BilVideo: Design and Implementation of a Video Database Management System [*]

Mehmet Emin Dönderler, Ediz Şaykol, Umut Arslan, Özgür Ulusoy and Uğur Güdükbay
E-mail: {*mdonder, ediz, aumut, oulusoy, gudukbay*}*@cs.bilkent.edu.tr*
Department of Computer Engineering, Bilkent University, Ankara, Turkey

Abstract. With the advances in information technology, the amount of multimedia data captured, produced, and stored is increasing rapidly. As a consequence, multimedia content is widely used for many applications in today's world, and hence, a need for organizing this data, and accessing it from repositories with vast amount of information has been a driving stimulus both commercially and academically. In compliance with this inevitable trend, first image and especially later video database management systems have attracted a great deal of attention, since traditional database systems are designed to deal with alphanumeric information only, thereby not being suitable for multimedia data.

In this paper, a prototype video database management system, which we call *BilVideo*, is introduced. The system architecture of *BilVideo* is original in that it provides full support for spatio-temporal queries that contain any combination of spatial, temporal, object-appearance, external-predicate, trajectory-projection, and similarity-based object-trajectory conditions by a rule-based system built on a knowledge-base, while utilizing an object-relational database to respond to semantic (keyword, event/activity, and category-based), color, shape, and texture queries. The parts of *BilVideo* (*Fact-Extractor*, *Video-Annotator*, its Web-based visual query interface, and its SQL-like textual query language) are presented, as well. Moreover, our query processing strategy is also briefly explained.

Keywords: video databases, multimedia databases, information systems, content-based retrieval, spatio-temporal relations, spatio-temporal query processing, video query languages.

1. Introduction

There is an increasing demand toward multimedia technology in recent years. As multimedia content (e.g. image, video, and audio) is widely used for many applications in today's world, a need for organizing this data, and accessing it from repositories with vast amount of information has been a driving stimulus both commercially and academically. In compliance with this inevitable trend, first image and especially later video database management systems have attracted a great deal of

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

attention, since traditional database systems are not suitable to be used for multimedia data.

In this paper, *BilVideo*, a prototype video database management system, is introduced. The architecture of *BilVideo* is original in that it provides full support for spatio-temporal queries that contain any combination of spatial, temporal, object-appearance, external-predicate, trajectory-projection, and similarity-based object-trajectory conditions by a rule-based system built on a knowledge-base, while utilizing an object-relational database to respond to semantic (keyword, event/activity, and category-based), color, shape, and texture queries. The knowledge-base of *BilVideo* contains a fact-base and a comprehensive set of rules implemented in Prolog. The rules in the knowledge-base significantly reduce the number of facts that need to be stored for spatio-temporal querying of video data [11]. Moreover, the system's response time for different types of spatio-temporal queries is at interactive rates. Query processor interacts with both the knowledge-base and object-relational database to respond to user queries that contain a combination of spatio-temporal, semantic, color, shape, and texture video queries. Intermediate query results returned from these two system components are integrated seamlessly by the query processor, and final results are sent to Web clients. *BilVideo* has a simple, yet very powerful SQL-like textual query language for spatio-temporal queries on video data [10]. For novice users, a visual query interface is provided. Both the query language and the visual query interface are currently being extended to support semantic, color, shape, and texture queries.

To the best of our knowledge, BilVideo is by far the most feature-complete video DBMS, as it supports spatio-temporal, semantic, color, shape, and texture queries in an integrated manner. Moreover, it is also unique in its support for retrieving any segment of a video clip, where the given query conditions are satisfied, regardless of how video data is semantically partitioned. To our knowledge, none of the video query systems available today can return a subinterval of a scene as part of a query result, simply because video features are associated with scenes defined to be the smallest semantic units of video data. In our approach, object trajectories, object-appearance relations, and spatio-temporal relations between video objects are represented as Prolog facts in a knowledge-base, and they are not explicitly related to semantic units of videos. Thus, *BilVideo* can return precise answers for user queries, when requested, in terms of frame intervals. Moreover, our assessment for the directional relations between two video objects is also novel in that two overlapping objects may have directional relations defined for them with respect to one another, provided that center points of the objects' Minimum Bounding Rectangles (MBRs) are different. It is because Allen's temporal interval algebra, [2], is not used as a basis for the directional relation definition in our approach: in order to determine which directional relation holds between two objects, center points of the objects' MBRs are used [11]. Furthermore, *BilVideo* query language provides three aggregate functions, *average*, *sum*, and *count*, which may be very attractive for some applications, such as sports analysis systems and mobile object tracking systems, to collect statistical data on spatio-temporal events.

The rest of the paper is organized as follows: A review of the research in the literature that is closely related to our work is given, in comparison to our work, in Section 2. Overall architecture of *BilVideo*, along with its knowledge-base structure, is briefly explained in Section 3. Section 4 presents the *Fact-Extractor* tool developed to populate the knowledge-base of the system with facts for spatio-temporal querying of video data. The tool also extracts color and shape histograms of objects, and stores them in the feature database for color and shape queries. The *Video-Annotator* tool that is used to annotate video clips for semantic content and to populate the system's feature database is introduced in Section 5. Section 6 presents the Web-based visual query interface. The system's SQL-like textual query language for spatio-temporal querying of video data is briefly explained in Section 7. In Section 8, we provide a short discussion on our query processing strategy, emphasizing on spatio-temporal query processing. Section 9 makes a discussion of the system's flexibility to support a broad range of applications. An example application of *BilVideo*, news archives search system, is also presented with some spatio-temporal queries in Section 9. We conclude stating our future work in Section 10.

2. Related Work

There are numerous content-based retrieval (CBR) systems, both commercial and academic, developed in recent years. However, most of these systems support only image retrieval. In this section, we restrict our discussion to the research in the literature mostly related to video modeling, indexing, and querying. A comprehensive review on the CBR systems in general can be found in [45, 49].

2.1. Spatio-Temporal Video Modeling

As mentioned in [41], there is a very limited number of proposals in the literature that take into account both spatial and temporal properties of video salient objects in an integrated manner. Some of the proposed index structures are *MR-tree*s and *RT-tree*s [48], *3D R-tree*s [42] and *HR-tree*s [31]. These structures are some adaptations of the well-known R-tree family. There are also quadtree-based indexing structures, such as *Overlapping Linear Quadtree*s [43], proposed for spatio-temporal indexing. All these approaches incorporate the MBR representation of spatial information within index structures. Thus, to answer spatio-temporal queries, spatial relations should be computed and checked for query satisfaction, which is a costly operation when performed during query processing. Our rule-based approach to model spatio-temporal relations in video data eliminates the need for the computation of relations at the time of query processing, thereby cutting down the query response time considerably. In our approach, a keyframe represents some consecutive frames in a video with no change in the set of spatial relations between video objects in the frames. Computed spatial relations for each keyframe are stored to model and query video data for spatio-temporal relations.

Li et al. describe an effort somewhat similar to our approach, where some spatial relations are computed by associated methods of objects while others may be derived using a set of inference rules [23]. Nonetheless, the system introduced in [21, 23, 25] does not explicitly store a set of spatio-temporal relations from which a complete set of relations between all pairs of objects can be derived by rules, and consequently, the relations which cannot be derived by rules are computed during query processing.

Sistla et al. propose a graph and automata based approach to find the minimal set of spatial relations between objects in a picture, given a set of relations that is a superset of the minimal set [38, 39]. The authors provide algorithms to find the minimal set from a superset, as well as to deduce all the relations possible from the minimal set itself for a picture. However, the directional relations are restricted to be defined only for disjoint objects as opposed to our approach, where overlapping objects may also have directional relations. Moreover, the set of inference rules considered is rather small compared to ours. The authors do not mention about any 3D relation, either. Furthermore, our fact-extraction algorithm is simpler, and it extracts spatio-temporal, appearance, and trajectory properties of objects from a video, even though we do not claim that it produces the minimal set of spatial relations in a video frame as they do for a picture.

In [7], a spatio-temporal semantic model for multimedia database systems is proposed. To model the semantic aspects of a multimedia presentation, browsing, and database searching, the augmented transition networks (ATNs), developed by Woods [47], are used. Temporal, spatial, and spatio-temporal relations of semantic objects are modeled via multimedia input streams, which are associated with subnetworks in ATNs. In the proposed model, each video frame has a subnetwork, which has its own multimedia string. Both subnetworks and their strings are created by the designer in advance for a class of applications, and spatio-temporal queries can be issued using a high-level query language such as SQL. The model supports spatio-temporal querying of video data; however, it provides a very coarse representation for the topological relations between objects, as there are only three types of topological relations supported, namely non-overlapping objects, partly overlapping objects, and completely overlapping objects. Moreover, the centroid points of the object MBRs are used for spatial reasoning, mapping an object to a point, which restricts the number of spatial relations that can be represented by the model. Furthermore, multimedia input strings are constructed for every frame by selecting a target object to determine the relative spatial positions of the other objects. Thus, for each target object in a frame, there will be a multimedia input string. This may drastically increase the complexity of query processing. In contrast, *BilVideo* uses a rule-based approach to model spatio-temporal relations between objects. Our approach not only yields considerable space savings, as only a subset of the entire relation set is stored for each video keyframe, but it also provides simple, yet very powerful query capabilities for *BilVideo*. *BilVideo* supports all possible spatial relations in 2D, and has a set of 3D relations defined for the third dimension. It also supports all temporal relations defined by Allen [2]. Thus, *BilVideo* allows users to specify spatio-temporal queries with much finer granularity, and the query results returned are more precise.

2.2. Semantic Video Modeling

A video database system design for automatic semantic content extraction, and semantic-based video annotation and retrieval with textual tags is proposed in [26]. Video semantic content is automatically extracted using low-level image features (color, shape, texture, and motion) and temporal diagrams constructed for videos and scenes. Shots/Scenes are tagged with textual descriptions, which are used for semantic queries. However, automatic extraction of semantic content and tagging shots/scenes with some textual descriptions with respect to the extracted information are limited to simple events/activities.

Hacid et al. propose a video data model that is based on logical video segment layering, video annotations, and associations between them [15]. The model supports retrieval of video data based on its semantic content. The authors also give a rule-based constraint query language for querying both semantic and video image features, such as color, shape, and texture. Color, shape, and texture query conditions are sent to IBM's QBIC system, whereas semantic query conditions are processed by FLORID, a deductive object-oriented database management system. A database in their model can essentially be thought of as a graph, and a query in their query language can be viewed as specifying constrained paths in the graph. *BilVideo* does not use a rule-based approach for semantic queries on video data. In this regard, our semantic video model diverts from that of Hacid et al.

There is also some research in the literature that takes into account audio and closed caption text stored together with video data for extracting semantic content from videos and indexing video clips based on this extracted semantic information. In [5], a method of event-based video indexing by means of intermodel collaboration, a strategy of collaborative processing considering the semantic dependency between synchronized multimodal information streams, such as auditory and textual streams, is proposed. The proposed method aims to detect interesting events automatically from broadcasted sports videos, and to give textual indexes correlating the events to shots. In [14], a digital video library prototype, called VISION, is presented. In VISION, videos are automatically partitioned into short scenes using audio and closed caption information. The resulting scenes are indexed based on their captions, and stored in a multimedia system. Informedia's news-on-demand system described in [16] also uses the same information (audio and closed caption) for automatic segmentation and indexing to provide efficient access to news videos. Satoh et al. propose a method of face detection and indexing by analyzing closed caption and visual streams [34]. However, all these systems and others that take into account audio and closed caption information stored with videos for automatic segmentation and indexing are application-dependent, whilst *BilVideo* is not.

2.3. Systems and Languages

QBIC: QBIC is a system primarily designed to query large online image databases [12]. In addition to text-based searches, QBIC also allows users to pose queries using sketches, layout or structural descriptions, color, shape, texture, sample images (Query by Example), and other iconic and graphical information. As the basis for content-based search, it supports color, texture, shape, and layout. QBIC provides some support for video data, as well [13]; however, this support is limited to the features used for image queries. Consequently, spatio-temporal relations between salient objects and semantic content of video data are not taken into account for video querying.

OVID: A paper by Oomoto and Tanaka [32] describes the design and implementation of a prototype video object database system, named OVID. Main components of the OVID system are VideoChart, VideoSQL, and Video Object Definition Tool. Each video object consists of a unique identifier, a pair of starting and ending video frame numbers for the object, annotations associated with the object as a set of attribute/value pairs, and some methods such as *play*, *inspect*, *disaggregate*, *merge* and *overlap*. Users may define different video objects for the same frame sequences, and each video object is represented as a bar chart on the OVID user interface VideoChart. The VideoChart is a visual interface to browse the video database, and manipulate/inspect the video objects within the database. The query language of the system, VideoSQL, is an SQL-like query language used for retrieving video objects. The result of a VideoSQL query is a set of video objects, which satisfy given conditions. Before examining the conditions of a query for each video object, target video objects are evaluated according to the interval inclusion inheritance mechanism. Nevertheless, the language does not contain any expression to specify spatial and temporal conditions on video objects. Hence, VideoSQL does not support spatio-temporal queries, which is a major weakness of the language.

AVIS: In [28], a unified framework for characterizing multimedia information systems, which is built on top of the implementations of individual media, is proposed. Some of user queries may not be answered efficiently using these data structures; therefore, for each media-instance, some feature constraints are stored as a logic program. Nonetheless, temporal aspects and relations are not taken into account in the model. Moreover, complex queries involving aggregate operations as well as uncertainty in queries require further work to be done. In addition, although the framework incorporates some feature constraints as facts to extend its query range, it does not provide a complete deductive system as we do.

The authors extend their work defining feature-subfeature relationships in [27]. When a query cannot be answered, it is relaxed by substituting a subfeature for a feature. In [1], a special kind of segment tree called *frame segment tree*, and a set of arrays to represent objects, events, activities, and their associations are introduced. The proposed model is based on the generic multimedia model described in [28]. Additional concepts introduced in the model are activities, events, and their associations with objects, thereby relating them to frame sequences. The proposed data model and algorithms for handling different types of semantic queries were implemented within a prototype, called Advanced Video Information System (AVIS). The idea of activities, events, and roles given in [1] is similar to that of *BilVideo*. However, objects have no attributes other than the roles defined for the events in [1]. In [18], an SQL-like video query language, based on the data model developed by Adalı et al. [1], is proposed. Nevertheless, the proposed query language does not provide any support for temporal queries on events. Nor does it have any language construct for spatio-temporal querying of video clips, since it was designed for semantic queries on video data. In our query model, temporal operators, such as *before*, *during*, etc., may also be used to specify order in time between events, just as they are used for spatio-temporal queries.

VideoQ: An object-oriented content-based video search engine, called VideoQ, is presented in [6]. VideoQ provides two methods for users to search for video clips. The first one is to use *keywords*, since each video shot is annotated. Moreover, video clips are also catalogued into a subject taxonomy, and users may navigate through the catalogue. The other method is a visual one, which extends the capabilities of the textual search. A video object is a collection of regions that are grouped together under some criteria across several frames. A region is defined as a set of pixels in a frame, which are homogeneous in the features of interest to the user. For each region, VideoQ automatically extracts the low-level features: *color*, *shape*, *texture*, and *motion*. These regions are further grouped into higher semantic classes, known as video objects. Motion is the key attribute in VideoQ, and the motion trajectory interface allows users to specify a motion trajectory for an object of interest. However, when a multiple-object query is submitted, VideoQ does not use the video objects' relative ordering in space and in time. Therefore, VideoQ does not support spatio-temporal queries on video data.

VideoSTAR: VideoSTAR proposes a generic data model that makes it possible sharing and reusing of video data [17]. Thematic indexes and structural components might implicitly be related to one another, since frame sequences may overlap and may be reused. Therefore, considerable processing is needed to explicitly determine the relations, making the system complex. Moreover, the model does not support spatio-temporal relations between video objects.

CVQL: A content-based logic video query language, CVQL, is proposed in [19]. Users retrieve video data specifying some spatial and temporal relationships for salient objects. An elimination-based preprocessing for filtering unqualified videos, and a behavior-based approach for video function evaluation are also introduced. For video evaluation, an index structure, called *M-index*, is proposed. Using this index structure, frame sequences satisfying a query predicate can be efficiently retrieved. Nonetheless, topological relations between salient objects are not supported, since an object is represented by a point in two-dimensional (2D) space. Consequently, the language does not allow users to specify topological queries. Nor does it support similarity-based object-trajectory queries.

MOQL and MTQL: In [24], multimedia extensions to Object Query Language (OQL) and TIGUKAT Query Language (TQL) are proposed. The extended languages are called Multimedia Object Query Language (MOQL) and Multimedia TIGUKAT Query Language (MTQL), respectively. The extensions made are spatial, temporal, and presentation features for multimedia data. MOQL has been used both in the Spatial Attributes Retrieval System for Images and Videos (STARS) [22] and an object-oriented SGML/HyTime compliant multimedia database system [33], developed at the University of Alberta.

MOQL and MTQL can be used for content-based spatial and temporal queries on video. Both languages allow for 3D-relation queries, as we call them, by *front*, *back*, and their combinations with other directional relations, such as *front_left*, *front_right*, etc. *BilVideo* query language has a different set of third-dimension (3D) relations, though. 3D relations supported by *BilVideo* query language are *infrontof*, *behind*, *strictlyinfrontof*, *strictlybehind*, *touchfrombehind*, *touchedfrombehind*, and *samelevel*. The moving object model integrated in MOQL and MTQL, [20], is also different from our model. *BilVideo* query language does not support similarity-based retrieval on spatial conditions while MOQL and MTQL do. Nonetheless, it does allow users to specify separate weights for the directional and displacement components of trajectory conditions in queries, which both languages lack.

Nabil et al. propose a symbolic formalism for modeling and retrieving video data by means of moving objects in video frames [30]. A *scene* is represented as a connected digraph, whose nodes are the objects of interest in the scene while edges are labeled by a sequence of spatio-temporal relations between two objects corresponding to the nodes. Trajectories are also associated with object nodes in the scene graph. A graph is precomputed for each scene in video data, and stored before query processing. For each user query, a query scene graph is constructed to match the query with the stored scene graphs. However, 3D relations are not addressed in [30]. The concepts used in the model are similar to those adopted in [20]; therefore, the same arguments we made for MOQL and MTQL also hold for the model proposed in [30].

There are also some commercial products, such as Virage's VideoLogger [46] and Convera's Screening Room [8]. VideoLogger segments the video, and generates a storyboard of keyframes that can be browsed, looking for changes in visual content. Moreover, faces, text, and numbers on frames can be identified, and spoken words, speaker names, and audio types can be converted into text from audio signals. VideoLogger also extracts closed caption text or teletex from the video signal. Screening Room by Convera basically provides features similar to those of VideoLogger, where visual features, closed caption text, and audio play an important role in indexing and querying video data. Nonetheless, neither VideoLogger nor Screening Room provides facilities to fully support spatio-temporal queries on video data.

2.4. Multimedia Content Description Interface (MPEG-7)

The Moving Picture Experts Group (MPEG) published a new standard, called *Multimedia Content Description Interface* (MPEG-7), in February 2002. An overview of the MPEG-7 standard can be found in [29]. MPEG-7 provides a set of descriptors (D) and Descriptions Schemes (DS). Descriptors are quantitative measures of audio-visual features, and Description Schemes define the structures of descriptors and their relationships. Audio-Visual (AV) material with associated MPEG-7 data can be indexed and searched for. Audio-visual material may include images, graphics, 3D models, audio, video, and information about their presentation (composition). MPEG-7 is currently the most complete description standard for multimedia data.

MPEG-7 standardizes how multimedia data is described, but does not attempt to address the issues related with feature extraction and querying of multimedia data. Thus, feature extraction and querying are outside the scope of MPEG-7. In short, MPEG-7 standardizes the exchange of descriptive information for multimedia data, but it is not suitable to serve as a multimedia data model for a database. Rather, MPEG-7 should be thought of as a standard that complements the multimedia databases, and helps build distributed multimedia systems. This role of the MPEG-7 standard is very important, as the issues relating to media delivery, e.g. Quality of Service (QoS), have generally been neglected in the literature so far.

BilVideo can be thought of as a complement to MPEG-7 by providing a novel architecture for a video database management system, along with novel data models and query construction/processing mechanisms to support a large variety of queries on video data including spatio-temporal, semantic, color, shape, and texture queries in an integrated manner.

3. BilVideo System Architecture

BilVideo is built over a client-server architecture as illustrated in Figure 1. The system is accessed on the Internet through its visual query interface. Users may query the system with sketches, and a visual query is formed by a collection of objects with some conditions, such as object trajectories with similarity measures, spatio-temporal orderings of objects, annotations, and events. Object motion is specified as an arbitrary trajectory for each salient object of interest, and annotations can be used for keyword-based video search. Users are able to browse the video collection before posing complex and specific queries. Furthermore, an SQL-like textual query language is also available for the users. Queries constructed by the visual query interface are first translated to their equivalent textual query language statements before being sent to the query server. In the heart of the system lies the query processor, which is responsible for processing and responding to user queries in a multi-user environment. The query processor communicates with a knowledge-base and an object-relational database. The knowledge-base stores fact-based meta data used for spatio-temporal queries, whereas semantic and histogram-based (color, shape, and texture) meta data is stored in the feature database maintained by the object-relational database. Raw video data and video data features are stored separately. Semantic meta data stored in the feature database is generated and updated by the Video-Annotator tool, and the facts-base is populated by the Fact-Extractor tool. The Fact-Extractor tool also extracts color and shape histograms of objects of interest in video keyframes to be stored in the feature database. In our histogram-based approach [37], three histograms -distance, angle and color- are used to store shape and color content of image and video data. Distance and angle histograms are filled with respect to the distribution of the pixels around the center of mass of the objects, hence the histogram-based approach resembles the human vision system. To the best of our knowledge, this is the first approach that employs these three histograms for object-based querying of image and video data by shape and color.

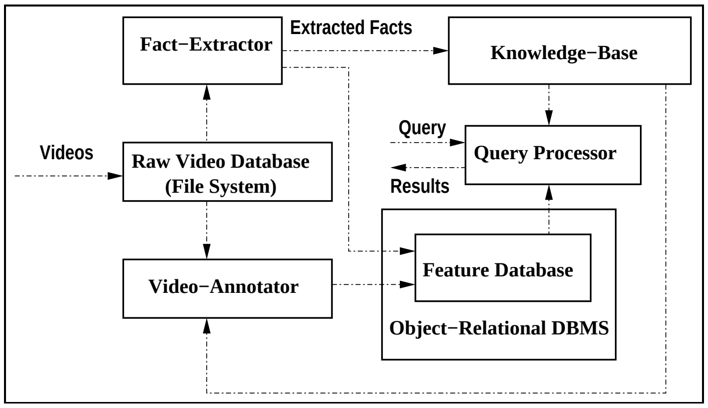


Figure 1. BilVideo System Architecture

3.1. Knowledge-Base Structure

In the knowledge-base, each fact[1] has a single frame number, which is of a keyframe. This representation scheme allows our inference engine Prolog to process spatio-temporal queries faster and easier in comparison to using frame intervals for the facts. It is because the frame interval processing to form the final query results is carried out efficiently by some optimized code outside the Prolog environment. Therefore, the rules used for querying video data, which we call *query rules*, have frame-number variables associated. A second set of rules, called *extraction rules*, was also created to work with frame intervals so as to extract spatio-temporal relations from video data. Extracted spatio-temporal relations are then converted to be stored as facts with frame numbers of the keyframes in the knowledge-base, and these facts are used by the query rules for query processing in the system.

In the knowledge-base, only are the basic facts stored, but not those that can be derived by rules according to our fact-extraction algorithm. Using a frame number instead of a frame interval introduces some space overhead, because the number of facts increases due to the repetitions of some relations for each keyframe over a frame interval. Nevertheless, it also greatly reduces the complexity of the rules, and improves the overall query response time. In our earlier work, we presented the results of our performance tests conducted on the knowledge-base of *BilVideo* to show that the system is scalable for spatio-temporal queries in terms of the number of salient objects per frame and the total number of frames

[1] Except for *appear* and *object-trajectory* facts, which have frame intervals as a component instead of frame numbers because of storage space, ease of processing, and processing cost considerations.

in a video clip [11]. The test results also demonstrate the space savings achieved due to our rule-based approach. Moreover, the system's response time for different types of spatio-temporal queries posed on the same data was at interactive rates. Details on the knowledge-base structure of *BilVideo*, types of the rules/facts used, their definitions, and our performance tests involving spatial relations on real and synthetic video data can be found in [11].

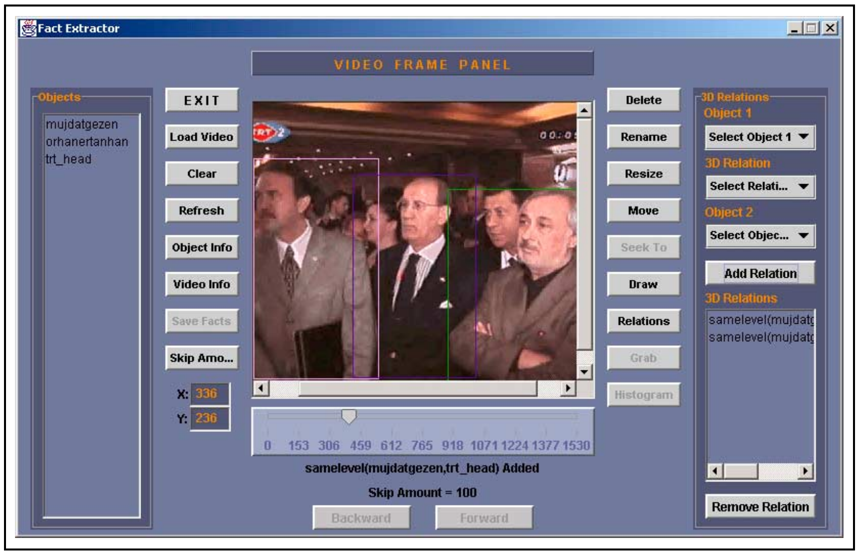
4. Fact-Extractor Tool

Fact-Extractor is used to populate the facts-base of *BilVideo*, and to extract color and shape histograms of the objects in video keyframes. Spatio-temporal relations between objects, object-appearance relations, and object trajectories are extracted semi-automatically. This data is stored in the facts-base as a set of facts representing the relations and trajectories, and it is used to query video data for spatio-temporal query conditions. Sets of facts are kept in separate facts-files for each video clip processed, along with some other video specific data, such as video length, video rate, keyframes list, etc., extracted automatically by the tool. Extracted color and shape histograms of salient objects are stored in the feature database to be used for color and shape queries. *Fact-Extractor* supports MPEG, AVI, and QT video formats.

The fact-extraction process is semi-automatic: currently, objects are manually specified in video frames by their 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 obtained 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 carries the set of 3D relations of a frame to 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, once the objects are identified by their MBRs. While computing the trajectories, the tool uses the center points of the object MBRs, and it is assumed that there is no camera movement. 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 currently 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 the process of object extraction, we developed an *Object-Extractor* utility module [36], which will be integrated into the *Fact-Extractor* tool very soon. This module is used to extract objects from video frames: the video frame from which the objects will be extracted passes through a color space conversion step, and the color vectors of all the pixels are transformed from RGB into HSV color space, since RGB color space is not perceptually uniform. Then, this data is used in the quantization step yielding 18 hue, 3 saturation, 3 value levels, and 4 gray levels as in VisualSeek [40] and VideoQ [6]. After this step, the median filtering algorithm is applied to eliminate the non-dominant color regions. We proposed a modified version of the well-known *Flood Fill* algorithm, and used it in the Object-Extractor module [36]. This modified algorithm, which we name *Flood Fill for Extraction*, is designed to specify object regions. The user clicks on an object, and this user-clicked pixel initiates the process, which forks into four branches corresponding to the four neighboring pixels (north, east, south, and west). As soon as the difference between the processed pixel's and the initiative pixel's colors exceeds a pre-specified threshold, the execution at a branch stops, and when all of the branches stop, the process ends. The user may continue to specify new initiative pixels clicking on some other regions on the object, if necessary. Our experimental results show that a few mouse clicks on an object suffice to extract it effectively, in most cases. When the *Object-Extractor* module is incorporated into the *Fact-Extractor* tool, it will automate the object MBR specification with only a few mouse clicks on objects.

The *Fact-Extractor* tool segments videos into shots, each represented by a single keyframe. This segmentation is based on spatial relationships between objects in video frames: videos are segmented into shots whenever the current set of relations between objects changes, and the frames, where these changes occur, are chosen as keyframes. The relations stored in the facts-base are those that are present in such keyframes, because the set of relations in a frame does not change from frame to frame in the same shot. Hence, *BilVideo* can support much finer granularity for spatio-temporal query processing in comparison to other systems, and this is independent of the semantic segmentation of videos: it allows users to retrieve any part of a video, where the relations do not change at all, as a result of a query.



mujdatgezen
orhanertanhan
trt_head

X: 336
Y: 236

samelevel(mujdatg
samelevel(mujdatg

samelevel(mujdatgezen,trt_head) Added

Skip Amount = 100

Figure 2. Fact-Extractor Tool

Fact-Extractor uses a heuristic algorithm to decide which spatio-temporal relations to store as facts in the knowledge-base. In this algorithm, objects at each frame are ordered with respect to the x-coordinates of the center points of their MBRs. Object index values are used as object labels after sorting, which is in ascending order. Then, objects are paired in relation to their labels. Directional and topological relations are computed for each possible object pair whose first object's label is smaller than that of the second object and whose label distance is one. The *label distance* of an object pair is defined to be the absolute value of the difference between the two object labels in the pair. Having processed all the pairs with the label distance one, the same operation is carried out for the pairs of objects whose label distance is two. This process is continued in the same manner and terminated when the distance reaches the number of objects in the frame. Details about our fact-extraction algorithm can be found in [11]. Figure 2 gives a snapshot of the *Fact-Extractor* tool.

5. Video-Annotator Tool

Video-Annotator is a tool developed for annotating video clips for semantic content, and populating the system's feature database with this data to support semantic queries [4]. The tool also provides facilities for viewing, updating, and deleting semantic data that has already been

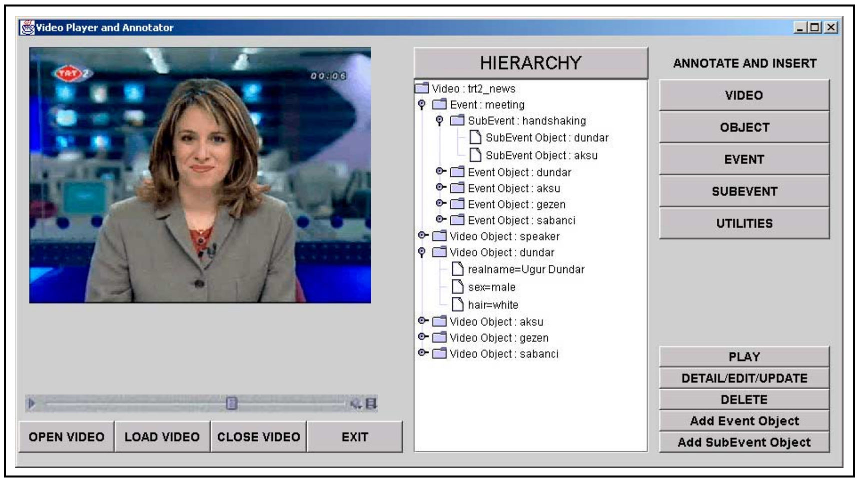


Figure 3. Video-Annotator Tool

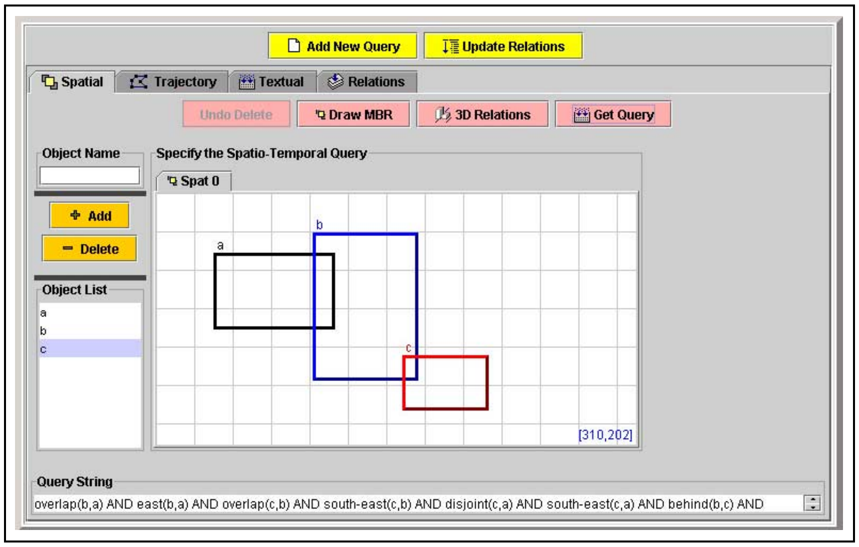
obtained from video clips and stored in the feature database. *Video Annotator* supports MPEG, AVI, and QT video formats. A snapshot of the tool is given in Figure 3.

Our semantic video model is inspired from that of AVIS [1], where activities, events, and their associations with objects are introduced, relating them to frame sequences, based on the generic multimedia model proposed in [28]. *BilVideo*'s 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 the descriptional data of video clips. Conditions may include title, length, producer, production year, category, and director information about a video. *Event/activity* queries are the most common queries among all, and can be used to retrieve videos by specifying events that occur at the *sequence* layer, 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 subevents associated with scenes. *Object* queries are used to retrieve videos by specifying semantic object features. As videos are annotated, video objects are also associated with some descriptional meta data.

Video consists of events, and activities are the abstractions of events. For example, wedding is an activity, but the wedding of Richard Gere and Julia Roberts in a movie is considered as an event, a specialization of activity wedding. Hence, activities can be thought of as classes while events constitute some instances (specializations) of these classes in videos. In our semantic model, a number of roles are defined for each activity. For example, activity murder is defined with two roles, *murderer* and *victim*. If the murder of Richard Gere by Julia Roberts is an event in a movie, then Richard Gere and Julia Roberts have the roles *victim* and *murderer*, respectively. Events may also have subevents defined for them, and these subevents are used to detail events and model the relationships between objects of interest. For example, a party event in a video may have a number of subevents, such as drinking, eating, dancing and talking. Moreover, the objects of interest in the party event may assume the roles *host* and *guest*. Objects are defined and assigned roles for an event; however, they are also associated with subevents defined for an event because actions represented by subevents, such as dancing and talking in the example given, are performed by those objects. Furthermore, subevents may overlap in time, as is the case for events. In our semantic video model, a video is segmented into sequences, which are in turn divided into scenes. This task is accomplished by specifying events and subevents, because events and subevents are associated with sequences and scenes, respectively. The order of annotation follows our hierarchical semantic model from top to bottom. In other words, video is annotated first as a whole entity, and the annotation of events with their corresponding subevents may be carried out afterwards. During this process, objects may be annotated whenever needed. Further information on the video annotation process, the *Video-Annotator*, and our relational database schema for storing semantic contents of videos can be found in [3].

6. Web-based User Interface

BilVideo can handle multiple requests over the Internet via a graphical query interface [35]. The interface is composed of query specification windows for *spatial* and *trajectory* queries. The specification and formation of these queries vary significantly, and hence, specific windows are created to handle each type. These two types of primitive queries can be combined with temporal predicates (*before*, *during*, etc.) to query temporal contents of videos. Specification of queries by visual sketches is much easier for novice users, and most of the relations are computed automatically based on these sketches.



Query String

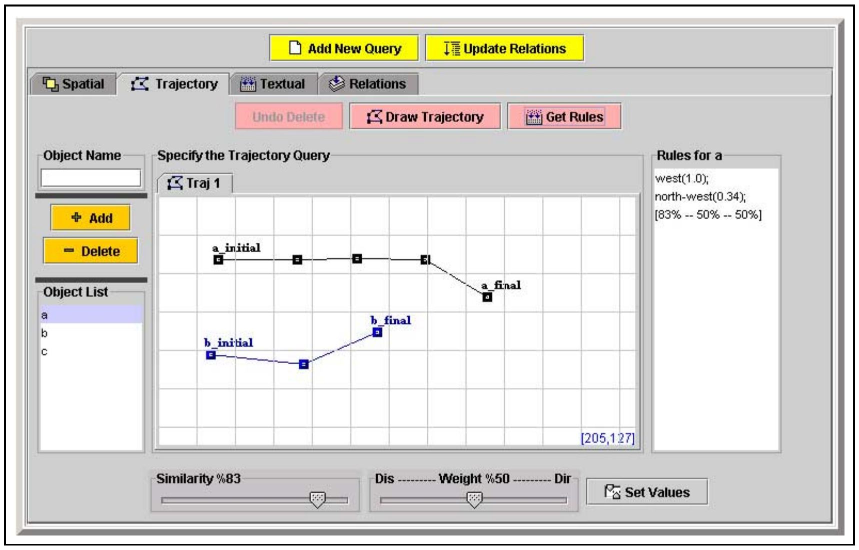
overlap(b,a) AND east(b,a) AND overlap(c,b) AND south-east(c,b) AND disjoint(c,a) AND south-east(c,a) AND behind(b,c) AND

Figure 4. Spatial Query Specification Window

6.1. Spatial Query Specification

Spatial content of a video keyframe is the relative positioning of its objects with respect to each other. This relative positioning consists of three sets of relations: *directional*, *topological* and *3D relations*. So as to query the spatial content of a keyframe, these relations have to be specified in the query within a proper combination.

In the spatial query specification window shown in Figure 4, objects are sketched by rectangles, representing the MBRs of the objects. Similar to the database population phase, the directional and topological relations between objects are extracted automatically from the objects' MBRs in the query specification phase. Since it is impossible to extract 3D relations from 2D data, users are guided to select proper 3D relations for object pairs. To provide flexibility, some facilities are employed in the window. Users may change the locations, sizes, and relative positions of the MBRs during the query specification. The spatial-relation extraction process takes place after the final configuration is formalized. Deleting or hiding an object modifies the relation set, and if this modification occurs after the extraction process, the relations associated with the deleted or hidden objects are removed accordingly from the set.



west(1.0);
north-west(0.34);
[83% -- 50% -- 50%]

Figure 5. Trajectory Query Specification Window

6.2. Trajectory Query Specification

Trajectory of an 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 5, users can draw trajectories of objects as a sequence of vertices. The trajectories are dynamic in the sense that any vertex can be deleted from, or a new vertex can be inserted to the trajectory of an object. Locations of the vertices can also be altered to obtain a desired trajectory.

Object-trajectory queries are similarity-based; therefore, users may specify a similarity value. Since an object trajectory contains lists of directions and displacements, weights can be assigned to each list. By default, both lists have equal weights (i.e., the weights are 0.5); however, users may modify these values that add up to 1. There are two sliders on the trajectory specification window (see lower part of Fig. 5): the first slider is for similarity value and its range varies from 0 to 100, where 100 corresponds to the exact match, and the other slider is for assigning weights. If the head of the slider used for weight specification is closer to the left end, directions become more important than displacements, and vice versa.

6.3. Final Query Formulation

Spatial and trajectory queries are specified in separate windows through the user interface. Each of these specifications forms a subquery, and these subqueries may be combined in the final query formulation window. This window contains all the specified subqueries, as well as the object-appearance relations for each object. Users can combine subqueries by logical operators (*and, or*) and temporal predicates (*before*, *during*, etc.). Except for the logical operator *not*, all temporal and logical operators are binary. After applying operators to subqueries, a new query is augmented to the list, and hierarchical combinations become possible. After the final query is formed, it can be sent to the query server. Furthermore, any subquery of the final query may also be sent to the query server at any time to obtain partial results if requested.

For an example query formulation using the visual subqueries in Figs. 4 and 5, the final query in textual form is given as

`appear(a) AND appear(b) AND appear(c) AND finishes(`S_1`,before(`T_a`,`T_b`))`

where T_a and T_b denote the query trajectories of `a` and `b`, respectively, and S_1 is given as

```
overlap(b,a) AND east(b,a) AND overlap(c,b) AND southeast(c,b) AND
disjoint(c,a) AND behind(b,c) AND southeast(c,a) AND samelevel(a,b).
```

7. Video Query Language

In this section, the textual query language of *BilVideo*, which is similar to SQL in structure, is presented. The language can currently be used for spatio-temporal queries that contain any combination of spatial (directional, topological, and 3D-relation), temporal (before, during, meets, etc.), object-appearance, external-predicate, trajectory-projection, and similarity-based object-trajectory conditions. As a work in progress, the language is being extended so that it could support semantic, color, shape, and texture queries as well in a unified and integrated manner. A semantic grammar for the language has already been defined [3], and it will be embedded to the current grammar (spatio-temporal) of the language. *BilVideo* textual query language has four basic statements for retrieving information on spatio-temporal query conditions:

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. Regardless of the target type specified, video identifiers for videos in which the conditions are satisfied are always returned as part of the query answer. 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 facts-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, and they can be defined recursively. Consequently, a condition may contain any combination of spatio-temporal subconditions. Further information about *BilVideo* textual query language is given in [10].

8. Query Processing

Figure 6 illustrates how the query processor communicates with Web clients and the underlying system components. Web clients make a connection request to the query request handler, which creates a process for each request, passing a new socket for communication between the process and the Web client. Then, the clients send user queries to the processes created by the query request handler. If the queries are specified visually, they are transformed into SQL-like textual query language expressions before being sent to the server. Having received the query from the client, each process calls the query processor, compiled as a library, with a query string and waits for the query answer. When the query processor returns, the process communicates the answer to the Web client issuing the query, and exits. The query processor first groups spatio-temporal, semantic, color, shape, and texture query conditions into proper types of subqueries. Spatio-temporal subqueries are reconstructed as Prolog-type knowledge-base queries, whereas semantic, color, shape, and texture subqueries are sent as regular SQL queries to an object-relational database. Intermediate results obtained are integrated by the query processor, and returned to the query request handler, which communicates the final results to Web clients.

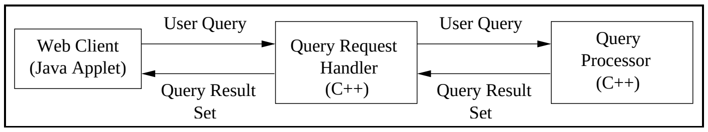


Figure 6. Web Client - Query Processor Interaction

The phases of query processing for spatio-temporal queries are *query recognition*, *query decomposition*, and *query execution*. In the phase of query recognition, the lexer partitions a query into tokens, which are then passed to the parser with possible values for further processing. The parser assigns structure to the resulting pieces, and creates a parse tree to be used as a starting point for query processing. The parse tree is traversed to construct a query tree in the next phase, called query decomposition phase. In this phase, queries are decomposed into three basic types of subqueries: *Prolog subqueries* (directional, topological, 3D-relation, external predicate, and object-appearance) that can be directly sent to the inference engine Prolog, *trajectory-projection subqueries* that are handled by the trajectory projector, and *similarity-based object-trajectory subqueries* that are processed by the trajectory processor. In the query execution phase, the query tree is traversed in postorder, executing each subquery separately and performing interval processing in internal nodes so as to obtain the final set of results.

One of the main challenges in query execution is to handle such user queries where the scope of a variable used extends to several subqueries after the query is decomposed. It is a challenging task because subqueries are processed separately, accumulating and processing the intermediate results along the way to form the final set of answers. Hence, the values assigned to variables for a subquery are retrieved and used for the same variables of other subqueries within the scope of these variables. Therefore, it is necessary to keep track of the scope of each variable for a query. This scope information is stored in a hash table generated for the variables. Dealing with variables makes the query processing much harder, but it also empowers the query capabilities of the system and yields much richer semantics for user queries. Currently, the query processor can handle a wide range of spatio-temporal queries, and we are working on extending it to support semantic, color, shape, and texture queries, as well. Details on the query processor, and some example queries that demonstrate how the query processor decomposes a spatio-temporal query into subqueries are given in [10].

8.1. Interval Processing

In *BilVideo* query model, intervals are categorized into two types: *non-atomic* and *atomic* intervals. If a condition holds for every frame of a part of a video clip, then the interval representing an answer for this condition is considered as a non-atomic interval. Non-atomicity implies that for every frame within an interval in question does the condition hold. Hence, the condition holds for any subinterval of a non-atomic interval, as well. This implication is not correct for atomic intervals, though. The reason is that the condition associated with an atomic interval does not hold for all its subintervals. Consequently, an atomic interval cannot be broken into its subintervals for query processing. On the other hand, subintervals of an atomic interval are populated for query processing, provided that conditions are satisfied in their range. In other words, the query processor generates all possible atomic intervals for which the given conditions are satisfied. This interval population is necessary since atomic intervals cannot be broken into subintervals, and all such intervals, where the conditions hold, should be generated for query processing. The intervals returned by the *Prolog subqueries* that contain directional, topological, object-appearance, 3D-relation, and external-predicate conditions are non-atomic, whereas those obtained by applying the temporal operators to the interval sets, as well as those returned by the similarity-based object-trajectory subqueries are atomic intervals. Since the logical operators *AND*, *OR* and *NOT* are considered as interval operators when their arguments contain intervals to process, they also work on intervals. Further information, along with some examples, about the interval processing and the semantics of the interval operators can be found in [10].

9. Application Areas

BilVideo is a full-fledged Web-based video database management system that supports spatio-temporal, semantic, color, shape, and texture queries on video data. There are only a few video database prototypes around developed either for academic or commercial purposes; nonetheless, they do only provide support for a rather small subset of the video features in comparison to *BilVideo*. 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 inputs queries from users. The visual query interface should be simple and easy-to-use, yet powerful enough to make use of all the capabilities of the underlying system.

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, color, shape, and texture video features. Furthermore, *BilVideo* query language provides a simple way to extend the system's query capabilities via *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.

9.1. An Example Application: News Archives Search[2]

In this section, we present an application, news archives search, for *BilVideo*. A news archives search system contains video clips of news broadcasts, and is used to retrieve specific news fragments based on some descriptions. The traditional approach for accomplishing this task is to provide some keywords that would describe semantic content of the news fragments for retrieval. For this, a traditional database system would suffice, since news fragments are indexed by some textual data. Nevertheless, spatio-temporal relations between objects, and object trajectories are not considered. Moreover, traditional database systems also lack of support for color, shape, and texture queries. Furthermore, the traditional approach might result in retrievals of some news fragments that are irrelevant to what the user wants to see, while also missing some others that are actually expected by the user. It is because keyword-based search is not powerful enough to formulate what the user has in his/her mind as a query. Hence, some other search mechanisms are also needed. In this regard, BilVideo fills up this gap by providing support for spatio-temporal, semantic, color, shape, and texture video queries. Users may also query news archives by some specific application-dependent predicates supported by the query language of BilVideo to retrieve precise answers to queries.

[2] This example application also appears in [9], which provides a general overview of BilVideo.

A fragment video clip captured from a news broadcast by a national Turkish TV channel was chosen as a basis for the spatio-temporal query examples given in this section. Facts representing the spatio-temporal relations between objects, object-appearance relations, and object trajectories were extracted and inserted into the knowledge-base prior to submitting the queries to the system. The following is a sample set of such relations extracted from a keyframe of this news fragment:

```
south(policevehicle, israeliflag)
overlap(policevehicle, israeliflag)
appear(israeliflag)
appear(policevehicle)
samelevel(israeliflag, policevehicle)
```

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 is used to search for video keyframes, where the only objects appearing are those specified. The predicate *right* is a directional predicate. *vid* is a unique video identifier assigned to the sample news video clip.

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 is used to search for video keyframes, where the objects specified are very close to each other. Here, the closeness is defined semantically as follows: if two objects are close, then their MBRs are not disjoint. This definition is given for this application and may change for others. The system can easily adapt to such changes via external predicates defined in the knowledge-base according to applications' specific needs. The predicate *infrontof* is a third-dimension (3D) predicate.

Query 3: “Retrieve the segments from the sample news clip, where a police vehicle moves toward west, together with an Israeli flag that is above the vehicle and overlaps with 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);
```

In this query, a similarity-based trajectory condition is given, along with directional and topological conditions. The interval operator *and* implies that all conditions are satisfied in the intervals returned to the user as segments, and that for all video frames in such segments, the flag is above the police vehicle and also it overlaps with the vehicle. The keywords `tgap` (time gap) and `repeat` are used for the trajectory condition to ensure that all segments in the clip that satisfy the given conditions, where the police vehicle may stop for at most 1 second at a time during its movement toward west, are returned as an answer to the query.

10. Future Work

The query language of *BilVideo* currently supports a broad range of spatio-temporal queries. However, it is assumed that the camera is fixed, which restricts the trajectory queries in *BilVideo* for some applications, such as sports event analysis systems. As a future work, we plan to investigate the effects of the camera motion in trajectory queries, and augment our trajectory model to account for the general global motions in video, such as panning and tilting. To enhance representation power, model parameters of global motion may be used as a feature vector, and this feature vector in turn may be used to classify the video sequences into static and motion sequences. Such a vector may be obtained using motion analysis algorithms. Motion models can range from a simple translation to complex planar parallax motion. The affine motion model with only six model parameters provides a good compromise between complexity and stability. We will also improve our algorithms developed to process trajectory-based queries to deal with camera motion.

BilVideo system architecture is designed to handle semantic, color, shape, and texture queries, as well, in addition to spatio-temporal queries. In order to provide support for color and shape queries, we propose a new approach to store and compare color and shape features of the salient objects in video keyframes. Our work on semantic modeling and querying of video data is ongoing. Furthermore, *BilVideo* query language is being extended to handle queries that contain not only spatio-temporal but also semantic, color, shape, and texture query conditions. Another important issue we are studying is the optimization of user queries[44]. Moreover, we are also working on enhancing the Web-based visual query interface of *BilVideo* for the semantic, color, shape, and texture video query specification support. We will integrate the interface to *BilVideo*, and make the system accessible on the Internet in future, when we complete our work on semantic, color, shape, and texture queries.

References

1. Adalı, S., K. Candan, S. Chen, K. Erol, and V. Subrahmanian: 1996, 'Advanced Video Information Systems: Data Structures and Query Processing'. *ACM Multimedia Systems* **4**, 172–186.
2. Allen, J.: 1983, 'Maintaining Knowledge About Temporal Intervals'. *Communications of ACM* **26**(11), 832–843.
3. Arslan, U.: 2002, 'A Semantic Data Model and Query Language for Video Databases'. M.S. Thesis, Dept. of Computer Eng., Bilkent University, Ankara, Turkey.
4. Arslan, U., M. Dönderler, E. Şaykol, Ö. Ulusoy, and U. Güdükbay: 2002, 'A Semi-Automatic Semantic Annotation Tool for Video Databases'. In: *Proc. of the Workshop on Multimedia Semantics (SOFSEM 2002).* Milovy, Czech Republic.
5. Babaguchi, N., Y. Kawai, and T. Kitahashi: 2002, 'Event Based Indexing of Broadcasted Sports Video by Intermodel Collaboration'. *IEEE Trans. on Multimedia* **4**(1), 68–75.
6. Chang, S. F., W. Chen, H. J. Meng, H. Sundaram, and D. Zhong: 1997, 'VideoQ: An Automated Content Based Video Search System Using Visual Cues'. In: *Proc. of ACM Multimedia.* Seattle, WA, pp. 313–324.
7. Chen, S. and R. L. Kashyap: 2001, 'A Spatio-Temporal Semantic Model for Multimedia Database Systems and Multimedia Information Systems'. *IEEE Trans. on Knowledge and Data Eng.* **13**(4), 607–622.
8. Convera, 'Screening Room Technical Overview'. http://www.convera.com.
9. Dönderler, M., E. Şaykol, Ö. Ulusoy, and U. Güdükbay: 2003, 'BilVideo: A Video Database Management System'. *IEEE Multimedia* **10**(1), 66–70.
10. Dönderler, M., Ö. Ulusoy, and U. Güdükbay: 2002a, 'Rule-Based Spatio-Temporal Query Processing for Video Databases'. Tech. Report BU-CE-0210, Dept. of Computer Eng., Bilkent University, Available at http://www.cs.bilkent.edu.tr/tech-reports/2002/BU-CE-0210.ps.gz (submitted to a journal).
11. Dönderler, M., Ö. Ulusoy, and U. Güdükbay: 2002b, 'A Rule-Based Video Database System Architecture'. *Information Sciences* **143**(1-4), 13–45.

12. Faloutsos, C., , W. Equitz, M. Flickner, W. Niblack, D. Petkovic, and R. Barber: 1994, 'Efficient and Effective Querying by Image Content'. *Journal of Intelligent Information Systems* **3**(3/4), 231–262.
13. Flickner, M., H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. G. abd J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker: 1995, 'Query by Image and Video Content: The QBIC System'. *IEEE Computer* **28**(9), 23–32.
14. Gauch, S., J. Gauch, and K. M. Pua, 'The VISION Digital Video Library Project'. *to appear in the Encyclopedia of Library and Information Science.*
15. Hacid, M., C. Decleir, and J. Kouloumdjian: 2000, 'A Database Approach for Modeling and Querying Video Data'. *IEEE Trans. on Knowledge and Data Eng.* **12**(5), 729–750.
16. Hauptmann, A. and M. Witbrock: 1997, *Informedia: News-on-Demand Multimedia Information Acquisition and Retrieval*, pp. 215–239. MIT Press.
17. Hjelsvold, R. and R. Midtstraum: 1994, 'Modelling and Querying Video Data'. In: *Proc. of the 20th Int. Conf. on VLDB.* Santiago, Chile, pp. 686–694.
18. Hwang, E. and V. Subrahmanian: 1996, 'Querying Video Libraries'. *Journal of Visual Communication and Image Representation* **7**(1), 44–60.
19. Kuo, T. and A. Chen: 2000, 'Content-Based Query Processing for Video Databases'. *IEEE Trans. on Multimedia* **2**(1), 1–13.
20. Li, J.: 1998, 'Modeling and Querying Multimedia Data'. Tech. Report TR-98-05, Dept. of Computing Science, The University of Alberta, Alberta, Canada.
21. Li, J., I. Goralwalla, M. Özsu, and D. Szafron: 1997a, 'Modeling Video Temporal Relationships in An Object Database Management System'. In: *Proc. of the Int. Symp. on Electronic Images: Multimedia Computing and Networking.* San Jose, CA, pp. 80–91.
22. Li, J. and M. Özsu: 1997, 'STARS: A Spatial Attributes Retrieval System for Images and Videos'. In: *Proc. of the 4th Int. Conf. on Multimedia Modeling.* Singapore, pp. 69–84.
23. Li, J., M. Özsu, and D. Szafron: 1996a, 'Spatial Reasoning Rules in Multimedia Management Systems'. In: *Proc. of Int. Conf. on Multimedia Modeling.* Toulouse, France, pp. 119–133.
24. Li, J., M. Özsu, D. Szafron, and V. Oria: 1997b, 'Multimedia Extensions to Database Query Languages'. Tech. Report TR-97-01, Dept. of Computing Science, The University of Alberta, Alberta, Canada.
25. Li, J. Z., M. T. Özsu, and D. Szafron: 1996b, 'Modeling of Video Spatial Relationships in an Object Database Management System'. In: *Proc. of the Int. Workshop on Multimedia DBMSs.* Blue Mountain Lake, NY, pp. 124–133.
26. Liu, Y. and F.Li: 2002, 'Semantic Extraction and Semantics-Based Annotation and Retrieval for Video Databases'. *Multimedia Tools and App.* **17**, 5–20.
27. Marcus, S. and V. Subrahmanian: 1996, 'Foundations of Multimedia Information Systems'. *Journal of ACM* **43**(3), 474–523.
28. Markus, S. and V. Subrahmanian: 1996, *Multimedia Database Systems: Issues and Research Directions (eds. V.S. Subrahmanian and S. Jajodia)*, Chapt. Towards a Theory of Multimedia Database Systems, pp. 1–35. Springer-Verlag.
29. Martinez, J.: 2001, 'Overview of the MPEG-7 Standard'. ISO/IEC JTC1/SC29/WG11 N4509. http://mpeg.telecomitalialab.com/standards/mpeg-7/mpeg-7.html.
30. Nabil, M., A. Ngu, and J.Shepherd: 2001, 'Modeling and Retrieval of Moving Objects'. *Multimedia Tools and Applications* **13**(1), 35–71.
31. Nascimento, M. and J. Silva: 1998, 'Towards Historical R-trees'. In: *Proc. of ACM Symposium on Applied Computing (ACM-SAC).* pp. 235–240.

32. Oomoto, E. and K. Tanaka: 1993, 'OVID: Design and implementation of a video object database system'. *IEEE Trans. on Knowledge and Data Eng.* **5**(4), 629–643.
33. Özsu, M., P. Iglinski, D. Szafron, S. El-Medani, and M. Junghanns: 1997, 'An Object-Oriented SGML/HYTIME Compliant Multimedia Database Management System'. In: *Proc. of ACM Multimedia.* Seattle, WA, pp. 233–240.
34. Satoh, S., Y. Nakamura, and T. Kanade: 1999, 'Name-it: Naming and Detecting Faces in News Videos'. *IEEE Multimedia* **6**(1), 22–35.
35. Şaykol, E.: 2001, 'Web-Based User Interface for Query Specification in a Video Database System'. M.S. Thesis, Dept. of Computer Eng., Bilkent University, Ankara, Turkey.
36. Şaykol, E., U. Güdükbay, and Ö. Ulusoy: 2001, 'A Semi-Automatic Object Extraction Tool for Querying in Multimedia Databases'. In: S. Adali and S. Tripathi (eds.): *7th Workshop on Multimedia Information Systems MIS'01, Capri, Italy.* pp. 11–20.
37. Şaykol, E., U. Güdükbay, and Ö. Ulusoy: 2002, 'A Histogram-Based Approach for Object-Based Query-by-Shape-and-Color in Multimedia Databases'. Tech. Report BU-CE-0201, Dept. of Computer Eng., Bilkent University, Available at http://www.cs.bilkent.edu.tr/tech-reports/2002/BU-CE-0201.ps.gz (submitted to a journal).
38. Sistla, A. and C. Yu: 1995, 'Similarity Based Retrieval of Pictures using Indices on Spatial Relationships'. In: *Proc. of the 21st VLDB Conf.* Zurich, Switzerland, pp. 619–629.
39. Sistla, A. and C. Yu: 2000, 'Reasoning About Qualitative Spatial Relationships'. *Journal of Automated Reasoning* **25**(4), 291–328.
40. Smith, J. R. and S. F. Chang: 1996, 'VisualSeek: A Fully Automated Content-Based Image Query System'. In: *Proc. of ACM Multimedia.* New York, NY, pp. 87–98.
41. Theodoridis, Y., J. Silva, and M. Nascimento: 1999, 'On the Generation of Spatio-temporal Datasets'. In: *Proc. of the 6th Int. Symp. on Large Spatial Databases (SSD).* Hong Kong, China.
42. Theodoridis, Y., M. Vazirgiannis, and T. Sellis: 1996, 'Spatio-temporal Indexing for Large Multimedia Applications'. In: *Proc. of the 3rd IEEE Conf. on Multimedia Computing and Systems (ICMCS).*
43. Tzouramanis, T., M. Vassilakopoulos, and Y. Manolopoulos: 1998, 'Overlapping Linear Quadtrees: A Spatio-temporal Access Method'. In: *Proc. of the 6th Int. ACM Workshop on Geographical Information Systems.* pp. 1–7.
44. Ünel, G., M. Dönderler, Ö. Ulusoy, and U. Güdükbay, 'An Efficient Query Optimization Strategy for Spatio-Temporal Queries in Video Databases'. *to appear in Journal of Systems and Software.*
45. Veltkamp, R. and M. Tanase: 2000, 'Content-Based Retrieval System: A Survey'. Tech. Report UU-CS-2000-34, Utrect University, The Netherlands.
46. Virage, 'VideoLogger'. http://www.virage.com.
47. Woods, W.: 1970, 'Transition Network Grammars for Natural Language Analysis'. *Communications of the ACM* **13**, 591–602.
48. Xu, X., J. Han, and W. Lu: 1990, 'RT-tree: An Improved R-tree Index Structure for Spatio-temporal Databases'. In: *Proc. of the 4th Int. Symp. on Spatial Data Handling (SDH).* pp. 1040–1049.
49. Yoshitaka, A. and T. Ichikawa: 1999, 'A Survey of Content-Based Retrieval for Multimedia Databases'. *IEEE Trans. on Knowledge and Data Eng.* **11**(1), 81–93.