Multimedia Systems
© Springer-Verlag 2002

Smart VideoText: a video data model based on conceptual graphs

F. Kokkoras[1], **H. Jiang**[2], **I. Vlahavas**[1,⋆], **A.K. Elmagarmid**[2], **E.N. Houstis**[2], **W.G. Aref**[2]

[1] Department of Informatics, Aristotle University of Thessaloniki, Thessaloniki, 54006 Greece (e-mail: {kokkoras,vlahavas}@csd.auth.gr)
[2] Computer Science Department, Purdue University, West Lafayette, IN 47907 USA (e-mail: {jiang,ake,enh,aref}@cs.purdue.edu)

Abstract. An intelligent annotation-based video data model called Smart VideoText is introduced. It utilizes the conceptual graph knowledge representation formalism to capture the semantic associations among the concepts described in text annotations of video data. The aim is to achieve more effective query, retrieval, and browsing capabilities based on the semantic content of video data. Finally, a generic and modular video database architecture based on the Smart VideoText data model is described.

Key words: Video databases, Conceptual graphs, Content-based video retrieval

1. Introduction

Video data, with its unique characteristics such as huge size, rich content, and its temporal and spatial nature, has posed many interesting challenges to the multimedia database research community. One critical problem is the modeling of video data for effective content-based indexing and user access capabilities such as querying, retrieval and browsing.

Video data can be modeled based on its visual content (such as color, motion, shape, and intensity) [13], audio content [2, 19, 24] and semantic content in the form of text annotations [9]. Because machine understanding of the video data is still an unsolved research problem, text annotations are usually used to describe the content of video data according to the annotator's understanding and the purpose of that video data. Although such content descriptions may be biased, incomplete and inaccurate, they still provide much of semantic content that cannot be obtained by current computer vision or voice recognition techniques. In general, computer vision techniques may aid in answering the question "what is in the video?" but cannot answer questions such as "what is happening in the video?" or "what is the video trying to tell us?". For example, the background information of a video stream cannot be obtained directly from the video but needs to be annotated.

Video annotation is suitable for applications such as distance learning and news video databases, but is inadequate for applications that require direct visual feature-based matching, such as face recognition. Visual content-based models are more appropriate for such applications [4].

Current video data models (such as Informedia [11], VideoText [7], VideoSTAR [9], and the algebraic model [30]), whether they use the video annotation layering (stratification) approach [7, 9, 30] or the keyword-based annotation approach [11] to represent video semantics, fail to model semantic relationships among the concepts expressed in the video. The importance of capturing video semantic associations lies in the fact that it can greatly improve the effectiveness of video querying by providing knowledge-based query processing. This is due to the fact that human beings always have multiple expressions or terms for the same or similar semantics. For example, "currency" and "euro" do not match syntactically but match conceptually. Furthermore, video semantic associations can be used for flexible, knowledge-based video browsing. Existing visual content-based video browsing approaches (e.g., see [22, 29, 32]) are mostly static. Visual representations such as the Video Scene Transition Graph [32] or salient images [29] can be precomputed using scene analysis and other image processing techniques. Video browsing based on semantic associations and background knowledge is far more flexible in the sense that navigation paths can be tailored easily to the specific needs of the user or the application. Despite its importance, the representation of semantics and semantic associations among video annotations has hitherto not been fully exploited.

In this paper, we introduce *Smart VideoText*, a knowledge- and annotation-based video data model. The goal is to achieve more effective querying, retrieval, and browsing capabilities based on the semantic associations that exist in the video data. Effectiveness is assessed by the degree of relevance of the query results to what the user has in mind. In this paper, this goal is achieved by integrating the VideoText video data model [9, 10] with the conceptual graph (CG) knowledge representation formalism [26] to model the semantic associations among video annotations. The semantic association knowledge, as well as additional information about video data, is encoded as CGs. The CGs

[⋆] Vlahavas was on sabbatical leave at Purdue University when this work was carried out.

are accompanied with a proper inference mechanism to support more flexible and effective video data access capabilities.

The rest of this paper is organized as follows. Section 2 introduces the current trends in information retrieval systems, the CG knowledge representation formalism, and a knowledge-based information retrieval model based on this formalism. Section 3 gives a summary of the current approaches to video data modeling. Section 3 also motivates and outlines our approach. The Smart VideoText data model is introduced in Sect. 4. In Sect. 5, different video data access methods supported by the model are discussed, with some examples. A system architecture based on the Smart VideoText model is proposed in Sect. 6. Finally, Sect. 7 contains concluding remarks and suggestions for future work.

2. Knowledge-based information retrieval

2.1. Trends in information retrieval

Information retrieval (IR) systems retrieve textual documents using a partial match between a user query and a proper document representation. There are three fundamental issues in building an IR system: (1) the choice of document representation; (2) the formulation of a query; and (3) a suitable matching function that determines the extent to which a document is relevant to a query. Different categories of IR models have been developed based on how these three issues are addressed in an IR system [5]. There are four main IR models: *the Boolean model*, *the cluster-based model*, *the probabilistic model*, and *the vector-space* model [18, 21, 31]. The efficiency of a model is usually determined by the so-called *precision* and *recall* measures. Precision is defined as the proportion of a retrieved set that is actually relevant. Recall is the proportion of all the relevant documents that are actually retrieved.

Recent research suggests that significant improvements in retrieval performance require techniques that, in some sense, “understand” the contents of the documents and the queries and can thus infer probable relationships between them. From this point of view, information retrieval is an inference process. The aim of the inference is to facilitate flexible matching between the terms or the concepts mentioned in the queries with those contained in the documents. The poor match between the vocabulary used to express queries and that used in the documents appears to be a major cause of poor recall. The recall of an IR model can be improved by applying domain knowledge in the conceptual representation and query processing without significantly degrading the precision.

One knowledge-based approach to information retrieval is described in [12] where the CG knowledge representation formalism is used to encode the semantic associations among the various terms within a collection of text documents. This approach leads to a knowledge-based hypertext model in which the links between text documents are implicitly defined through the relationships among the entries (concepts, conceptual relations, concept-type hierarchy etc.) of the knowledge base (KB). This approach can be viewed as a knowledge-based IR model (KBIR). A more elaborated approach on using CGs for KBIR is described in [17], where CGs are stored in Web-accessible documents and provide semantic knowledge about them.

2.2. Conceptual graphs: primitives and definition

The elements of CG theory [26] are *concept-types*, *concepts*, *relation-types* and *relations*. Concept-types represent classes of entity, attribute, state and event. Concept-types can be merged in a lattice whose partial ordering relation can be interpreted as a categorical generalization relation. A concept is an instantiation of a concept-type and is usually denoted by a concept-type label inside a box. To refer to specific individuals, a referent field is added to the concept – [book:*] (a book), [book:{*}@3] (three books), etc. Relations are instantiations of relation-types and show the relation between concepts. They are usually denoted as a relation label inside a circle. Each relation is constrained to which concepts it can connect. A CG is a finite, connected, bipartite graph consisting of concept and relation nodes (Fig. 1). Each relation is linked only to its requisite number of concepts and each concept to zero or more relations. CGs represent information about typical objects or classes of objects in the world and can be used to define new concepts in terms of old ones.

In the CG formalism, every context (situation, proposition, etc.) is a concept. Thus, contexts are represented as concepts (*contextual concepts*) whose referent field contains a nested CG. A number of operations (restriction, join, contraction, expansion, etc.) are also defined on CGs, by which one can derive allowable CGs from a *canonical basis* [26]. The canonical basis is a set of CGs from which all other CGs are derivable and it is manually constructed. The canonical formation rules enforce constraints on meaningfulness; they do not allow nonsensical graphs to be created from meaningful ones. A *maximal join* is a join of two CGs followed by a sequence of restrictions, internal joins and simplifications so that the maximum amount of matching and merging of the original graphs is achieved. Deduction with CGs is performed via a top-down resolution algorithm. A query expressed as a CG can be answered either by a direct matching with a CG of the KB or an indirect matching using inference rules.

The CG model of knowledge representation is a practical way to express a large amount of pragmatic information through assertions. All of the algorithms defined on CGs are domain-independent and every semantic domain can be described through a purely declarative set of CGs. CGs have the same model-theoretic semantics as KIF (Knowledge Interchange Format) and are currently being standardized.

3. Video data models

A video data model is a representation of video data based on its characteristics and content, as well as the applications it is intended for. Some desired capabilities of a video data model include multi-level video data abstraction; video annotation support; spatial and temporal relation support; and video data independence. Video data models can be based on the idea of video segmentation or video annotation layering [4].

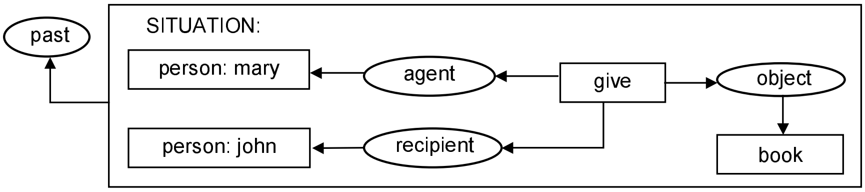


Fig. 1. CG of the senctence “Mary gives John a book”

3.1. Segmentation-based models

For a given video stream, segmentation-based models [6, 7, 28] usually use scene change detection algorithms [8] to parse and segment the video stream into a set of basic indexing units called *shots*. These shots can be matched or classified against a set of domain-specific templates (patterns) to extract higher-level semantics (such as CNN Headline News) and structures (such as episode) contained in the video. A hierarchical video stream representation can thus be built through this process. More recent techniques [23] perform video optical character recognition on news video to improve the overall understanding of the content.

The main advantage of these models is that the video indexing process can be fully automated and that they support visual content-based video data access. However, they also have some limitations, such as the following:

- Template matching has limited applicability because the similarity measure between two frame images is, in general, ill-defined and limited.
- Video streams that do not have a well-defined structure suffer from lack of applicability. For example, in a video stream of a class lecture, where there is no clear visual structure in terms of shots, segmentation using scene-change detection algorithms is difficult.
- Very limited semantics can be extracted from the template matching process which is application-specific.

3.2. Annotation-based models

The basic idea of the annotation-based models is to layer the content information (depicted by annotations) on top of the video streams, rather than segment the video data into shots. Each annotation is associated with a logical video segment, which is, in general, a subset of a video stream and is defined by the starting and ending frame numbers. Logical video segments can be overlapped or nested [14, 15, 30] in an arbitrary manner.

One of the earliest annotation-based models is the stratification model proposed by Davenport et al. [3, 25], which is based on the idea of annotation layering. Other annotation-based models, such as the generic video data model [7] and the Algebraic Video model [30], have been developed since then, while recent techniques [16] use hybrid methods (i.e. segmentation and anotation) for better content-based video indexing. Annotation layering and the notion of logical video segment have the following advantages:

- Various video access granularities can be supported, i.e. annotations can be made on logical video segments of any length, from a single frame to the whole video stream.
- The annotation information can be managed by previously existing, sophisticated information retrieval and database techniques.
- Various annotations can be linked to the same logical segment of video data to provide multiple views of the video data [30].
- Annotations can be added and deleted independent of the underlying video streams. This supports dynamic and incremental creation and modifications of video annotations.
- Video retrieval and queries based on semantic content via manual annotation can be performed at a level that current image processing and computer vision techniques cannot achieve.

An annotation-based video data model called *VideoText* [9, 10] integrates IR and video databases to support text video annotations. This model incorporates all possible interval relationships between two logical video segments. It supports IR operators like AND, OR, and NOT, as well as user queries that are based on temporal adjacency (ADJ) and interval relations (OVERLAP, AFTER, etc.) among logical video segments. The VideoText model is briefly overviewed in the next section.

3.3. The VideoText data model

The VideoText model [9] is a video data model based on the concepts of logical video segment and free text video annotations with arbitrary mapping between them. The model is defined as

$$VT = (V, T, Map)$$

where V is a set of video streams $v^i \in \hat{I}V, i = 1, \ldots, n$, and also a set of logical video segments; a *logical video segment* is a consecutive video frame sequence $[f^i_j, f^i_{j+1}, \ldots, f^i_k]$ that has meaning by itself for indexing and query purposes. A logical video segment can span from a single frame to the whole video and can also overlap with other logical video segments in arbitrary ways. T is a set of video annotations. A video annotation is a free text segment that describes the content of the corresponding logical video segment. *Map* is the mapping that defines the relationship between logical video segments and video annotations. The relationship is, in general, many-to-many, that is, a logical video segment can correspond to multiple annotations (different user views), and an annotation can be assigned to several logical video segments (annotation sharing).

Since V and T are relatively independent of each other, the model also supports dynamic creation and incremental updates of the video annotations. The VideoText model can be used to implement a video database system with a modular architecture that consists of a video data storage subsystem, an information retrieval subsystem and an integrator subsystem for coordination purposes.

The VideoText model defines a query language that supports queries based on semantic content (video annotation documents) of video data. This query language uses Boolean (AND, OR and NOT), temporal adjacency (ADJ) and interval operators (DURING, OVERLAP, etc.). The VideoText query language enables users to formulate complicated video queries which involve temporal characteristics of video data. One such query expression is:

((((George AND chair) AND (chair ADJ window))
OVERLAPS raining) BEFORE Chicago)

This query expression requires finding video segments whose annotations contain “George”, “chair”, “window”, with “chair” appearing before “window”. Also, the segments should overlap with a video clip that contains the annotation term “raining” and precedes a video clip with the annotation term “Chicago”.

3.4. Our approach

Although the VideoText model captures the temporal characteristics of the video data, it does not consider the semantic associations among video annotations. A similar problem exists in other annotation-based models [7, 30]. Semantic association, which refers to the complex relationships between different concepts and words, is important due to the fact that human beings tend to express the same or similar meaning in multiple ways or through different concepts and/or words. Modeling such associations will greatly improve the effectiveness of the video query, retrieval and browsing capabilities. For example, a user query such as “Greece AND currency” is semantically related to a logical video segment which is annotated with “. . . European Union . . . euro . . . ”. The reason is that the term “Greece” refers to a country of the “European Union” and “euro” is one type of “currency”.

Our approach proposed in this paper is a knowledge-based video data model called *Smart VideoText*. It extends the VideoText model by utilizing the CG knowledge representation formalism to capture the semantic associations among the concepts described in the video annotations. This will enable the basic annotation-based video data model to provide functionality beyond the simple operator-based video query and retrieval. Namely, the CG layer allows hypertext-like browsing and natural language querying on the video data based on the semantic relationship among video clips or logical video segments. Furthermore, the effectiveness of the operator-based retrieval will be greatly improved because the CG layer will provide semantic term matching. The details of our approach are presented in the next section.

4. The Smart VideoText model

This section describes a new video data model, called *Smart VideoText*. It extends the ideas of the VideoText model by applying KBIR techniques to capture and model the semantic associations in the video annotations and support intelligent video query, retrieval and browsing based on the semantic content of the video data.

4.1. Definition of the model

The Smart VideoText model can be defined as a 5-tuple:

$$SmartVideoText = (V, Map_1, T, Map_2, KB)$$

where V is a set of video streams ($v^i \in V, i = 1, \dots, n$) and also a set of logical video segments. Logical video segments are denoted as $[v^i_{j-k}]$, where j and k are the starting and ending frames, respectively, of the logical video segment which is part of the video stream i. T is a set of video annotations ($t^i_m \in T$), where t^i_m is a text segment that describes the contents of the logical video segments of video stream i. The mapping relation Map_1 defines the correspondence between annotations and logical video segments. For instance, the mapping $Map_1([t^3_5], [v^3_{40-980}])$ defines a one-to-one relationship between video annotation 5 of video stream 3 and the frame sequence 40 to 980 of the same video stream. This mapping is, in general, a many-to-many relationship since, in this way, an annotation could be shared among logical video segments or a logical video segment could have multiple annotations (perhaps by different users) to fulfill different application needs and to reflect possible different understanding of the same video data. In addition, logical video segments implicitly define the temporal relationship between any given two annotations within the same video.

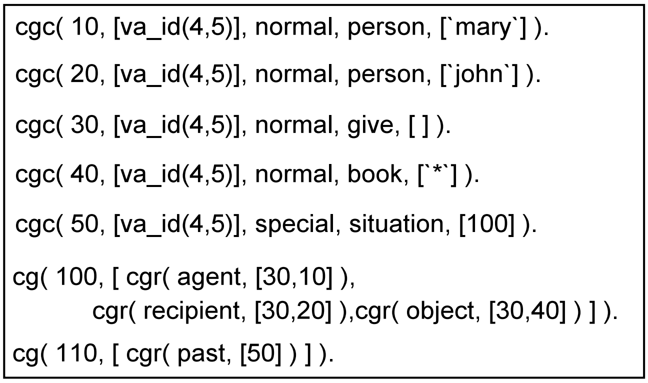
KB is the knowledge base that is encoded according to the CG knowledge representation scheme. It includes system knowledge, application knowledge and domain knowledge. More details are given in Sect. 4.2. Finally, Map_2 is a mapping relation that maps a subset (application knowledge) of the KB to video annotations T since only this knowledge is directly derived from the video annotations. Map_2 is a many-to-many relationship according to the mapping scheme described in Sect. 4.2.

4.2. Video knowledge representation

The KB in the Smart VideoText model logically consists of three parts, namely system, application, and domain knowledge:

- *System knowledge*: this mostly includes rules about how to handle CGs (formation rules, inference rules, etc.).
- *Application knowledge*: this is the knowledge related to the content of the video database; it is derived from video annotations.
- *Domain knowledge*: this includes knowledge related to but not explicitly defined in the video database, such as the type hierarchy and concept definitions which are expressed as CGs.

Although all three parts are used during knowledge-based video access, only the application knowledge is involved in the mapping Map_2 since only this knowledge is explicitly derived from video annotations. The extent of the automation of this knowledge derivation process varies, and



```
cgc( 10, [va_id(4,5)], normal, person, [`mary`] ).

cgc( 20, [va_id(4,5)], normal, person, [`john`] ).

cgc( 30, [va_id(4,5)], normal, give, [ ] ).

cgc( 40, [va_id(4,5)], normal, book, [`*`] ).

cgc( 50, [va_id(4,5)], special, situation, [100] ).

cg( 100, [ cgr( agent, [30,10] ),
        cgr( recipient, [30,20] ),cgr( object, [30,40] ) ] ).

cg( 110, [ cgr( past, [50] ) ] ).
```

Fig. 2. "Mary gave John a book" expressed in the notation of the KB

it is considered, in general, a semi-automatic one [27]. Usually, well-organized and structured source text documents allow more automation to be achieved.

The basic building blocks in the knowledge base are CGs, concepts, and conceptual relations. We present the definition of these terms in the Smart VideoText data model using a Prolog-like notation. Conceptual Graphs are defined as predicates of the form:

$$cg(ID, RelationList)$$

where ID is a unique identifier associated with this CG and $RelationList$ is a Prolog list that stores the conceptual relations of the specific CG.

A conceptual relation is defined as:

$$cgr(RelationName, ConceptIDs)$$

where $ConceptIDs$ is a list of concept identifiers that this specific conceptual relation joins and $RelationName$ is the name of the conceptual relation.

Concepts are represented as predicates of the form

$$cgc(ID, VideoAnnotationIDList, Context,$$
$$ConceptName, ReferentField)$$

where ID is a unique identifier associated with this concept; $VideoAnnotationIDList$ is a Prolog list of identifiers of the video annotations containing this concept; $Context$ is either $normal$ for the case of normal concepts or $special$ for the case of contextual concepts; $ConceptName$ is the type-name of a normal concept or the context name of a contextual concept (situation, proposition, etc.); and $ReferentField$ is a Prolog list that holds the referent fields of the specific concepts as they appear in the various video annotations. Each referent field is also a Prolog list. For instance, for the concept [Book:{*}@3] (three books), the $ReferentField$ has the value [{*}@3]. Although this naive representation of the $ReferentField$ argument of the cgc tuple provides the functionality required to establish the Smart VideoText model, it can be further improved, since it is a crucial factor in the calculation of the semantic distance between two concepts. A more sophisticated approach can be found in [20].

The concept and the relation type hierarchy are defined using is_a relationships. For example, $is\_a(car, vehicle)$ denotes that car is a kind of vehicle. Note that other kinds of hierarchical relations such as equivalent, scope and association are not required to be explicitly defined since they can be expressed using relations of the CG formalism.

The above Prolog-like notation can handle complex CGs without information repetition since concepts are indexed separately. Furthermore, a CG can be traversed starting from any of its concepts [20].

It is obvious that the mapping Map_2 is included in the above representations of the various elements in the CG formalism. The $VideoAnnotationIDList$ argument in the $cgc/5$ predicate is a list that holds this information as tuples of the form $va\_id(i, j)$. Such a binary tuple in a cgc means that, the concept represented by this cgc is mapped to the video annotation j of the logical video stream i. This argument has the value $va\_id(-1, -1)$ and $va\_id(0, 0)$ for concepts belonging to the system and domain knowledge respectively, since these are not derived from video annotations.

In Fig. 2, we illustrate the concepts, conceptual relations and CGs for the example of Fig. 1 in the way they are stored in the KB of the Smart VideoText model. Note that $Context$ in $cgc/5$ is a flag; when it has the value *special*, the next argument defines the context of a concept (*situation* here) and not a usual concept-type name.

5. Video data access in Smart VideoText database

Smart VideoText model supports multiple-strategy, knowledge-based video data access which includes operator-based queries, natural language queries and hypertext-like video browsing, based on semantic associations provided by the KB.

5.1. Operator-based video queries

An operator-based user query is an expression formed through terms and zero or more operators. Terms are the strings that a user wants to find in the annotations of the target logical video segments. The operators define the relationships among the terms and can be Boolean (AND etc.), temporal (ADJ) and/or interval operators (AFTER, DURING, etc.). The detailed description and syntax of such operator-based query expressions can be found in [9]. A video query in the Smart VideoText model, which is an extension of the VideoText query, is defined as:

$$Q(Expression, Scope, KBFlag, MaxResults)$$

where $Expression$ is the VideoText query expression; $Scope$ defines the granularity of the answer and can be video streams (v) or logical video segments (s); $KBFlag$ is a flag (true/false) denoting whether to use the knowledge base or not; and $MaxResults$ is the maximum number of returned query results. Notice that if $Expression$ contains interval operators, then $Scope$ is always (s). This stems from the nature of this category of operators.

The utilization of the KB is expected to increase the effectiveness of the operator-based video data access. This is due to the fact that exact term matching suffers in the following cases:

- *Poor recall*: in this case, useful logical video segments are not retrieved because their annotations contain a synonym or something semantically similar, rather than the exact terms presented in the video query.
- *Poor precision*: too many video annotations contain the given term(s), but not all the retrieved logical video segments are actually semantically relevant to the video query.

The use of the KB (particularly the concept and relation-type hierarchy) in the video data model alleviates the poor recall problem. For example, consider a user interested in video lectures about the "rumen" (part of the digestive system in cows), who is getting no results due to poor recall. A generalization of the term "rumen" to the term "digestive system" based on the concept type hierarchy can probably produce some results. A discussion about how the KB can be used to improve the video query precision is given in Sect. 5.2.

The existence of the KB in the Smart VideoText model provides two modes of operator-based query evaluation: *raw term matching* and *semantic term matching*. The mode is determined by the value of the $KBFlag$ argument (false or true, respectively) in a Smart VideoText query expression. The first case is straightforward: a text annotation (and its logical video segment) is considered as an answer to a video query if the annotation contains all the query terms and satisfies all the constraints introduced by the operators. In the second case, a *similarity measure* is required to be able to determine the extent to which two concepts may be labeled "similar". Calculation of the similarity of two concepts depends upon the prior identification of appropriate "sources" of similarity associated with the concepts. Such a source is the concept-type hierarchy. Having the form of a lattice, the type hierarchy of a CG system allows the computation of the distance between related nodes in the lattice, which is often called *semantic distance*. The semantic distance can be used as the measure to rank the results of an operator-based video query.

For a semantic match, if two concepts are syntactically different from each other but belong to the same branch of a concept-type hierarchy, the more specific one can be repeatedly generalized to shorten the semantic distance between them. Between two semantic matches, the one that uses fewer successive generalizations is more important since the semantic distance between this one and the matching concept is shorter. Thus it has a higher rank. We restrict ourselves to generalization since specialization does not always preserve truth [26]. For example, specializing the concept [building] to [hospital] is not correct in all contexts. Polysemy cases (e.g., whether *bank* is a financial institution or a river bank) are dissolved based on the different conceptual definitions of the polysemy terms, together with the neighboring text of the annotation in which the polysemy term occurs. This text helps select the right CG definition automatically. If this is not possible, the user manually dissolves the ambiguity.

The operator-based query evaluation is performed by recursively decomposing the query into subexpressions and processing them along the lines described in [9]. The only difference is that in the Smart VideoText model, if a query term does not match a term in a video annotation, an attempt to generalize the query term using the concept-type hierarchy is performed. On successful generalization, the matching process is repeated, this time for the term, which is the result of the generalization. This is depicted in Fig. 3. The user decides whether to use raw or semantic term matching in a video query. When enabled, the semantic term matching method is automatically invoked when the raw term matching fails to give results or the number of results is below a user-defined threshold. In addition, the query evaluation process is recursive.

5.2. Natural language video queries

Since the derivation of CGs from natural language text is not fully automated, video queries expressed in natural language are partially supported in the form of query templates. A query template corresponds to a predefined, semi-structured CG in which the user is requested to precisely define it by either specifying one or more concepts or replacing a generic referent marker of a concept with a more specific one.

An example of a query template coupled with a semi-structured CG is given in Fig. 4. The user fills in the empty fields of either a natural language query or a CG expression (these empty fields are represented as variables in *italics* in Fig. 4) to make the question complete. In both cases, the result is a CG that the system tries to "prove" with the contents (CGs, concepts, etc.) of the KB. Upon successful matching, the query CG (or possibly a modified version of it due to an expansion operation, for example) will be augmented with icons that will allow the user to invoke a video player to play the related video clip(s). This is possible through the mapping schemes Map_1 and Map_2 that map CGs and concepts to logical video segments.

It is useful to distinguish between *directly related* videos and *indirectly related* ones. A logical video segment is said to be directly related to a user query, if its derived knowledge matches syntactically the information need of the user, that is, without any use of other knowledge. Consequently, for indirectly related logical video segments, additional knowledge has been used (for example, the concept-type hierarchy) to complete the match which, as a result, is a semantic match. The Dempster–Shafer theory [1] can be used to combine various "sources" of similarity evidence associated with CGs to compute the *total similarity* between two CGs. Such "sources" can be the maximal join, matching between relations, concepts and conceptual referents, the concept-type hierarchy, and the ratio of arcs in the maximal join CG to the total number of arcs in the larger of the two CGs that participate in the maximal join operation [1]. The size of a CG is equal to the number of arcs that join its building blocks, i.e., its concepts and relations. The contribution from any of the above sources of evidence of similarity can be equal or weighted. In general, the total similarity is defined as:

$$TotalSimilarity = w_1 \times Evidence_1 + w_2 \times Evidence_2 + \ldots + w_N \times Evidence_N$$

where w_i are the weights and $\sum w_i = 1, i = 1, \ldots, N$. According to [1], this combined similarity allows for superior retrieval to that obtained by any individual form of evidence.

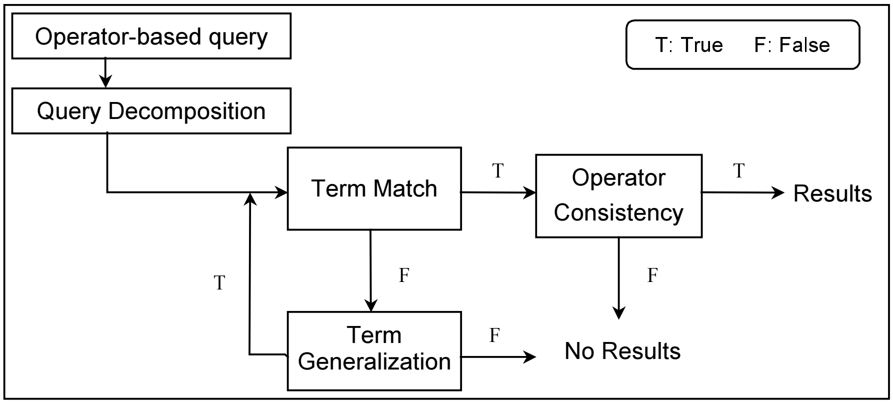


Fig. 3. Term Generalization provides the semantic term matching in the operator-based query

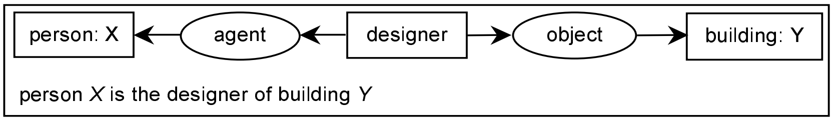


Fig. 4. A quaery template for the designer X of some building Y

This $TotalSimilarity$ value is used to rank the results in the case of natural language video queries.

Since the end-user is assumed to be unfamiliar with the CG representation formalism, a sophisticated graphical user interface (GUI) is necessary to make natural language video queries as user-friendly a process as possible. In a real application with a clearly defined content, a preexisting, controlled vocabulary can further alleviate the problem of user unfamiliarity with the CG formalism.

The KB can be used in both operator-based and natural language queries to improve precision. If a Smart VideoText query has produced too many answers, it is possible to use this knowledge to construct system queries – queries that are constructed by the system and are presented to the user – to improve the precision of the returned logical video segments. For example, if searching for video clips about cars has returned too many logical video segments, then, the system should try to collect various instances of cars and ask the user if he/she is interested in any particular brand name or model. According to the notation we have adopted for the concepts, such an operation is straightforward as long as the information required for this task is enclosed into the referent fields of concepts.

5.3. Video browsing

The Smart VideoText model supports an efficient and effective way for the user to browse a large collection of video data as well as the corresponding video annotations based on their semantic content. This is achieved by using the semantic associations among the concepts of video annotations and dynamically representing them as “hyperlinks” between the corresponding logical video segments. The users can follow these links when they browse the video database.

A term in a video annotation, which is also a concept in the KB, could serve as a hypertext-like link that points to a logical video segment that contains the same concept or a semantically similar one. For instance, two video annotations containing the term “Mars” are directly associated (so are their corresponding logical video streams), but any of them is indirectly related to a third annotation containing the term “Pathfinder” and vice versa. Thus, semantic associations among the concepts in the KB can be the vehicles to browse over semantically related video clips and video annotations. This functionality is supported by the mappings Map_1, Map_2 and the other KB components in the Smart VideoText model.

Figure 5 gives an example of the browsing capabilities of the Smart VideoText model. Assume that there are three logical video segments $LVS_i(i = 1, 2, 3)$, with corresponding video annotations $VA_i(i = 1, 2, 3)$. $CG_i(i = 1, 2, 3)$ are conceptual graphs derived from those annotations and are part of the KB. Assume further that the first set of data, LVS_1, VA_1, and CG_1, is presented to the user. The user can either ask the system to find information related to one of the designers (say designer b) or ask for information related to skyscrapers. All can be done in a hypertext-like fashion, i.e., some elements displayed have the potential to function as hypertext links to other video data. The existence and functionality of these hyperlinks are provided by the model rather than being explicitly defined by the database user. Hence, these KB-based links are generated “on the fly” rather than being explicitly predefined and fixed as in hypertext documents like the HTML pages on the World Wide Web.

In the example of Fig. 5, a request for additional or related information about designer b can lead the user to VA_2 or CG_2 through a direct concept matching over [Person] and a membership check over the referents. At any time, the user can also ask the system to play the corresponding logical video segment, LVS_2.

On the other hand, an information request about skyscrapers will drive the user to the third set of data (VA_3, CG_3, and LVS_3) given the fact that the concept-type hierarchy in the domain KB includes a relation like $is\_a(skyscraper, building)$. Moreover, from the third set of data, the system can suggest the second set as closely related information.

VA_1 of LVS_1: *"Three famous skyscraper designers."*

CG_1: [Person:{a,b,c}] ← (agent) ← [designer] → (object) → [skyscraper]

[famous] ← (attrib) ↵

VA_2 of LVS_2. *"b, the designer of the Sears Tower."*

CG_2: [Person:b] ← (agent) ← [designer] → (object) → [skyscraper: Sears Tower]

VA_3 of LVS_3: *"The Sears Tower. Once the highest building."*

CG_3: [[building: Sears Tower] ← (attrib) → [highest]] ← (past)

skyscraper < building

LVS_i: Logical Video Segment i
VA_i: Video Annotation i
CG_i: Conceptual Graph i

Fig. 5. Video annotations of some logical video segments and their corresponding CGs

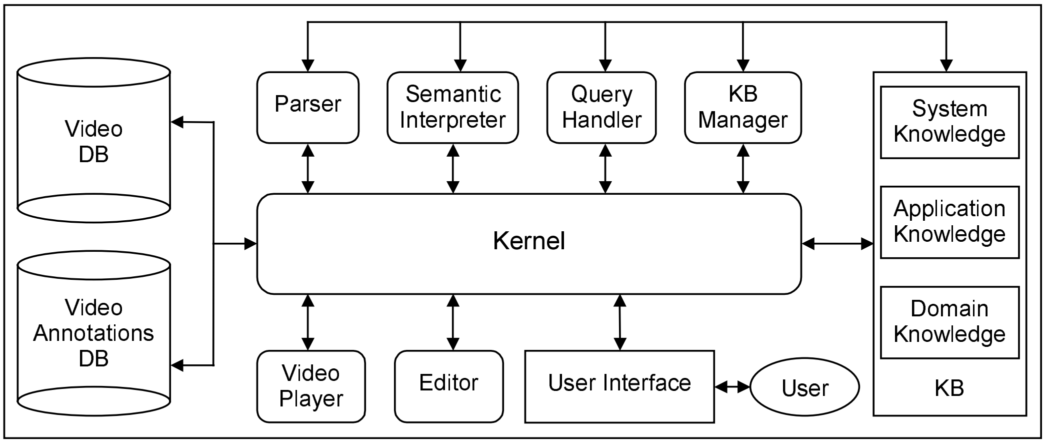


Fig. 6. Smart VideoText system architecture

6. Smart VideoText system architecture

The Smart VideoText model introduces a modular architecture for implementing video databases. Although video segments, text annotations and application knowledge are related in the data model, they can be managed by a different component of a Smart VideoText based system. Such a system is outlined in Fig. 6 and its components are described below.

The *video database* stores the video data, while the *video annotation database* stores annotations of the video data. Each annotation is stored together with a reference to the logical video stream it annotates.

The *system knowledge* includes the canonical formation rules and knowledge that is supposed to be application-independent. *Application knowledge* consists of CGs, concept-types, concepts, and relations that are derived from video annotations. Although multiple application knowledge bases can be allowed (one for each application), only one of them is used at any time. This means that knowledge consistency checking is performed at the application level. *Domain knowledge* includes the concept-type hierarchy, together with the concept and conceptual relation definitions of the various concepts and conceptual relations used in the application. In other words, it includes all the knowledge related to the application, but not explicitly defined in the video annotations.

A *kernel* integrates the various modules into a single video database system. The kernel includes a Prolog inference engine that is responsible for the inference.

Among the *modules* for the user/author are a *parser*, a *semantic interpreter*, a *query handler*, a *knowledge manager*, a *linkage assistant*, an *editor*, and a *video player*. The *parser* uses syntactic rules to generate parse trees corresponding to all sentences of the video annotations (or those selected by the user), as well as to the queries expressed in natural language. The *semantic interpreter* translates the above trees into CGs and, in the case of knowledge construction, inserts them into the KB if they fulfill the canonical formation rules. In the case of a natural language query, the resulting CG is passed to the query handler module. In both cases, the semantic interpreter can be manually forced to abandon some parts of a sentence that are of no interest.

The *query handler* lets the user construct queries concerning the video database. A query is expressed as a CG and is constructed by the user either directly or indirectly. In the first case, the user selects the appropriate items (concepts and relations) from a combination of selection components (menus, listboxes, etc.). It is also possible for the user to select a query from a set of previously defined query templates, expressed in natural language. For each of them, there exists a preconstructed CG that is used to answer the query. Creating a CG-query indirectly means that the user expresses it in natural language using predefined sentence parts. Constraints on the use of these sentence parts ensure the syntactic validity of the final query statement. The proper system modules (the parser and the semantic interpreter) convert these parts into CG form. The *knowledge manager* supports operations such as reviewing the KB and asserting the knowledge in any of the three parts of the KB. Review of the KB is performed at the CG level. The user opens the KB file and displays any CG in graphical form. CGs that satisfy the user-defined criteria can also be displayed. These criteria are filters that allow the user to inspect CGs with a common property. For example, in a geographical video database application, the user may want to see all the CGs concerning the population of capital cities in order to update the population numbers. Furthermore, the knowledge base can be modified if necessary. For example, when a video annotation is updated or deleted, the corresponding CGs in the KB are also modified or deleted by the system. In addition, knowledge consistency techniques are applied to ensure that the KB remains consistent after modifications.

The main advantage of the above architecture is its modularity, which stems directly from the video content and the knowledge representation methods that are used. The KB-related modules are independent of each other as long as they conform with the representation standards of CGs, which are well-founded. Hence, advances in any of these domains could be easily utilized.

7. Conclusion and future work

In this paper, we have proposed an annotation-based video data model called Smart VideoText. This model utilizes the conceptual graph knowledge representation formalism to capture and represent the semantic associations among the concepts described in video annotations. The model can support dynamic video annotation manipulation, annotation sharing and multiple interpretations of the same logical video segment. The model also includes a query language that supports complicated video queries based on the semantic content of the video data. Operator-based (Boolean, temporal, and interval query operators) as well as natural language video queries are supported. The CG layer also allows transparent, hypertext-like content-based browsing on the video data. Furthermore, the functionality of the operator-based video query and retrieval is significantly enhanced because the CG layer provides semantic term matching. In this paper, we have concentrated mainly on the video data model and its knowledge-based aspects.

We are currently in the process of implementing the Smart VideoText model along the lines of the system architecture proposed in Fig. 6. The system's prototype is primarily based on the COMFRESH system [12], and provides web-based video access. More issues concerning implementation aspects, such as query optimization and various performance measurements, will be studied in the near future.

Applying the ideas of Smart VideoText model to other media forms (e.g., sound and graphics) is fairly straightforward since our model deals directly with an intermediate, textual representation of the content of such data. The inference mechanism is not aware of the media form of the raw data; it deals only with the text annotations that describe their content. Thus, extending our model to multimedia documents is feasible. This is very important since, given the popularity of the World Wide Web and the constantly increasing amount of multimedia data available online, knowledge-based information retrieval systems play an important role in the effort to implement more efficient and effective multimedia information systems.

References

1. F. Brkich and R. Wilkinson. A conceptual graph matching algorithm for information retrieval. First Australian Conceptual Structures Workshop, University of New England, Armidale, New South Wales, 1994
2. M.G. Brown, J.T. Foote, G.J.F. Jones, K. Sparck and S.J. Young. Open-vocabulary speech indexing for voice and video mail retrieval. In Proceedings of the 4th ACM International Multimedia Conference, pp. 307–316, 1996
3. G. Davenport, T.G.A. Smith and N. Pincever. Cinematic primitives for multimedia. IEEE Comput Graph Appl 00:67–74, 1991
4. A.K. Elmagarmid, H. Jiang and ♣ Video database system: issues, products and applications. Kluwer Academic Publishers, 1996
5. N. Fuhr Models for integrating retrieval and database systems. IEEE Data Eng Bull 19, 1996
6. A. Hampapur, R. Jain and T. Weymouth. Digital video indexing in multimedia systems. In Proceedings of the Workshop on Indexing and Reuse in Multimedia Systems, 1994
7. R. Hjelsvold and R. Midtstraum. Modeling and querying video data. In Proceedings of the 20th International Conference on Very Large Data Bases, 1994
8. H. Jiang, A. Helal, A.K. Elmagarmid and A. Joshi. Scene change detection techniques for video database systems. ACM Multimedia Syst 6:186–195, 1998
9. H. Jiang, D. Montesi and A.K. Elmagarmid. VideoText database systems. In Proceedings of the 4th IEEE International Conference on Multimedia Computing and Systems, pp. 334–351, 1997
   H. Jiang, D. Montesi and A.K. Elmagarmid. Integrated video and text for content-based access to video databases. J Multimedia Tools Appl 9:♣, 1999
10. T. Kanade, M. Sirbu, M. Mauldin, S. Stevens, H. Wactlar, R. Reddy and D. Tygar. In Informedia digital video library system: annual progress report. Computer Science Department, Carnegie Mellon University, Pittsburgh, 1997
11. F. Kokkoras and I. Vlahavas. COMFRESH: a common framework for expert systems and hypertext. Inform Processing Mgmt 31:593–604, 1995
12. Z. Lei and Y.T. Lin. 3D shape inferencing and modeling for semantic video retrieval. In Proceedings of Multimedia Storage and Archiving Systems, pp. 224–235, 1996
13. T.D.C. Little and A. Ghafoor. Interval-based conceptual model for time-dependent multimedia data. IEEE Trans Know Data Eng 5:551–563, 1993
14. T.D.C. Little, G. Ahanger, R.J. Folz, J.F. Gibbon, F.W. Reeve, D.H. Schelleng and D. Venkatesh. (1993) A digital on-demand video service supporting content-based queries. Proceedings of 1st ACM International Conference on Multimedia, Anaheim, CA, pp. 427–436, 1993
15. S. Liou, R. Hjelsvold, R. Depommier, and A. Hsu. Efficient and reliable digital media archive for content-based retrieval. ACM Multimedia Syst 7:256–268, 1999
16. P. Martin and P. Eklund. Embedding knowledge in web documents. In Proceedings of the 8th International WWW Conference. Int J Comput Telecommun Netw, special issue, 1999
17. M.J. McGill and G. Salton. Introduction to modern information retrieval. McGraw-Hill, 1983
18. N.V. Patel and I.K. Sethi. Audio characterization for video indexing. In IST/SPIE Proceedings: Storage and Retrieval for Image and Video Databases IV, San Jose, CA, 1995
19. H. Petermann, L. Euler and K. Bontcheva. CGPro – A PROLOG implementation of conceptual graphs. Technical Report, University of Hamburg, FBI-HH-M-251/95, 1995
20. C.J. van Rijsbergen. Information retrieval, 2nd edn. Butterworths, 1997

21. Y. Rui, T.S. Huang and S. Mehrotra. Exploring video structure beyond the shots. In Proceedings of IEEE International Conference on Multimedia Computing and Systems, pp. 243–251, 1998
22. T. Sato, T. Kanade, E. Hughes, M. Smith, and S. Satoh. Video OCR: indexing digital news libraries by recognition of superimposed captions. ACM Multimedia Syst 7: 385–395, 1999
23. M.A. Smith and A. Hauptmann. Text, speech, and vision for video segmentation: the informedia project. In AAAI Fall 1995 Symposium on Computational Models for Integrating Language and Vision, 1995
24. T.G.A. Smith and G. Davenport. The stratification system: a design environment for random access video. Workshop on Networking and Operating System Support for Digital Audio and Video, ACM, 1992
25. J.F. Sowa. Conceptual structures: information processing in minds and machines. Addison-Wesley, 1984
26. J.F. Sowa and E.C. Way. Implementing a semantic interpreter using conceptual graphs. IBM J Res Dev 30:57–69, 1986
27. D. Swanberg, C.F. Shu and R. Jain. Knowledge guided parsing in video database. In Electronic imaging: science and technology, San Jose, California, IST/SPIE, 1993
28. L. Teodosio and W. Bender. Salient video stills: content and context preserved. In Proceedings of ACM Multimedia ’93, pp. 39–46, 1993
29. R. Weiss, A. Duda and D. Gifford. Content-based access to algebraic video. In IEEE International Conference Multimedia Computing and Systems, Los Alamitos, CA, 1994
30. S.K.M. Wong and Y.Y. Yao. On modeling information retrieval with probabilistic inference. ACM Trans Inform Syst 13:38–68, 1995
31. M.M. Yeung, B.L. Yeo and B. Liu. Extracting story units from long programs for video browsing and navigation. In Proceedings of the 3rd IEEE International Conference on Multimedia Computing and Systems, 1996