Editor: Tiziana Catarci
University of Rome

BilVideo: A Video Database Management System

Mehmet Emin Dönderler, Ediz Şaykol, Özgür Ulusoy, and Uğur Güdükbay
Bilkent University, Ankara, Turkey

The BilVideo video database management system provides integrated support for spatio-temporal and semantic queries for video.[1] A knowledge base—consisting of a fact base and a comprehensive rule set implemented in Prolog—handles spatio-temporal queries. These queries contain any combination of conditions related to direction, topology, 3D relationships, object appearance, trajectory projection, and similarity-based object trajectories. The rules in the knowledge base significantly reduce the number of facts representing the spatio-temporal relations that the system needs to store. A feature database stored in an object-relational database management system handles semantic queries.

To respond to user queries containing both spatio-temporal and semantic conditions, a query processor interacts with the knowledge base and object-relational database and integrates the results returned from these two system components.

Because of space limitations, here we only discuss the Web-based visual query interface and its fact-extractor and video-annotator tools. These tools populate the system's fact base and feature database to support both query types.

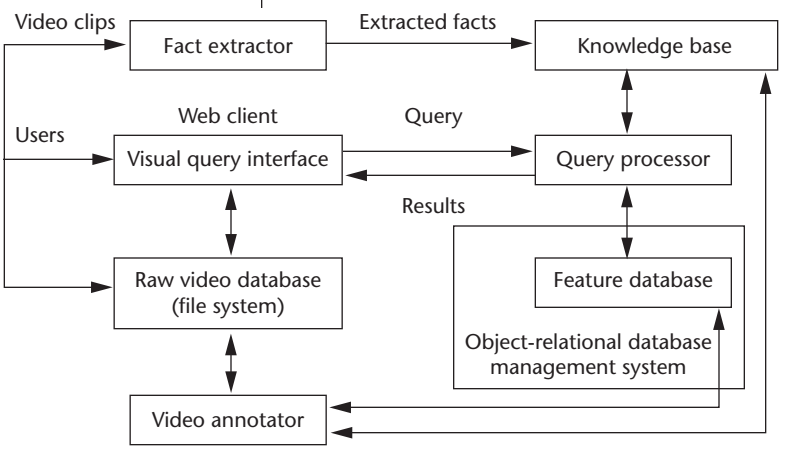

Figure 1. BilVideo's system architecture.

System architecture

We built BilVideo over a client–server architecture, shown in Figure 1. Users access BilVideo over the Internet through a Java client applet. The "Query Types" sidebar discusses the forms of user queries that BilVideo's architecture supports. The heart of the system is the query processor, which runs in a multithreaded environment.

The query processor communicates with a feature database and the knowledge base, where the system stores semantic and fact-based metadata, respectively. The system stores raw video data and its features in a separate database. The feature database contains video semantic properties to support keyword-, activity/event-, and category-based queries. The video-annotator tool, which we developed as a Java application, generates and maintains the features. As mentioned previously, the knowledge base supports spatio-temporal queries and the facts base is populated by the fact-extractor tool, which is also a Java application.

Fact-extractor tool

The fact-extraction process is semiautomatic: users manually specify objects in video frames by their minimum bounding rectangles (MBRs). Using object MBRs, this process automatically computes a set of spatio-temporal relations (directional and topological). The rules in the knowledge base eliminate redundant relations; therefore, this set contains only the relations that Prolog can't derive by using the rules. For 3D relations, extraction doesn't occur automatically because the fact-extraction tool can't extract 3D-object coordinates from video frames. Hence, users must manually enter these relations for each object pair of interest. The fact-extractor tool performs an interactive conflict check for 3D relations and keeps a frame's 3D-relations set

1070-986X/03/$17.00 © 2003 IEEE

Published by the IEEE Computer Society

intact for the next frame. This lets users apply any changes in 3D relations by editing this set in the next frame. The tool also automatically extracts object trajectories and object-appearance relations for each object. Moreover, users need not redraw object MBRs for each frame; instead, they can resize, move, and delete MBRs. When exiting the tool after saving the facts, the system stores some configuration data in the knowledge base if the video isn't yet entirely processed. This lets users continue processing the same video clip at a later time from the point where the original search left off. Figure 2 shows a view of the fact-extractor tool.

Because users manually draw object MBRs, the system can't tolerate erroneous MBR specifications, although, in many cases, small errors don't affect the set of relations computed. To automate this process, we developed an object-extractor utility module, shown in Figure 3.[2] When we embed this module into the fact-extractor tool, users will be able to specify object MBRs by clicking the mouse on objects.

Video-annotator tool

The video-annotator tool, shown in Figure 4 (next page), extracts semantic data from video clips and stores it in the feature database. The tool lets users view, update, and delete semantic data already extracted from video clips. 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. When we complete the semantic query processor, BilVideo will answer video, event/activity, and object queries by using this semantic data model. Video queries retrieve videos based on descriptive data (annotations). Conditions could include title, length, producer, production year, category, and director. Event/activity queries retrieve videos by specifying events that occur at the semantic layer sequence (because events are associated with sequences). However, this query type might also return a particular scene (or scenes) because events may have subevents associated with scenes. Object queries retrieve videos by specifying semantic object features. Because videos are annotated, video salient objects are also associated with some descriptive metadata.

Web-based user interface

BilVideo handles multiple requests received over the Internet through a graphical query interface, developed as a Java applet.[3] The interface has query specification windows for different types of queries, that is, spatial and trajectory. Because video has a time dimension, these two types of primitive queries can combine with temporal predicates to query temporal contents.

Query Types

BilVideo-supported queries might be constructed with user-drawn sketches. A collection of objects with some conditions forms a visual query. Such conditions could include object trajectories with similar measurements, spatio-temporal ordering of objects, annotations, or events. Searches specify object motion as an arbitrary trajectory and annotations support keyword-based searching. Users can browse the video collection before giving complex and specific queries. A text-based Structured Query Language-like language is also available.[1]

Reference

1. M.E. Dönderler, Ö. Ulusoy, and U. Güdükbay, "Rule-Based Spatio-Temporal Query Processing for Video Databases," submitted for publication.

Figure 2. Fact extractor.

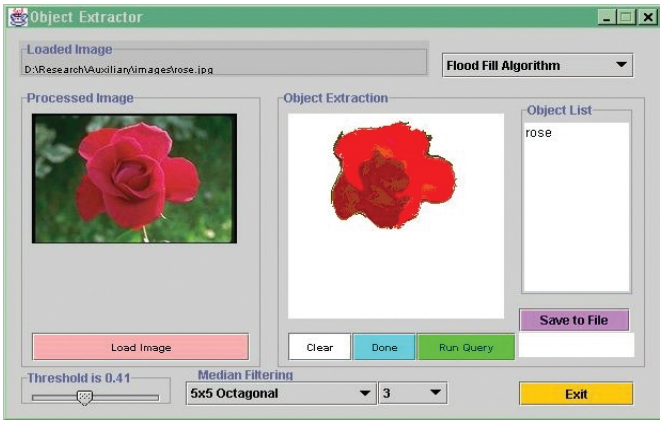

Figure 3. Object extractor.

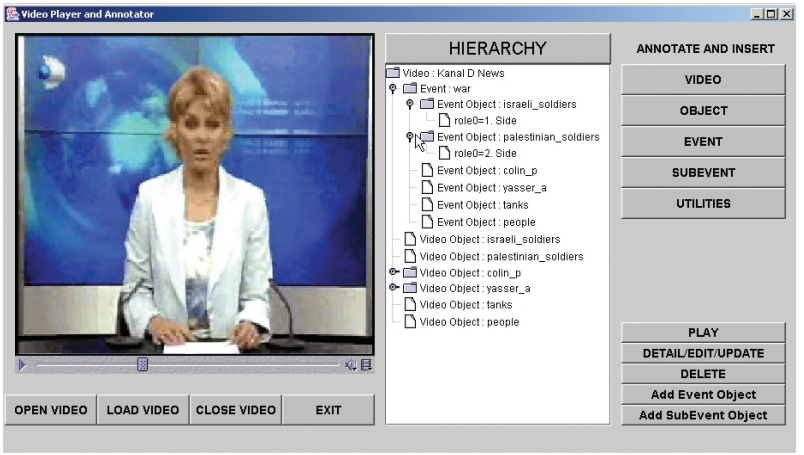

Figure 4. Video-annotator tool.

Spatial query specification

Spatial content of a video keyframe is the relative positioning of the keyframe's salient objects with respect to each other. This relative positioning consists of three sets of relations: directional, topological, and 3D. To query a keyframe's spatial content, users must specify these relations within a proper combination. Obviously, users should construct this combination with a logical connector AND; thus, all video frame(s) returned must contain these relations. In the spatial query specification window, shown in Figure 5, users sketch objects as rectangles. These rectangles represent object MBRs. Hence, each object is enclosed by its MBR during the database population phase, and the system extracts a keyframe's spatial content based on object MBRs. Similarly, the system automatically extracts directional and topological relations between objects in the query specification phase. Because it's impossible to extract 3D relations from 2D data, the system directs users to select appropriate 3D relations for object pairs.

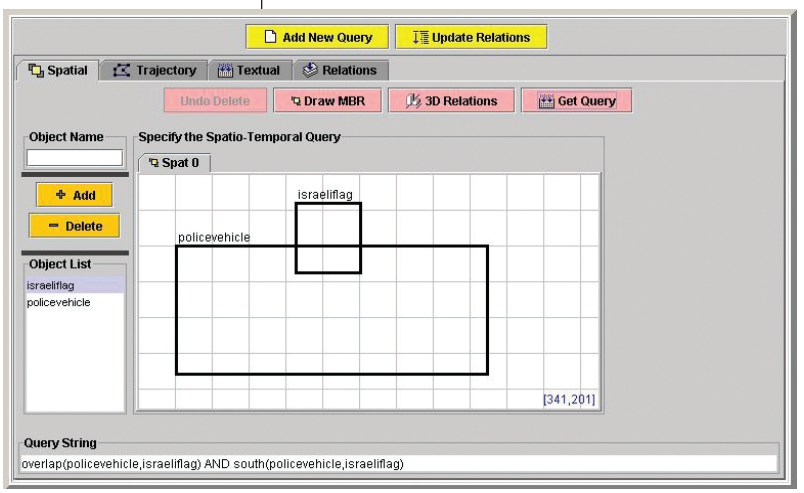

Figure 5. Spatial query specification.

Trajectory query specification

Trajectory of a salient object is a path of vertices corresponding to the object's locations in different video keyframes. Displacement values and directions between consecutive keyframes (vertices) define the trajectory fact of an object. In the trajectory query specification window, users can draw trajectories of objects, as Figure 6 shows. The trajectories displayed are dynamic; users can delete or insert any vertex to a trajectory. Users can also change vertex locations 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.

Final query formulation

The user interface specifies spatial and trajectory queries in different windows. Each of these specifications forms a subquery, and these subqueries combine in the final query formulation window, as Figure 7 shows. This window contains all specified subqueries and object-appearance relations for each object. Users can combine subqueries by logical operators (AND, OR) and temporal predicates (before, during, and so on). Except for the logical operator NOT, all temporal and logical operators are binary. If the user gives more than two subqueries as arguments to binary operators, the system combines them as cumulative pairs with the operators. After applying operators to subqueries, a new query is augmented to the list, and successive and hierarchical combinations become possible. The query interface sends the final query to the query processor. Furthermore, the user can send subqueries of the final query to the query processor at any time, which will provide partial results.

Example application

A news archive search system contains video clips of news broadcasts and can retrieve specific news fragments based on descriptions given as query conditions. The traditional approach to this task requires searching for keywords that describe the semantic content of the news fragments. For this, a traditional database system would suffice, because these systems would index news fragments by some textual data. However, traditional database systems don't consider spatio-temporal relations between objects and object trajectories. They also don't support low-level video queries (for example, color, shape, and texture). Furthermore, the traditional approach might result in

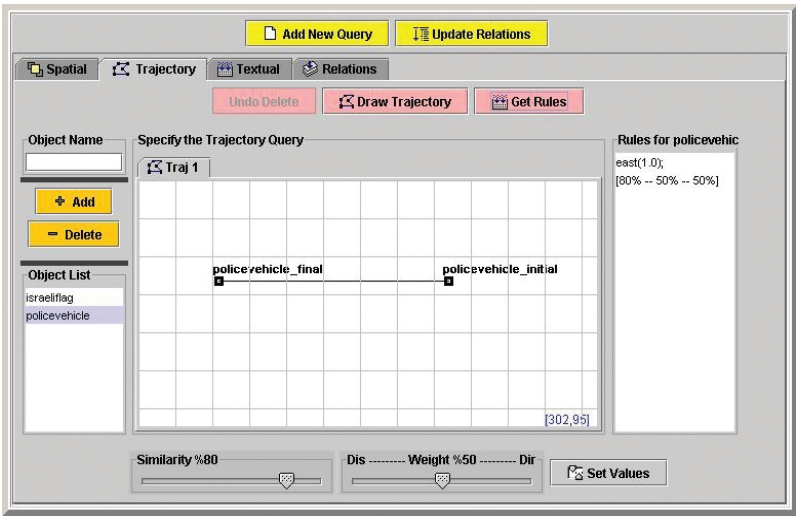

Figure 6. Trajectory query specification.

retrievals of irrelevant news fragments, while missing others that users would expect. Keyword-based searching isn't powerful enough to formulate what users have in mind as a query. Consequently, BilVideo with its support of spatio-temporal and semantic video queries provides a more useful search mechanism. To retrieve precise answers, users can also query news archives by some specific application-dependent predicates supported by BilVideo's query language. In the future, BilVideo will support some low-level features as well.[4]

As a basis for a spatio-temporal query example, we use a fragment video clip captured from news broadcast by a national Turkish TV station (Kanal D). We extracted facts representing the spatio-temporal relations between objects, object-appearance relations, and object trajectories and inserted them into the knowledge base prior to submitting the queries. Figure 8 (next page) shows a screen capture of the spatial relations dialog box of the fact-extractor tool for a keyframe in this news fragment, after the frame was processed. Queries 1 through 3 give examples of different types of spatio-temporal queries.

Query 1: Retrieve the segments from the sample news clip, where Arafat and Powell appear together alone (no other object of interest is in the scene), and Powell is to the right of Arafat.

```
select segment from vid
where appear_alone(arafat, powell) and
right(powell, arafat);
```

In this query, *appear_alone* is an external (application-dependent) predicate. It searches for video keyframes where only specified objects appear. The predicate *right* is a directional predicate. *Vid* is a unique video identifier assigned to the sample news video clip.

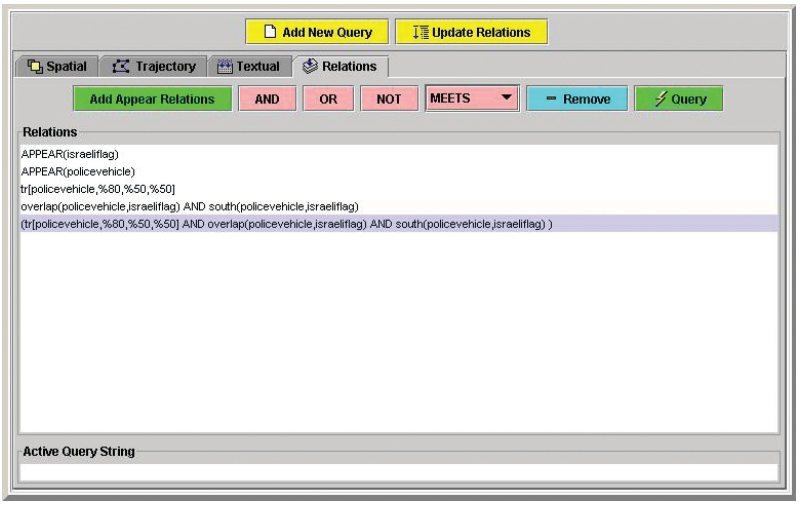

Figure 7. Final query formulation.

Query 2: Retrieve the segments from the sample news clip, where Turkish Prime Minister Ecevit and Turkish Foreign Affairs Minister Cem appear together close to each other, and Ecevit is to the right of and in front of Cem.

```
select segment from vid
where right(ecevit, cem) and
infrontof(ecevit, cem) and
close(ecevit, cem);
```

In this query, *close* is an external predicate. It

Figure 8. Spatial relations for a keyframe.

searches for video keyframes with specified objects in close proximity to each other. Here, we semantically define the closeness as follows: If two objects are close, then their MBRs aren't disjoint. This definition might change for other applications. The system can easily adapt to such changes through external predicates defined in the knowledge base according to application-specific needs. The predicate *infrontof* is a 3D predicate.

Query 3: Retrieve the segments from the sample news clip, where a police vehicle moves west, together with an Israeli flag that is above the vehicle and overlaps it, given a similarity threshold value of 0.8 and an allowed time gap value of 1 second.

```
select segment from vid
where (tr(policevehicle, [[west]])
sthreshold 0.8 tgap 1)
repeat and overlap(israeliflag, policevehicle)
and above(israeliflag, policevehicle);
```

This query contains similarity-based trajectory, directional, and topological conditions. The interval operator AND implies that the intervals returned as segments have all conditions satisfied, and that for all video frames in such segments, the flag is above and overlaps the police vehicle. The keywords *tgap* (time gap) and *repeat* indicate the trajectory condition. These conditions ensure that the query returns all clip segments satisfying the given conditions—that is, the police vehicle can only stop for at most 1 second at a time during its movement west.

Conclusions

BilVideo doesn't target a specific application, thus, it can support any application with video data searching needs. Moreover, BilVideo's query language provides a simple way to extend the system's query capabilities, through the use of external predicates. This feature makes BilVideo application independent, but we can easily fine-tune it to the specific needs of different applications, without sacrificing performance. Users can add application-dependent rules and/or facts to the knowledge base. Applications suited for BilVideo include sporting-event analysis, object-movement tracking (medical, biological, astrophysical, and so on), and video archives.

Our work on semantic query execution and query by low-level properties is ongoing. We're studying the optimization of user queries. Some tutorial video clips that demonstrate the visual query interface and tools are available on *MultiMedia*'s Web site at http://computer.org/multimedia/mu2003/u1toc.htm. We're currently working on the integration of the Web-based query interface and the query processor, so BilVideo isn't yet available on the Internet. **MM**

Acknowledgment

The Scientific and Technical Research Council of Turkey (TÜBITAK) supports this work under project code 199E025.

References

1. M.E. Dönderler, Ö. Ulusoy, and U. Güdükbay, "A Rule-Based Video Database System Architecture," *Information Sciences*, vol. 143, no. 1-4, 2002, pp. 13-45.
2. E. Saykol, U. Güdükbay, and Ö. Ulusoy, "A Semi-automatic Object Extraction Tool for Querying in Multimedia Databases," *Proc. 7th Workshop Multimedia Information Systems*, 2001, pp. 11-20; http://www.cs.bilkent.edu.tr/~ediz/bilmdg/papers/mis01.pdf.
3. E. Saykol, *Web-Based User Interface for Query Specification in a Video Database System*, master's thesis, Dept. of Computer Eng., Bilkent Univ., Ankara, Turkey, 2001.
4. E. Saykol, U. Güdükbay, and Ö. Ulusoy, *A Histogram-Based Approach for Object-Based Query-by-Shape-and-Color in Multi-Media Databases*, tech. rep. BU-CE-0201, Bilkent Univ., Ankara, Turkey, 2002.

Readers may contact Uğur Güdükbay at Bilkent Univ., Dept. of Computer Engineering, 06533 Bilkent, Ankara, Turkey, email gudukbay@cs.bilkent.edu.tr.

Readers may contact editor Tiziana Catarci at the Dept. of Information Systems, Univ. of Rome "La Sapienza," Via Salara 113, 00198 Rome, Italy, email catarci@dis.uniroma1.it.