, $< u_1, u_2, \dots >$
Step 5. Rank prob(r) for all $r \in R$.
        If prob(r)>0, then return r.Pname
```

For each event inference query, we find a set of evidence events first, and then apply the causal inference algorithm [18] of Bayesian networks to evaluate the probabilities of the other events in an event hierarchy.

- ***Temporal query***

Temporal queries take temporal information as inputs. We make use of a two dimensional index structure to evaluate temporal queries, such as R*-tree, SS-tree and SR-tree [13]. The keys are the frame intervals of a given object/event. Hence, a temporal query can be viewed as a range query in a two dimensional space.

4. Implementation

We have implemented a prototype content-based video retrieval system based on the proposed model on the MS-Windows environment. The overall architecture illustrated in Fig. 4 consists of three major modules: *interface*, *content management*, and *storage modules*. In the interface module, *authoring tool* extracts objects and events from video data, and allows users to annotate their content. Notice that users need not remember the exact syntax of annotation language for the authoring interfaces will direct users to process it, illustrated from Fig. 5 to Fig. 8. *Query tool* shown as Fig. 9 is responsible for the correctness of SQL-like queries by parsing users' inputs.

```
<expression> ::= <expression> <bool 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>
<bool op> ::= AND | OR
<temporal op> ::= START, FINISH, BEFORE, MEET,
                  OVERLAP, DURING, EQUAL,
                  INTERSECT
<compare op> ::= ≈ | = | < | > | ≦ | ≧ | ⊂ | ⊃ | ⊆ | ⊇
<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 the domain variables of video, events, and objects individually defined in the From clause. The keyword *CONTAIN* denotes the relations among them. The query unit, *A CONTAIN B*, means *A* is the entity where *B* appears.

This query language makes use of domain names as inputs because users are usually familiar with the domains of interesting things. For example, it is more natural 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 stated later can be identified by its query format. This helps query processors analyze the inputs.

3.2 Query Processing

A semantic information query is to inquire the information about objects, events, and their relationships. We classify it into several more detailed queries as follows:

- ***Elementary domain query***

This kind of query is to find all objects/events of the same domain.

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

Query lang:
```
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 $t.Pname$
```

Several traditional index structures facilitate the step 1 and 2, such as B+ tree and hashing.

- ***Elementary content query***

An elementary content query is to find 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 $prob(r)$ is a
            probability denoting how r satisfies the condition Clist
            and can be treated as a similarity function
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 problem of matching annotation structures with query forms.

- ***Object-event relation query***

These kinds of queries correspond to elementary content queries except their conditions must be associated with object/event identifiers. They are to find the objects occurring in a given event or to find the events in which a given object appears.

Example: find the students who are playing basketball in the video *campus*.

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

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 = $Vname$
```

Presentation tool shown as Fig. 10 permits 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. *Feature manager* manages the audio-visual features of extracted events and objects. *Query processor* primarily implements the algorithms described in Section 3.2. In the storage module, we adopt CTSS [3] developed by our lab. at Chiao Tung University, Taiwan, which provides the capabilities of BLOB, clustering, indices, video data compression/decompression, and recovery.

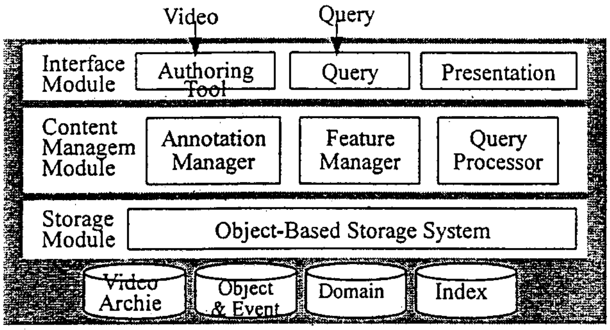


Fig. 4: The architecture of the prototype system.

5. Conclusion

In this paper, we have proposed a video data model, which is comprised of an object net and an event hierarchy. Unlike previous methods [1,4,12,17,20,21], the proposed model clearly describes the structures of objects and events in video data and the relationships among them to provide the advantages of sharability. The division of objects and events helps users understand video content directly, abstract important features early, and query/browse video information detailedly. This model is rather generalized into the applications, where a set of objects interact with one another, such as a movie with lots of actors, a soccer game with lots of players, and so on. To improve content-based access, event hierarchies integrate inheritance by context with conditional dependence relationships. The experimental results [23] show that this policy outperforms the existing ones substantially in maximizing the criteria of recall and precision. Moreover, an event hierarchy can be extended to represent multi-level abstractions of multi-dimensional data like images. The integrated policy can be applied as well.

Regarding semantic annotations, conventional approaches either cannot describe complex scenarios [1,9,17,21] or suffer from computation complexity [4,12,14]. We propose a nested annotation language to support a great expressive power, which needs not analyze free text statistically or parse natural language smartly. It can be decomposed and processed efficiently.

Our further work is to plan to combine graphical user interfaces with SQL-like query language so as to allow users to query semantic content of video data and to indicate visual features and spatial relations of video entities at the same time. We also intend to extend the data model to support fuzzy queries and apply our system to more applications in the real world.

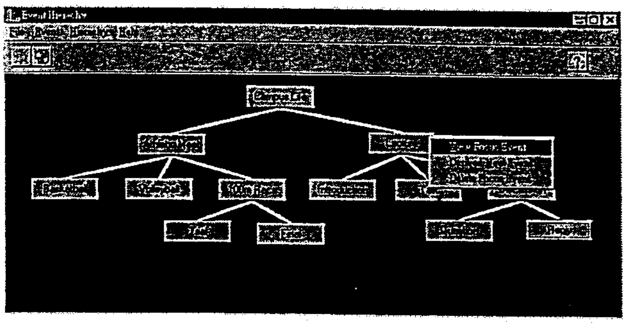
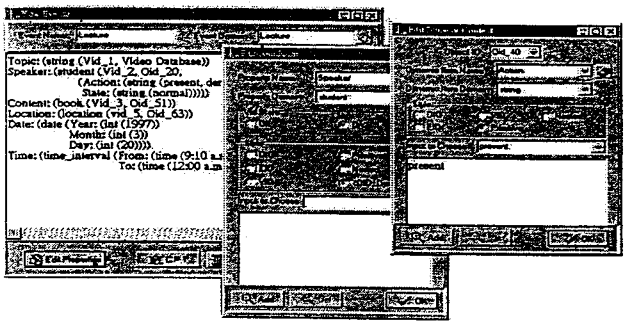


Fig. 5: The authoring interface of event hierarchies.



```
Topic: (string (Vid_1, Video Database))
Speaker: (student (Vid_2, Oid_20,
                  (Action: (string (present, der
                   State: (string (normal))))
Content: (book (Vid_3, Oid_51))
Location: (location (vid_5, Oid_63))
Date: (date (Year: (int (1997))
             Month: (int (3))
             Day: (int (20)))).
Time: (time_interval (From: (time (9:10 a.
                       To: (time (12:00 a.m
```

Fig. 6: An example of event *Lecture* annotation.

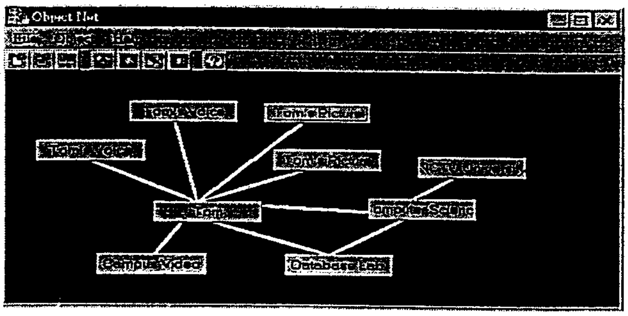
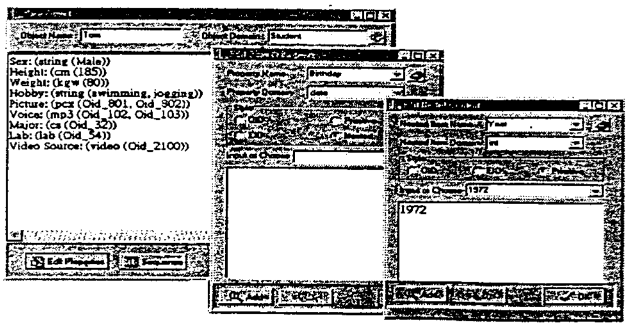
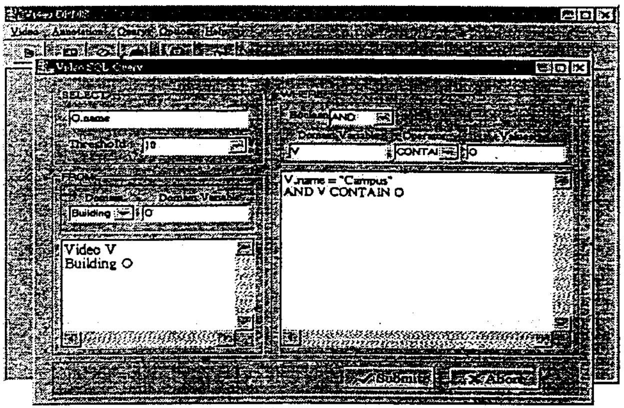


Fig. 7: The authoring interface of object nets.



```
Sex: (string (Male))
Height: (cm (185))
Weight: (kgw (80))
Hobby: (string (swimming, jogging))
Picture: (pcx (Oid_801, Oid_802))
Voice: (mp3 (Oid_102, Oid_103))
Major: (cs (Oid_32))
Lab: (lab (Oid_34))
Video Source: (video (Oid_2100))
```

Fig. 8: An example of object *Tom* annotation.



O.name

10

Building O

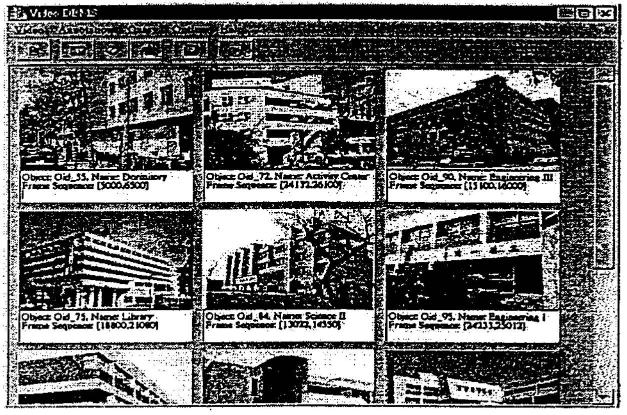
Video V
Building O

AND

V CONTAI O

V.name = "Campus"
AND V CONTAIN O

Fig. 9: An example query of finding out all buildings.



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. 10: The Query results of searching the object *Building*.

References

[1] S. Adali, K. S. Candan, S. S. Chen, K. Erol, and V. S. Subrahmanian, "The Advanced Video Information System: Data Structure and Query Processing," Multimedia Systems, vol. 4, pp. 172-186, 1996.

[2] M. L. Cascia and E. Ardizzone, "JACOB: Just A Content-Based Query System for Video Databases," In Proc. of ICASSP-96, pp. 7-10, May 1996.

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

[4] T. S. Chua and L. Q. Ruan, "A Video Retrieval and Sequencing System," ACM Transactions on Information Systems, vol. 13, no. 4, pp. 373-407, Oct. 1995.

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

[6] N. Dimitrova, T. McGee, and H. Elenbaas, "Video Keyframe Extraction and Filtering: A Keyframe is not a Keyframe to Everyone," In Proc. of the Sixth International Conference on Information and Knowledge Management, p.113-120, 1997.

[7] A. K. Elmagarmid, H. Jiang, A. A. helal, A. Joshi, and M. Ahmed, Video Database Systems, Kluwer Academic Publishers, 1997.

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

[9] R. Hjelsvold and R. Midtstraum, "Databases for Video Information Sharing," In Proc. of SPIE – The International Society for Optical Engineering, pp. 268-279, 1995.

[10] T. S. Huang, S. Mehrotra, and K. Ramchandran, "Multimedia Analysis and Retrieval System Project," In Proc. of 33rd Annual Clinic on Library Application of Data Processing – Digital Image Access and Retrieval, 1996.

[11] E. Hwang and V. S. Subrahmanian, "Query Video Libraries," Journal of Visual Communication & Image Representation, vol. 7, no. 1, pp. 44-60, Mar 1996.

[12] H. Jiang, D. Montesi, and A. K. Elmagarmid, "VideoText Database Systems," In Proc. of the 1997 IEEE International Conference on Multimedia Computing and Systems, pp. 344-351, 1997.

[13] N. Katayama and S. Satoh, "The SR-tree: An Index Structure for High-Dimensional Nearest Neighbor Queries," In Proc. of ACM SIGMOD, vol. 26, no. 2, pp. 369-380, June 1997.

[14] Y. B. Kim and M. Shibata, "Content-Based Video Indexing and Retrieval – A Natural Language Approach," IEICE Transactions on Information and Systems, vol. E79-D, no. 6, pp. 695-705, June 1996.

[15] J. C. M. Lee, Q. Li, and W. Xiong, "VIMS: A Video Information Management System," Multimedia Tools and Applications, vol. 4, pp. 7-28. 1997.

[16] T. D. C. Little, and A. Ghafoor, "Interval-Based Conceptual Models for Time-dependent Multimedia Data," IEEE Transactions on Knowledge and Data Engineering, vol. 5, no. 4, pp. 551-563, Aug. 1993.

[17] E. Oomoto and K. Tanaka, "OVID: Design and Implementation of a Video-Object Database System," IEEE Transaction on Knowledge and Data Engineering, vol. 5, No. 4, pp. 629-642, Aug. 1993.

[18] S. J. Russell and R. Norvig, Artificial Intelligence, Prentice-Hall, 1995.

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

[20] 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.

[21] R. Weiss, A. Duda, and D. K. Gifford. "Content-Based Access to Algebraic Video," IEEE International Conference Multimedia Computing and Systems, pp. 140-151, 1994.

[22] J. K. Wu, A. D. Narasimhalu, B. M. Mehtre, C. P. Lam, and Y.J. Gao, "CORE: a Content-based Retrieval Engine for Multimedia Information Systems," Multimedia Systems, vol. 3, pp. 25-41, 1995.

[23] W. P. Yang and P. J. Cheng, "Design and Analysis of a video database management system," Technical Report, NSC88-2213-E-009-016, National Chiao Tung University, Hsinchu, Taiwan, R.O.C., 1998.

[24] A. Yoshitaka, S. Kishida, M. Hirakawa, and T. Ichikawa, "Knowledge-Assisted Content-Based Retrieval for Multimedia Databases," IEEE Multimedia, vol. 1, no. 4, pp. 12-21, 1994.