A MPEG-7 Compatible Video Retrieval System with Integrated Support for Complex Multimodal Queries

Muhammet Başştan, Hayati Çam, Uğur Güdükbay, Özgür Ulusoy
Bilkent University
Department of Computer Engineering
Ankara, Turkey
(bastan,hayati,gudukbay,oulusoy)@cs.bilkent.edu.tr

April 28, 2009

Abstract

We present *BilVideo-7*, an MPEG-7 compatible, distributed video database management system that supports complex multimodal queries in an integrated way. An MPEG-7 profile is developed to represent the videos by decomposing them into *Shots, Keyframes, Still Regions* and *Moving Regions*. The MPEG-7 compatible XML representations of videos according to this profile are obtained by the MPEG-7 compatible video feature extraction and annotation tool of *BilVideo-7*, and stored in a native XML database. Users can formulate *text-based semantic, color, texture, shape, location, motion* and *spatio-temporal* queries on an intuitive, easy-to-use *Visual Query Interface*, whose *Composite Query Interface* can be used to specify very complex queries containing any type and number of video segments with their descriptors. The multi-threaded *Query Processing Server* parses queries into subqueries and executes each subquery in a separate thread. Then, it fuses subquery results in a bottom-up manner to obtain the final query result. The whole system is unique in that it provides very powerful querying capabilities with a wide range of descriptors and multimodal query processing in an MPEG-7 compatible interoperable environment. We present sample queries to demonstrate the capabilities of the system.

1 Introduction

Early prototype multimedia database management systems used the query-by-example (QBE) paradigm to respond to user queries. Users needed to formulate their queries by providing examples or sketches. The Query-by-keyword (QBK) paradigm, on the other hand, has emerged due to the desire to search multimedia content in terms of semantic concepts using keywords or sentences rather than low-level multimedia descriptors. This is because it is much easier to formulate some queries by keywords, which is also the way text retrieval systems work. However, some queries are still easier to formulate by examples or sketches (e.g., the trajectory of a moving object). Moreover, there is the so-called "semantic gap" problem, the disparity between low-level representation and high-level semantics, which makes it very difficult to build multimedia systems capable of supporting keyword-based semantic queries effectively with an acceptable number of semantic concepts. The consequence is the need to support both query paradigms in an integrated way.

Another important issue to be considered in today's multimedia management systems is interoperability. This is especially crucial for distributed architectures if the system is to be used by multiple heterogeneous clients. Therefore, MPEG-7 [1, 2] standard as the multimedia content description interface can be employed to address this issue.

The design of a retrieval system is directly affected by the type of queries to be supported. Types of descriptors and the granularity of the representation determine the system's performance in terms of speed and accuracy. As the level of detail in the representation increases more detailed queries can be answered by the system. However, both the database size and system response time increase. Therefore, the system should be designed according to the type of queries to be supported, and representation granularity should be selected accordingly. Below, we give some example video query types that might be attractive for most users, but which also are not supported by the existing systems all together in an MPEG-7 compatible framework.

- *Content-based queries by examples.* The user may specify an image, an image region or a video segment and the system returns video segments similar to the input query.
- *Text-based semantic queries.* Queries may be specified by a set of keywords corresponding to high-level semantic concepts and relations between them.
- *Spatio-temporal queries.* Queries related to spatial and temporal locations of objects and video segments within the video.
- *Composite queries.* These queries may contain any combination of other simple queries.

The user composes the query (hence the name 'composite' query) by putting together image/video segments and specifying their properties, and then asks the system to retrieve similar ones from the database. This type of queries is especially desirable to formulate very complex queries easily.

We developed *BilVideo-7* as a powerful MPEG-7 compatible, distributed video database system to support such multimodal queries in an integrated way. We designed an MPEG-7 profile for video representation which enables detailed queries on videos, and used our MPEG-7 compatible video feature extraction and annotation tool to obtain the MPEG-7 compatible video representations according to this profile. The *Visual Query Interface* of *BilVideo-7* is an easy-to-use and powerful query interface to formulate complex multimodal queries easily, with support for a comprehensive set of MPEG-7 descriptors. Queries are processed on the multi-threaded *Query Processing Server* with a multimodal query processing and subquery result fusion architecture, which is also suitable for parallelization.

The name *BilVideo-7* is reminiscent of BilVideo [3], which was a prototype video database system that supported keyword-based spatio-temporal queries using a knowledge-base and a Prolog inference engine. *BilVideo-7* is a new system, developed from scratch and is different from BilVideo in terms of MPEG-7 compatibility (BilVideo was not MPEG-7 compatible), video data model, multimodal query processing, query formulation capabilities and wide range of MPEG-7 descriptors supported.

The rest of the paper is organized as follows. Section 2 gives an overview of MPEG-7 and reviews some of the available MPEG-7 compatible systems. Section 3 describes how video is represented in *BilVideo-7*. Section 4 presents the distributed, client-server architecture of *BilVideo-7*. Section 5 explains how queries are processed by the *Query Processing Server*. Section 6 gives implementation details for the whole system. Section 7 presents example queries and results returned by the system. Finally, Section 8 concludes with a discussion and possible future extensions.

2 Related Work

2.1 MPEG-7 Standard

MPEG-7 is an ISO/IEC standard developed by MPEG (Moving Picture Experts Group), the committee that also developed the standards MPEG-1, MPEG-2 and MPEG-4 [1, 2]. Different

from the previous MPEG standards, MPEG-7 is designed to describe the content of multimedia. It is formally called "Multimedia Content Description Interface."

MPEG-7 offers a comprehensive set of audiovisual description tools in the form of Descriptors (D) and Description Schemes (DS) that describe the multimedia data, forming a common basis for applications and enabling efficient and effective access to the data. The Description Definition Language (DDL) is based on W3C XML with some MPEG-7 specific extensions, such as vectors and matrices. Therefore, MPEG-7 documents are XML documents that conform to particular MPEG-7 schemas for describing multimedia content. Descriptors describe features, attributes or groups of attributes of multimedia content. Description Schemes describe entities or relationships pertaining to multimedia content. They specify the structure and semantics of their components, which may be Description Schemes, Descriptors or datatypes.

The eXperimentation Model (XM) software is the framework for all the reference code of the MPEG-7 standard. It implements the normative components of MPEG-7. MPEG-7 standardizes multimedia content description but it does not specify how the description is produced. It is up to the developers of MPEG-7 compatible applications how the descriptors are extracted from the multimedia, provided that the output conforms to the standard. MPEG-7 Visual Description Tools consist of basic structures and Descriptors that cover the following basic visual features for multimedia content: *color, texture, shape, motion, localization,* and *face recognition.*

Color Descriptors: Color Structure Descriptor (CSD) represents an image by both color distribution and spatial structure of color. *Scalable Color Descriptor (SCD)* is a Haar transform based encoding of a color histogram in HSV color space. *Dominant Color Descriptor (DCD)* specifies up to eight representative (dominant) colors in an image or image region. *Color Layout Descriptor (CLD)* is a compact and resolution-invariant color descriptor that efficiently represents spatial distribution of colors. *Group-of-Frame or Group-of-Picture Descriptor (GoF/GoP)* is used for the color-based features of multiple images or multiple frames in a video segment. It is an alternative to single keyframe based representation of video segments. The descriptor is obtained by aggregating the histograms of multiple images or frames and representing the final histogram with Scalable Color Descriptor.

Texture Descriptors: Edge Histogram Descriptor (EHD) specifies the spatial distribution of edges in an image. *Homogeneous Texture Descriptor (HTD)* characterizes the texture of a region using mean energy and energy deviation from a set of frequency channels, which are modeled with Gabor functions. *Texture Browsing Descriptor (TBD)* characterizes textures perceptually in terms of regularity, coarseness and directionality.

Shape Descriptors: Contour Shape Descriptor (CShD) describes the closed contour of a 2-D

region based on a Curvature Scale Space (CSS) representation of the contour. *Region Shape Descriptor (RSD)* is based on the Angular Radial Transform (ART) to describe shapes of regions composed of connected single or multiple regions, or regions with holes. It considers all pixels constituting the shape, including both boundary and interior pixels.

Motion Descriptors: Motion Activity (MAc) captures the notion of 'intensity of action' or 'pace of action' in a video sequence. *Camera Motion* describes all camera operations like translation, rotation, focal length change. *Motion Trajectory (MTr)* is the spatio-temporal localization of one of the representative points (e.g., center of mass) of a moving region. *Parametric Motion* characterizes the motion of an arbitrarily shaped region over time by one of the classical parametric motion models (translation, rotation, scaling, affine, perspective, quadratic) [4].

Localization Descriptors: Region Locator specifies locations of regions within images using a box or polygon. *Spatio-temporal Locator* specifies locations of video segments within a video sequence spatio-temporally.

Face Recognition Descriptor is a Principal Component Analysis (PCA) based descriptor that represents the projection of a face onto a set of 48 basis vectors that span the space of all possible face vectors.

In MPEG-7, the semantic content of multimedia (e.g., objects, events, concepts) can be described by text annotation (free text, keyword, structured and dependency structure) and/or semantic entity and semantic relation tools. Free text annotations describe the content using unstructured natural language text (e.g., Barack Obama visits Turkey in April). Such annotations are easy for humans to understand but difficult for computers to process. Keyword annotations use a set of keywords (e.g., Barack Obama, visit, Turkey, April) and easier to process by computers. Structured annotations strike a balance between simplicity (in terms of processing) and expressiveness. They consist of elements each answering one of the following questions: who, what object, what action, where, when, why and how (e.g., who: Barack Obama, what action: visit, where: Turkey, when: April). Dependency structure represents the linguistic structure of an annotation based on a linguistic theory called dependency grammar that explains a sentence's grammatical structure in terms of dependencies between its elements.

More detailed descriptions about semantic entities such as objects, events, concepts, places and times can be stored using semantic entity tools. The semantic relation tools describe the semantic relations between semantic entities using the normative semantic relations standardized by MPEG-7 (e.g., agent, agentOf, patient, patientOf, result, resultOf, similar, opposite, user, userOf, location, locationOf, time, timeOf) or by non-normative relations [1].

The semantic tools of MPEG-7 provide methods to create very brief or very extensive semantic

descriptions of multimedia content. The choice of which description tool is to be used in a system is affected by the type of semantic queries to be supported and by the annotation tool to be used. Some of the descriptions can be obtained automatically while most of them require manual labeling. Automatic speech recognition (ASR) text can be used as free text annotations to describe video segments. Keyword and structured annotations can be obtained automatically to some extent using state-of-the-art auto-annotation techniques. Description of semantic entities and relations between them cannot be obtained automatically with the current-state-of-the-art, therefore, considerable amount of manual work is needed for this kind of semantic annotation.

2.2 MPEG-7 Compatible Systems

Although MPEG-7 was announced in 2001, only a few MPEG-7 compatible multimedia systems have been developed so far. In this section, we review some of the existing systems for image and video.

The comprehensiveness and flexibility of MPEG-7 allow its usage in a broad range of applications, but also increase its complexity and adversely affects interoperability. To overcome this problem, profiling has been proposed. An MPEG-7 profile is a subset of tools defined in MPEG-7, providing a particular set of functionalities for one or more classes of applications. In [5], an MPEG-7 profile is proposed for detailed description of audiovisual content that can be used in a broad range of applications.

An MPEG-7 compatible Database System extension to Oracle DBMS is proposed in *MPEG-7 MMDB* [6]. The resulting system is demonstrated by audio and image retrieval applications. In [7], algorithms for the automatic generation of three MPEG-7 DSs are proposed: (1) *Video Table of Contents DS*, for active video browsing, (2) *Summary DS*, to enable the direct use of meta data annotation of the producer, and (3) *Still Image DS*, to allow interactive content-based image retrieval. Tseng *et al.* [8] address the issues associated with designing a video personalization and summarization system in heterogeneous usage environments utilizing MPEG-7 and MPEG-21. In [9], an MPEG-7 compatible description of video sequences for scalable transmission and reconstruction is presented. In [10], a method for automatically extracting motion trajectories from video sequences and generation of MPEG-7 compatible XML descriptions is presented within the context of sports videos.

IBM's *VideoAnnEx Annotation Tool* [11] enables users to annotate video sequences with MPEG-7 metadata. Each shot in the video sequence is represented by a single keyframe and can be annotated with static scene descriptions, key object descriptions, event descriptions and other

custom lexicon sets that may be provided by the user. The tool is limited to concept annotation and cannot extract low-level MPEG-7 descriptors from the video. The *M-OntoMat-Annotizer* [12] software tool aims at linking low-level MPEG-7 visual descriptions to conventional semantic web ontologies and annotations. The visual descriptors are expressed in *Resource Description Framework (RDF)*. The IFINDER system [13] is developed to produce limited MPEG-7 representation from audio and video by speech processing, keyframe extraction and face detection. *COSMOS-7* [14] defines its own video content model and converts the representation to MPEG-7 for MPEG-7 conformance. It models content semantics (object names, events, etc.), spatial and temporal relations between objects using what are called m-frames (multimedia frames). *ERIC7* [15] is a software test-bed that implements Content-Based Image Retrieval (CBIR) using image-based MPEG-7 color, texture and shape descriptors. *Caliph & Emir* [16] are MPEG-7 based Java prototypes for digital photo and image annotation and retrieval, supporting graph-like annotations for semantic meta data and content-based image retrieval using MPEG-7 descriptors.

Available MPEG-7 compatible systems described above have two major problems. (1) Most of them use a coarse image or video representation, extracting low-level descriptors from whole images or video frames and annotating them, but ignoring region-level descriptors. This coarse representation in turn limits the range of queries the users can perform on these systems. (2) The user cannot perform complex multimodal queries on these systems by combining several descriptors in different modalities. *BilVideo-7* addresses these two major problems by adopting an MPEG-7 profile with a more detailed video representation and using a multimodal query processing and bottom-up subquery result fusion architecture to support complex multimodal queries with a comprehensive set of MPEG-7 descriptors.

3 MPEG-7 Compatible Representation of Video

The first step in constructing an MPEG-7 compatible video management system is to decide what kind of queries will be supported and then to design an MPEG-7 profile accordingly. The representation of video is crucial since it directly affects the system's performance. There is a trade-off between the accuracy of representation and the speed of access: more detailed representation will enable more detailed queries but will also result in longer response time during retrieval. Keeping these factors in mind, we decided to use the MPEG-7 profile shown in Figure 1. This is adapted from the detailed audiovisual profile proposed in [5] to represent image, audio and video collections. Our profile corresponds to the video representation portion of the detailed audiovisual profile, with our own interpretation of what to represent with

Keyframes, Still and *Moving Regions* so that our system can support the wide range of queries it is designed for. First, audio and visual data are separated using Media Source Decomposition. Then, visual content is hierarchically decomposed into smaller structural and semantic units. An example of video decomposition according to this profile is shown in Figure 2. Please see Section 4.1 for the details of how the MPEG-7 representation of a video is obtained using our MPEG-7 compatible video feature extraction and annotation tool.

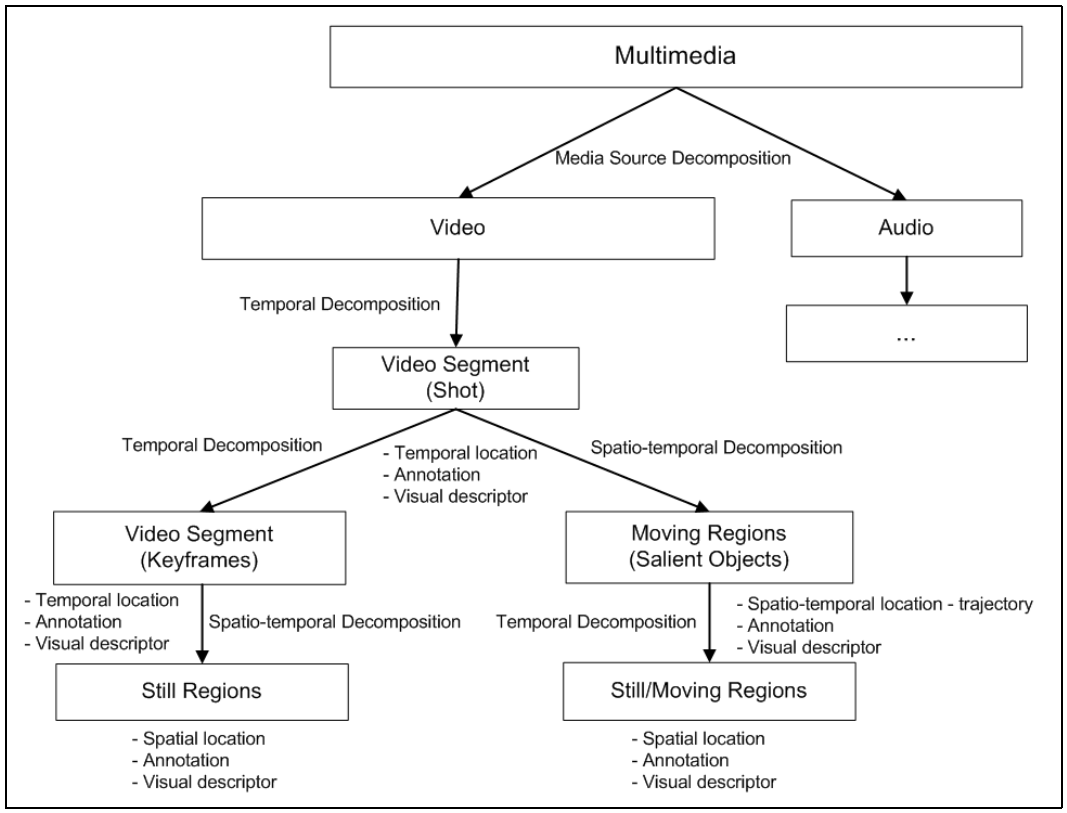

Figure 1: MPEG-7 Profile used in *BilVideo-7*.

Temporal Decomposition of video into shots. Video is partitioned into non-overlapping video segments called shots, each having a temporal location (start time and duration), semantic annotation to describe the objects and/or events in the shot with free text, keyword and structured annotation and visual descriptor (e.g., motion, GoF/GoP descriptors). A *shot* is a sequence of frames captured by a single camera in a single continuous action. *Shot boundaries* are the transitions between shots. They can be abrupt (cut) or gradual (fade, dissolve, wipe, morph).

Temporal Decomposition of shots. The background content of the shots does not change much, especially if the camera is not moving. This static content can be represented with a single keyframe or a few keyframes if there is a considerable amount of change in the visual appearance (e.g., in case of camera motion). Therefore, each shot is decomposed into smaller, more ho-

mogeneous video segments (keysegments) which are represented by keyframes. Each keyframe is described by a temporal location, semantic annotation and a set of visual descriptors. The visual descriptors are extracted from the frame as a whole.

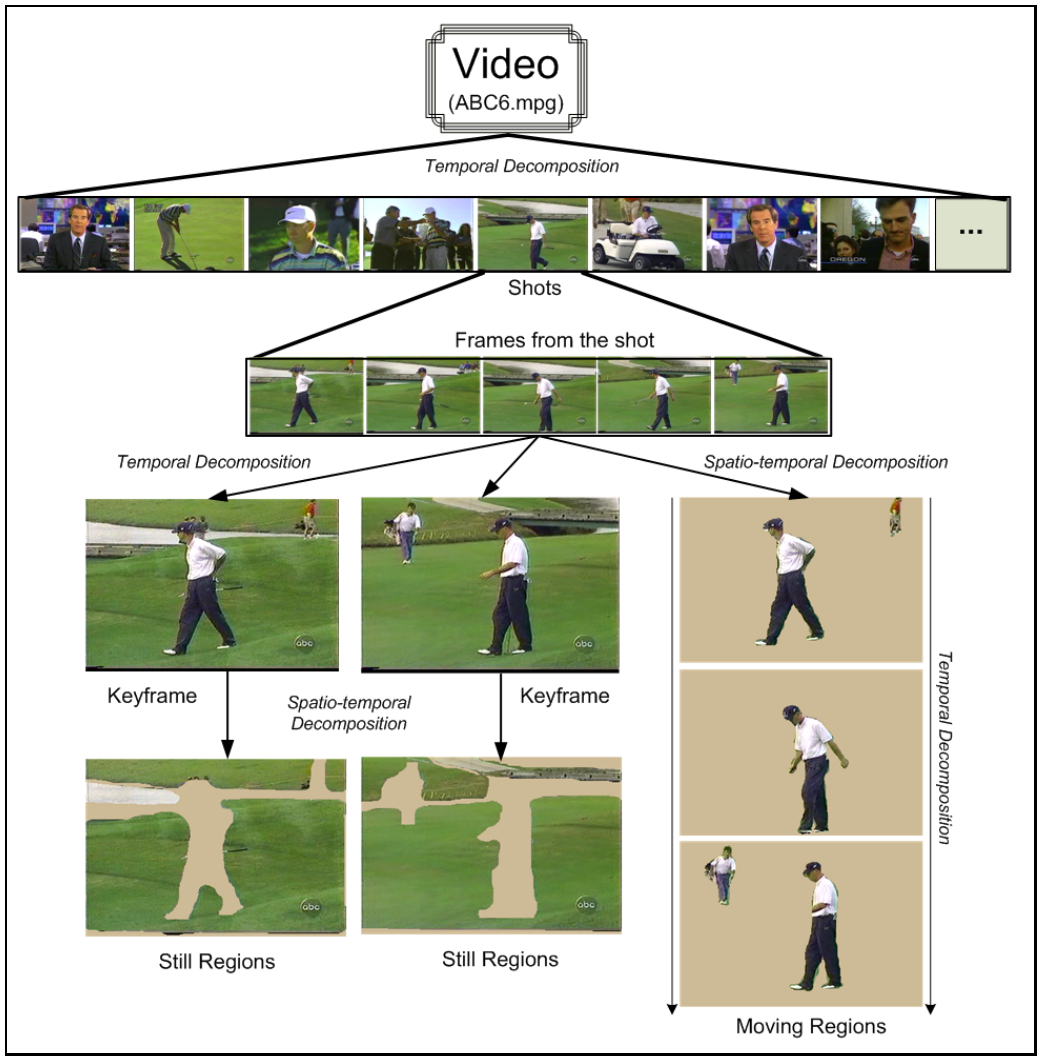

Figure 2: MPEG-7 decomposition of a video according to the MPEG-7 profile used in *BilVideo-7* (Figure 1). Low-level color, texture and shape descriptors of the *Still* and *Moving Regions* are extracted from the selected arbitrarily shaped regions, but the locations of the regions are represented by their MBRs.

Each keyframe is also decomposed into a set of non-overlapping *Still Regions* (*Spatio-temporal Decomposition*) to be able to keep more detailed region-based information in the form of spatial location by the Minimum Bounding Rectangle (MBR) of the region, semantic annotation and region-based visual descriptors.

Spatio-temporal Decomposition of shots into Moving Regions. Each shot is also decomposed into a set of *Moving Regions* to represent the dynamic and more important content of the shots

corresponding to the salient objects. Hence, more information can be stored for *Moving Regions* to enable more detailed queries about salient objects. The term "*Moving Regions*", as used in MPEG-7, is somewhat confusing in this context. The objects do not need to be moving to be qualified as *Moving Regions*; they should only be salient. Hence, a salient stationary object in a shot is represented with a *Moving Region* [17]. *Faces* are also represented with *Moving Regions*, having an additional visual descriptor: Face Recognition Descriptor.

Since the position, shape, motion and visual appearance of the salient objects may change throughout the shot, descriptors sampled at appropriate time points should be stored. The trajectory of an object is represented by the *Motion Trajectory* descriptor. The MBRs and visual descriptors of the region throughout the shot are stored by temporally decomposing the object into *Still Regions*. A new sample is taken at any time point (key time point) at which there is a certain amount of change in the descriptor values compared to the previous time point.

Video Segment: From here on, we refer to the building blocks of a video, *Shots, Keyframes, Still Regions* and *Moving Regions*, as *video segments*.

4 System Architecture

BilVideo-7 has a distributed, client-server architecture (Figure 3). Users formulate queries on *BilVideo-7* clients, which communicate with the *BilVideo-7 Query Processing Server* using an XML-based query language over TCP/IP. The *Query Processing Server* parses queries into subqueries, retrieves the required data from the XML database using XQuery [18] for each subquery, executes subqueries, fuses results of subqueries and sends query results back to the clients.

4.1 MPEG-7 Compatible Feature Extraction and Annotation of Videos

MPEG-7 representations of videos are obtained using the MPEG-7 compatible video feature extraction and annotation tool (Figure 4). Currently, the tool is operated manually to obtain the MPEG-7 representations according to the MPEG-7 profile given in Figure 1. Videos, along with shot boundary information, are loaded and then processed on a shot-by-shot basis. Users can manually select *Keyframes*, *Still Regions* and *Moving Regions* and then annotate the *Video, Shots, Keyframes, Still Regions* and *Moving Regions* with free text, keyword and structured annotations. The MPEG-7 visual descriptors (color, texture, shape, motion, localization) for

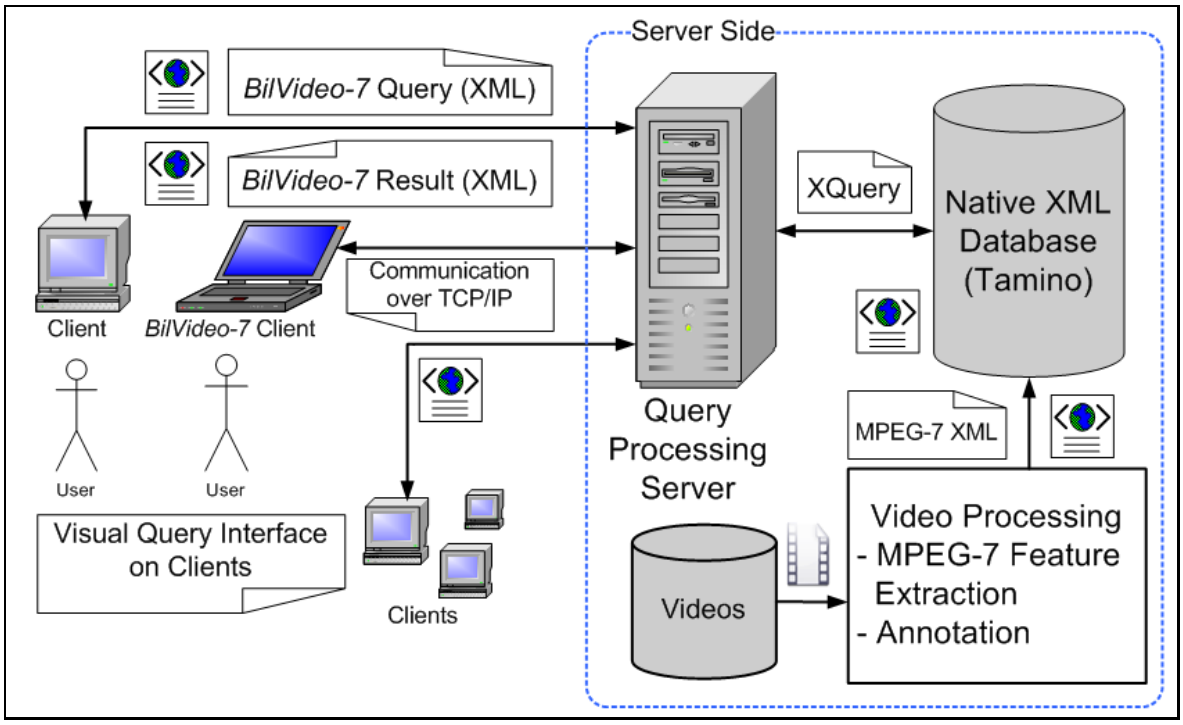

Figure 3: Distributed, client-server architecture of *BilVideo-7*.

the selected video segments are computed by the tool, using an MPEG-7 feature extraction library adapted from MPEG-7 XM Reference Software [19]. The user can select the set of visual descriptors to describe each type of video segment (e.g., any subset of CSD, SCD, DCD, CLD, EHD, HTD to describe the keyframes). The semantic content is described by text annotations (free text, keyword and structured annotation), which strike a good balance between simplicity (in terms of manual annotation effort and processing during querying) and expressiveness. The output is saved as an MPEG-7 compatible XML file to be stored in the XML database. The tool is still being improved to handle audio, video and image data, and will become a full-fledged MPEG-7 compatible multimedia feature extraction and annotation tool with as much automatic processing capabilities as possible so that manual processing time, human subjectivity and error-proneness can be reduced.

4.2 Visual Query Interface

Users formulate their queries on *BilVideo-7* client's graphical user interface, *Visual Query Interface* (Figure 5). These queries are converted into *BilVideo-7Query* format in XML and sent to the *BilVideo-7 Query Processing Server* over TCP/IP. The query results are displayed to the user as a list of video segment intervals in ranked order, from where the segments can be selected and viewed.

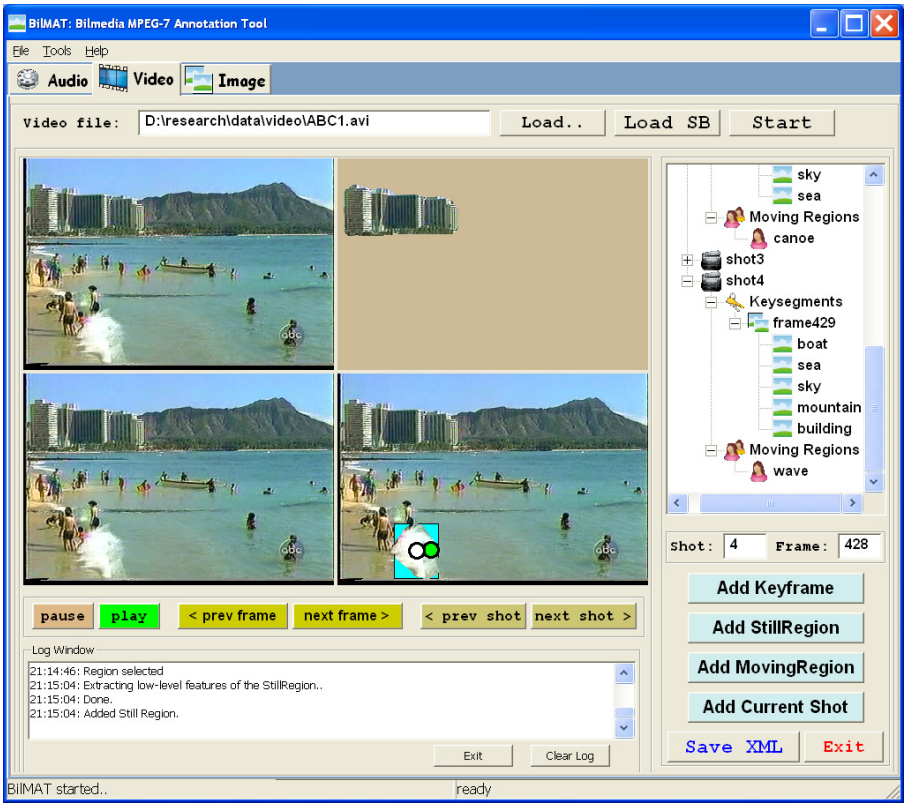

Figure 4: MPEG-7 compatible video feature extraction and annotation tool, which is used to obtain the MPEG-7 compatible representations of videos according to the MPEG-7 profile in Figure 1. In the graphical user interface, the current video frame is shown at the top left, latest processed frame is at the bottom left, latest selected region is at the top right, and selected Moving Regions along with their trajectories are at the bottom right. Selected video segments (*Shots, Keyframes, Still Regions, Moving Regions*) are shown on the right in a hierarchical tree view reflecting the structure of the video.

The Visual Query Interface of *BilVideo-7* client provides an intuitive, easy-to-use query formulation interface and consists of several tabs, each for a different type of query with a comprehensive set of descriptors and query options. As shown in Figure 5 (Spatial Query Interface), the query formulation tabs are on the left, the query result list is displayed at the top right, the query results can be viewed on the media player at the bottom right, and messages are displayed at the bottom left. The user can select the media type, return type and maximum number of results to be returned from the toolbar at the top. The return type of a query can be one of the following: *Video, Supershot, Shot, Subshot*. If *Video* is selected as the return type, whole videos matching the query are returned; if *Shot* is selected, the query result list consists of *Shots*. *Subshots* are video segments contained in the *Shots*, such as *Keysegments* and *Moving Regions* and *Supershots* are consecutive *Shots* satisfying the query. In the following, all the *BilVideo-7* client query formulation interfaces are described briefly.

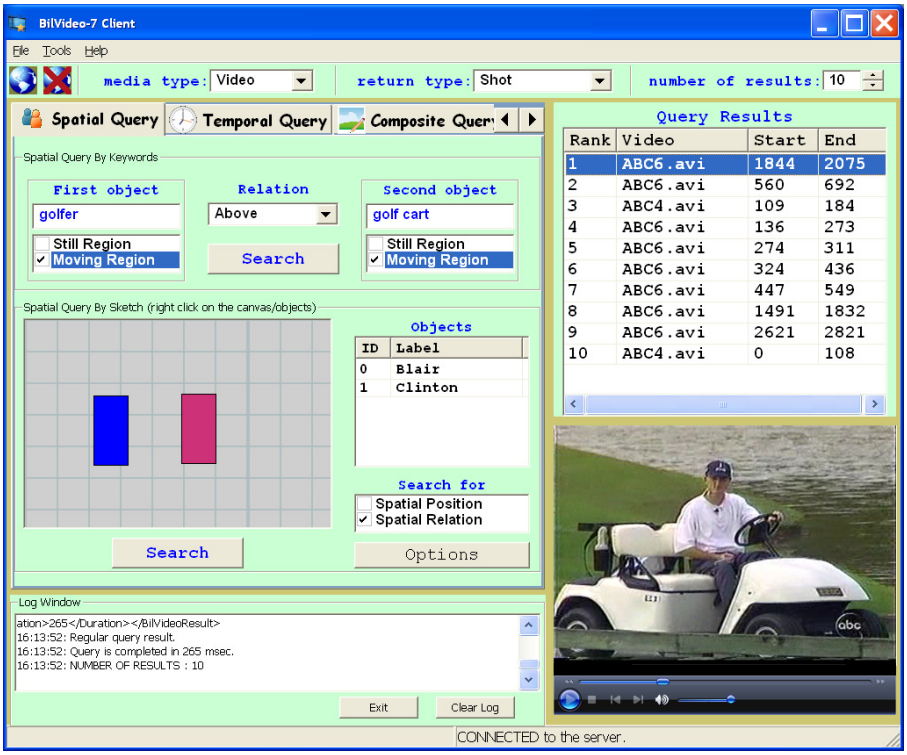

Figure 5: *BilVideo-7* client *Visual Query Interface*. This screenshot shows the *Spatial Query Interface*. For the other query interfaces please see [20].

Video Table of Contents (VideoToC) is a useful facility to let the user browse through the video collection in the database, to see the contents of each video in a hierarchical tree view reflecting the structure of the MPEG-7 representation of the video in XML format and to see the high-level semantic concepts in the collection and in each video separately (Figure 6). The user can browse through each video in the collection and see all the *Shots, Keyframes, Still Regions* and *Moving Regions* as well as the semantic concepts they are annotated with and their temporal location (Media Time) in the video.

Textual Query Interface enables the user to formulate high-level semantic queries quickly by entering keywords and specifying the type of video segment (*Shot, Keyframe, Still Region, Moving Region*) and annotation (free text, keyword, structured) to search in.

Color, Texture, Shape Query Interface is used for querying video segments by MPEG-7 color, texture and shape descriptors. The input media can be a video segment, a whole image or an image region. The descriptors need to be extracted from the selected input media. Instead of uploading the input media to the server and extracting the descriptors there, we have chosen to extract the descriptors on the client, form the XML-based query expression containing the descriptors and send the query to the server. Therefore, the MPEG-7 feature extraction module

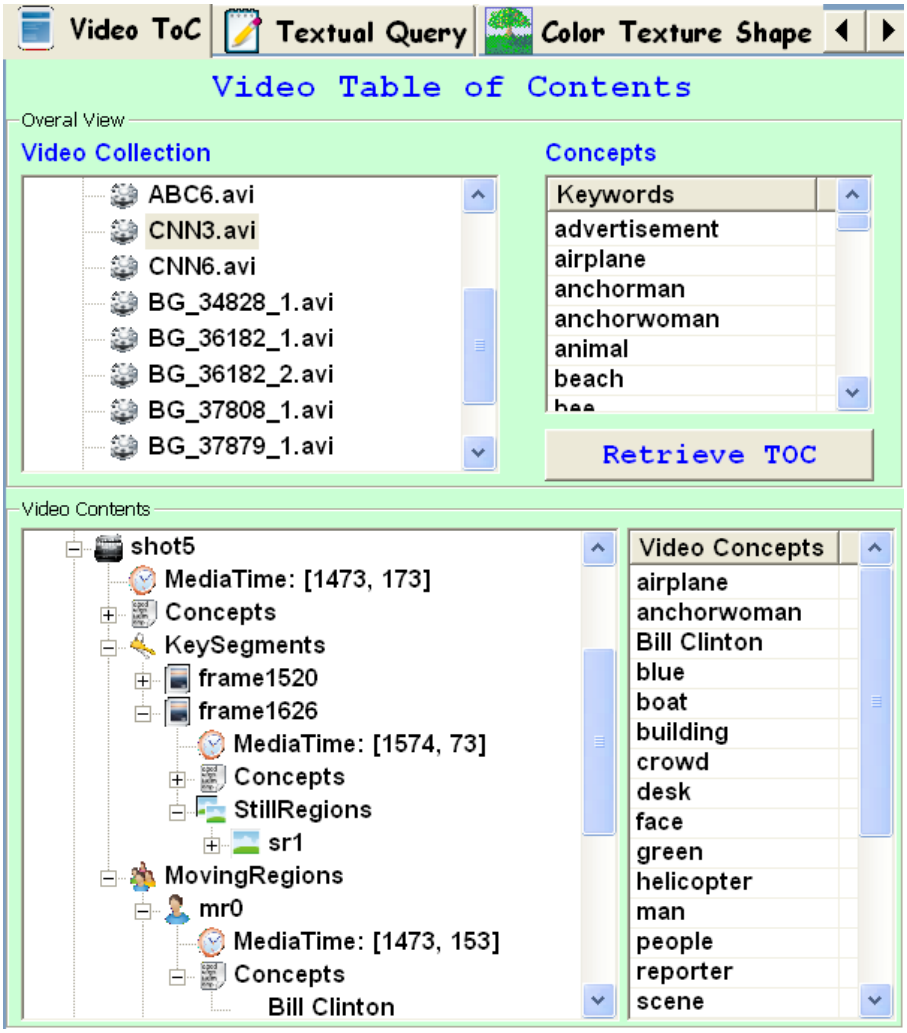

Figure 6: Video Table of Contents (VideoToC) interface of a *BilVideo-7* client. The whole video collection and concepts are shown at the top, details for each video are shown at the bottom.

is integrated with *BilVideo-7* clients. The user also specifies which type of video segments to search in, and also other query options such as weights and thresholds for each type of descriptor. Sections 5.2 and 5.3 describe in detail the query processing, distance computation between the descriptors and fusion of subquery results.

Motion Query Interface is for the formulation of motion activity and motion trajectory queries. Trajectory points are entered with the mouse. The user can optionally specify keywords for the *Moving Region* for which the trajectory query will be performed.

Spatial Query Interface enables the user to formulate spatial queries for *Still* and *Moving Regions* using either keywords and a set of predefined spatial relations (left, right, above, below, east, west, etc.) or by sketching the minimum bounding rectangles (MBR) of objects

with the mouse, and if desired, giving labels to them. Since spatial queries are valid for *Still* and *Moving Regions*, region types (*Still/Moving Region*) should also be selected along with other query options. It is possible to query objects based on location, spatial relations or both.

Temporal Query Interface is very similar to spatial query interface; this time, the user specifies temporal relations between video segments (*Shots, Keyframes, Still Regions, Moving Regions*) either by selecting from a predefined list (before, after, during, etc.) or by sketching the temporal positions of the segments with the mouse.

Composite Query Interface is for *composing* a query using any combination of textual, color, texture, shape, motion, spatial and temporal queries with any number and type of video segments. This is the most powerful query interface and it enables the user to formulate very complex queries easily. The query is composed by putting together *Shots, Keyframes, Still Regions* and *Moving Regions* and specifying their properties as text-based semantic annotations, visual descriptors, location, spatial and temporal relations. Using this interface, the user can describe a video segment or a scene and ask the system to retrieve similar video segments.

XQuery Interface is more suited to experienced users who can write XQueries to search in the database. This is the most flexible interface and the user can specify a wide range of queries.

5 Query Processing

Query processing is done on the *Query Processing Server*, which is a multi-threaded server side component that listens to a configured TCP port and accepts incoming clients. Clients send queries in XML-based *BilVideo-7Query* format [20]. When the query execution is completed, query results are sent back to the originating client in XML-based *BilVideo-7Result* format, which contains a list of video segments (video name, start time, end time) in ranked order.

5.1 Storing MPEG-7 Compatible XML Representations

The output of the MPEG-7 compatible video feature extraction and annotation tool is an XML file for each video. Conceptually, there are two different ways to store XML documents in a database. The first way is to map the data model of the XML document to a database model and convert XML data according to this mapping. The second way is to map the XML model into a fixed set of persistent structures (a set of tables for *elements, attributes, text,* etc.) designed to hold any XML document. Databases that support the former method are called *XML-enabled* databases, whereas databases that support the latter are called *native*

XML databases (NXD). XML-enabled databases map instances of the XML data model to instances of their own data model (relational, hierarchical, etc). Native XML databases use the XML data model directly [21]. As a result, it is more convenient and natural to use a native XML database to store the MPEG-7 descriptions. *BilVideo-7* uses a native XML database (Tamino [22]) along with the standard XQuery to execute queries in the database.

5.2 Multi-threaded Query Execution

Each incoming query is parsed into subqueries and executed in a multi-threaded fashion, with one thread for each type of subquery, as shown in Figure 7. Queries with the same subquery type are accumulated in a queue and executed on a first-in-first-out (FIFO) basis. For example, subqueries for color descriptors (CSD, SCD, DCD, etc.) are added to the end of queue of *Color Query Executor* thread and executed in this order. One XQuery is formed and executed for each video segment (*Shot, Keyframe, Still Region, Moving Region*) and for each type of subquery. The XML database returns the XQuery results (which are parsed to extract the actual data – the descriptors) in XML format. Textual queries are the easiest to execute since the XML database can handle textual queries and no further processing is needed for the similarity computation. However, the database cannot handle the similarity queries for low-level descriptors. That is, the distance (or similarity) between a query descriptor and a descriptor in the database cannot be computed by the database. Therefore, the corresponding query execution thread retrieves the relevant descriptors from the database for the video segment in the subquery (e.g., Color Structure descriptors for *Keyframes*) and computes the distance between the descriptors of the query and the database.

The distance measures suggested by MPEG-7 authors for each descriptor are implemented in MPEG-7 XM Reference Software [19] but they are not normative. The evaluation of distance measures for a set of MPEG-7 descriptors, presented in [23], shows that although there are better distance measures (e.g., pattern difference, Meehl index), the recommended distance measures of MPEG-7 are among the best. Therefore, we adapted the distance measures from the XM Reference Software implementation. In the future, other distance measures will also be investigated.

Distance Metrics. In the following, we briefly describe the distance metrics adapted from MPEG-7 XM software (for more explanations and details please see [1, 19]). Q refers to a descriptor in the query, D to a descriptor in the database and d is the computed distance between the descriptors.

L_1-norm is used to compute the distance between Color Structure, Scalable Color, GoF/GoP,

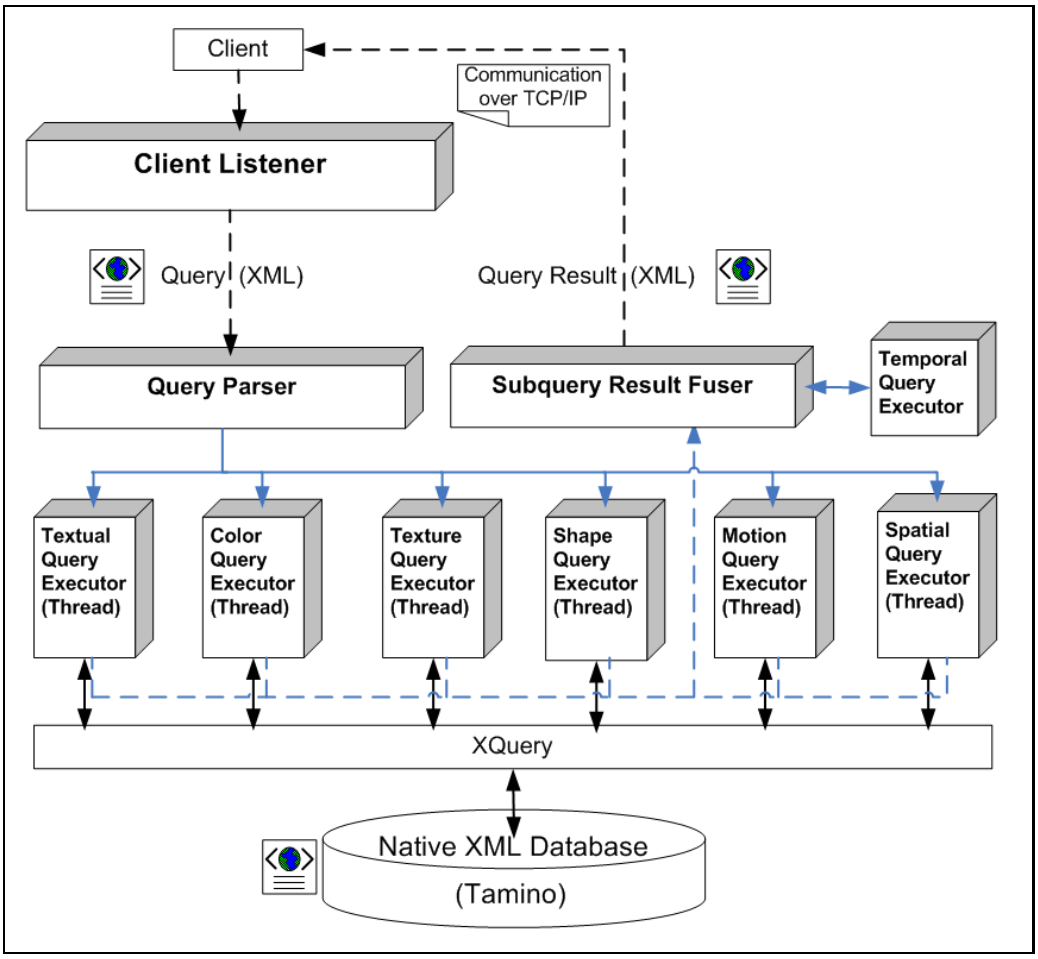

Figure 7: The framework of the *Query Processing Server*. XML-based queries coming from the clients are parsed into subqueries and each type of subquery is executed in a separate thread (e.g., color subqueries – CSD, SCD,. . . – in Color Query Executor thread, texture subqueries – EHD, HTD – in Texture Query Executor), each having a queue of subqueries which are executed on a first-in-first-out (FIFO) basis. Subquery results are fused in a bottom-up manner (Figure 9) and the final result is returned to the client.

Region Shape descriptors.

$$d_{L_1}(Q, D) = \sum_i |Q(i) - D(i)|$$

The distance between two Color Layout descriptors, $Q = \{QY, QCr, QCb\}$ and $D = \{DY, DCr, DCb\}$, is computed by

$$d(Q,D) = \sqrt{\sum_i w_{yi}(QY_i - DY_i)^2} + \sqrt{\sum_i w_{bi}(QCb_i - DCb_i)^2} + \sqrt{\sum_i w_{ri}(QCr_i - DCr_i)^2}$$

where the subscript i represents the zigzag-scanning order of the coefficients and the weights ($w_{yi},\ w_{bi},\ w_{ri}$) are used to give more importance to the lower frequency components of the descriptor.

The distance between two Dominant Color descriptors Q and D (without using the spatial coherency and optional color variance) is computed by

$$\begin{aligned} Q =& \{(c_{qi}, p_{qi}, v_{qi}), s_q\}, i = 1, 2, \ldots, N_q \\ D =& \{(c_{dj}, p_{dj}, v_{dj}), s_d\}, j = 1, 2, \ldots, N_d \\ d^2(Q,D) =& \sum_{i=1}^{N_q} p_{qi}^2 + \sum_{j=1}^{N_d} p_{dj}^2 - \sum_{i=1}^{N_q}\sum_{j=1}^{N_d} 2a_{qi,dj}p_{qi}p_{dj} \end{aligned}$$

where $a_{q,d}$ is the similarity coefficient between two colors c_q and c_d,

$$a_{q,d} = \begin{cases} 1 - d(c_q, c_d)/d_{max}, & d(c_q, c_d) \le T_c \\ 0, & d(c_q, c_d) > T_c \end{cases}$$

where $d(c_q, c_d) = \|c_q - c_d\|$ is the Euclidean distance between two colors c_q and c_d; T_c is the maximum distance for two colors to be considered similar and $d_{max} = \alpha T_c$. The recommended value for T_c is between 10 and 20 in CIE-LUV color space and between 1.0 and 1.5 for α.

The distance between the Edge Histogram descriptors of two images Q and D is computed by adapting the L_1-norm as

$$d(Q,D) = \sum_{i=0}^{79} |h_Q(i) - h_D(i)| + 5\sum_{i=0}^{4} \left|h_Q^g(i) - h_D^g(i)\right| + \sum_{i=0}^{64} \left|h_Q^s(i) - h_D^s(i)\right|$$

where $h_Q(i)$ and $h_D(i)$ represent the histogram bin values of image Q and D, $h_Q^g(i)$ and $h_D^g(i)$ for global edge histograms, and $h_Q^s(i)$ and $h_D^s(i)$ for semi-global edge histograms.

The distance between two Homogeneous Texture descriptors Q and D (full layer – using both energy and energy deviation) is computed by

$$d(Q,D) = w_{dc}|Q(0)-D(0)| + w_{std}\,|Q(1) - D(1)| + \sum_{n=0}^{RD-1}\sum_{m=0}^{AD-1} w_e(n)\,|Q(n\cdot AD+m+2) - D(n\cdot AD+m+2)| + w_{ed}(n)\,|Q(n\cdot AD+m+32) - D(n\cdot AD+m+32)|$$

where w_{dc}, w_e and w_{ed} are the weights; the *Radial Division*, $RD = 5$ and *Angular Division*, $AD = 6$. Matching with this distance metric is not scale and rotation invariant.

The distance between two face recognition descriptors Q and D is computed as follows.

$$d(Q,D) = \sum_{i=0}^{47} w_i (Q(i) - D(i))^2$$

The intensity of Motion Activity is a scalar value, therefore, the distance is computed simply by taking the difference between two descriptor values in the query and the database. When the spatial localization of motion activity is given, Euclidean distance between the vectors is used. The distance between two object trajectories T_Q and T_D is computed as a weighted average of distances between object positions d_P, speeds d_S and accelerations d_A.

$$d(T_Q,T_D) = \frac{w_P d_P(T_Q,T_D) + w_S d_S(T_Q,T_D) + w_A d_A(T_Q,T_D)}{w_P + w_S + w_A}$$
$$d_P(T_Q,T_D) = \sum_i \frac{(x_{qi} - x_{di})^2 + (y_{qi} - y_{di})^2}{\Delta ti}$$

with similar definitions for d_S and d_A.

For spatial position queries, Euclidean distance between the center points of objects' MBRs is used. The definition of distance computation for Contour Shape descriptor is rather long, and therefore, not included here. If multiple instances of a descriptor are available for a *Moving Region* to account for the change in its appearance througout the shot, the distance is computed for all the instances and the minimum is taken.

If the computed distance for a video segment in the database is greater than the user-specified distance threshold for the query video segment and descriptor (e.g., for *Keyframe* with CSD, if $d(Q,D)/d_{max} > T_{Keyframe,CSD}$) that segment is discarded. Otherwise, the similarity, $s(Q,D)$,

between two descriptors Q and D is computed as

$$s(Q, D) = 1 - d(Q, D)/d_{max}, \quad 0 \leq s(Q, D) \leq 1.0$$

where $d(Q, D)$ is the distance between descriptors Q and D, d_{max} is the maximum possible distance for the type of descriptor in the computation. The maximum distance for each descriptor is computed by taking the maximum distance from a large set of descriptors extracted from video segments (*Shots, Keyframes, Moving Regions, Still Regions*).

Spatial Query Processing. Spatial locations of Still Regions and Moving Regions are stored in the database by their MBRs, without any preprocessing to extract and store the spatial relations between the regions. Therefore, spatial similarity between regions is computed at query execution time. This is computationally more expensive but it provides a more flexible and accurate matching for spatial position and spatial relation queries.

For each *Still Region* or *Moving Region* in the query, first, queries related to the properties of the region (textual, color, texture, shape, location, motion) are executed as described above. Then, the resulting video segments undergo spatial query processing to compute the spatial similarities between them. We use the spatial similarity matching approach described in [24] because of its efficiency and robustness. First, the vectors connecting the center points of objects' MBRs, $\overrightarrow{Q_{xy}}$ and $\overrightarrow{D_{ij}}$, are computed. $\overrightarrow{Q_{xy}}$ is the query vector connecting the center points of query objects' MBRs, Q_x and Q_y; $\overrightarrow{D_{ij}}$ is the vector connecting the center points of database objects' MBRs, D_i and D_j, as shown in Figure 8. Then, the pairwise spatial similarity is computed by the cosine of the angle between the vectors $\overrightarrow{Q_{xy}}$ and $\overrightarrow{D_{ij}}$, using vector dot product:

$$d(Q_{xy}, D_{ij}) = \cos\theta = \frac{\overrightarrow{Q_{xy}} \bullet \overrightarrow{D_{ij}}}{\left|\overrightarrow{Q_{xy}}\right| \left|\overrightarrow{D_{ij}}\right|}, \quad 0 \leq \theta \leq \pi, \quad -1 \leq d(Q_{xy}, D_{ij}) \leq +1$$

The output value is in the range [-1, +1], with +1 indicating identical spatial relation and -1 opposite spatial relation. The text-based spatial queries are executed in the same way, by converting each spatial relation query to a unit vector. For instance, Q_x *right* Q_y (Q_x is to the right of Q_y) query is converted to a query vector $\overrightarrow{Q_{xy}} = [-1, 0]$, from Q_x to Q_y. Similarly, query Q_x *left* Q_y is converted to $\overrightarrow{Q_{xy}} = [1, 0]$ and query Q_x *above* Q_y to $\overrightarrow{Q_{xy}} = [0, -1]$.

Multiple MBRs are stored in the database for *Moving Regions* to keep track of their locations. The spatial similarity is computed for all the MBRs and the maximum similarity value is taken

as the final similarity.

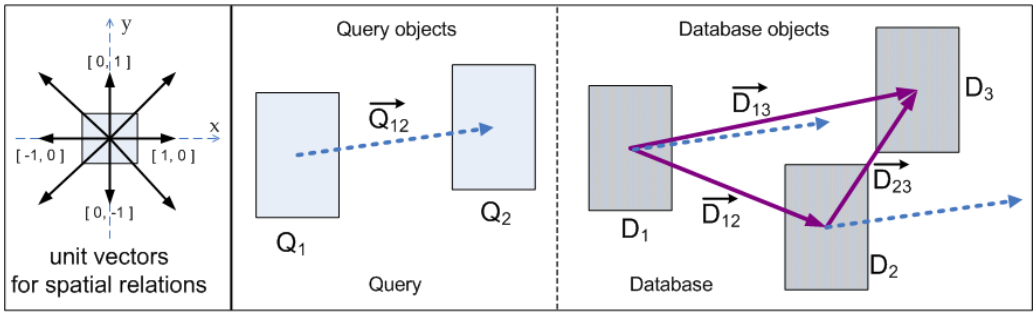

Figure 8: Spatial query processing by vector dot product between the vectors connecting centers of objects' MBRs. In the sketch-based spatial query in the middle, the query is represented with the vector $\overrightarrow{Q_{12}}$, from the center of object Q_1's MBR to the center of object Q_2's MBR. The spatial relation between the database objects D_1 and D_3 is the most similar to the spatial relation between query objects Q_1 and Q_2. Text-based queries (*right, left, above, below, etc.*) are converted to unit vectors as shown on the left.

Temporal queries, if any, are executed after spatial queries by checking if the list of video segments satisfies the temporal relations specified in the query. Spatial queries implicitly enforce a temporal relation between *Still* and *Moving Regions*, since they must co-appear on a scene for a certain time interval in the video to satisfy the spatial relations.

5.3 Fusion of Subquery Results for Multimodal Query Processing

When multiple descriptors, possibly in different modalities, are specified for a query video segment, each is executed as a separate subquery resulting in a list of video segments with similarity values. These subquery results must be fused to come up with the final query result. This is done in a bottom-up manner as shown in Figure 9. Each node in the tree has an associated weight and threshold, which can be specified by the user during query specification. The similarity at each node is computed as the weighted average of the similarities of its children and the fusion process continues upward in the tree until the final query result is obtained.

To illustrate the fusion process, consider a composite query consisting of a *Keyframe* and a *Moving Region*, similar to the one in Figure 13. Suppose that the user specifies CSD and SCD as color descriptors, EHD and HTD as texture descriptors and *golf green* as semantic descriptor for the *Keyframe* and CSD, EHD, RSD and spatial location for the *Moving Region*. Hence there are two video segments in the query and 9 subqueries, 5 for the *Keyframe* (CSD, SCD, EHD, HTD and semantic) and 4 for the *Moving Region* (CSD, EHD, RSD and spatial location). First, 9 subqueries are executed by the respective query execution threads (Figure 7)

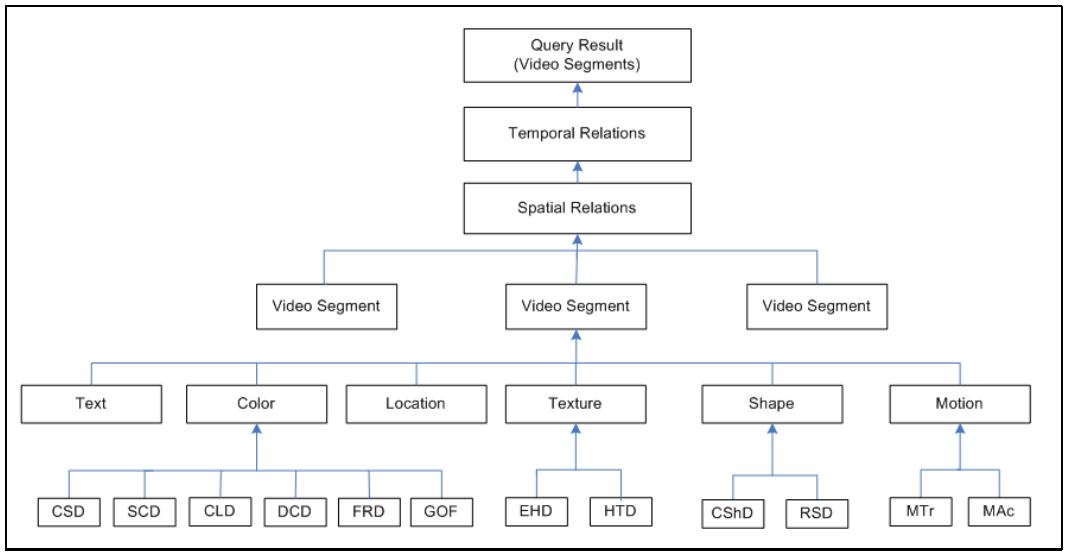

Figure 9: Subquery results are fused in a bottom-up manner. Each node has an associated weight and threshold. The similarity of a video segment at each node is computed as the weighted average of the similarities of its children.

using the distance metrics defined above. At the end of this first stage, we obtain 5 lists of *Keyframes*, 4 lists of *Moving Regions*, each element of the lists having a similarity value. Then, we merge these lists starting with the descriptors in the leaf nodes of Figure 9 (e.g., CSD, SCD and EHD, HTD) and move up in the tree. For the *Keyframe*, we fuse the results from CSD and SCD subqueries to obtain the color similarity, the results from EHD and HTD subqueries to obtain the texture similarity using

$$s_{i,Color} = \frac{w_{Keyframe,CSD}\ s_{i,CSD} + w_{Keyframe,SCD}\ s_{i,SCD}}{w_{Keyframe,CSD} + w_{Keyframe,SCD}}$$
$$s_{i,Texture} = \frac{w_{Keyframe,EHD}\ s_{i,EHD} + w_{Keyframe,HTD}\ s_{i,HTD}}{w_{Keyframe,HTD} + w_{Keyframe,HTD}}$$

where $s_{i,Color}$ is the color similarity for the i^{th} *Keyframe*, $w_{Keyframe,CSD}$ is the weight for CSD and so on. If the similarity of Keyframe i is less than the threshold specified by the user, it is discarded. At this point we have 3 lists of *Keyframes* having similarity values for color, texture, semantic (text). We fuse these 3 lists to obtain the final list of *Keyframes*.

$$s_i = \frac{w_{Keyframe,Color}\ s_{i,Color} + w_{Keyframe,Texture}\ s_{i,Texture} + w_{Keyframe,Text}\ s_{i,Text}}{w_{Keyframe,Color} + w_{Keyframe,Texture} + w_{Keyframe,Text}}$$

The subquery results for the *Moving Region* are fused in the same way. If there are also spatial

or temporal relation subqueries, they are executed and similarity values of the video segments are updated in the same way. Finally, we obtain N_{vs} lists of video segments, where N_{vs} is the number of video segments in the query. The final query result is obtained by fusing these lists using the same weighted average approach as above and sorting the list in descending order of similarity. The final query result is sent to the client.

6 Implementation Details

The system is implemented in C++. Graphical user interfaces are created with open-source, cross-platform C++ GUI library wxWidgets [25]. Open Source Computer Vision Library (OpenCV) [26] and FFmpeg [27] are used to handle (read, decode, copy, save, etc.) the image and video data. The MPEG-7 compatible video feature extraction and annotation tool uses the MPEG-7 feature extraction library that we adapted from the MPEG-7 XM Reference Software [19]. XML data is handled with open-source Xerces-C++ XML Parser library [28]. Finally, Tamino [22] is used as the native XML database to store the MPEG-7 XML descriptions of videos. The system can use any XML database that supports XQuery.

7 Sample Queries

In this section, we present some example queries performed on a video data set consisting of 14 video sequences with 25 thousand frames from TRECVID 2004 and 2008 data sets [29], consisting of news, documentary, educational and archiving program videos. We obtained the MPEG-7 representations of the videos with our MPEG-7 compatible video feature extraction and annotation tool. The return type of queries is selected as *Shot* during query formulation, i.e., the query result returned by the system is a list of shots in ranked order. Each shot in the query result list is shown with a representative keyframe in the following figures. For more query examples, please see *BilVideo-7* website [20].

Two spatial query examples are shown in Figure 10. The first query at the top searches for the video segments in which a golfer is *above* a golf cart. The query is formulated as a text-based spatial relation query, "golfer *above* golf cart", in the *Spatial Query Interface* (Figure 5). The system successfully returns three relevant video segments that exactly match the spatial query condition. The fourth result contains a golfer but no golf cart and spatial condition is not satisfied. Therefore, its rank is lower than the first three. The second query is formulated by drawing two rectangles on the sketch-based *Spatial Query Interface* and labeling them as

Figure 10: Spatial query examples. Queries are formulated in the *Spatial Query Interface*, Figure 5. The query at the top is a text-based spatial relation query, "golfer *above* golf cart"; the query at the bottom is a sketch-based spatial query formulated by drawing two rectangles and labeling them as *Clinton* and *Blair*. Numbers show the rank of retrieval.

Figure 11: Image-based low-level query example. Query image is represented with Color Structure and Dominant Color descriptors.

Clinton and *Blair*. The spatial query condition is satisfied exactly in the first two video segments returned, while it is not satisfied in the last two, but *Clinton* and *Blair* appear together. This is a desirable result of our bottom-up fusion algorithm; as the number of satisfied query conditions for a video segment decreases the video segment's similarity also decreases, ranking lower in the query result. As a result, the ranking approach is effective and it produces query results that are close to our perception.

The query in Figure 11 is an image-based low-level query, in which the query image is represented with Color Structure and Dominant Color descriptors. The query in Figure 12 is a

Figure 12: Region-based low-level query example. Query image region is represented with Color Structure and Region Shape descriptors, and searched in *Moving Regions*.

region-based low-level query, in which the query region is represented with Color Structure and Region Shape descriptors. Both query results are satisfactory considering the types of descriptors used.

Figure 13: Composite query example. Keyframe is represented with Dominant Color and *golf green*. Moving Region is represented with Color Structure, Region Shape and *golfer*.

The query shown in Figure 13 is a composite query, in which high-level semantics in the form of keyword annotations and low-level descriptors (DCD, CSD, RSD) are used together to describe the query video segments. Moreover, there are two different types of video segments in the query: *Keyframe* and *Moving Region*. Similarly, the query in Figure 14 is also a composite query consisting of two *Still Regions* and one *Moving Region* with descriptors. Using this query, the user can access the video segments in which an *airplane or boat or helicopter* appears in a scene having regions as described by the *Still Regions* in the query. As the query results show, our system can handle such queries effectively. The number and type of video segments

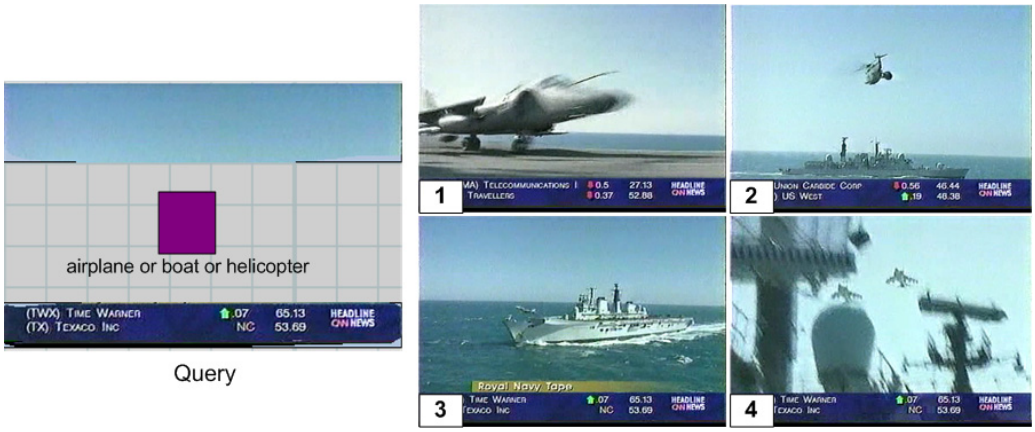

Figure 14: Composite query example. Two *Still Regions* at the top and at the bottom are represented with Color Structure and Edge Histogram descriptors. The *Moving Region* in the middle is represented with semantic concepts *airplane or boat or helicopter*.

in the query, as well as the descriptors used to describe them are not limited. This makes the composite queries very flexible and powerful, enabling the user to formulate very complex queries easily. To our knowledge, our system is unique in supporting such complex queries.

Query type	*Description (Segments and descriptors)*	*Response time (sec)*
Text-based semantic query	Keyframe (keyword)	0.125
Text-based semantic query	Moving Region (keyword)	0.125
Text-based semantic query	Keyframe (keyword), Moving Region (keyword)	0.188
Color query	Keyframe (CSD)	0.141
Texture query	Keyframe (HTD)	0.125
Color + Texture query	Keyframe (CSD+HTD)	0.172
Shape query	Moving Region (RSD)	0.156
Spatial query	Text-based, 2 Still Regions	0.172
Spatial query	Text-based, 2 Moving Regions	0.187
Spatial query	Sketch-based, 2 Moving Regions	0.187
Composite query in Figure 13	Keyframe (DCD+keyword), Moving Region (CSD+RSD+keyword)	0.438
Composite query in Figure 14	2 Still Regions (CSD+EHD), Moving Region (keyword)	0.391

Table 1: Response times (in seconds) for different types of queries. *Query Processing Server* and Tamino XML database are installed on a notebook PC with Intel Core 2, dual-core 2.0 GHz processors and 2.0 GB of RAM, running Windows XP. The client connects to the server and executes the queries described in the table.

Table 1 presents query execution times for several queries. The execution time is measured as

the difference between the arrival and completion times of a query. The query execution time is proportional to the number of subqueries (number of video segments and descriptors in the query), database size (number of video segments in the database), the sizes of the descriptors and the complexity of the matching algorithm (distance metric). As expected, queries involving low-level descriptors take longer to execute compared to text-based queries since the distance computation between the low-level descriptors are computationally more expensive. The multi-threaded query processing architecture provides some degree of parallelism and shortens the query execution times when the subqueries are executed in separate threads. For instance, a *Keyframe* query with CSD takes 0.141 seconds and a *Keyframe* query with HTD takes 0.125 seconds to execute, while a *Keyframe* query with CSD and HTD descriptors takes 0.172 seconds to execute, which is less than the serial execution times of CSD and HTD queries (0.266 seconds). This is also demonstrated in the two composite queries in the table.

8 Conclusions and Future Work

We described our prototype MPEG-7 compatible video database system, *BilVideo-7*, that supports different types of multimodal queries in an integrated way. To our knowledge, *BilVideo-7* is the most comprehensive MPEG-7 compatible video database system currently available, in terms of the wide range of MPEG-7 descriptors and manifold querying options. The MPEG-7 profile used for the representation of the videos enables the system to respond to complex queries with the help of the flexible query processing and bottom-up subquery result fusion architecture. The user can formulate very complex queries easily using the *Visual Query Interface*, whose *Composite Query Interface* is novel in formulating a query by describing a video segment as a composition of several video segments along with their descriptors. The broad functionality of the system is demonstrated with sample queries which are handled effectively by the system. The retrieval performance depends very much on the MPEG-7 descriptors and the distance measures used. The low-level MPEG-7 descriptors have been found effective, consistent with our observations, and therefore, widely used by the researchers in the computer vision, pattern recognition and multimedia retrieval communities. We will investigate distance measures other than the ones recommended by MPEG-7 [23].

The multi-threaded query execution architecture is suitable for parallelization. This is required for video databases of realistic size to keep the response time of the system at interactive rates. In a parallel architecture, each query processing node may keep the data for a subset of descriptions (e.g., text, color, texture, shape) and execute only the relevant subqueries. A central Query Processor can coordinate the operation of query processing nodes.

The major bottleneck for the system is the generation of the MPEG-7 representations of videos by manual processing, which is time consuming, error-prone and which also suffers from human subjectivity. This hinders the construction of a video database of realistic size. Therefore, our current focus is on equipping the MPEG-7 compatible video feature extraction and annotation tool with as much automatic processing capabilities as possible to reduce manual processing time, error and human subjectivity during region selection and annotation.

Finally, future versions of *BilVideo-7* will also support representation and querying of audio and image data. The multimodal query processing architecture makes it easy to add new descriptors in new modalities (e.g., audio descriptors). Images can be considered to be a special case of *Keyframes* which are decomposed into *Still Regions*, and hence can be supported easily.

9 Acknowledgments

The authors would like to thank Rana Nelson for proofreading this manuscript, and anonymous reviewers for their constructive comments.

References

[1] B. S. Manjunath, P. Salembier, and T. Sikora, Eds., *Introduction to MPEG-7: Multimedia Content Description Interface*. England: WILEY, 2002.

[2] "MPEG-7 Web Site." [Online].
Available: http://www.chiariglione.org/mpeg/standards/mpeg-7/mpeg-7.htm

[3] M. E. Dönderler, E. Saykol, U. Arslan, Ö. Ulusoy, and U. Güdükbay, "BilVideo: Design and Implementation of a Video Database Management System," *Multimedia Tools and Applications*, vol. 27, no. 1, pp. 79–104, 2005.

[4] S. Jeannin and A. Divakaran, "MPEG-7 Visual Motion Descriptors," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 6, pp. 720–724, Jun 2001.

[5] W. Bailer and P. Schallauer, "Detailed Audiovisual Profile: Enabling Interoperability Between MPEG-7 Based Systems," *Proceedings of the 12th International Multi-Media Modelling Conference Proceedings*, pp. 217–224, January 2006.

[6] M. Döller and H. Kosch, "The MPEG-7 Multimedia Database System (MPEG-7 MMDB)," *The Journal of Systems and Software*, vol. 81, no. 9, pp. 1559–1580, 2008.

[7] Y. Rui, "MPEG-7 Enhanced Ubi-Multimedia Access – Convergence of User Experience and Technology," in *Proceedings of the First IEEE International Conference on Ubi-Media Computing*, 31 August 2008, pp. 177–183.

[8] B. Tseng, C.-Y. Lin, and J. Smith, "Using MPEG-7 and MPEG-21 for Personalizing Video," *IEEE Multimedia*, vol. 11, no. 1, pp. 42–52, January-March 2004.

[9] O. Steiger, A. Cavallaro, and T. Ebrahimi, "MPEG-7 Description for Scalable Video Reconstruction," EPFL, Tech. Rep., 2004. [Online].
Available: http://infoscience.epfl.ch/record/87042/files/Steiger2004_740.pdf

[10] H. Yi, D. Rajan, and L.-T. Chia, "Automatic Generation of MPEG-7 Compliant XML Document for Motion Trajectory Descriptor in Sports Video," *Multimedia Tools and Applications*, vol. 26, no. 2, pp. 191–206, 2005.

[11] "IBM VideoAnnEx Annotation Tool." [Online].
Available: http://www.research.ibm.com/VideoAnnEx

[12] K. Petridis, D. Anastasopoulos, C. Saathoff, N. Timmermann, Y. Kompatsiaris, and S. Staab, "M-OntoMat-Annotizer: Image Annotation Linking Ontologies and Multimedia Low-Level Features," in *Proceedings of the 10th International Conference on Knowledge-Based Intelligent Information and Engineering Systems (KES'2006), Lecture Notes in Computer Science*, vol. 4253. Springer, 2006, pp. 633–640.

[13] J. Löffler, K. Biatov, C. Eckes, and J. Köhler, "IFINDER: An MPEG-7-Based Retrieval System for Distributed Multimedia Content," in *Proceedings of the Tenth ACM International Conference on Multimedia*. New York, NY, USA: ACM, 2002, pp. 431–435.

[14] A. Gkoritsas and M. Angelides, "COSMOS-7: A Video Content Modeling Framework for MPEG-7," *Proceedings of the 11th International Multimedia Modelling Conference (MMM)*, pp. 123–130, January 2005.

[15] L. Gagnon, S. Foucher, and V. Gouaillier, "ERIC7: An Experimental Tool for Content-Based Image Encoding and Retrieval Under the MPEG-7 Standard," in *Proceedings of the Winter International Synposium on Information and Communication Technologies (WISICT '04)*. Trinity College Dublin, 2004, pp. 1–6.

[16] M. Lux, J. Becker, and H. Krottmaier, "Caliph&Emir: Semantic Annotation and Retrieval in Personal Digital Photo Libraries," in *Proceedings of the CAiSE '03 Forum at 15th Conference on Advanced Information Systems Engineering*, Velden, Austria, June 2003, pp. 85–89.

[17] M. Baştan, U. Güdükbay, and Özgür Ulusoy, "Segmentation-Based Extraction of Important Objects from Video for Object-Based Indexing," *Proceedings of the IEEE International Conference on Multimedia and Expo*, pp. 1357–1360, June 2008.

[18] "XQuery 1.0: An XML Query Language." [Online]. Available: http://www.w3.org/TR/xquery

[19] "ISO Publicly Available Standards." [Online].
Available: http://standards.iso.org/ittf/PubliclyAvailableStandards/index.html

[20] "BilVideo-7: MPEG-7 Compatible Video Database System Web Site." [Online].
Available: http://www.cs.bilkent.edu.tr/~bilmdg/bilvideo-7

[21] G. Pavlović-Lažetić, "Native XML Databases vs. Relational Databases in Dealing with XML Documents," *Kragujevac Journal of Mathematics*, vol. 30, pp. 181–199, 2007.

[22] "Tamino: Software AG XML Data Management Product." [Online].
Available: http://www.softwareag.com/Corporate/products/wm/tamino/default.asp

[23] H. Eidenberger, "Distance measures for MPEG-7-based retrieval," in *Proceedings of the 5th ACM SIGMM International Workshop on Multimedia Information Retrieval (MIR '03)*, 2003, pp. 130–137.

[24] J. Z. Li and M. T. Ozsu, "STARS: A Spatial Attributes Retrieval System for Images and Videos," in *Proceedings of 4th International Conference on Multimedia Modeling*, 1997, pp. 69–84.

[25] "wxWidgets: Open Source Cross-Platform GUI Library." [Online].
Available: http://www.wxwidgets.org

[26] "OpenCV: Open Source Computer Vision Library." [Online].
Available: http://opencvlibrary.sourceforge.net

[27] "FFmpeg: Cross-platform Audio Video Library." [Online].
Available: http://ffmpeg.mplayerhq.hu

[28] "Xerces-C++ XML Parser." [Online]. Available: http://xerces.apache.org/xerces-c

[29] A. F. Smeaton, P. Over, and W. Kraaij, "Evaluation campaigns and TRECVid," in *Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval (MIR)*, New York, NY, USA, 2006, pp. 321–330.