A VIDEO DATABASE MANAGEMENT SYSTEM

Mehmet Emin Dönderler, Ediz Şaykol, Özgür Ulusoy and Uğur Güdükbay

Department of Computer Engineering, Bilkent University
06533 Bilkent, Ankara, Turkey
{mdonder,ediz,oulusoy,gudukbay}@cs.bilkent.edu.tr

ABSTRACT

We develop a Web-based video database management system, named *BilVideo*, providing an integrated support for both spatio-temporal and semantic (keyword, event/activity and category-based) queries on video data. In this paper, we introduce the architecture of *BilVideo* and its SQL-like textual video query language that has the capability to handle a broad range of spatio-temporal queries, which may contain any combination of directional, topological, 3D-relation, object-appearance, trajectory-projection and similarity-based object-trajectory conditions. Furthermore, we present the Web interface of *BilVideo* and its tools, *Fact-Extractor* and *Video-Annotator*, which are used to populate the facts-base and feature database of the system to support spatio-temporal and semantic video queries, respectively. The goal of our project is to provide support for any application with spatio-temporal and semantic video query requirements; hence, *BilVideo* is application-independent, but yet can easily be tailored for specific needs of such applications through the definition of *external predicates*.

Keywords: Spatio-temporal relations, content-based retrieval, video databases, multimedia databases, video query languages.

1. INTRODUCTION

We develop a video database management system, named *BilVideo*, which provides an integrated support for both spatio-temporal and semantic queries on video data [1]. A spatio-temporal query may contain any combination of directional, topological, 3D-relation, object-appearance, trajectory-projection and similarity-based object-trajectory conditions. *BilVideo* handles spatio-temporal queries using a knowledge-base, which consists of a fact-base and a comprehensive set of rules implemented in Prolog, while semantic queries are handled by an object-relational database. The rules in the knowledge-base significantly reduces the number of facts that need to be stored for spatio-temporal querying of video data whilst keeping the query response time reasonably short [1]. The query processor interacts with both of the knowledge-base and object-relational database to respond to user queries that contain a combination of spatio-temporal and semantic queries. Intermediate query results returned from these two system components are integrated seamlessly by the query processor and sent to Web clients. We also present an SQL-like textual video query language for spatio-temporal queries on video data [2] as well as the Web-interface of *BilVideo* and its tools, *Fact-Extractor* and *Video-Annotator*, which are used to populate the facts-base and feature database of the system to support spatio-temporal and semantic video queries, respectively.

The goal of our project is to support any application with spatio-temporal and semantic query requirements on video data; therefore, *BilVideo* is application-independent. However, it can easily be tailored according to specific requirements of such applications through the definition of *external predicates* without any loss in performance. Thus, we intend to cooperate with an organization to realize our project with an application that needs spatio-temporal and semantic video query capabilities.

The rest of the paper is organized as follows: Overall architecture of *BilVideo* is briefly introduced in Section 2. Sections 3 and 4 present the tools *Fact-Extractor* and *Video-Annotator*, respectively. Section 5 provides a brief discussion on the Web query interface of *BilVideo*. Our SQL-like textual video query language for spatio-temporal querying of video data is introduced in Section 6. Section 7 makes a short discussion of the system's flexibility to support a broad range of applications. We conclude the paper stating our ongoing work in Section 8.

2. BILVIDEO SYSTEM ARCHITECTURE

Figure 1 illustrates the architecture of our Web-based video database management system, *BilVideo*. It is built over a client-server architecture and users access the video database on the Internet through a Java client applet. The system can be queried with user-drawn sketches. A visual query is formed by a collection of objects with some conditions,

This work is supported by the Scientific and Research Council of Turkey (TÜBİTAK) under Project Code 199E025.

such as object trajectories with similarity measure, spatio-temporal ordering of objects, annotations and events. Object motion is specified as an arbitrary trajectory and annotations are used for keyword-based search. Users can browse the video collection before giving complex and specific queries, as well. A text-based SQL-like query language is also available for the users. In the heart of the system lies the query processor that is responsible for processing user queries in a multi-threaded environment. It communicates with an object-relational database and the knowledge-base, where semantic and fact-based meta data is stored, respectively. Raw video data and video data features are stored separately. The feature database contains semantic properties of videos used for keyword, activity/event and category-based queries. These features are generated and maintained by the *Video-Annotator* tool developed as a Java application. The knowledge-base is used to respond to spatio-temporal queries and the facts-base is populated by the *Fact-Extractor* tool, which is a Java application as well.

3. FACT-EXTRACTOR TOOL

We have developed a tool, *Fact-Extractor*, to extract spatio-temporal relations between salient objects, object trajectories and object-appearance relations in video clips. Relations extracted are stored in the knowledge-base as facts, which are used to query video data for spatio-temporal conditions.

The fact-extraction process is semi-automatic: objects are manually specified in video frames by their minimum bounding rectangles (MBRs). Using the object MBRs, a set of spatio-temporal relations (directional and topological) is automatically computed. The rules in the knowledge-base are used to eliminate redundant relations; therefore, the set contains only the relations that cannot be derived by the rules. For 3D-relations, extraction cannot be done automatically because 3D-coordinates of the objects cannot be extracted from video frames. Hence, these relations are entered manually for each object-pair of interest and the relations that can be derived by the rules are eliminated automatically. The tool performs an interactive conflict-check for 3D-relations and keeps the set of 3D-relations of a frame intact for the next frame so that the user may apply any changes in 3D-relations by editing this set in the next frame. Object trajectories and object-appearance relations are also extracted automatically for each object. Moreover, object MBRs need not be redrawn for each frame since MBR resizing, moving and deletion facilities are available. When exiting the tool after saving the facts, some configuration data is also stored in the knowledge-base if the video is not entirely processed yet so that the user may continue processing the same video clip later on from where it was left off. Since object MBRs are drawn manually by users, there is a space for erroneous MBR specification although in many cases small errors do not affect the set of relations computed. To automate this process, an *Object-Extractor* utility module has been developed [3]. We plan to embed this module into our *Fact-Extractor* tool to help users specify object MBRs with a few mouse clicks on objects. Figure 2 gives a snapshot of the *Fact-Extractor* tool.

4. VIDEO-ANNOTATOR TOOL

We have also developed a tool, *Video-Annotator*, to extract semantic data from video clips to be stored in the feature database to query video data for its semantic content. The tool also provides facilities for viewing, updating and deleting semantic data that has already been extracted from video clips and stored in the feature database. A snapshot of the tool is given in Figure 3.

Our semantic video hierarchy contains three levels: *video*, *sequence* and *scene*. *Videos* consist of *sequences* and *sequences* contain *scenes* that need not be consecutive in time. With this semantic data model, we plan to answer three types of queries: *video*, *event/activity* and *object*. *Video* queries can be used for retrieving videos based on descriptional data (annotations) of video clips. Conditions may include title, length, producer, production year, category and director information about a video clip. *Event/activity* queries are the most important and powerful type of queries among all and they can be used to retrieve videos by specifying events that occur at semantic layer *sequence* because events are associated with sequences. However, a particular scene or scenes of an event can also be returned as an answer to a semantic query when requested because events may have sub-events associated with scenes. *Object* queries are used to retrieve videos by specifying semantic object features. As videos are annotated, video salient objects are also associated with some descriptional meta data.

5. WEB-BASED USER INTERFACE

BilVideo can handle multiple requests over the Internet through a graphical query interface developed as a Java applet [4]. The interface is composed of query specification windows for different types of queries: *spatial* and *trajectory*. Since video has a time dimension, these two types of primitive queries can be combined with temporal predicates to query temporal contents of videos.

5.1. Spatial Query Specification

Spatial content of a video keyframe is the relative positioning of its salient objects with respect to each other. This

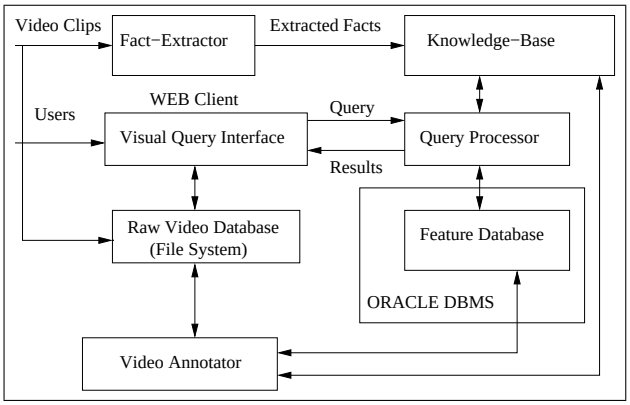


Figure 1: BilVideo System Architecture

Figure 2: Fact-Extractor Tool

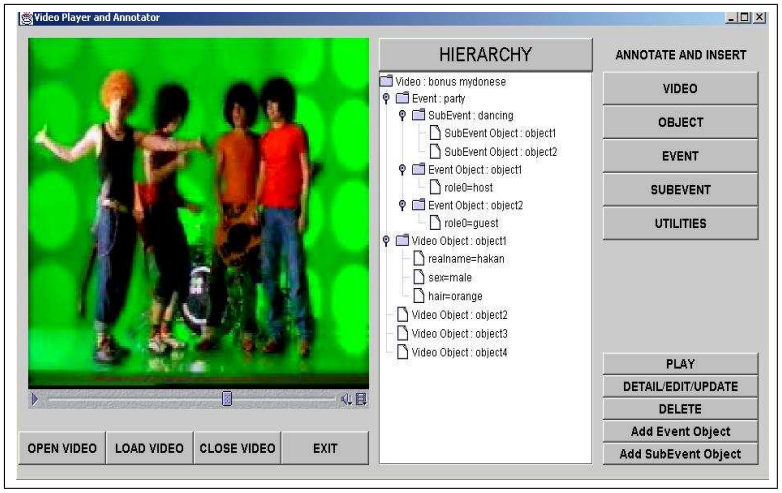


Figure 3: Video-Annotator Tool

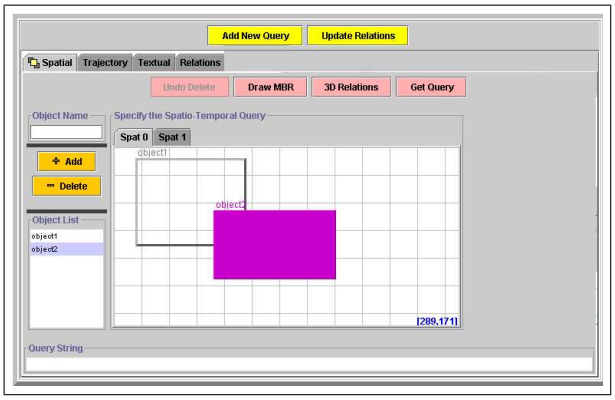
relative positioning consists of three separate sets of relations: directional, topological and 3D-relations. In order to query the spatial content of a keyframe, these relations have to be specified in the query within a proper combination. Obviously, this combination should be constructed with the logical connector *and*; thus, all the relations have to be present in the video frame(s) returned as a result. In the spatial query specification window shown in Figure 4 (a), salient objects are sketched by rectangles, which represent MBRs of the objects. Hence, each object is enclosed by its MBR during the database population phase and spatial content of a keyframe is extracted according to object MBRs. Similar to the database population phase, the directional and topological relations between objects are extracted automatically in the query specification phase. Since it is impossible to extract 3D-relations from 2D-data, users are guided to select appropriate 3D-relations for salient-object pairs.

5.2. Trajectory Query Specification

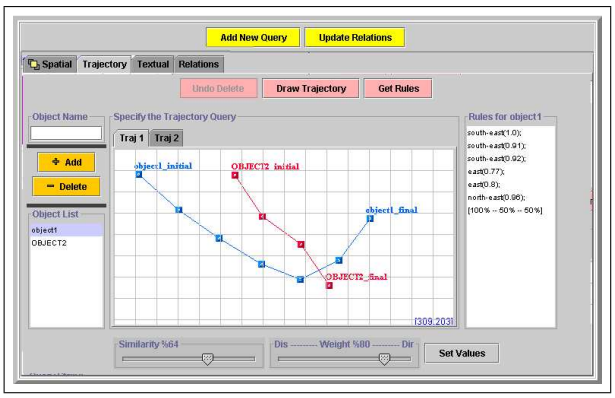
Trajectory of a salient object is described as a path of vertices corresponding to the locations of the object in different video keyframes. Displacement values and directions between consecutive keyframes (vertices) are used in defining the trajectory fact of an object. In the trajectory query specification window shown in Figure 4 (b), users can draw trajectories of salient objects. The trajectories displayed are dynamic in a sense that any vertex can be deleted from or a new vertex can be inserted to a trajectory. Locations of the vertices can also be changed to obtain a desired trajectory. Object-trajectory queries are similarity-based; therefore, users specify a similarity value, between 0 and 100, where the value 100 implies an exact match.

5.3. Final Query Formulation

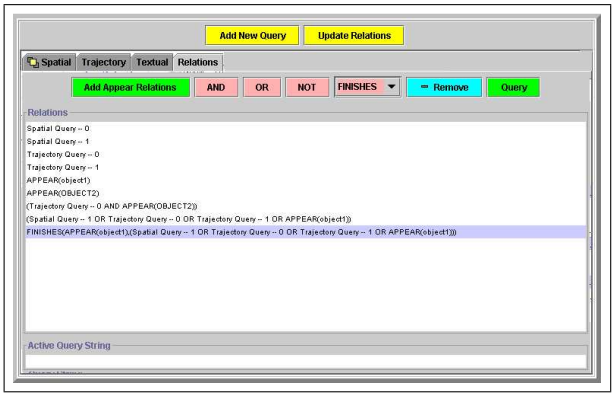
In the user interface, spatial and trajectory queries are specified in different windows. Each of these specifications forms a sub-query and these sub-queries are combined in the final query formulation window given in Figure 4 (c). This window contains all the specified sub-queries as well as object-appearance relations for each object. Users can combine sub-queries by logical operators (and, or) and temporal predicates (before, during, etc.). Except for the logical operator *not*, all temporal and logical operators are binary. If more than two sub-queries are given as arguments to binary operators, they are combined as cumulative pairs with the operators. After applying operators to sub-queries, a new query is augmented to the list, and successive and hierarchical combinations become possible. After the final query is formed, it can be sent to the query processor. Furthermore, any sub-query of the final query may also be sent to the query processor at any time to obtain partial results if requested.



(a) Spatial Query Specification



(b) Trajectory Query Specification



(c) Final Query Formulation

Figure 4: Web-Based User Interface

6. VIDEO QUERY LANGUAGE

In this section, we present a new video query language that is similar to SQL in structure. The language can currently be used for spatio-temporal queries that contain any combination of directional, topological, 3D-relation, object-appearance, trajectory-projection and similarity-based object-trajectory conditions. As a work in progress, the language is being extended so that it could support semantic video queries as well in a unified and integrated manner. The language has four basic statements for retrieving information:

select *video* **from all** [**where** *condition*];
select *video* **from** *videolist* **where** *condition*;
select *segment* **from** *range* **where** *condition*;
select *variable* **from** *range* **where** *condition*;

Target of a query is specified in **select** clause. A query may return videos (*video*) or segments of videos (*segment*), or values of variables (*variable*) with/without segments of videos where the values are obtained. Aggregate functions, which operate on segments, may also be used in **select** clause. Variables might be used for object identifiers and trajectories. Moreover, if the target of a query is videos (*video*), users may also specify the maximum number of videos to be returned as a result of a query. If the keyword **random** is used, video fact-files to process are selected randomly in the system, thereby returning a random set of videos as a result. The range of a query is specified in **from** clause, which may be either the entire video collection or a list of specific videos. Query conditions are given in **where** clause. In our query language, *condition* is defined recursively, and consequently, it may contain any combination of spatio-temporal conditions.

Supported Operators: The language supports a set of logical and temporal operators to be used in query conditions. Logical operators are *and*, *or* and *not* while temporal operators are *before*, *meets*, *overlaps*, *starts*, *during*, *finishes* and their inverse operators.

The language also has a trajectory-projection operator, *project*, which can be used to extract sub-trajectories of video objects on a given spatial condition. The condition is local to *project* and it is optional. If it is not given, entire object trajectories rather than sub-trajectories of objects are returned.

The language has two operators, "=" and "!=", to be used for assignment and comparison. Operator "!=" is used for inequality comparison while operator "=" may take on different semantics depending on its arguments. If the left argument of operator "=" is an unbound variable, it is treated as the assignment operator. Otherwise, it is considered as the equality-comparison operator. These semantics were adopted from the Prolog language.

Operators that perform interval processing are called *interval operators*. Hence, all temporal operators are interval operators. Logical operators are also considered as interval operators when they are not part of a maximal sub-query and if their arguments contain intervals. A *maximal sub-query* is defined as a longest sub-query of a given query that can be processed by Prolog without changing the semantics of the original query.

In the language, precedence values of logical, assignment and comparison operators follow their usual order. Logical operators assume the same precedence values defined for them when they are considered as interval operators, as well. Temporal operators are given a higher priority over logical operators when determining the arguments of operators and they are left associative as are logical operators.

The query language also provides a keyword, *repeat*, that can be used in conjunction with a temporal operator, such as *before*, *meets*, etc., or a trajectory condition. Video data may be queried by repetitive conditions in time using *repeat* with an optional repetition number given. If a repetition number is not given with *repeat*, then, it is considered indefinite, thereby causing the processor to search for the largest intervals in a video, where the conditions given are satisfied at least once over time.

Aggregate Functions: The query language has three aggregate functions, *average*, *sum* and *count*, which take a set of intervals (segments) as input and return a time value in minutes for each video clip satisfying given conditions. *Average* returns a time value that is the average of the time durations of all intervals found for a video clip whereas *sum* and *count* are used to calculate the total time duration for and the total number of all such intervals, respectively. These aggregate functions might be very useful to collect statistical data for some applications, such as sports event analysis systems, motion tracking systems, etc.

External Predicates: The proposed query language is generic and designed to be used for any application that requires spatio-temporal query processing capabilities. The language has a condition type *external* defined for application-dependent predicates, which we call *external predicates*. This condition type is generic; consequently, a query may contain any application-dependent predicate in *where* clause of the language with a name different from any predefined predicate and language construct, and with at least one argument that is either a variable or a constant (atom).

External predicates are processed just like spatial predicates as part of maximal subqueries. If an external predicate is to be used for querying video data, facts and/or rules related to the predicate should be added to the knowledge-base beforehand.

In our design, each video segment returned as an answer to a user query has an associated importance value ranging between 0 and 1, where 1 denotes an exact match. The results are ordered with respect to these importance values in descending order.

7. APPLICATION AREAS

With this project, we target for a full-fledged Web-based video database management system to support spatio-temporal (directional, topological, 3D-relation, trajectory-projection, object-appearance and similarity-based object-trajectory), semantic (keyword, activity/event and category-based) and some low-level (color, shape and texture) queries on video data. There are only a few video database prototypes around developed for either academic or commercial purposes; nonetheless, they do only provide support for a rather small subset of the video features we plan to incorporate into *BilVideo*, and hence, their success in returning what the user has actually in mind for his/her query is very limited. Moreover, their support for visual query specification is also not as powerful as that of *BilVideo*, which is very important because the success rate of a video database system also depends on how it acquires the query parameters from users. The visual interface should be simple and easy-to-use, but yet sophisticated and powerful enough to make use of all the capabilities the underlying system offers.

BilVideo does not target a specific application area, and thus, it can be used to support any application, where vast amount of video data needs to be searched by spatio-temporal, semantic and some low-level (color, shape and texture) video features. Furthermore, our video query language provides a simple way to extend the system's query capabilities through *external predicates*, which makes *BilVideo* application-independent but yet easily fine-tunable for specific needs of such applications without much effort and without any loss in performance at all. This can be achieved by adding to the knowledge-base some application-dependent rules and/or facts that will be used for queries. Some example applications that might be supported are sports event analysis systems (soccer, basketball, etc.), object movement tracking systems (medical, biological, astrophysical, etc.) and video archive search systems (movie retrieval, digital libraries, news retrieval, etc.). Specifically, some emerging applications in such areas as digital culture, tourism, entertainment, education and e-commerce may greatly benefit from *BilVideo* using it as their underlying video database management system.

8. ONGOING WORK

The query language of *BilVideo* currently supports a broad range of spatio-temporal video queries. However, the system architecture was designed to handle semantic queries, as well. We also plan to support querying of video data by some low-level properties (color, shape and texture). In order to provide support for shape and color queries, we propose a new approach to store and compare shape and color features of video keyframes and salient objects [5]. Our work on semantic modeling and querying of video data is ongoing. Furthermore, the query language is currently being extended to handle queries that contain not only spatio-temporal but also semantic query conditions. Another important issue we are studying is the optimization of user queries. In an ideal environment, our query language will establish the basis for a visual query interface and serve as an embedded language for users because some video queries are much easier to specify visually. Hence, we will enhance query specification capabilities of the Web-based user interface of *BilVideo* in compliance with the features supported by its textual video query language and integrate the interface with *BilVideo* in future.

Some tutorial video clips that demonstrate the usage of the visual query interface and the tools of *BilVideo* are available on the Internet at *http://www.cs.bilkent.edu.tr/∼oulusoy/BilVideo*.

9. REFERENCES

[1] M.E. Dönderler, Ö. Ulusoy, and U. Güdükbay, "A rule-based video database system architecture," *Information Sciences*, vol. 143, no. 1-4, pp. 13–45, 2002.

[2] M.E. Dönderler, Ö. Ulusoy, and U. Güdükbay, "Rule-based spatio-temporal query processing for video databases," *submitted journal paper*.

[3] E. Şaykol, U. Güdükbay, and Ö. Ulusoy, "A semi-automatic object extraction tool for querying in multimedia databases," in *7th Workshop on Multimedia Information Systems MIS'01, Capri, Italy*, S. Adali and S. Tripathi, Eds., November 2001, pp. 11–20.

[4] E. Şaykol, "Web-based user interface for query specification in a video database system," M.S. thesis, Department of Computer Engineering, Bilkent University, Ankara, Turkey, September 2001.

[5] E. Şaykol, U. Güdükbay, and Ö. Ulusoy, "A histogram-based approach for object-based query-by-shape-and-color in multimedia databases," Tech. Rep. BU-CE-0201 and also submitted journal paper, Bilkent University, January 2002.