A Database Approach for Modeling and Querying Video Data[*]

Cyril Decleir
LISI-INSA, Bât. 501
20, avenue Albert Einstein
F-69621 Villeurbanne
cdecleir@lisi.insa-lyon.fr

Mohand-Said Hacid[†]
LuFG Theoretical Computer Science
RWTH Aachen
Ahornstraße 55, 52074 Aachen, Germany
hacid@cantor.informatik.rwth-aachen.de

Jacques Kouloumdjian
LISI-INSA, Bât. 501
20, avenue Albert Einstein
F-69621 Villeurbanne
koulou@lisi.insa-lyon.fr

Abstract

Indexing video data is essential for providing content based access. In this paper, we consider how database technology can offer an integrated framework for modeling and querying video data. As many concerns in video (e.g., modeling and querying) are also found in databases, databases provide an interesting angle to attack many of the problems. From a video applications perspective, database systems provide a nice basis for future video systems. More generally, database research will provide solutions to many video issues even if these are partial or fragmented. From a database perspective, video applications provide beautiful challenges. Next generation database systems will need to provide support for multimedia data (e.g., image, video, audio). These data types require new techniques for their management (i.e., storing, modeling, querying, etc.). Hence new solutions are significant.

This paper develops a data model and a rule-based query language for video content based indexing and retrieval. The data model is designed around the object and constraint paradigms. A video sequence is split into a set of fragments. Each fragment can be analyzed to extract the information (i.e., symbolic descriptions) of interest that can be put into a database. This database can then be searched to find information of interest. Two types of information are considered: (1) the entities (i.e., objects) of interest in the domain of a video sequence, (2) video frames which contain these entities. To represent these information, our data model allows facts as well as objects and constraints. We present a declarative, rule-based, constraint query language that can be used to infer relationships about information represented in the model. The language has a clear declarative and operational semantics.

1. Introduction

With recent progress in compression technology, it is possible for computer to store huge amount of pictures, audio and even video. If such media are widely used in today's communication (e.g., in the form of home movies, education and training, scholarly research, and corporate enterprise solutions), efficient computer exploitation is still lacking. Many databases should be created to face the increasing development of advanced applications, such as video on demand, video/visual/multimedia databases, monitoring, virtual reality, internet video, interactive TV, video conferencing and video email, etc. Though only a partial list, these advanced applications need to integrate video data for complex manipulations.

Video analysis and content retrieval based on semantics require multi-disciplinary research effort in areas such as computer vision, image processing, data compression, databases, information systems, etc. (see [26]). Therefore, video data management poses special challenges which call for new techniques allowing an easy development of

[*]This work is partially supported by France Télecom (through CNET/CCETT) research contract no. 96-ME-17.

[†]Work supported in part by the Deutscher Akademischer Austausch Dienst (DAAD) grant A/98/05793.

applications. Facilities should be available for users to view video material in a non-sequential manner, to navigate through sequences, to build new sequences from others, etc. To facilitate retrieval, all useful semantic objects and their features appearing in the video must be appropriately indexed. The use of keywords or free text [28] to describe the necessary semantic objects is not sufficient [9]. Additional techniques are needed. As stated in [9], the issues that need to be addressed are: (1) the *representation of video information in a form that facilitates retrieval and interaction*, (2) the organization of this information for efficient manipulation, and (3) the user-friendly presentation of the retrieved video sequences. Being able to derive an adequate content description from a video, however, does not guarantee a satisfactory retrieval effectiveness, it is only a necessary condition to this end. It is mandatory the video data model be powerful enough to allow both the expression of sophisticated content representation and their proper usage upon querying a video database. For example, the time-dependent nature of video is of considerable importance in developing adequate data models and query languages.

Many features of database systems seem desirable in a video context: secondary storage management, persistence, transactions, concurrency control, recovery, versions, etc. In addition, a database support for video information will help sharing information among applications and make it available for analysis. The advantages (in general and for video in particular [22, 15]) of database technology, such as object-oriented databases, are (1) the ability to represent complex data, and (2) more openness to the external world (e.g., Web, Java, CORBA, languages bindings) than traditional database systems. However, as existing database technology is not designed to manage digital video as first class media, new techniques are required for organizing, storing, manipulating, retrieving by content, and automatic processing and presentation of visual content. Although some tools for video exploitation are available, their use often amounts to displaying video in sequence, and most modeling methods have been developed for specific needs. In many cases, query languages concentrate on extraction capabilities. Queries over video data are described only by means of a set of pre-defined, ad hoc operators, often incorporated to SQL, and are not investigated in *theoretical framework*. One can argue that logic-based database query languages appropriately designed to support video specific features should form a sound basis for query languages.

From database point of view, video data presents an interesting challenge. Future database systems must cover the range of tasks associated with the management of video content including feature extraction, indexing, querying, and developing representation schemes and operators. For example, the data model should be expressive enough to capture several characteristics inherent to video data, such as movements, shapes, variations, events, etc. The query language should allow some kind of reasoning, to allow, for example, virtual editing [19], and should be able to perform exact as well as partial or fuzzy matching (see [4]).

Despite the consensus of the central role video databases will play in the future, there is little research work on finding *semantic foundations for representing and querying video information*. This paper is a contribution in this direction. The framework presented here integrates formalisms developed in constraint, object and sequence databases. We propose a hybrid data model for video data and a declarative, rule-based, constraint query language, that has a clear declarative and operational semantics. We make the following contributions:

1. We develop a simple video data model on the basis of relation, object and constraint paradigms. Objects of interest and relationships among objects can be attached to a generalized interval[1] either through attribute/value pairs or relations.

2. We propose a declarative, rule-based, constraint query language that can be used to infer relationships from information represented in the model, and to intentionally specify relationships among objects. It allows a high level specification of video data manipulations.

The model and the query language use the point-based approach to represent periods of time associated with generalized intervals. First-order queries can then be conveniently asked in a much more declarative and natural way [27]. There has been some previous research on the power of constraints for the implicit specification of temporal data [8].

The model and the query language will be used as a core of a video document archive prototype by both a television channel and a national audio-visual institute. *To the best of our knowledge, this is the first proposal of a formal rule-based query language for querying video data*.

Paper outline: This paper is organized as follows. Section 2 discusses related work. Section 3 presents some useful definitions. Section 4 formally introduces the video data model. Section 5 describes the underlying query language. Section 6 draws conclusions.

2. Related Work

With the advent of multimedia computers (PCs and workstations), the world-wide web, and standard and powerful compression techniques[2], it becomes possible to digitize and store common human media, such as pictures,

[1] A generalized interval is a set of pairwise non overlapping fragments in a video sequence.

[2] Such as *MPEG-I* [10] [11] and its successors.

sounds, and video streams worldwide. Nevertheless, storing is the minimal function we are to expect from a computer, its power should also be aimed at *content indexing* and *retrieval*. Two main approaches have been experienced: fully automated content indexing approach, and the approach based on human-machine interaction. Some fully automated research systems have been developed, among others, *VIOLONE* [29] and *JACOB* [18]. However, because of the weakness of content analysis algorithms, they focus on a very specific exploitation. On the other hand, much more aided video content indexing systems have been designed, among others, *OVID* [22], *AVIS* [1], or *VideoStar* [14].

In the context of image and video data, queries can be formulated using several techniques, which fall broadly into two categories: *textual* and *visual*. Several systems have been developed to retrieve visual data based on color, shape, size, texture, image segments, keyword, relational operators, objects, and bibliographic data (see, among others, [6, 7, 13, 16, 21]). In this paper, we focus on textual languages.

The work presented here is closest to and complements the ones in [20, 1, 22, 14].

Meghini [20] proposed a retrieval model for images based on first-order logical language which spans along four main dimensions: visual, spatial, mapping and content. Queries on images can address anyone of these dimensions or any combination of them. In the proposed model, objects cannot be characterized by attributes. Every entity is described by means of relations (predicates). For example, objects' shapes cannot be stated in a declarative and equational manner, as it is the case in our model.

Oomoto and Tanaka [22] proposed a schema-less video-object data model. They focus on the capabilities of object-oriented database features (their extension) for supporting schema evolution and to provide a mechanism for sharing some descriptive data. A video frame sequence is modeled as an object with attributes and attribute values to describe its contents. A semantically meaningful scene is a sequence of (not always continuous) video frames. An interval is described by a pair of a starting frame and an ending frame. It denotes a continuous sequence of video frames. They introduced the notion of inheritance based on the interval inclusion relationship. By means of this notion different video-objects may share descriptional data. Several operations, such as interval projection, merge and overlap are defined to compose new objects from other objects. They provide the user with the SQL-based query language VideoSQL for retrieving video-objects. This model does not allow the description and the definition of relationships among objects within a video-object. The content of a video-object is described in terms of attribute-values of this video-object. Semantic objects are considered as values of attributes of video-objects.

Adali et al. [1] have developed a formal video data model, and they exploit spatial data structures for storing such data. They emphasized some kinds of human-level information in video: objects of interest, activities, events and roles. A given video is divided into a sequence of frames which constitute logical divisions of the video. Associated with each object/event is a set of frame-sequences. Each frame sequence can be viewed as a frame segment. Events are characterized by a set of attributes describing their context. For example, the event give party may be characterized by the multi-valued attribute *host* whose values are *Philip* and *Brandon* and the attribute guest whose value is *Rupert*. In this framework, objects other than events have no complex structure. The only relationships among objects are those given implicitly through the description of events. They also developed a simple SQL-like video query language which can be used to retrieve videos of interest and extracts from them the relevant segments of the video that satisfy the specified query conditions.

These two proposals provide an interval-based approach to represent the periods of time associated with frames of interest in a video sequence.

Hjelsvold and Midtstraum [14] proposed a generic video data model. Their proposal combines ideas from the stratification [2] and the segmentation [9] approaches. The objective is to develop a framework where structuring, annotations, sharing and reuse of video data become possible. Their model is built upon an enhanced-ER model. A simple SQL-like video query language with temporal interval operators (e.g., *equals*, *before*, etc.) is provided. This work concentrates on the structural part of a video in order to support video browsing. Thematic indexing is based on annotations, which give a textual description of the content of frame sequences. In contrast, we allow a more elaborated and structured description of the content of frame sequences.

With regard to the modeling, we extended these works by allowing the description of the contents of video sequences by means of first class citizen objects of the data model and by relating them to each others either through attributes or through explicit relation names, leading to more expressive relationships to link objects. Hence, video frames (which we call generalized intervals) as well as semantic objects (objects of interest in a generalized interval) are modeled and manipulated at the same level. Special queries, like spatial and temporal ones, can be expressed in a much more declarative manner. Generalized intervals, as well as semantic objects and relationships among these elements can be described, making a video sequence appropriate for different applications.

3. Basic Definitions

This section provides the preliminary concepts that will be used to design the video data model and the underlying rule-based, constraint query language.

Definition 1 (Concrete Domains) *A concrete domain $\mathcal{D} = (dom(\mathcal{D}), pred(\mathcal{D}))$ consists of:*

- *the domain $dom(\mathcal{D})$,*
- *a set of predicate symbols $pred(\mathcal{D})$, where each predicate symbol $P \in pred(\mathcal{D})$ is associated with an arity n and an n-ary relation $P^{\mathcal{D}} \subseteq dom(\mathcal{D})^n$,*

An example of a concrete domain is the set of (nonnegative) integers with comparisons $(=, <, \leq, \geq, >)$.

In the following, we assume *entailment* of conjunctions or disjunctions over $pred(\mathcal{D})$ is decidable.

Definition 2 (Dense Linear Order Inequality Constraints) *Dense order inequality constraints are all formulas of the form $x\theta y$ and $x\theta c$, where x, y are variables, c is a constant, and θ is one of $=, <, \leq$ (or their negation $\neq, \geq, >$). We assume that these constants are interpreted over a countably infinite set $\mathcal{D}$ with a binary relation which is a dense order. Constants, $=$, $<$, and $\leq$ are interpreted respectively as elements, equality, the dense order, and the irreflexive dense order of the concrete domain $\mathcal{D}$.*

Complex constraints are built from primitive (atomic) constraints by using logical connectives. We use the special symbol, $\Rightarrow$, to denote the entailment between constraints, that is, if c_1 and c_2 are two constraints, we write $c_1 \Rightarrow c_2$ for c_1 entails c_2. $c_1 \Rightarrow c_2$ is *satisfiable* if and only if the constraint $c_1 \wedge \neg c_2$ is *unsatisfiable*.

Techniques for checking satisfiability and entailment for order constraints over various domains have been studied. Regarding expressive power and complexity of linear constraint query languages, see [12].

Definition 3 (Set-Order Constraints) *Let $\mathcal{D}$ be a domain. A set-order constraint is one of the following types:*

$$c \in \widetilde{X}, \widetilde{X} \subseteq s, s \subseteq \widetilde{X}, \widetilde{X} \subseteq \widetilde{Y}$$

where c is a constant of type $\mathcal{D}$, s is a set of constants of type $\mathcal{D}$, and $\widetilde{X}, \widetilde{Y}$ denote set variables that range over finite sets of elements of type $\mathcal{D}$.

Our set-order constraints are a restricted form of set constraints [5], involving $\in$, $\subseteq$, and $\supseteq$, but no set functions such as $\cup$ and $\cap$.

Note that the constraint $c \in \widetilde{X}$ is a derived form since it can be rewritten as $\{c\} \subseteq \widetilde{X}$.

Satisfaction and entailment of conjunctions of set-order constraints can be solved in polynomial-time using a quantifier elimination algorithm given in [25].

This class of constraints play an important role in declaratively constraining query answers.

Definition 4 (Time Interval) *An interval i is considered as an ordered pair of real numbers (x_1, x_2), $x_1 \leq x_2$. This definition refers to the predicate $\leq$ of the concrete domain $\mathbb{R}$. If t is a time variable, then an interval (x_1, x_2) can be represented by the conjunction of the two primitive dense linear order inequality constraints $x_1 \leq t$ and $t \leq x_2$.*

Definition 5 (Generalized Time Interval) *A generalized time interval, or simply a generalized interval, is a set of pairwise non overlapping intervals. Formally, a generalized time interval can be represented as a disjunction of time intervals.*

4. Video Data Model

To naturally capture the entities and relationships among entities within a video sequence, we resort to the following basic paradigms:

- *Objects and object identity* Objects are entities of interest in a video sequence. In our model, we refer to objects via their logical object identities, which are nothing but syntactic terms in the query language. Any logical oid uniquely identifies an object. In this paper, we will be using the word ”object identity” (or even ”object”) to refer to ids at logical level. We have essentially two types of objects: (1) generalized interval objects, which are abstract objects resulting from splitting a given video sequence into a set of smaller sequences; (2) semantic objects which are entities of interest in a given video sequence.
- *Attributes* Objects are described via attributes. If an attribute is defined for a given object, then it also has a value for that object.
- *Relations* It has been argued many times that objects do not always model real world in the most natural way, and there are situations when the use of relations combined with objects leads to more natural representation. Although relations can be encoded as objects, this is not the most natural way of handling relations and so we prefer to have relations as first-class language constructs.

We assume the existence of the following countably infinite and pairwise disjoint sets of atomic elements:

- relation names $\mathcal{R} = \{R_1, R_2, \ldots\}$;
- attributes $\mathcal{A} = \{A_1, A_2, \ldots\}$;
- (atomic) constants $\mathcal{D} = \{d_1, d_2, \ldots\}$;
- object identities or oid's $\mathcal{ID} = \{id_1, id_2, \ldots\}$. In the following, we distinguish between object identities for entities and object identities for generalized intervals.

Furthermore, in order to be able to associate a time interval to a generalized interval object, we allow a restricted form of dense linear order inequality constraints to be values of attributes. We define the set $\tilde{C}$ whose elements are:

- Primitive (atomic) constraints of the form $t\theta c$ where t is a variable, c is a constant, and θ is one of $<, =, >$;
- conjunctions, and disjunctions of primitive constraints.

Definition 6 (value) *The set of values is the smallest set containing $\mathcal{D} \cup \mathcal{ID} \cup \tilde{C}$ and such that, if $v_1, \ldots, v_n$ $(n \geq 1)$ are values, then so is $\{v_1, \ldots, v_n\}$.*

Definition 7 (Video Object) *A video object (denoted* v-object*) consists of a pair (oid, v) where:*

- *oid is an object identifier which is an element of $\mathcal{ID}$;*
- *v is an m-tuple $[A_1 : v_1, \ldots, A_m : v_m]$, where A_i $(i \in [1, m])$ are distinct attribute names in $\mathcal{A}$ and v_i $(i \in [1, m])$ are values.*

If $o = (oid, v)$ with $v = [A_1 : v_1, \ldots, A_n : v_n]$, then $attr(o)$ denotes the set of all attributes in v (i.e. $\{A_1, \ldots, A_n\}$), and $value(o)$ denotes the value v, that is, $v = value(o)$. The value v_i is denoted by $o.A_i$.

5. Rule-Based, Constraint Query Language

In this section, we present the declarative, rule-based query language that can be used to reason with facts and objects in our video data model. The language consists of two constraint languages on top of which relations can be defined by means of definite clauses.

This language has a model-theoretic and fix-point semantics based on the notion of *extended active domain* of a database. The extended domain contains all generalized interval objects and their concatenations. The language has an interpreted function symbol for building new generalized intervals from others (by concatenating them). A constructive term has the form $I_1 \otimes I_2$ and is interpreted as the concatenation of the two generalized intervals I_1 and I_2.

The extended active domain is not fixed during query evaluation. Instead, whenever a new generalized interval object is created (by the concatenation operator, $\otimes$), the new object and the ones resulting from its concatenation with already existing ones are added to the extended active domain.

5.1. Syntax

To manipulate generalized intervals, our language has an interpreted function symbol for constructing[3] complex term. Intuitively, if I_1 and I_2 are generalized intervals, then $I_1 \otimes I_2$ denotes the concatenation of I_1 and I_2.

The language of terms uses three countable, disjoint sets:

1. A set $\mathbb{D}$ of constant symbols. This set is the union of three disjoint sets:
   - $\mathbb{D}_1$: a set of atomic values,
   - $\mathbb{D}_2$: a set of entities, also called object entities,
   - $\mathbb{D}_3$: a set of generalized interval objects.
2. A set $\mathcal{V}$ of variables called object and value variables, and denoted by $X, Y, \ldots$;
3. A set $\tilde{V}$ of variables called generalized interval variables, and denoted by $S, T, \ldots$;

If I_1 and I_2 denote generalized interval objects, generalized interval variables, or constructive interval terms, then $I_1 \otimes I_2$ is a *constructive* interval term.

In the following, the concatenation operator is supposed to be defined on $\mathbb{D}_3$, that is $\forall e_1, e_2 \in \mathbb{D}_3$, $e_1 \otimes e_2 \in \mathbb{D}_3$. The structure of the resulting element $e = e_1 \otimes e_2$ is defined from the structure of e_1 and e_2 as follows:
Let $e_1 = (id_1, v_1)$ and $e_2 = (id_2, v_2)$. Then $e = (id, v)$, is such that:

- $id = f(id_1, id_2)$. Here we follow the idea of [17] that the object id of the object generated from e_1 and e_2 should be a function of id_1 and id_2.
- $attr(e) = attr(e_1) \cup attr(e_2)$.
- $\forall A_i \in attr(e), e.A_i = e_1.A_i \cup e_2.A_i$.

Note that $I_1 \otimes I_1 \equiv I_1$. This means that if I is obtained from the concatenation of I_1 and I_2, then the result of the concatenation of I with I_1 or I_2 is I. This leads to the termination of the execution of constructive rules (see below the definition of constructive rule).

Definition 8 (Predicate symbol) *We define the following predicate symbols:*

[3] For concatenating generalized intervals.

- *each* $\mathsf{P} \in \mathcal{R}$ *with arity* n *is associated with a predicate symbol* P *of arity* n,
- *a special unary predicate symbol* $\mathsf{Interval}$. *It can be seen as the class of all generalized interval objects.*
- *a special unary predicate symbol* Object. *It can be seen as the class of all objects other than generalized interval objects.*

Definition 9 (Atom) *If* P *is an* n*-ary predicate symbol and* $t_1, \ldots, t_n$ *are terms, then* $\mathsf{P}(t_1, \ldots, t_n)$ *is an atom. If* O *and* O' *denote objects or object variables,* Att *and* Att' *are attribute names, and* c *is a constant value, then* $O.Att\theta c$ *and* $O.Att\ \theta\ O'.Att'$ *where* θ *is one of* $=, <, \leq$ *(or their negation* $\neq, \geq, >$*) are called inequality atoms.*

Definition 10 (Rule) *A* rule *in our language has the form:*

$$r : H \leftarrow L_1, \ldots, L_n, c_1, \ldots, c_m$$

where H *is an atom,* $n, m \geq 0$, $L_1, \ldots, L_n$ *are (positive) literals, and* $c_1, \ldots, c_m$ *are constraints.*

Optionally, a rule can be named as above, using the prefix "r : ", where r is a constant symbol. We refer to A as the head of the rule and refer to $L_1, \ldots, L_n, c_1, \ldots, c_m$ as the body of the rule.

Note that we impose the restriction that constructive terms appear only in the head of a rule, and not in the body. A rule that contains a constructive term in its head is called a constructive rule.

Recall that we are interested in using order constraints, that is arithmetic constraints involving $<, >$, but no arithmetic functions such as $+, -, *$, and set-order constraints, a restricted form of set constraints involving $\in, \subseteq$, and $\supseteq$, but no set functions such as $\cup$ and $\cap$.

Definition 11 (Range-restricted Rule) *A rule* r *is said to be range-restricted if every variable in the rule occurs in a body literal. Thus, every variable occurring in the head occurs in a body literal.*

Definition 12 (Program) *A program is a collection of range-restricted rules.*

Definition 13 (Query) *A query is of the form:*

$$Q :?q(\bar{s})$$

where q *is referred to as the query predicate, and* $\bar{s}$ *is a tuple of constants and variables.*

Example Let us give some simple examples of queries. In the following, uppercase letters stand for variables and lowercase letters stand for constants.
The query "list the objects appearing in the domain of a given sequence g" can be expressed by the following rule:

$$q(O) \leftarrow Interval(g), Object(O), O \in g.entities$$

In this example, g is a constant and O is the output variable. Here, we suppose that for a given generalized interval, the set-valued attribute "entities" gives the set of semantic objects of interest in that generalized interval. This query involves an atomic (primitive) constraint. To compute the answer set to the query, we need to check the satisfiability of the constraint $O \in g.entities$ after O being instantiated.
The query "list all generalized Intervals where the object o appears" can be expressed as:

$$q(G) \leftarrow Interval(G), Object(o), o \in G.entities$$

The query "does the object o appear in the domain of a given temporal frame $[a, b]$" can be expressed as:

$$\begin{aligned} q(o) \leftarrow\ & Interval(G), Object(o), o \in G.entities, \\ & G.duration \Rightarrow (t > a \wedge t < b) \end{aligned}$$

where t is a temporal variable. This query involves one primitive constraint $o \in G.entities$, and a complex arithmetic constraint $G.duration \Rightarrow (t > a \wedge t < b)$. To compute the answer set to the query, we need to check satisfiability of these two constraints.
The query "list all generalized intervals where the objects o_1 and o_2 appear together" can be expressed as:

$$\begin{aligned} q(G) \leftarrow\ & Interval(G), Object(o_1), Object(o_2), \\ & o_1 \in G.entities, o_2 \in G.entities \end{aligned}$$

or equivalently by:

$$\begin{aligned} q(G) \leftarrow\ & Interval(G), Object(o_1), Object(o_2), \\ & \{o_1, o_2\} \subseteq G.entities \end{aligned}$$

The query "list all pairs of objects, together with their corresponding generalized interval, such that the two objects are in the relation "Rel" within the generalized interval", can be expressed as:

$$\begin{aligned} q(O_1, O_2, G) \leftarrow\ & Interval(G), Object(O_1), Object(O_2), \\ & O_1 \in G.entities, O_2 \in G.entities, Rel(O_1, O_2, G) \end{aligned}$$

The query "find the generalized intervals containing an object O whose value for the attribute A is val" can be expressed as:

$$q(G) \leftarrow Interval(G), Object(O), O \in G.entities, O.A = val$$

5.2. Inferring new relationships

Rules can be used to infer (specify) new relationships, as facts, between existing objects.

Example Suppose we want to define the relation *contains*, which holds for two generalized interval objects G_1 and G_2 if the time interval associated with G_1 overlaps the time interval associated with G_2. This can be expressed as follows:

$$contains(G_1, G_2) \leftarrow \\ Interval(G_1), Interval(G_2), G_2.duration \Rightarrow G_1.duration$$

G_1 and G_2 are in the relation $contains$ if the constraint ($duration\text{-}filler$) associated with G_2 entails the one associated with G_1.

If we want to define the relation $same\text{-}object\text{-}in$ of all pairs of generalized intervals with their common objects, we write the following rule:

$$same - object - in(G_1, G_2, O) \leftarrow \\ Interval(G_1), Interval(G_2), Object(O), \\ O \in G_1.entities, O \in G_2.entities$$

The following rule constructs concatenations of generalized intervals that have some objects, say o_1 and o_2 in common.

$$concatenate - Gintervals(G_1 \otimes G_2) \leftarrow \\ Interval(G_1), Interval(G_2), Object(o_1), Object(o_2), \\ \{o_1, o_2\} \subseteq G_1.entities, \{o_1, o_2\} \subseteq G_2.entities$$

Our language has a declarative model-theoretic and an equivalent fix-point semantics.

For Datalog with set order constraints, queries are shown to be evaluable bottom-up in closed form and to have DEXPTIME-complete data complexity [24]. For a rule language with arithmetic order constraints, the answer to a query can be computed in PTIME data complexity [25]. As a consequence, we obtain a lower bound complexity for query evaluation in our rule based query language.

6. Conclusion and Future Work

There is a growing interest in video databases. We believe that theoretical settings will help understanding related modeling and querying problems. This will lead to the development of powerful systems for managing and exploiting video information.

In this paper, we have addressed the problem of developing a video data model and a formal, rule-based, constraint query language that allow the definition and the retrieval by content of video data. The primary motivation of this work was that objects and time intervals are relevant in video modeling and the absence of suitable supports for these structures in traditional data models and query languages represent a serious obstacle.

The data model and the query language allow (a) an abstract representation of the visual appearance of a video able to support modeling and retrieval techniques; (b) a semantic data modeling styled representation of the video content, independent from how the content information is obtained; (c) a relational representation of the association between objects within a video sequence.

This paper makes the following contributions. (1) We have developed a simple and useful video data model that integrates relations, objects and constraints. Objects allow to maintain an object-centered view inherent to video data. Attributes and relations allow to capture relationships between objects. It simplifies the indexing of video sequences. (2) We have developed a declarative, rule-based, constraint query language to reason about objects and facts, and to build new sequences from others. This functionality can be useful in virtual editing for some applications. The language provides a much more declarative and natural way to express queries.

Due to the complex nature of video queries, the query language presents a facility that allows a user to construct queries based on previous queries. In addition, as all properties inherent to image data are also part of video data, the framework presented here naturally applies to image data.

There are many interesting directions to pursue.

- An important direction of active research is to extend our framework to incorporate abstraction mechanisms such as *classification*, *aggregation*, and *generalization*.

- Another important direction is to study the problem of sequence presentation. Most existing research systems use template-based approach [3] to provide the automatic sequencing capability. With this approach, a set of sequencing templates is predefined to confine the user's exploration to a certain sequencing order. The problem is that this approach is domain-dependent and relies on the availability of a suitable template for a particular query. We believe that a framework based on declarative graphical (visual) languages [23] will offer more possibilities and flexibility in the specification of sequence presentations.

We are investigating these important research directions.

References

[1] S. Adali, K. S. Candan, S.-S. Chen, K. Erol, and V. S. Subrahmanian. Advanced video information system: Data structures and query processing. *ACM-Springer Multimedia System*, 4:172–186, 1996.

[2] T. G. Aguierre Smith and G. Davenport. The stratification system, a design environment for random access video. Technical report, The Media Lab, M.I.T, 1992.

[3] T. G. Aguierre-Smith and N. C. Pincever. Parsing movies in context. In *Proceedings of USENIX (Summer 1991), USENIX Association, Berkeley, California*, pages 157–168, 1991.

[4] G. Ahanger, D. Benson, and T. Little. Video query formulation. In W. Niblack and R. C. Jain, editors, *Storage and retrieval for image and video database III (SPIE'95), San Jose, California*, pages 280 – 291, Feb. 1995.

[5] A. Aiken and E. L. Wimmers. Solving systems of set constraints (extended abstract). In I. C. S. Press, editor, *Proceedings of the 7th Annual IEEE Symposium on Logic in Computer Science, Santa Cruz, California*, pages 329–340, 1992.

[6] A. D. Bimbo, M. Campanai, and P. Nesi. A three-dimensional iconic environment for image database querying. *IEEE Trans. on Software Engineering*, 19(1):997–1011, 1993.

[7] N. Chang and K. Fu. Picture query languages for pictorial data-base systems. *Computer*, 14(11):23–33, Nov. 1981.

[8] J. Chomicki and T. Imielinski. Relational specifications of infinite query answers. *ACM Transactions on Database Systems*, 18(2):181–223, June 1993.

[9] T.-S. Chua and L.-Q. Ruan. A video retrieval and sequencing system. *ACM Transactions on Information Systems (TOIS)*, 13(4):373–407, Oct. 1995.

[10] D. L. Gall. Mpeg: a video compression standard for multimedia applications. *Communications of the ACM*, 1991.

[11] D. L. Gall. The mpeg compression algorithm: a review. *Communications of the ACM*, 1991.

[12] S. Grumbach, J. Su, and C. Tollu. Linear constraint query languages expressive power and complexity. In D. Leivant, editor, *Proceedings of the International Workshop on Logic and Computational Complexity (LCC'94), Indianapolis, IN, USA*, pages 426–446, Oct.

[13] K. Hirata and T. Kato. Query by visual example. In *Proceedings of the Third International Conference on Extended Database Technology (EDBT'92), Vienna, Austria*, pages 56–71, Mar. 1992.

[14] R. Hjelsvold and R. Midtstraum. Modelling and querying video data. In *Proceedings of the 20th International Conference on Very Large Databases (VLDB'94), Santiago, Chile*, 1994.

[15] L. Huang, J. C.-M. Lee, Q. Li, and W. Xiong. An experimental video database management system based on advanced object-oriented techniques. In I. K. Sethi and R. C. Jain, editors, *Storage and retrieval for image and video database IV (SPIE'96), San Jose, California*, pages 158 – 169, Feb. 1996.

[16] T. Joseph and A. cardenas. Picquery: A high level query language for pictorial database management. *IEEE Trans. on Software Engineering*, 14(5):630–638, May 1988.

[17] M. Kifer and J. Wu. A logic for object-oriented logic programming (maier's o-logic revisited). In *Proceedings of the 1989 Symposium on Principles of Database Systems (PODS'89), Philadelphia, Pennsylvania*, pages 379–393, Mar. 1989.

[18] M. La Cascia and E. Ardizzone. Jacob : Just a content-based query system for video databases. In *ICASSP-96, Atlanta*, 1996.

[19] W. E. Mackay and G. Davenport. Virtual video editing in interactive multimedia applications. *Communications of the ACM*, 32(7):802–810, July 1989.

[20] C. Meghini. Towards a logical reconstruction of image retrieval. In I. K. Sethi and R. C. Jain, editors, *Storage and retrieval for image and video database IV (SPIE'96), San Jose, California*, pages 108–119, Feb. 1996.

[21] W. Niblack, R. Barber, W. Equitz, M. Flickner, D. Petkovic, and P. Yanker. The qbic project: Querying images by content using color, texture, and shape. In *IS&T/SPIE Symposium on Electronic Imaging;Science and Technology, San Jose, California*, Feb. 1993.

[22] E. Oomoto and K. Tanaka. Ovid: Design and implementation of a video-object database system. *IEEE Transactions on Knowledge and Data Engineering*, 5(4):629–643, Aug. 1993.

[23] J. Paredaens, P. Peelman, and L. Tanca. G-log: A declarative graphical query languages. In C. Delobel, M. Kifer, and Y. Masunaga, editors, *Proceedings of the Second International Conference on Deductive and Object-Oriented Databases (DOOD'91), Munich, Germany*, volume 566 of *LNCS*, pages 108–128, Dec. 1991.

[24] P. Z. Revesz. Datalog queries of set constraint databases. In G. Gottlob and M. Y. Vardi, editors, *Proceedings of the 5th International Conference on Database Theory (ICDT'95), Prague, Czeck Republic*, pages 425–438, Jan. 1995. LNCS 893.

[25] D. Srivastava, R. Ramakrishnan, and P. Z. Revesz. Constraint objects. In *Proceedings of the Second International Workshop on Principles and Practice of Constraint Programming (PPCP'94)*, number 874 in LNCS, pages 218–228. Springer Verlag, 1994.

[26] A. TOIS. Special issue in video information systems. *ACM Transactions on Information Systems (TOIS)*, 13(4), Oct. 1995.

[27] D. Toman. Point vs. interval-based query languages for temporal databases. In *Proceedings of the 1996 Symposium on Principles of Database Systems (PODS'96), Montréal, Canada*, pages 58–67, June 1996.

[28] Y. Tonomura. Video handling based on structured information for hypermedia systems. In *Proceedings of the International Conference on Multimedia Information Systems'91, Singapore*, pages 333–344. McGraw Hill, New York, 1991.

[29] A. Yoshitaka, Y. Hosoda, M. Yoshimitsu, M. Hirakawa, and T. Ichikawa. Violone : Video retrieval by motion example. *Journal of Visual languages and Computing*, 7:423–443, 1996.