BilVideo-7: VIDEO PARSING, INDEXING AND RETRIEVAL

A DISSERTATION SUBMITTED TO

THE DEPARTMENT OF COMPUTER ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCE

OF BİLKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

By

Muhammet Baştan

July, 2010

ABSTRACT

BilVideo-7: VIDEO PARSING, INDEXING AND RETRIEVAL

Muhammet Baştan
Ph.D. in Computer Engineering
Supervisors: Assoc. Prof. Dr. Uğur Güdükbay and Prof. Dr. Özgür Ulusoy
July, 2010

Video indexing and retrieval aims to provide fast, natural and intuitive access to large video collections. This is getting more and more important as the amount of video data increases at a stunning rate. This thesis introduces the BilVideo-7 system to address the issues related to video parsing, indexing and retrieval.

BilVideo-7 is a distributed and MPEG-7 compatible video indexing and retrieval system that supports complex multimodal queries in a unified framework. The video data model is based on an MPEG-7 profile which is designed 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, 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 formulate very complex queries containing any type and number of video segments with their descriptors and specifying the spatio-temporal relations between them. The multi-threaded query processing server parses incoming 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 and sends the result to the originating client. 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.

Keywords: MPEG-7, video processing, video indexing, video retrieval, multimodal query processing.

ÖZET
BilVideo-7: VİDEO ÇÖZÜMLEME, İNDEKSLEME VE ERİŞİMİ

Muhammet Baştan
Bilgisayar Mühendisliği, Doktora
Tez Yöneticileri: Doç. Dr. Uğur Güdükbay ve Prof. Dr. Özgür Ulusoy
Temmuz, 2010

Video indeksleme ve erişimi sistemleri büyük çaptaki video verilerine hızlı, doğal ve kolay bir şekilde ulaşılabilmesini amaçlar. Son zamanlarda video arşivlerinin çok hızlı büyümesiyle bu sistemlerin önemi daha da artmıştır. Bu tez, video çözümleme, indeksleme ve erişimi konularında yeni yöntemler öneren BilVideo-7 sistemini sunmaktadır.

BilVideo-7, karmaşık çok kipli video sorgularını aynı anda destekleyen, dağıtık mimariye sahip MPEG-7 uyumlu bir video indeksleme ve erişimi sistemidir. Video veri modeli bir MPEG-7 profili üzerine bina edilmiş olup, videolar bu profile uygun olarak çekimlere, anahtar karelere, durağan ve hareketli bölgelere ayrılmaktadır. Videoların bu veri modeline uygun XML gösterimleri, BilVideo-7'nin MPEG-7 uyumlu video öznitelik çıkarma ve etiketleme yazılımı yardımıyla elde edilip XML veritabanında saklanmaktadır. Kullanıcılar, görsel sorgulama arayüzünü kullanarak metin, renk, doku, biçim, konum, hareket ve uzamsal-zamansal sorguları kolay bir şekilde yapabilmektedir. Kompozit sorgu arayüzü ise, kullanıcıların, istenilen sayıda video parçasını ve betimleyicisini bir araya getirip aralarındaki uzamsal-zamansal ilişkileri belirleyerek, oldukça karmaşık, çok kipli sorguları kolayca formüle edebilmesini sağlamaktadır. Sorgular, çok izlekli bir sorgu işleme sunucusu tarafından işlenmekte; istemcilerden gelen sorgular önce alt sorgulara ayrılmakta ve herbir sorgu, kendi sorgu tipine ait biz izlek tarafından işlenmektedir. Daha sonra, alt sorgu sonuçları birleştirilerek nihai sorgu sonucu elde edilip istemciye geri gönderilmektedir. Sistemin bir bütün olarak özgünlüğü, MPEG-7 uyumlu bir ortamda, detaylı bir video veri modeli, çok sayıda betimleyici ve çok kipli sorgu işleme özelliği ile güçlü bir video indeksleme ve sorgulama sistemi olmasıdır.

Anahtar sözcükler: MPEG-7, video işleme, video indeksleme, video sorgulama, çok kipli sorgu işleme.

Acknowledgement

I would like to express my sincere gratitude to my supervisors Assoc. Prof. Dr. Uğur Güdükbay and Prof. Dr. Özgür Ulusoy for their support, guidance, encouragement and patience during this thesis work.

I am grateful to Prof. Dr. Adnan Yazıcı, Asst. Prof. Dr. Pınar Duygulu Şahin and Asst. Prof. Dr. Sinan Gezici for reviewing this thesis. I am also grateful to Asst. Prof. Dr. Ali Aydın Selçuk, Asst. Prof. Dr. İbrahim Körpeoğlu, Prof. Dr. Fazlı Can, Asst. Prof. Dr. Selim Aksoy, Assoc. Prof. Dr. Uğur Doğrusöz, Prof. Dr. Cevdet Aykanat and Asst. Prof. Dr. İlyas Çiçekli, from whom I have learned a lot.

I would like to thank all the friends in Bilkent, in particular, my housemates in 18-2: Ali Cevahir, Murat Ak, Hüseyin Gökhan Akçay, Hasan Güner (the nightbird), Esat Belviranlı and Bayram Boyraz; my dear companions: Tahir Malas, Uygar Aydemir, Mehmet Uç, Rıfat Özcan, Serkan Bütün, Enver Paşa (Kayaaslan), Cem Aksoy, Ali Adalı, Ömer Faruk Oran, Sayım Gökyar, Talha Erdem, Gürkan Polat, İbrahim Onaran, Uğur Töreyin, Kadir Akbudak, Tolga Özaslan, Abdullah Bülbül, Akif Burak Tosun, Mehmet Zahid Ateş, Mesut Göksu, Fatih Genişel, Mücahid Kutlu, Şükrü Torun and Bilal Turan; my diligent research colleagues: Onur Küçüktunç, Hayati Çam (sadly, passed away in a traffic accident on May 2009), Fatih Çakır, Serkan Genç and Erdem Sarıyüce; my former research colleagues: Tolga Can, Esra Ataer, Emel Doğrusöz, Gökberk Cinbiş, Nazlı İkizler-Cinbiş, Selen Pehlivan and Derya Özkan; and other nice friends: Ata Türk, Sengör Abi, Alper Rıfat Uluçınar, Onur Önder, Alptuğ Dilek, Ateş Akaydın, Funda Durupınar, Çağlar Arı and Kıvanç Köse. I must also mention my good-never-old friends Mustafa Kılınç, Ahmet Öztürk and Çetin Göztepe.

I sincerely thank my family for their everlasting support. I am indebted to *Hocabey* and Doğramacı family for establishing and maintaining this university.

Finally, I would like to thank TÜBİTAK BİDEB for their financial support during my Ph.D. study.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

YouTube[1] is currently the world's largest online video sharing site. Today, 24 hours of video are being uploaded to YouTube every minute [1], with over 2 billion views a day. In 2008, it was estimated that there were over 45,000,000 videos on YouTube, with a rate of increase of 7 hours of video per minute [2]. Other online video repositories, on-demand Internet TV, news agencies, etc. all add to the astounding amount and growth of video data, which needs to be indexed, and when requested, presented to the users that may be using various client software residing on various platforms. This is where multimedia database management systems are brought into play.

Early prototype multimedia database management systems used the query-by-example (QBE) paradigm to respond to user queries [3, 4, 5]. 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

[1]http://www.youtube.com

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 so that users can formulate queries containing both high-level semantic and low-level descriptors.

Another important issue to be considered in today's multimedia systems is interoperability: the ability of diverse systems and organizations to work together (interoperate)[2]. This is especially crucial for distributed architectures if the system is to be used by multiple heterogeneous clients. Therefore, MPEG-7 [6] standard as the multimedia content description interface can be employed to address this issue.

The design of a multimedia indexing and retrieval system is directly affected by the type of queries to be supported. Specifically for a video indexing and retrieval system, types of descriptors and the granularity of the representation determine the system's performance in terms of speed and effective retrieval. Below, we give some example video query types that might be attractive for most users, but which also are not all together supported by the existing systems in an interoperable 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.

[2]http://en.wikipedia.org/wiki/Interoperability

Especially noteworthy is the composite query type, since it encompasses the other query types and enables the formulation of very complex video queries that would otherwise be very difficult, if not impossible, to formulate. However, the video data model, query processing and query interface should be so designed that such queries can be supported.

This dissertation introduces the BilVideo-7 [7, 8, 9] video parsing, indexing and retrieval system to address the above-mentioned issues within the domain of video data.

1.2 Introducing BilVideo-7

BilVideo-7 is a comprehensive, MPEG-7 compatible and distributed video database system to support multimodal queries in a unified video indexing and retrieval framework. The video data model of BilVideo-7 is designed in a way to enable detailed queries on videos. 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 MPEG-7 compatible video representations according to the adopted data model is obtained using the MPEG-7 compatible video feature extraction and annotation tool of BilVideo-7.

We next highlight the prominent features of BilVideo-7, which render it unique as a complete video parsing, indexing and retrieval system and also emphasize the contributions of this thesis.

- Composite queries. This is one of the distinctive features of BilVideo-7. Users can compose very complex queries by describing the scenes or video segments they want to retrieve by assembling video segments, images, image regions and sketches, and then specifying their properties by high-level or low-level MPEG-7 descriptors. Figures 5.2 and 7.4 show examples of such queries.

- Video data model. In contrast to simple keyframe-based video representation that is prevalent in the literature, BilVideo-7 uses a more detailed video representation to enable more advanced queries (e.g., composite queries).
- Multi-modal query processing. The query processing with a bottom-up subquery result fusion architecture (Chapter 5) enables a seamless support for multimodal queries. Moreover, it is easy to add new modalities, which is important for the extendibility of the system.
- MPEG-7 compatibility. The data model of BilVideo-7 is based on an MPEG-7 profile. Videos are decomposed into Shots, Keyframes, Still Regions and Moving Regions, which are represented with a wide range of high- and low-level MPEG-7 descriptors. This in turn provides manifold query options for the users. MPEG-7 compatibility is crucial for the interoperability of systems and is getting more and more important as the use of different types of platforms gets more widespread.
- Distributed architecture. BilVideo-7 has a distributed, client-server architecture (Figure 4.1). This distributed architecture allows all the online components, i.e., client (visual query interface), query processing server and XML database, to reside on different machines; this is important for the construction of realistic, large-size systems.
- Multi-threaded query execution. The query processing server parses the incoming queries into subqueries and executes each type of subquery in a separate thread (Section 5.2, Figure 5.3). Multi-modal query processing and multi-threaded query execution are closely related and this architecture is also very suitable for parallelization for the construction of a realistic system.
- MPEG-7 compatible feature extraction and annotation. BilVideo-7 has an MPEG-7 compatible video parsing, feature extraction and annotation tool, BilMAT (Chapter 6), to obtain the MPEG-7 compatible XML representations of the videos according to the the detailed data model. This is expected to fill a gap in the literature.
- Visual query interface. BilVideo-7 clients' visual query interface provides an intuitive, easy-to-use query interface (Figure 4.2) with manifold querying and

browsing capabilities: video table of contents (VideoTOC), XQuery; textual, color, texture, shape, motion, spatial, temporal and composite queries.

1.3 Organization of the Dissertation

The rest of the dissertation is organized as follows. Chapter 2 reviews the related work on video database systems and MPEG-7. Chapter 3 describes the video data model of BilVideo-7. Chapter 4 presents the distributed, client-server architecture and main software components of the system. Chapter 5 focuses on the query processing on the server side. Chapter 6 elaborates on video parsing, feature extraction and annotation to obtain the MPEG-7 representations of the videos. Chapter 7 demonstrates the capabilities of BilVideo-7 with sample queries. Finally, Chapter 8 concludes the dissertation with possible future directions.

Chapter 2

Related Work

2.1 Image and Video Retrieval Systems

In this section, we review some of the prominent image/video indexing and retrieval systems; the MPEG-7 compatible systems are discussed in Section 2.3.

QBIC (Query by Image Content) system [10, 3] was developed by IBM to explore content-based image and video retrieval methods. QBIC was designed to allow queries on large image and video databases based on example images, sketches, selected color/texture patterns, and camera and object motion. Videos are represented by shots, representative frames (r-frames) and moving objects.

PicToSeek [11] is a web-based image database system for exploring the visual information on the web. The images are automatically collected from the web and indexed based on invariant color and shape features, which are later used for object-based retrieval.

SIMPLIcity (Semantics-sensitive Integrated Matching for Picture LIbraries) [12, 13] is an image retrieval system, which uses semantic classification methods and integrated region matching based on image segmentation. Images are represented by a set of regions, corresponding to objects, with color, texture, shape, and location features. and classified into semantic categories, such as textured-nontextured and graph-photograph. The similarity between images is computed using a region-matching scheme that integrates properties of all the regions in the image.

Photobook [14] is a system to enable interactive browsing and searching of images and image sequences. It relies on image content rather than text annotations and uses an image compression technique to reduce images to a small set of coefficients. VisualSEEk [5] is an image database system that supports color and spatial queries on images with a sketch-based query interface.

STARS [15] is an object oriented multimedia (image, video) database system to support a combination of text- and content-based retrieval techniques with special focus on spatial queries. VideoQ [4] is a content-based video search system that supports sketch-based queries formulated on a visual query interface running on a web browser. The data model is based on video objects which are represented and queried by low-level color, texture, shape and motion (trajectory) features.

BilVideo [16, 17] is a prototype video database management system that supports spatio-temporal queries that contain any combination of spatial, temporal, object-appearance and trajectory queries by a rule-based system built on a knowledge-base. The knowledge-base 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. BilVideo has an SQL-like textual query language, as well as a visual query interface for spatio-temporal queries. The query interface is later improved to enable natural language queries [18].

The system described in [19] proposes a fuzzy conceptual data model to represent the semantic content of video data. It utilizes the Unified Modeling Language (UML) to represent uncertain information along with video specific properties. It also presents an intelligent fuzzy object-oriented database framework, which provides modeling of complex and rich semantic content and knowledge of video data including uncertainty, for video database applications. The fuzzy conceptual data model is used in this framework and it supports various types of flexible queries related to video data such as (fuzzy) semantic, temporal, and (fuzzy) spatial queries.

The aim of Video Google [20, 21, 22] is to retrieve the shots and keyframes of a video containing a user-specified object/region, similar to web search engines, such as Google, that retrieve text documents containing particular words.

VITALAS [23] is a video indexing and retrieval system that allows users to perform text-based keyword/concept queries, low-level visual similarity queries and combination of high-level and low-level queries. MediaMill [24] is one of the successfull video retrieval systems supporting high-level queries by automatically obtained semantic concept descriptions, speech transcript based queries and low-level visual similarity queries. The system has effective visualization and browsing interfaces for interactive video retrieval.

There are several survey articles reviewing multimedia information retrieval systems. Early content-based image retrieval systems are described by Smeulders *et al.* [25] and Veltkamp *et al.* [26]. More recent image and video retrieval systems are reviewed in [27, 28, 29, 30].

2.2 MPEG-7 Standard

MPEG-7 [6] 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. 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. 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 data types. 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 [31] in XML Schema Document (XSD) [32] format for describing multimedia content.

The eXperimentation Model (XM) software [33] 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,* and *localization* [6, 34].

2.2.1 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. *Face Recognition Descriptor (FRD)* 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.

2.2.2 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.

2.2.3 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.

2.2.4 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) [35].

2.2.5 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.

2.2.6 Semantic Descriptors

In MPEG-7, the semantic content of multimedia (e.g., objects, events, concepts) can be described by text annotation (free text, keyword, structured) 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 are 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).

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 [6].

The semantic tools of MPEG-7 provide methods to create very brief or very extensive semantic descriptions of multimedia content. Some of the descriptions can be obtained automatically while most of them require manual labeling. Speech transcript text obtained from automatic speech recognition (ASR) tools 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.

2.2.7 MPEG Query Format

In 2007, MPEG-7 adopted a query format, MPEG Query Format (MPQF) [36], to provide a standard interface between clients and MPEG-7 databases for multimedia content retrieval systems. The query format is based on XML and consists of three main parts: (1) Input query format defines the syntax of query messages sent by a client to the server and supports different types of queries: query by free text, query by description, query by XQuery, spatial query, temporal query, etc. (2) Output query format specifies the structure of the result set to be returned. (3) Query management tools are used to search and choose the desired services for retrieval.

2.3 MPEG-7 Compatible Systems

The comprehensiveness and flexibility of MPEG-7 allow its usage in a broad range of applications, but also increase its complexity and adversely affect 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 [37], 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* [38]. The resulting system is demonstrated by audio and image retrieval applications. In [39], 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. In [40], an MPEG-7 compatible description of video sequences for scalable transmission and reconstruction is presented. In [41], 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.

Tseng *et al.* [42] address the issues associated with designing a video personalization and summarization system in heterogeneous usage environments utilizing MPEG-7 and MPEG-21. The system has a three-tier architecture of server, middleware and client. The server maintains the content as MPEG-7 and MPEG-21 metadata descriptions. The client communicates with the server to send user queries, retrieve and display the personalized contents. The middleware selects, adapts and delivers the summarized media to the user.

An MPEG-7 compatible, web-based video database management system is presented in [43]. The system supports semantic description of video content (objects, agent objects, activities and events) and faclitates content-based spatio-temporal queries on video data. In [44], an XML-based content-based image retrieval system is presented. It combines three visual MPEG-7 descriptors: DCD, CLD and EHD. The system supports high dimensional indexing using an index structure called *M-Tree* and uses an Ordered Weighted Aggregation (OWA) approach to combine the distances of the three descriptors.

IBM's *VideoAnnEx Annotation Tool* [45] enables users to annotate video sequences with MPEG-7 metadata. Each shot 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* [46] 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 [47] is developed to produce limited MPEG-7 representation from audio and video by speech processing, keyframe extraction and face detection. COSMOS-7 system [48] 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 is called m-frames (multimedia frames).

ERIC7 [49] is a software test-bed that implements Content-Based Image Retrieval (CBIR) using image-based MPEG-7 color, texture and shape descriptors. *Caliph &*

Emir [50] 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 (CLD, DCD, SCD, EHD).

2.4 Evaluation of Existing Systems

The 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. (2) The user cannot perform complex multimodal queries by combining several video segments and descriptors in different modalities. BilVideo-7 addresses these two major problems by adopting an MPEG-7 profile with a more detailed video representation (Section 3.2) and using a multimodal query processing and bottom-up subquery result fusion architecture to support complex multimodal queries (e.g., composite queries – see Chapter 7 for examples) with a comprehensive set of MPEG-7 descriptors.

Chapter 3

Video Data Model

3.1 Introduction

A video is a sequence of frames which are structured to represent scenes in motion. Figure 3.1 broadly depicts the structural and semantic building blocks of a video. 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). A *scene* is a logical grouping of shots into a semantic unit. This structure is important in designing the video data model.

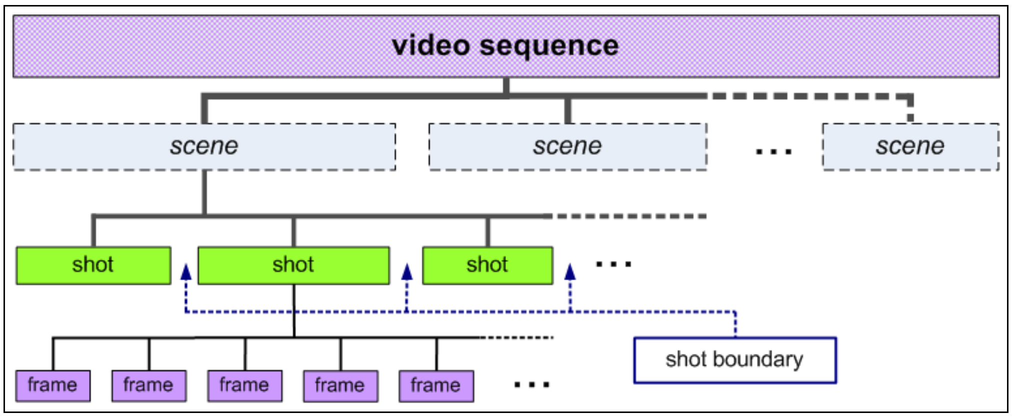


Figure 3.1: Building blocks of a video.

The first step in constructing a multimedia indexing and retrieval system is to decide what kind of queries will be supported and then design the data model accordingly. This is crucial since the data model directly affects the system's performance in terms of querying power. For instance, considering a video indexing and retrieval system, if the videos are represented by only shot-level descriptors, we cannot perform frame or region based queries. Similarly, if video representation does not include object-level details, we cannot perform queries including objects and spatio-temporal relations between them. 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.

3.2 Video Decomposition and Representation

As a video indexing and retrieval system, BilVideo-7 takes into consideration the above mentioned factors for the design of its video data model. That is, the data model should have enough detail to support all types of queries the system is designed for and it should also enable quick response time during retrieval. Hence, the data model should strike a balance between level of detail in representation and retrieval speed.

As an MPEG-7 compatible video indexing and retrieval system, the data model of BilVideo-7 is represented by the MPEG-7 profile depicted in Figure 3.2. First, audio and visual data are separated (Media Source Decomposition [6]). Then, visual content is hierarchically decomposed into smaller structural and semantic units: Shots, Keysegments/Keyframes, Still Regions and Moving Regions. An example of video decomposition according to this profile is shown in Figure 3.3.

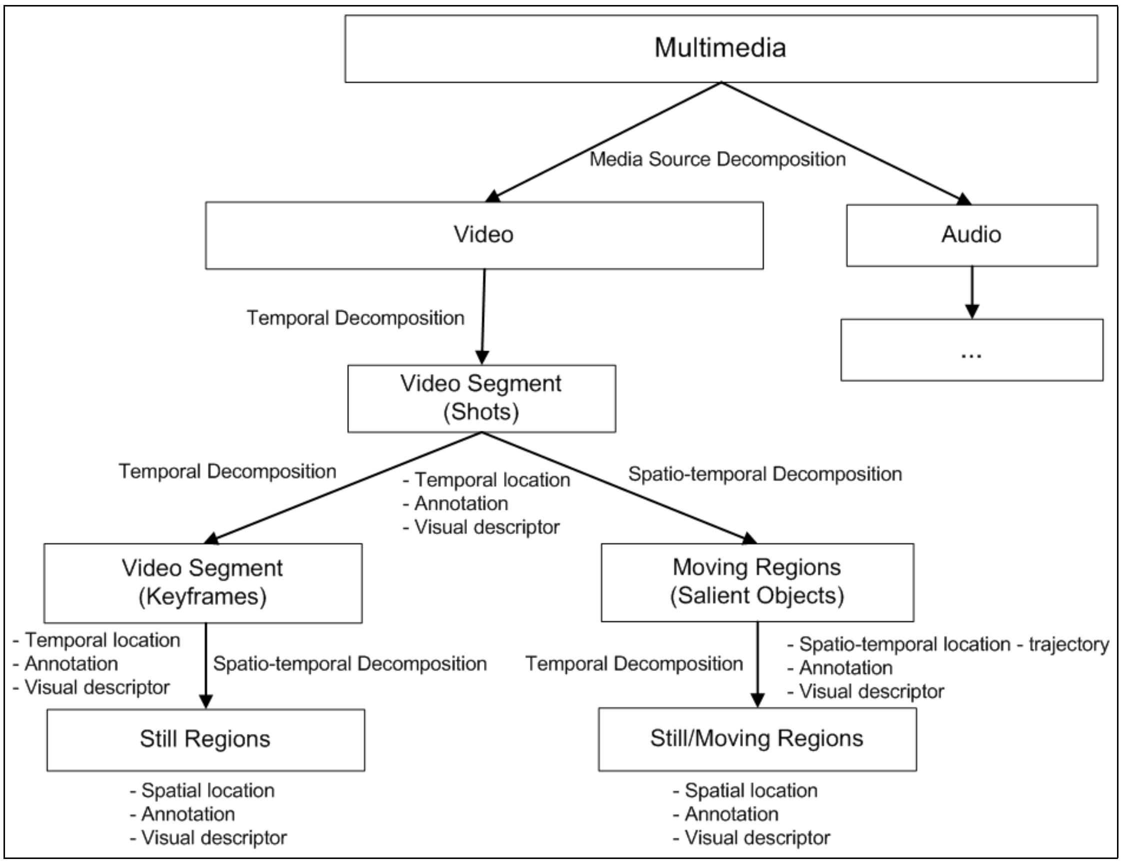


Figure 3.2: MPEG-7 profile used to model the video data.

3.3 Temporal Decomposition

Video is temporally decomposed into non-overlapping video segments called *Shots*, each having a temporal location (start time, duration), annotation to describe the objects and/or events with free text, keyword and structured annotations, and visual descriptors (e.g., motion, GoF/GoP).

The background content of the Shots does not change much, especially if the camera is not moving. This static content can be represented by a single *Keyframe* or a few Keyframes. Therefore, each Shot is temporally decomposed into smaller, more homogeneous video segments (*Keysegments*) which are represented by Keyframes. Each Keyframe is described by a temporal location, annotations and a set of visual descriptors. The visual descriptors are extracted from the frame as a whole.

3.4 Spatio-temporal Decomposition

Each Keyframe in a Shot is decomposed into a set of *Still Regions* (*Spatio-temporal Decomposition*) to keep more detailed region-based information in the form of spatial location by the MBRs of the region, annotation and region-based visual descriptors. These Still Regions are assumed to be valid for the duration of the Keysegment that is represented by this Keyframe.

Each Shot is decomposed into a set of *Moving Regions* to represent the dynamic and more important content of the Shots corresponding to the salient objects. This is to store more information about salient objects and keep track of the changes in their position and appearance throughout the Shot so that more detailed queries regarding them can be performed. We represent all salient objects with Moving Regions even if they are not moving. *Faces* are also represented by Moving Regions, having an additional visual descriptor: Face Recognition Descriptor.

To keep track of the changes in position, shape, motion and visual appearance of the salient objects, we sample and store descriptor values at time points when there is a predefined amount of change in the descriptor values. The trajectory of a salient object is represented by the *Motion Trajectory* descriptor. The MBRs and visual descriptors of the object throughout the Shot are stored by temporally decomposing the object into *Still Regions*.

Notation: From here on, we refer to Shots, Keyframes, Still Regions and Moving Regions, as *video segments*. Throughout the text, we capitalize these terms to comply with the MPEG-7 terminology.

3.5 Summary and Discussion

To summarize the video data model of BilVideo-7, each video consists of a set of Shots. Each Shot consists of a set of Keysegments and Moving Regions. Keysegments are represented by Keyframes which are composed of a set of Still Regions. Keyframes and Still Regions are used to represent mainly the static background content of Shots, while Moving Regions act as the salient objects in the scene.

The summarized data model is a generic data model expressed in MPEG-7 for a general purpose video indexing and retrieval system, e.g., a system for TV news videos. The representation is coarse at shot level, and it gets finer and finer for Keyframes, Still Regions and Moving Regions. The detail level can be easily adjusted to better suit to different application domains. For example, if shot and keyframe level queries are enough for a particular application domain, then the region-level description (Still and Moving Regions) can be omitted during the creation of MPEG-7 compatible XML representations of videos. On the other hand, if the foreground salient objects and faces are of primary interest, as in a surveillance system for security purposes, the Moving Regions may be represented with greater detail, while the shot and keyframe level descriptions may be kept at a minimum or even entirely omitted. The omission is not necessary, but should be preferred to save online/offline processing time and storage.

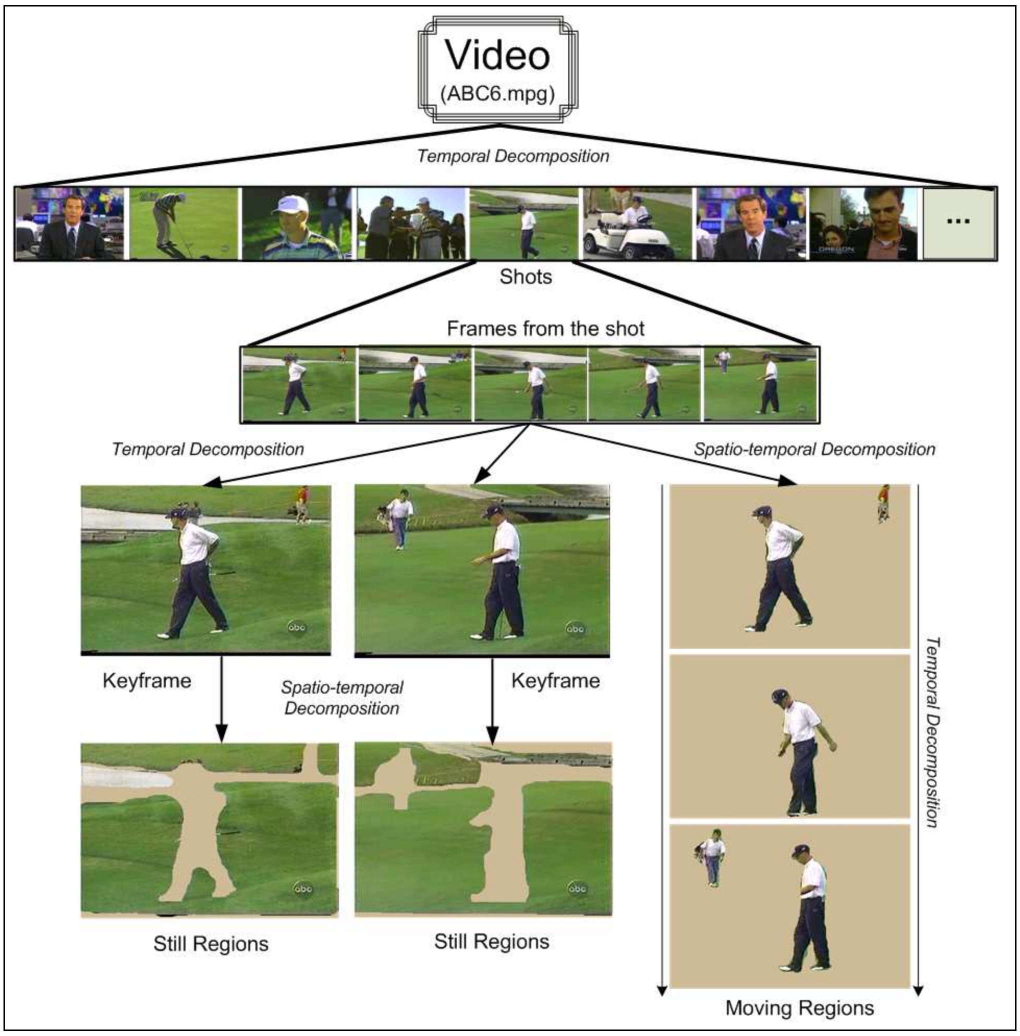


Figure 3.3: MPEG-7 decomposition of a video according to the MPEG-7 profile used in BilVideo-7. Low-level color, texture and shape descriptors of the Still and Moving Regions are extracted from the selected arbitrarily shaped regions, but the locations of the regions are represented by their Minimum Bounding Rectangles (MBR).

Chapter 4

System Architecture

4.1 Overview

BilVideo-7 has a distributed, client-server architecture as shown in Figure 4.1. Videos are processed offline and their MPEG-7 compatible XML representations are stored in an XML database. Users formulate their queries on BilVideo-7 clients' *visual query interface* (Section 4.4), which communicate with the BilVideo-7 *query processing server* over TCP/IP, using an XML-based query language (Section 4.5). The query processing server communicates with the XML database to retrieve the required data, executes queries and sends the query results back to the client.

This distributed architecture allows all the online components, i.e., client, query processing server and XML database, to reside on different machines; this is important for the construction of realistic, large-size systems. Furthermore, the query processing server and XML database can have a distributed architecture to allow for faster query processing and hence shorter query response times.

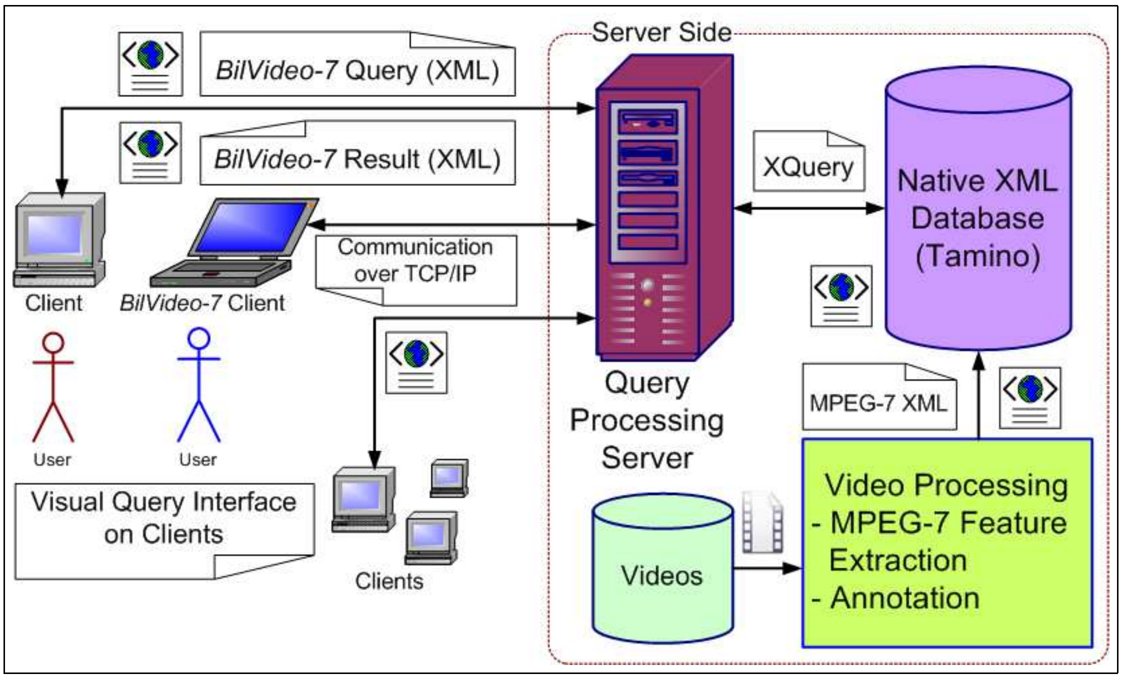


Figure 4.1: Distributed, client-server architecture of BilVideo-7.

4.2 Feature Extraction and Annotation

Videos should first undergo an offline processing stage to obtain their MPEG-7 compatible XML representations. This processing is to decompose a video into its structural and semantic building blocks (Shots, Keysegments/Keyframes, Still Regions and Moving Regions), extract the low-level MPEG-7 descriptors and annotate them with high-level semantic concepts, according to the adopted video data model. Chapter 6 focuses on video parsing, feature extraction and annotation for the MPEG-7 compatible representations of videos.

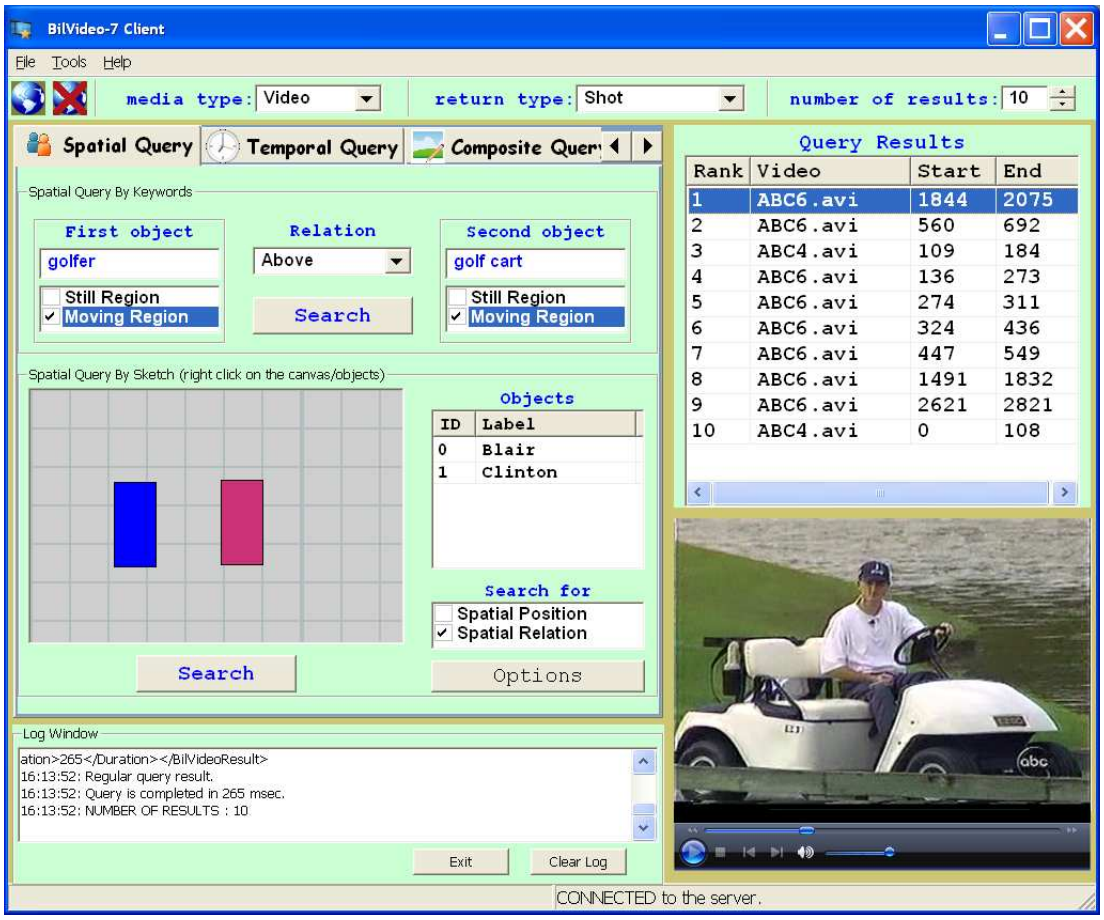
4.3 XML Database

MPEG-7 compatible representations of videos are obtained as XML files conforming to the MPEG-7 schema [31]. 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)* [51]. 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 [52]. As a result, it is more convenient and natural to use a native XML database to store the MPEG-7 descriptions. Therefore, BilVideo-7 uses a native XML database, Tamino [53], along with the standard W3C XQuery [54] to execute its queries in the database.

4.4 Visual Query Interface

Users formulate queries on BilVideo-7 clients' visual query interface, which provides an intuitive, easy-to-use query formulation interface (Figure 4.2). The graphical user interface consists of several tabs, each for a different type of query: textual query, color-texture-shape query, motion query, spatial query, temporal query, composite query, XQuery and video table of contents. As shown in Figure 4.2, 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 in the log window at the bottom left.

The user can select the media type, return type (video, video segment, shot, shot segment) and maximum number of results to be returned, from the toolbar at the top. The user can provide weights and distance/similarity thresholds for each video segment, each descriptor (e.g., CSD, HTD) and query type (e.g., color, texture, motion) in the query to have more control over query processing. Hence, the weights and thresholds can be tuned by the user according to the query results to obtain better results. Chapter 5 describes the details of how the weights and thresholds are used in query processing and in fusing the subquery results. The queries are converted into BilVideoQuery format (Section 4.5) in XML and sent to the BilVideo-7 query processing server.



media type: Video
return type: Shot
number of results: 10

First object: golfer
☐ Still Region
☑ Moving Region

Relation: Above

Second object: golf cart
☐ Still Region
☑ Moving Region

ID	Label
0	Blair
1	Clinton

Search for:
☐ Spatial Position
☑ Spatial Relation

Rank	Video	Start	End
1	ABC6.avi	1844	2075
2	ABC6.avi	560	692
3	ABC4.avi	109	184
4	ABC6.avi	136	273
5	ABC6.avi	274	311
6	ABC6.avi	324	436
7	ABC6.avi	447	549
8	ABC6.avi	1491	1832
9	ABC6.avi	2621	2821
10	ABC4.avi	0	108

```
ation>265</Duration></BilVideoResult>
16:13:52: Regular query result.
16:13:52: Query is completed in 265 msec.
16:13:52: NUMBER OF RESULTS : 10
```

CONNECTED to the server.

Figure 4.2: BilVideo-7 client visual query interface. The queries are formulated on the query formulation area on the left, result list is shown at the top right, the query results can be viewed on the media player at the bottom right and messages to the user are shown at the bottom left.

4.4.1 Video Table of Contents

Video Table of Contents (VideoToC) is a useful facility to let the user browse through the video collection in the database. The contents of each video is shown in a hierarchical tree view reflecting the structure of the MPEG-7 representation of the video in XML format. As shown in Figure 4.3, all the videos in the database are displayed at the top, along with all the high-level semantic concepts which are used to annotate the videos. The user can view the contents and list of high-level semantic concepts of each video at the bottom. The user can browse through the video 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.

4.4.2 Textual Query Interface

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 (Figure 4.4). The user can also formulate more detailed keyword-based queries to search in structured annotations.

4.4.3 Color, Texture, Shape Query Interface

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 (Figure 4.5). To be able to execute a query for the input media, 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 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 (Chapter 6) is integrated into BilVideo-7 clients. The user also specifies the type of video segments to search in, and also other query options, such as weights and thresholds for each type of descriptor.

4.4.4 Motion Query Interface

Motion Query Interface is for the formulation of Motion Activity and Motion Trajectory queries. Trajectory points are entered using the mouse (Figure 4.6). The user can optionally specify keywords for the Moving Region for which the trajectory query will be performed. Motion Activity queries can be specified by providing intensity of the motion activity or by a video segment from which the motion activity descriptor will be computed. The search can be based on motion intensity and/or spatial/temporal localization of motion intensity.

4.4.5 Spatial Query Interface

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. – Figure 4.7, top) or by sketching the minimum bounding rectangles (MBR) of objects using the mouse (Figure 4.7, bottom), and if desired, giving labels to them. It is possible to query objects based on location, spatial relations or both. The sketch-based query interface is more powerful in terms of expressing the spatial relations between the regions.

4.4.6 Temporal Query Interface

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 temporal relations such as before, after, during (Figure 4.8, top) or by sketching the temporal positions of the segments using the mouse (Figure 4.8, bottom).

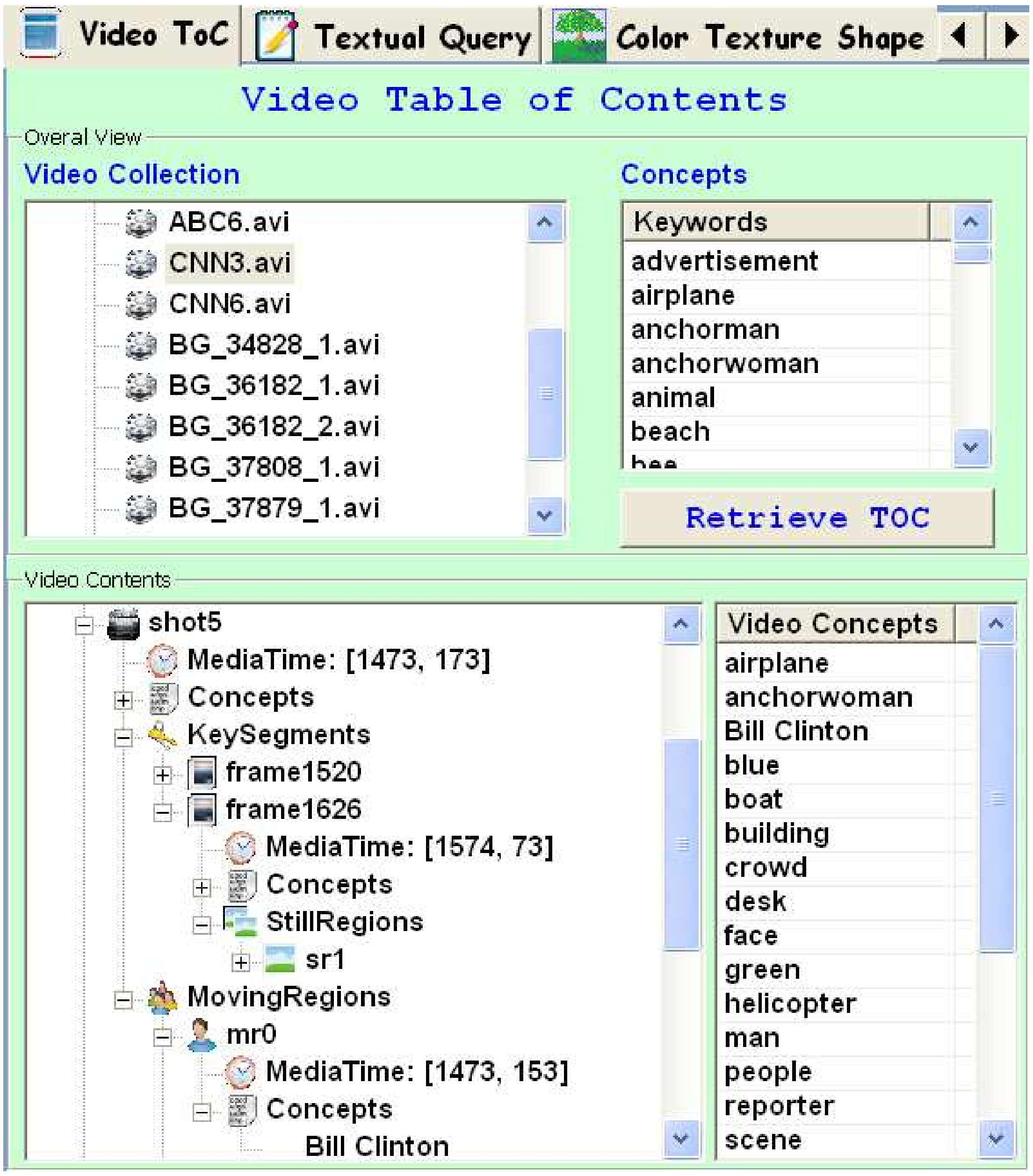
James F. Allen introduced the Allen's Interval Algebra for temporal reasoning in 1983 [55]. It defines possible relations between time intervals and provides a composition table that can be used as a basis for reasoning about temporal intervals. The temporal query interface provides the 13 base temporal relations defined by James F. Allen: *before, after, equal, meets, met-by, overlaps, overlapped-by, during, includes, starts, started-by, finishes, finished-by*. The user can select one of these relations from the pull-down list to formulate his query. The sketch-based query interface is more powerful in terms of expressing the temporal relations between the video segments.

4.4.7 Composite Query Interface

Composite Query Interface is the most powerful query interface and enables the user to formulate very complex queries easily (Figure 4.9). The query is composed by putting together any number of 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.

4.4.8 XQuery Interface

XQuery Interface is more suited to experienced users who can formulate their queries in W3C standard XQuery language to search in the database (Figure 4.10). This provides a direct access to the XML database, but XQuery provides only access to the data and cannot handle, for instance, similarity-based low-level descriptor (color, texture, shape, etc.) queries. Providing XQuery support may be useful in two ways. (1) It provides a very flexible query interface for text-based queries, or queries related to the contents of the database. (2) If a client does not use the visual query interface of BilVideo-7, it can use its own query interface and convert queries to XQuery or XML-based query language of BilVideo-7. Then, it can post-process and present the query results to the user on its own graphical user interface.



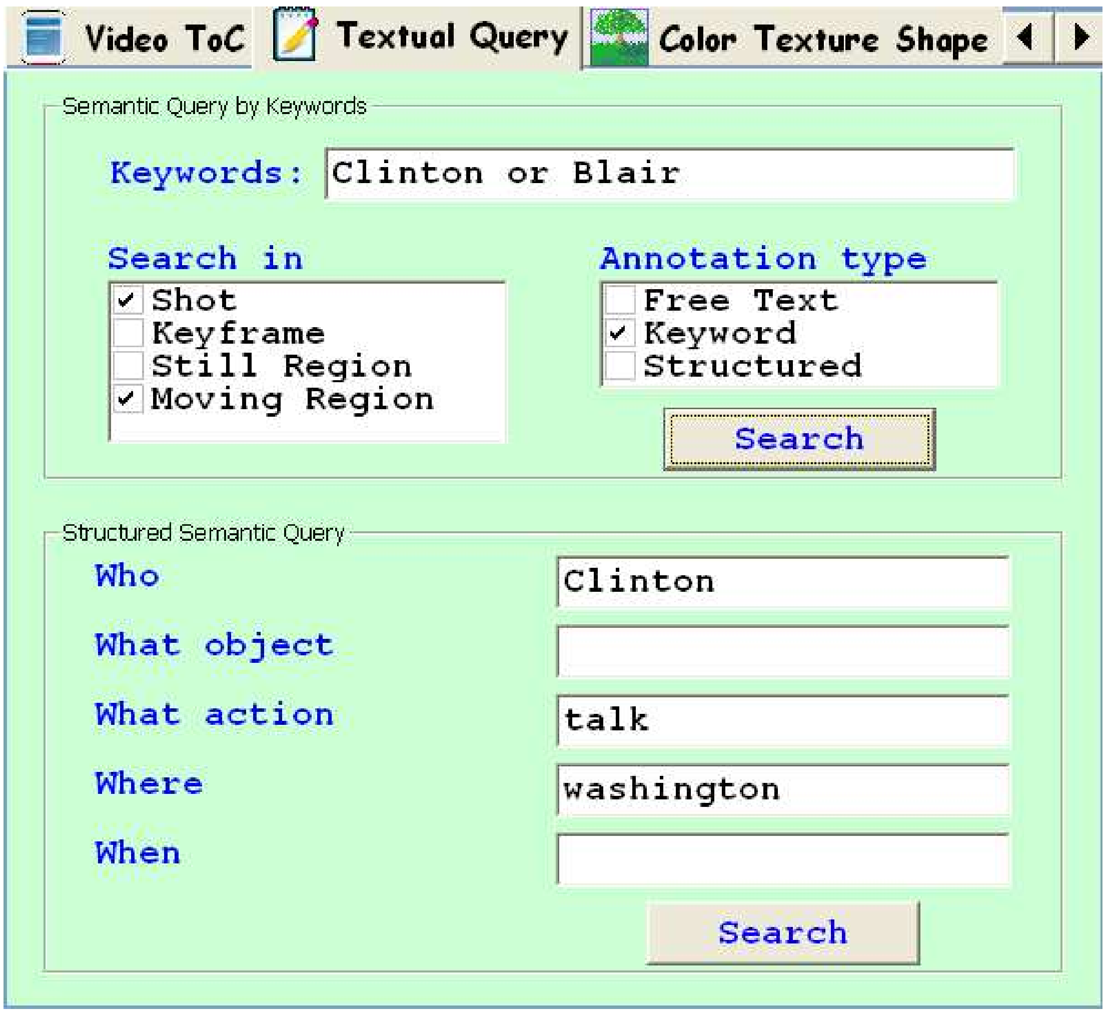
ABC6.avi
CNN3.avi
CNN6.avi
BG_34828_1.avi
BG_36182_1.avi
BG_36182_2.avi
BG_37808_1.avi
BG_37879_1.avi

Keywords
advertisement
airplane
anchorman
anchorwoman
animal
beach
bee

```
shot5
    MediaTime: [1473, 173]
    Concepts
    KeySegments
        frame1520
        frame1626
            MediaTime: [1574, 73]
            Concepts
            StillRegions
                sr1
    MovingRegions
        mr0
            MediaTime: [1473, 153]
            Concepts
                Bill Clinton
```

Video Concepts
airplane
anchorwoman
Bill Clinton
blue
boat
building
crowd
desk
face
green
helicopter
man
people
reporter
scene

Figure 4.3: Video table of contents (VideoToC) interface of a BilVideo-7 client. The whole video collection and concepts are shown at the top details of each video are shown at the bottom.



Semantic Query by Keywords

Keywords: Clinton or Blair

Search in
- [✓] Shot
- [ ] Keyframe
- [ ] Still Region
- [✓] Moving Region

Annotation type
- [ ] Free Text
- [✓] Keyword
- [ ] Structured

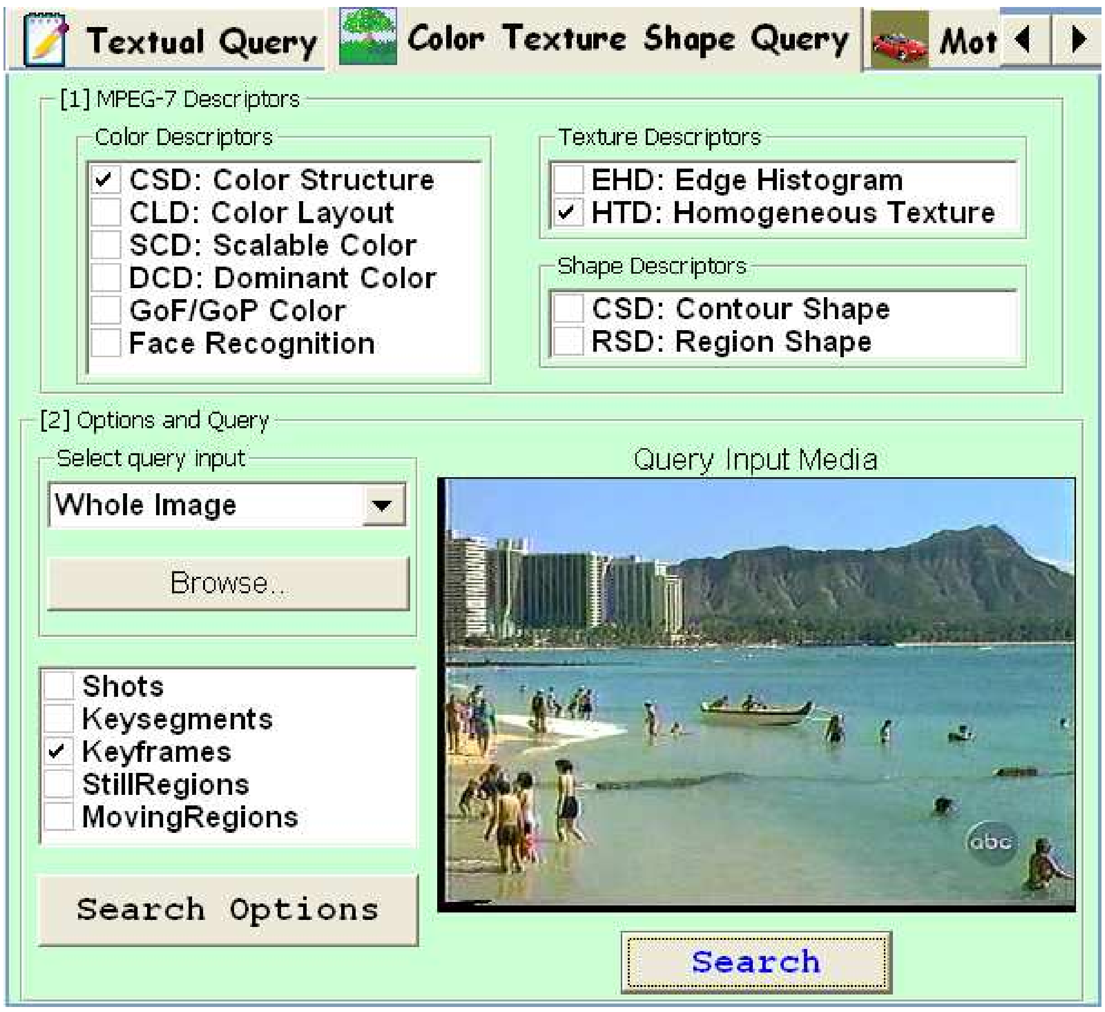
Search

Structured Semantic Query

Who	Clinton
What object	
What action	talk
Where	washington
When	

Search

Figure 4.4: BilVideo-7 client textual query interface.



[1] MPEG-7 Descriptors

Color Descriptors

- [✓] CSD: Color Structure
- [ ] CLD: Color Layout
- [ ] SCD: Scalable Color
- [ ] DCD: Dominant Color
- [ ] GoF/GoP Color
- [ ] Face Recognition

Texture Descriptors

- [ ] EHD: Edge Histogram
- [✓] HTD: Homogeneous Texture

Shape Descriptors

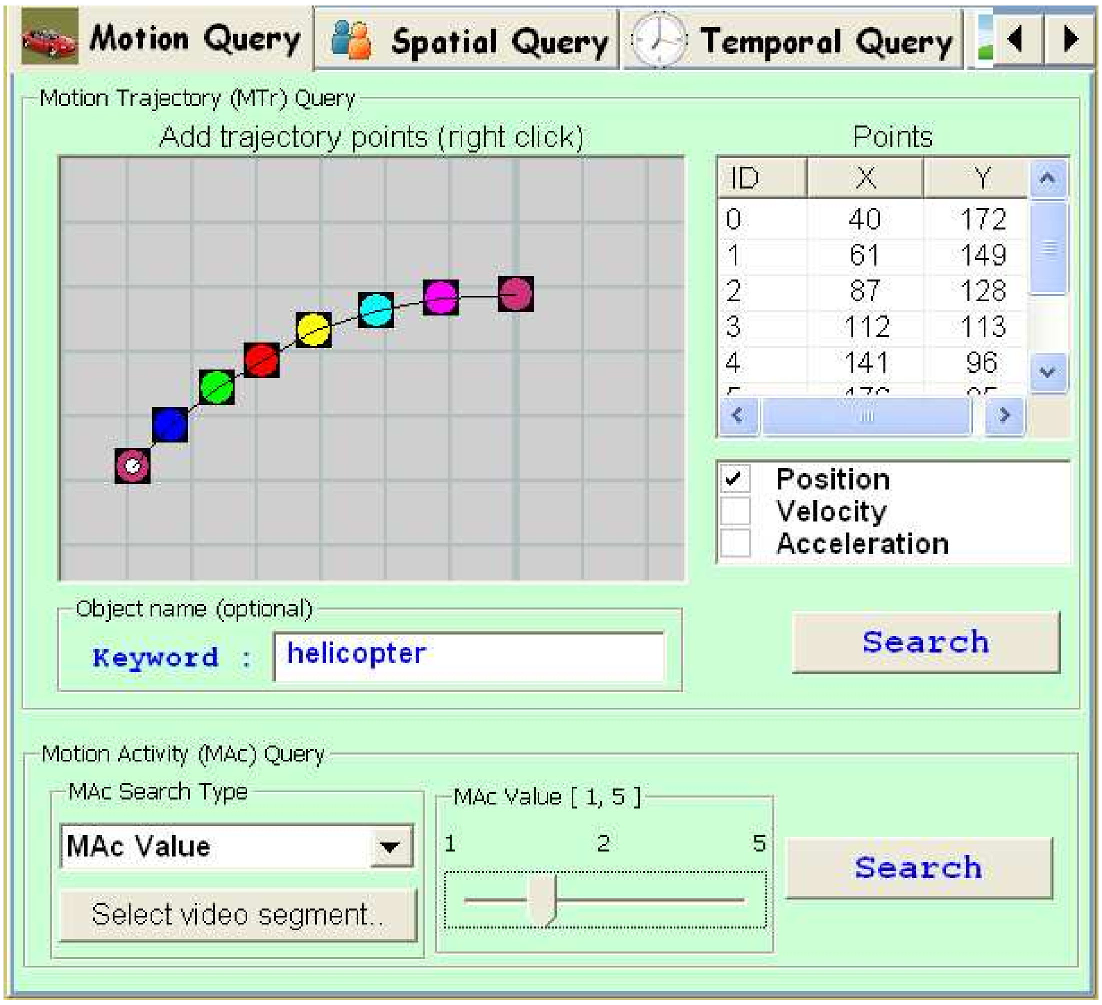
- [ ] CSD: Contour Shape
- [ ] RSD: Region Shape

[2] Options and Query

Select query input: Whole Image

- [ ] Shots
- [ ] Keysegments
- [✓] Keyframes
- [ ] StillRegions
- [ ] MovingRegions

Figure 4.5: BilVideo-7 client color, texture, shape query interface.



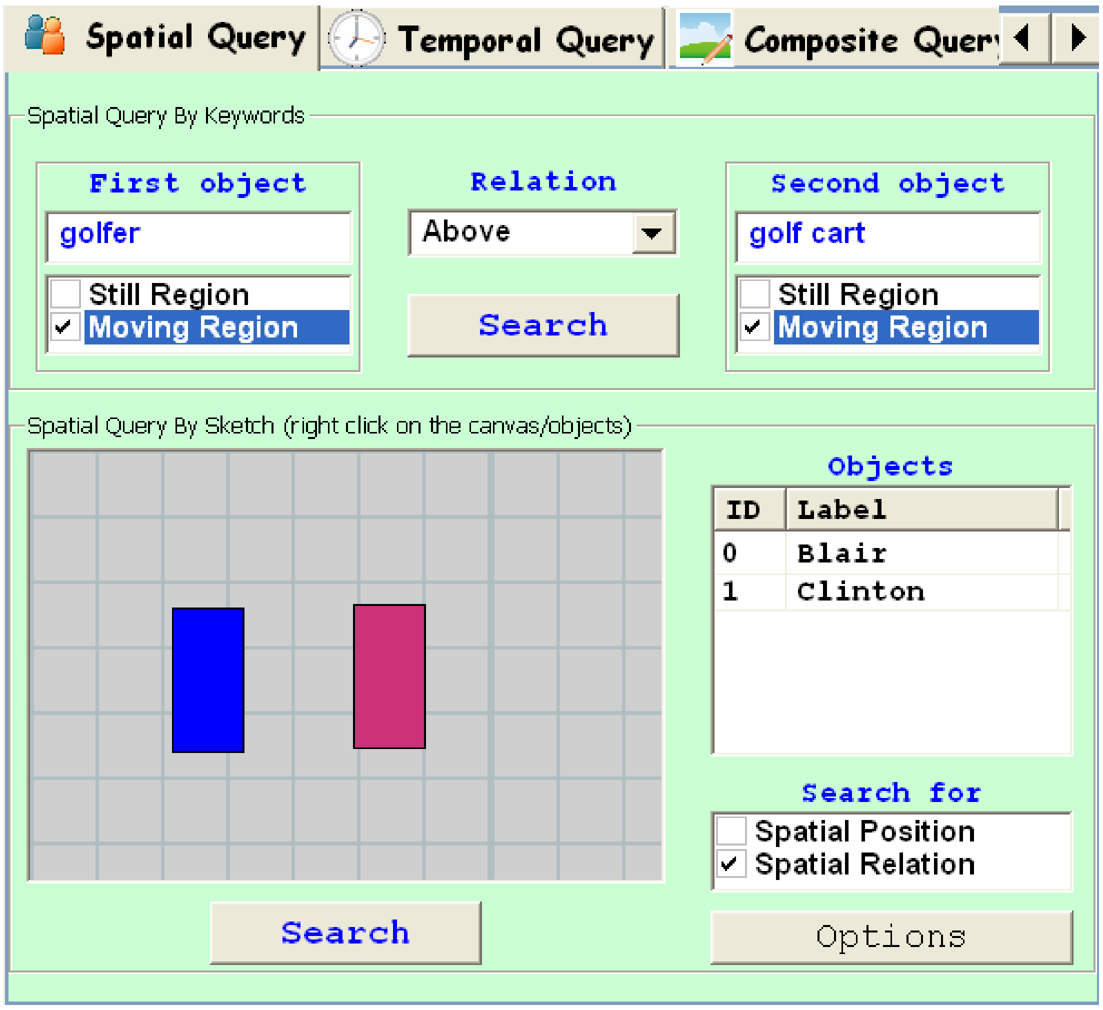
ID	X	Y
0	40	172
1	61	149
2	87	128
3	112	113
4	141	96

[✓] Position
[ ] Velocity
[ ] Acceleration

Keyword : helicopter

MAc Search Type: MAc Value

Figure 4.6: BilVideo-7 client motion query interface. Motion Trajectory queries are formulated at the top; Motion Activity queries are formulated at the bottom.


Spatial Query | Temporal Query | Composite Query

Spatial Query By Keywords

First object: golfer
- [ ] Still Region
- [✓] Moving Region

Relation: Above

Search

Second object: golf cart
- [ ] Still Region
- [✓] Moving Region

Spatial Query By Sketch (right click on the canvas/objects)

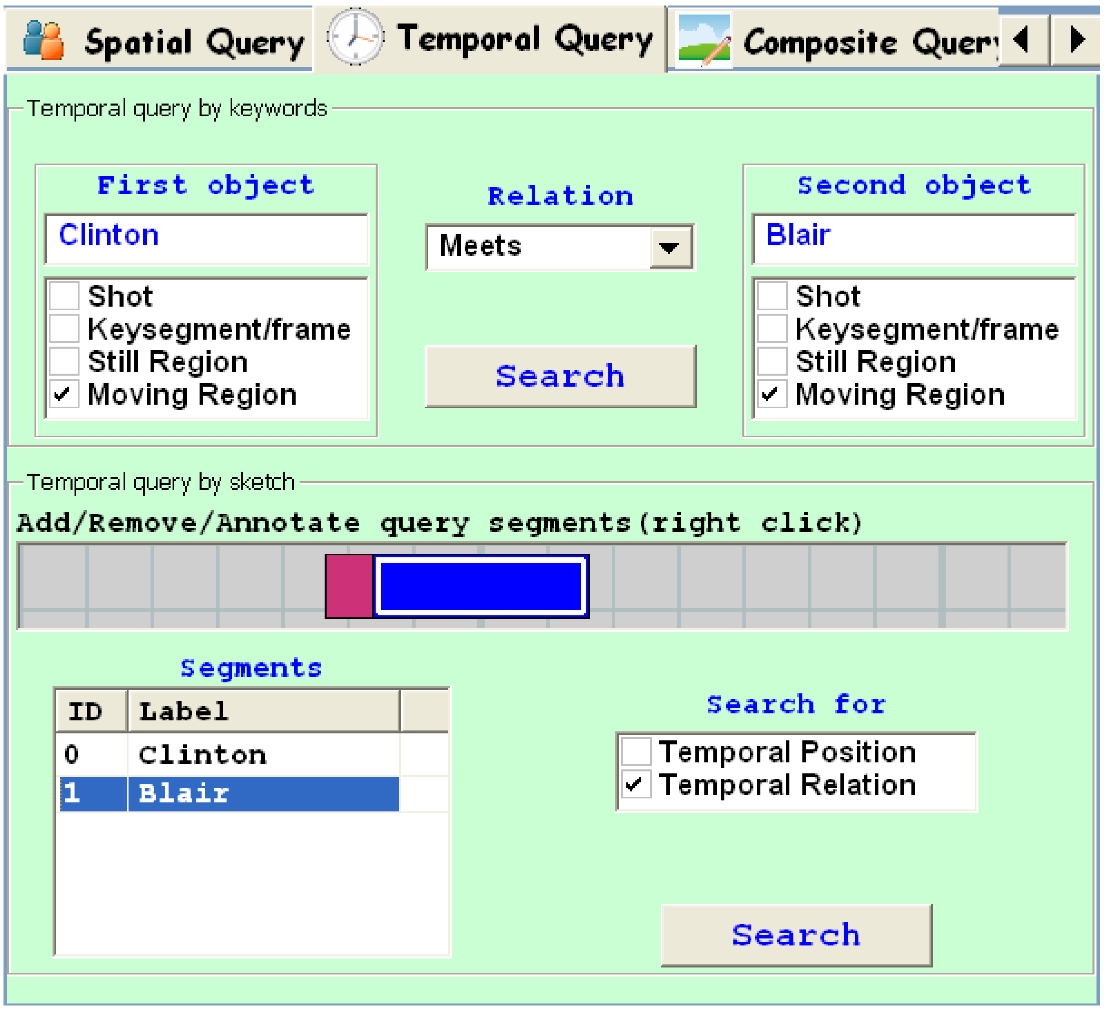
Objects

ID	Label
0	Blair
1	Clinton

Search for
- [ ] Spatial Position
- [✓] Spatial Relation

Search | Options

Figure 4.7: BilVideo-7 client spatial query interface. Spatial relations between two Still/Moving Regions can be selected from the pull-down list at the top. Sketch-based queries can be formulated at the bottom.



First object: Clinton
- [ ] Shot
- [ ] Keysegment/frame
- [ ] Still Region
- [✓] Moving Region

Relation: Meets

Second object: Blair
- [ ] Shot
- [ ] Keysegment/frame
- [ ] Still Region
- [✓] Moving Region

Add/Remove/Annotate query segments(right click)

ID	Label
0	Clinton
1	Blair

Search for
- [ ] Temporal Position
- [✓] Temporal Relation

Figure 4.8: BilVideo-7 client temporal query interface. Temporal relations between video segments can be selected from the pull-down list at the top. Sketch-based queries can be formulated at the bottom.

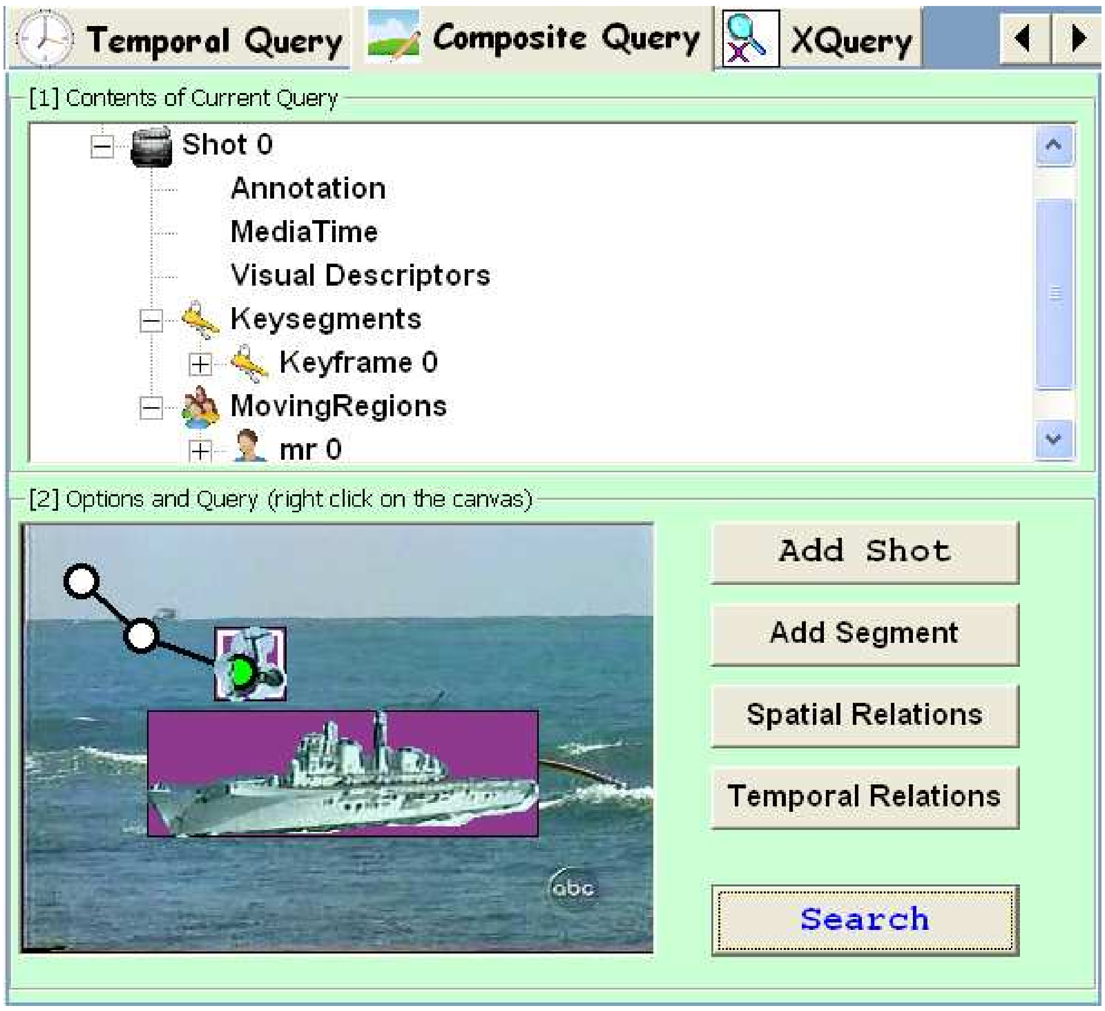
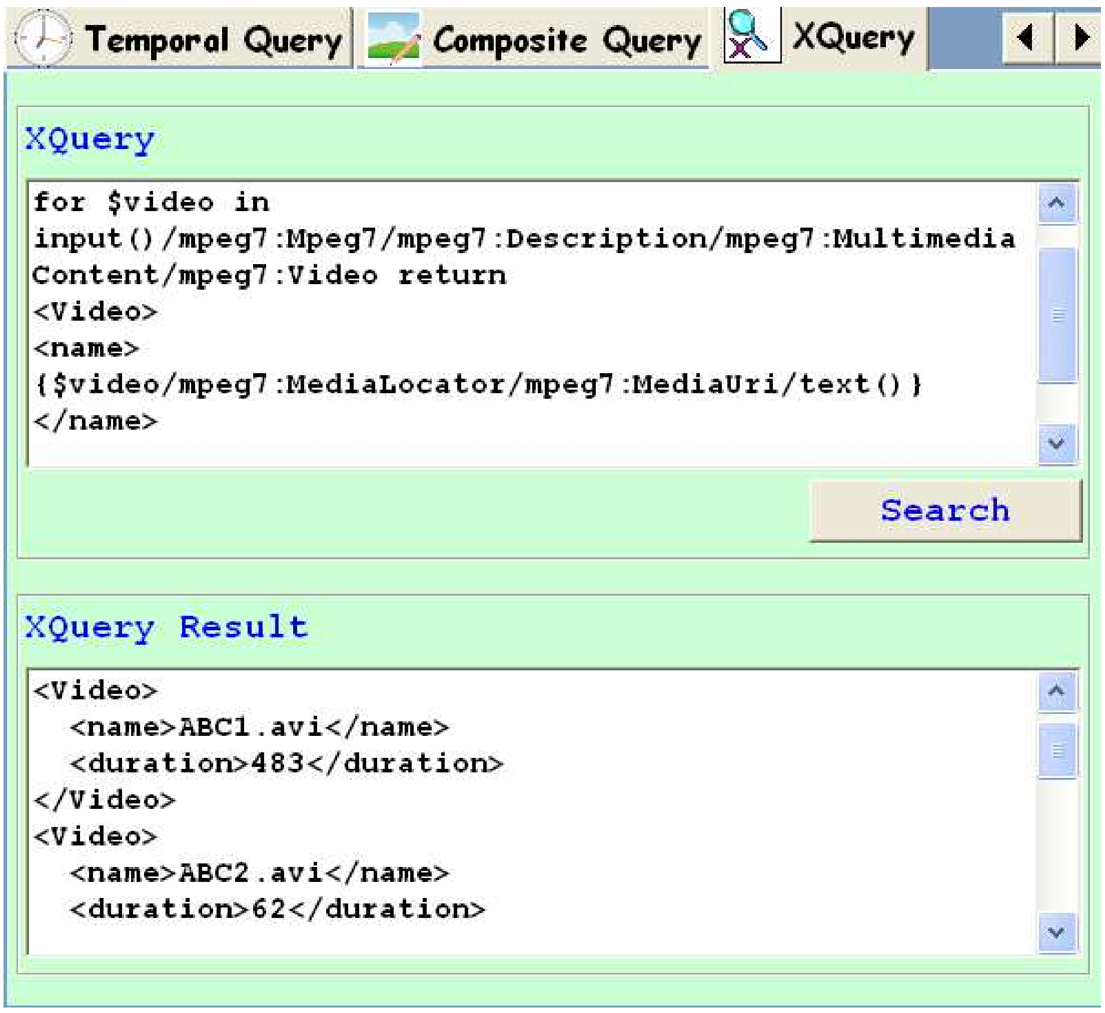


Figure 4.9: BilVideo-7 client composite query interface.



```
for $video in
input()/mpeg7:Mpeg7/mpeg7:Description/mpeg7:Multimedia
Content/mpeg7:Video return
<Video>
<name>
{$video/mpeg7:MediaLocator/mpeg7:MediaUri/text()}
</name>
```

```
<Video>
  <name>ABC1.avi</name>
  <duration>483</duration>
</Video>
<Video>
  <name>ABC2.avi</name>
  <duration>62</duration>
```

Figure 4.10: BilVideo-7 client XQuery interface.

4.5 XML-based Query Language

We need a query language for the communication between the clients and the server. Since MPEG-7 uses XML as its Description Definition Language (DDL), and video representations in XML format are kept in a native XML database, it is most appropriate to use an XML-based query language. This language is transparent to the user, since queries are formulated on the visual query interface. However, any client with its own query formulation interface can convert its queries to this format and execute the queries on the system.

Current version of BilVideo-7 does not support MPQF query language (Section 2.2.7) since it is not possible to formulate some of the BilVideo-7 queries in MPQF (e.g., spatial queries by location). The format of the BilVideo-7's XML-based query language is as follows.

```
<BilVideoQuery attributes='general query options'>

   <VideoSegment attributes='subquery options'>
      <Textual attributes='subquery options'>SubQuery</Textual>
      <Location attributes='subquery options'>SubQuery</Location>
      <Color attributes='subquery options'>SubQuery</Color>
      <Texture attributes='subquery options'>SubQuery</Texture>
      <Shape attributes='subquery options'>SubQuery</Shape>
      <Motion attributes='subquery options'>SubQuery</Motion>
   </VideoSegment>

   <VideoSegment attributes='subquery options'>
      SubQuery
   </VideoSegment>
   ...
   ...
   <Spatial attributes='subquery options'>SubQuery</Spatial>

   <Temporal attributes='subquery options'>SubQuery</Temporal>

   <TOC attributes='subquery options'>SubQuery</TOC>

   <XQUERY>SubQuery</XQUERY>

</BilVideoQuery>
```

As shown above, the query may consist of a list of VideoSegments along with their descriptors and Spatial and/or Temporal queries, if any, or a single TOC (Video Table of Contents) or XQuery query. The Spatial and Temporal queries references the VideoSegments already described by their unique segment IDs. Note that our XML-based query language is very similar to MPQF.

4.6 Query Processing Server

The query processing server accepts incoming clients and replies to their queries. First, it parses the queries that are in XML format into subqueries which are composed of a single query video segment and a single descriptor, e.g., a Keyframe with Color Structure Descriptor (CSD), a Moving Region with Region Shape Descriptor (RSD). Then, it retrieves the required data from the XML database using XQuery, executes each subquery and fuses the results of all subqueries to obtain a single list of video segments as the query result. Finally, it ranks the video segments in the query result according to their similarities to the query and sends the result back to the originating client. Chapter 5 is dedicated to discuss the query processing in detail.

Chapter 5

Query Processing

This chapter focuses on query processing on the BilVideo-7 Query Processing Server. We first describe the multi-threaded query execution architecture, then give the details of how different types of queries are processed, and finally explain the subquery result fusion strategy that enables complex queries.

5.1 Overview

BilVideo-7 clients connect to the query processing server to execute their queries. The query processing server is a multi-threaded server side component that listens to a configured TCP port, accepts incoming clients and processes their queries (Figure 4.1). Clients send their queries in the XML-based *BilVideoQuery* format (see Section 4.5) and receive query results in XML-based *BilVideoResult* format, which contains a list of video segments (video name, start time, end time) in ranked order.

Definition 5.1.1 (Simple Query)**.** A query is a *simple query* if it contains only one query segment with only one descriptor.

For example, a Shot with GoF, a Keyframe with HTD, a Moving Region with CSD queries are all simple queries.

Definition 5.1.2 (Composite Query)**.** A query is a *composite query* if it contains multiple query segments or multiple descriptors.

For example, a Shot with GoF + MAc, a Keyframe with SCD + EHD + text, a Still Region and a Moving Region with spatial relation queries are all composite queries. The query in Figure 5.2 is also a composite query.

5.2 Multi-threaded Query Execution

The query processing server receives queries in XML-based BilVideoQuery format from the clients and parses each incoming query into subqueries, which are simple queries (see Definition 5.1.1). Then, it executes the subqueries in a multi-threaded fashion, with one thread for each type of subquery, as shown in Figure 5.3. Queries with the same subquery type (e.g., color) 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 the queue of *Color Query Executor* thread and executed in this order. This is the current implementation in BilVideo-7, however, other possibilities of multi-threaded query processing also exist, such as a separate thread for each type of descriptor, in which case the number of threads will be much higher.

One XQuery is formed and executed on the XML database for each subquery, consisting of a single video segment and a single descriptor (e.g., Keyframe with CSD). The XML database returns the XQuery results in XML format, which are parsed to extract the actual data (the descriptors). The descriptors undergo further processing for distance/similarity computation to obtain the subquery result. If there are spatial relation queries between Still/Moving Regions, and/or temporal relation queries between video segments (Shot, Keyframe, Still/Moving Region), they are executed after the execution of the subqueries related to the high/low-level descriptions of the video segments. Subquery results must be fused to obtain the final query result; this is discussed in Section 5.3.

An illustrative query example, as formulated on the client visual query interface,

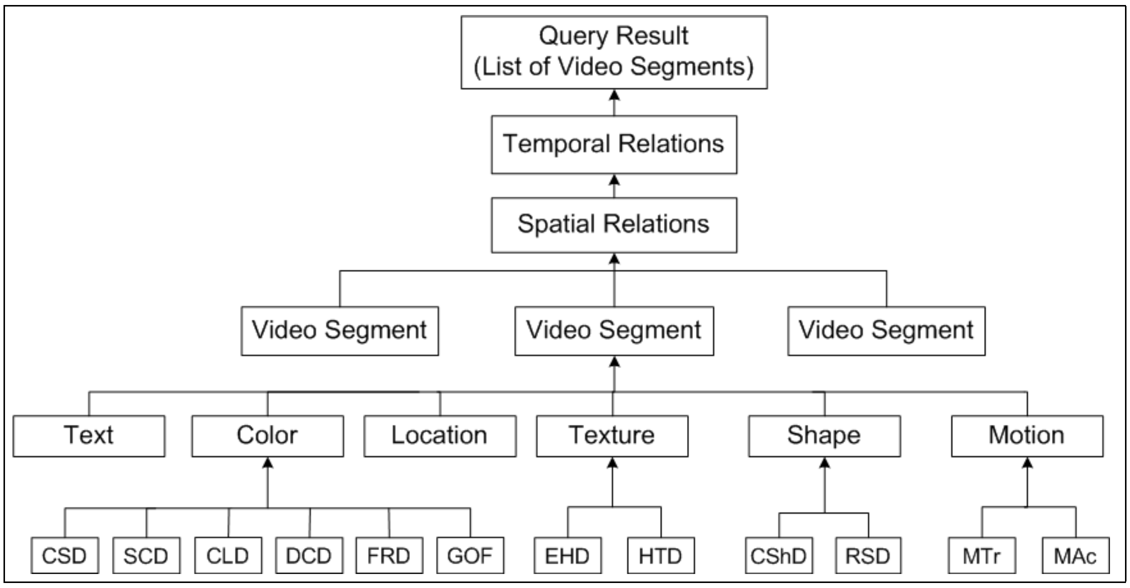


Figure 5.1: Subquery results are fused in a bottom-up manner. Each node has an associated weight and threshold. The similarity of a video segment at each node is computed as the weighted average of the similarities of its children.

is shown in Figure 5.2. This is a composite query having three video segments (one Keyframe, one Still Region and one Moving Region) with various descriptors. When the user presses the “Search” button on the Composite Query Interface (Figure 4.9), the specified descriptors are extracted from the Keyframe, Still Region and Moving Region and using the other query options (weights, thresholds, etc.) the query is assembled into an XML string and sent to the server. The query processing server parses this query into 6 (simple) subqueries: (1) Still Region with HTD, (2) Keyframe with DCD, (3) Keyframe with CSD, (4) Keyframe with text, (5) Moving Region with CSD, (6) Moving Region with MTr. Then, the query processing proceeds as described in the previous paragraph and in the following sections.

5.2.1 Similarity Computation

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 similarity between the descriptors in a query and the descriptors 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., CSD for Keyframes) and computes their distances to the query.

The distance measures suggested by MPEG-7 authors for each descriptor are implemented in MPEG-7 XM Reference Software [33] but they are not normative, i.e., any other suitable distance measure can also be used without breaking the MPEG-7 compatibility of the system. An evaluation of distance measures for a set of MPEG-7 descriptors [56] shows that although there are better distance measures such as pattern difference and Meehl index, the distance measures recommended by MPEG-7 are among the best. Therefore, we adapted the distance measures from the XM Reference Software implementation. In the following sections, we summarize the adapted distance metrics. More detailed information on MPEG-7 distance measures can be found in [6, 33, 56].

The user specifies a set of weights and thresholds at query formulation time. 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.

5.2.2 VideoTOC and Textual Query Processing

Video table of contents (VideoTOC) interface (Figure 4.3) requests (1) the video collection and high-level semantic concepts in the XML database, and (2) the contents of a video, which are retrieved from the database with XQuery and send back to the client in XML format.

Textual queries can be handled by the database. MPEG-7 allows the specification of confidence scores for text annotations which can be taken as the similarity value during query processing if the annotation matches with the query.

5.2.3 Color, Texture, Shape Query Processing

Low-level color, texture and shape queries may originate either from the color, texture, shape query interface (Figure 4.5) or the composite query interface (Figure 4.9). These queries are executed by the respective color, texture and shape execution threads, which are responsible for executing a simple subquery (e.g., Keyframe with CSD) at a time. The distances between the descriptor in the query and the descriptors in the database should be computed using suitable distance measures.

In the following, we briefly describe the distance measures adapted from MPEG-7 XM software for color, texture and shape descriptors. 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, Region Shape descriptors.

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

The distance between two Color Layout descriptors, $Q = \{QY, QCb, QCr\}$ and $D = \{DY, DCb, DCr\}$, 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 $d(c_q,c_d) = \left\| c_q - c_d \right\|$ 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 two Edge Histogram descriptors 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

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

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.

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

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 changes in its descriptors throughout the shot, the distance is computed for all the instances and the minimum is taken.

5.2.4 Motion Query Processing

Motion query execution thread handles the Motion Activity and Motion Trajectory queries. 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.

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

with similar definitions for d_S and d_A [6].

5.2.5 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 them. 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 [15] because of its efficiency and robustness. First, the vectors connecting the center points of objects' MBRs, $\mathbf{Q_{xy}}$ and $\mathbf{D_{ij}}$, are computed as shown in Figure 5.4. Then, the pairwise spatial similarity is computed by the cosine of the angle θ between the vectors $\mathbf{Q_{xy}}$ and $\mathbf{D_{ij}}$, using vector dot product:

$$d(\mathbf{Q_{xy}},\mathbf{D_{ij}}) = \cos\theta = \frac{\mathbf{Q_{xy}} \cdot \mathbf{D_{ij}}}{|\mathbf{Q_{xy}}|\,|\mathbf{D_{ij}}|},\ \ 0 \le \theta \le \pi,\ \ -1 \le d(\mathbf{Q_{xy}},\mathbf{D_{ij}}) \le +1$$

The output value is in the range [-1, +1], with +1 indicating identical spatial relation and -1 opposite spatial relation. In Figure 5.4, 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.

The text-based spatial queries (*right, left, above, below, etc.*) are executed in the same way, by converting each spatial relation query to a unit vector (Figure 5.4, left). For instance, Q_x *right* Q_y (Q_x is to the right of Q_y) query is converted to a query vector $\mathbf{Q_{xy}} = [-1, 0]$, from Q_x to Q_y.

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 5.5 illustrates the spatial relation query processing between a Still Region and a Moving Region.

5.2.6 Temporal Query Processing

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.

5.2.7 Composite Query Processing

In Section 5.1, we defined the composite query as a query that contains multiple query segments or multiple descriptors. When received by the query processing server, the composite queries are decomposed into a set of 'simple' subqueries and executed separately. The subquery results are fused (similar to late fusion [57]) in a bottom-up manner as explained below, in Section 5.3. This flexible query processing architecture enables the easy formulation of complex queries.

5.3 Fusion of Subquery Results

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 5.1 and illustrated in Figure 5.6. Referencing Figure 5.1, each node in the tree has an associated weight and a threshold, which can be specified by the user during query formulation.

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. This is similar to the sum rule of combining classifier outputs, which is shown to be more resilient to errors compared to, for instance, the product rule [58, 59]. Moreover, this simple approach provides a very flexible query processing architecture to support complex multimodal queries seamlessly and to add new modalities and descriptors easily.

To illustrate the fusion process, consider a composite query consisting of a Keyframe with color (CSD and DCD), texture (EHD and HTD) and text-based semantic (keyword *golf green*) descriptors. The query processor parses this query into 5 subqueries (CSD, DCD, EHD, HTD and text), executes each and produces 5 lists of Keyframes from database with similarity values. Then, it fuses color (CSD, DCD) and texture (EHD, HTD) subquery results to obtain the color and texture similarities of each Keyframe.

$$
\begin{aligned}
s_{i,Color} &= \frac{w_{Keyframe,CSD}\, s_{i,CSD} + w_{Keyframe,DCD}\, s_{i,DCD}}{w_{Keyframe,CSD} + w_{Keyframe,DCD}} \\
s_{i,Texture} &= \frac{w_{Keyframe,EHD}\, s_{i,EHD} + w_{Keyframe,HTD}\, s_{i,HTD}}{w_{Keyframe,EHD} + w_{Keyframe,HTD}}
\end{aligned}
$$

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 and text. We fuse these 3 lists to obtain the final list of Keyframes.

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

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.

5.4 Discussion

Multithreading provides some degree of parallel execution on multi-core processors, and hence reduces query execution time. Query processing for a multimedia retrieval system is computationally costly. To keep the response time of the system at interactive rates, especially for large databases, a truly parallel system should be employed. 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 multimodal query processing and bottom-up subquery result fusion architecture make it possible to add new modalities easily. For instance, current BilVideo-7 implementation can easily be extended to support queries related to the audio content of the videos. In such a case, audio and visual segment queries can be executed first, and final list of video segments can be merged after temporal query processing (if any).

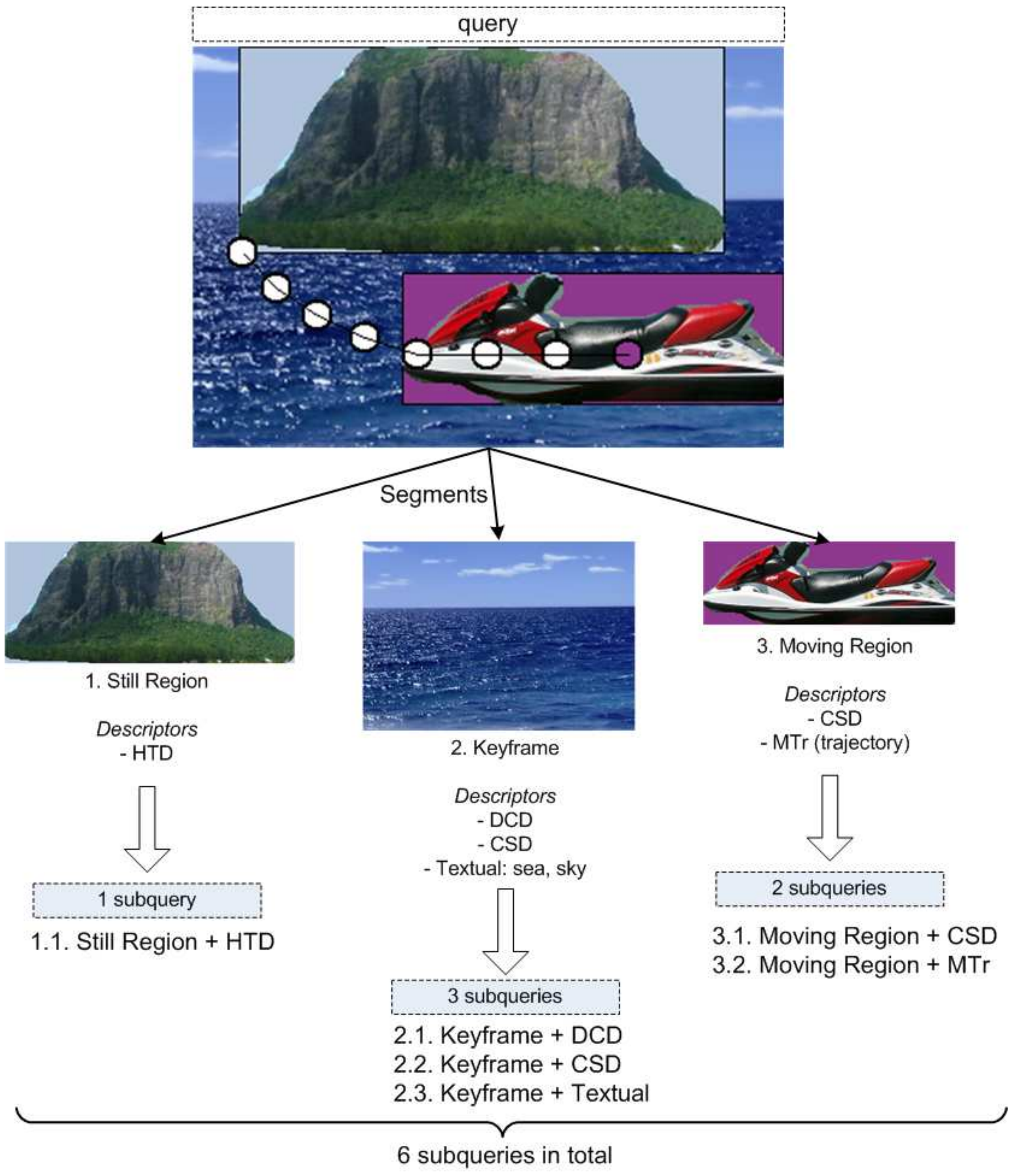


Figure 5.2: Interpretation of the input queries on the query processing server. Composite queries are parsed into several subqueries, which are all simple queries.

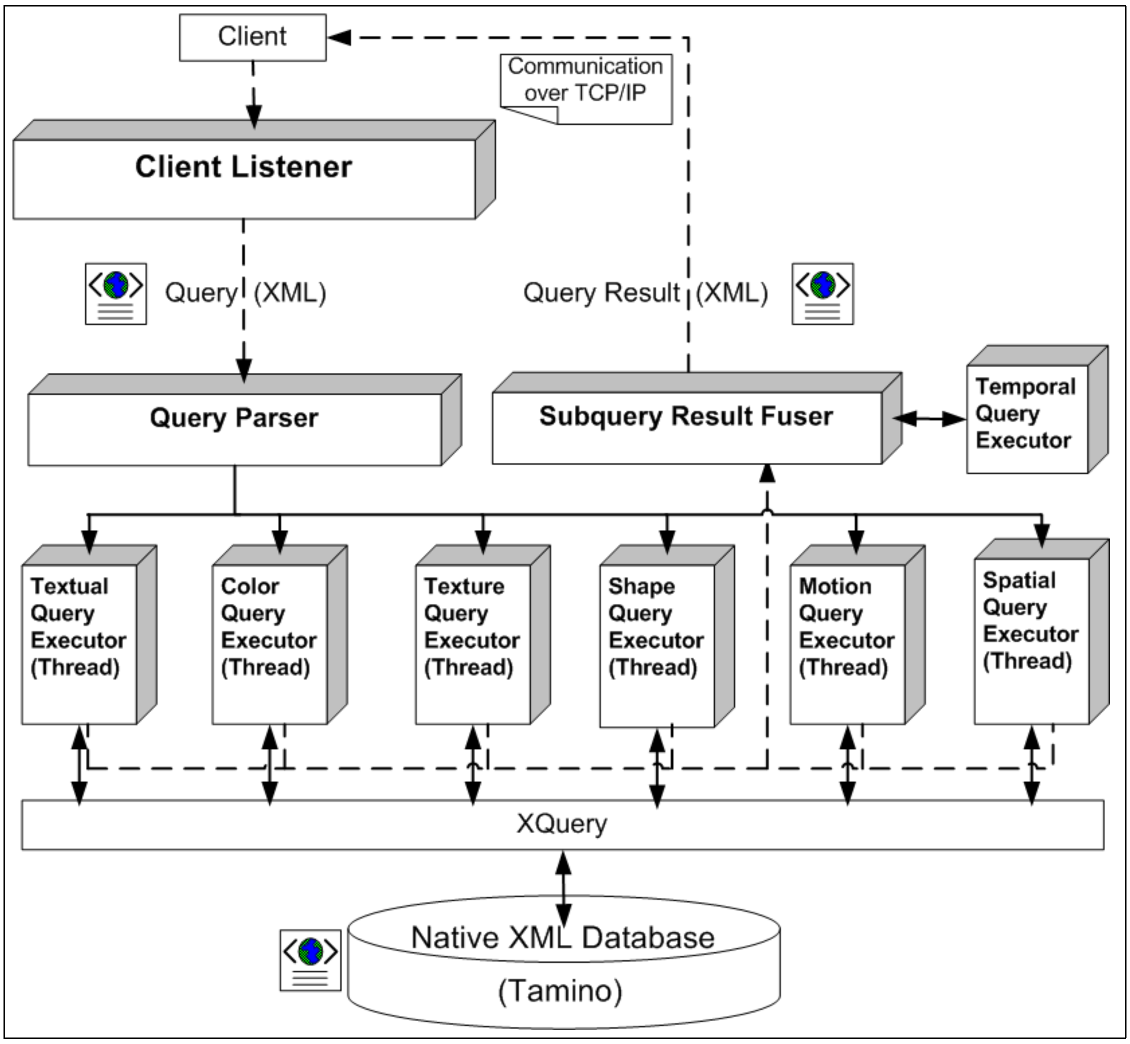

Client
Communication over TCP/IP
Client Listener
Query (XML)
Query Result (XML)
Query Parser
Subquery Result Fuser
Temporal Query Executor
Textual Query Executor (Thread)
Color Query Executor (Thread)
Texture Query Executor (Thread)
Shape Query Executor (Thread)
Motion Query Executor (Thread)
Spatial Query Executor (Thread)
XQuery
Native XML Database
(Tamino)

Figure 5.3: 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. Subquery results are fused in a bottom-up manner (Figure 5.1) and the final result is returned to the client.

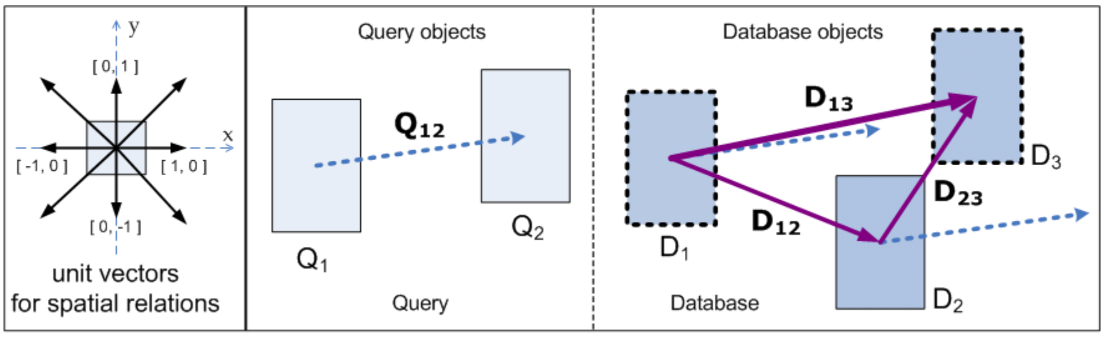


Figure 5.4: Spatial query processing by vector dot product between the vectors connecting centers of objects' MBRs. In the sketch-based spatial query in the middle, the query is represented with the vector $\mathbf{Q_{12}}$, from the center of object Q_1 to the center of object Q_2. The spatial relation 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.

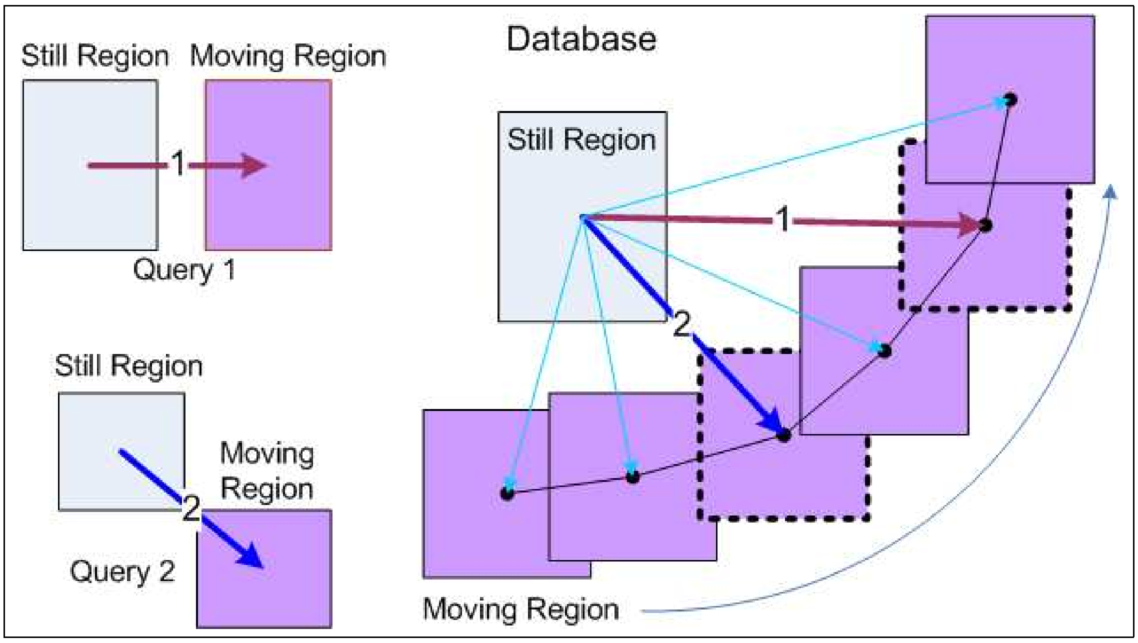


Figure 5.5: Spatial query processing between Still and Moving Regions.

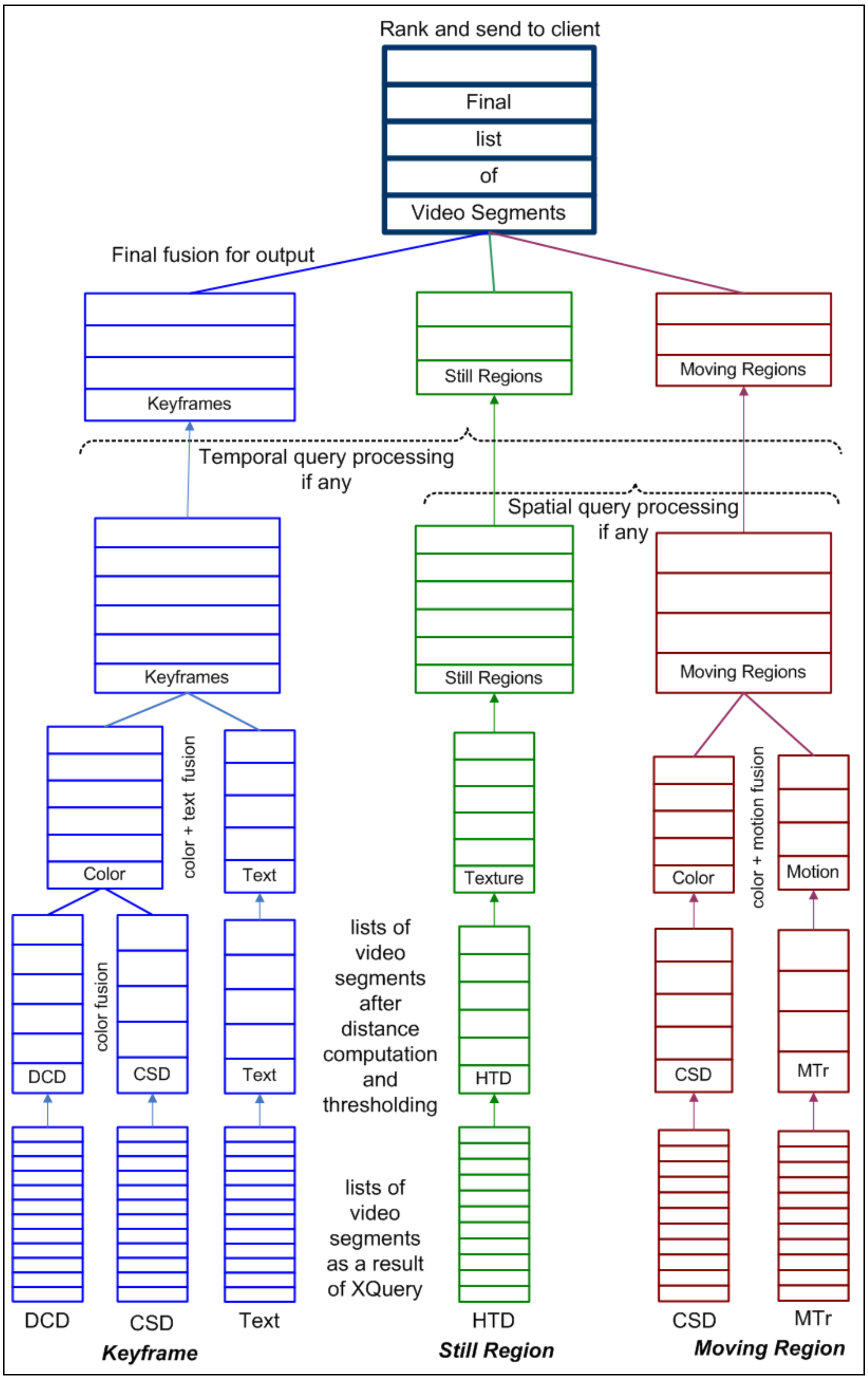


Figure 5.6: Fusion of subquery results illustrated, for the query in Figure 5.2.

Chapter 6

From Videos to MPEG-7 Representations

This chapter discusses the issues related to how to obtain the MPEG-7 compatible XML representations of the videos. We first present our video feature extraction and annotation tool, BilMAT, and then explore possible ways of automatizing the process of video-to-MPEG-7 conversion as much as possible.

6.1 Introduction

Videos should undergo an offline processing stage to obtain their MPEG-7 compatible XML representations which are then stored in the database. This processing (1) decomposes the videos into its structural and semantic building blocks (Shots, Keyframes, Still Regions and Moving Regions) according to the adopted data model, (2) extracts low-level descriptors from them and (3) annotates them with high-level semantic concepts. The extraction of low-level descriptors (color, texture, shape, motion, etc.) from the video segments is done automatically. What remains is the decomposition into and annotation of video segments, which can be achieved in one of the following three modes of video processing.

- Manual processing: both video decomposition and annotation are done manually.
- Semi-automatic processing: some human assistance is required for either video decomposition or annotation.
- Automatic processing: video decomposition and annotation are performed automatically, without the need for any human help.

6.2 Semi-automatic Video Parsing

The term *video parsing* is borrowed from the work of Tu *et al.* [60], where it is used as *image parsing*: decomposing an image into its constituent visual patterns and producing a scene representation in a "parsing graph" similar to parsing sentences in speech and natural language. Likewise, video parsing is defined as the process of decomposing a video into its constituent parts according to the video data model adopted. In our case, the constituent parts are Shots, Keyframes, Still Regions and Moving Regions, according to our data model described in Chapter 3.

There are several annotation tools in the literature. LabelMe [61, 62] is a web-based image annotation tool to label objects in images by specifying the objects' boundaries with polygons and providing keyword labels. The resulting labels are stored on the LabelMe server in XML format. Similarly, LabelMe video [63, 64] is a web-based video annotation tool to label objects and events. The aim is to produce large amounts of annotated ground truth image/video data that can be used in training for the recognition of objects, scenes, actions, etc. The Graphical Annotation Tool (GAT) [65] is a region-level annotation tool for images, producing MPEG-7/XML outputs. Recently, Amazon Mechanical Turk [66, 67] has become a popular way of obtaining high-quality ground truth annotations at a low cost, according to the annotation protocol defined by the initiator.

MuLVAT [68] is a video annotation tool, which uses structured knowledge, in the form of XML dictionaries, combined with a hierarchical classification scheme to attach semantic labels to video segments at various level of granularity. There are also MPEG-7 compatible tools (VideoAnnEx [45], Caliph & Emir [50], M-OntoMat-Annotizer [46], ERIC7 [49]) which were discussed in Chapter 2. None of these tools is sufficient for the task of MPEG-7 compatible video feature extraction and annotation in BilVideo-7, which lead us to develop a new tool, BilMAT, described next.

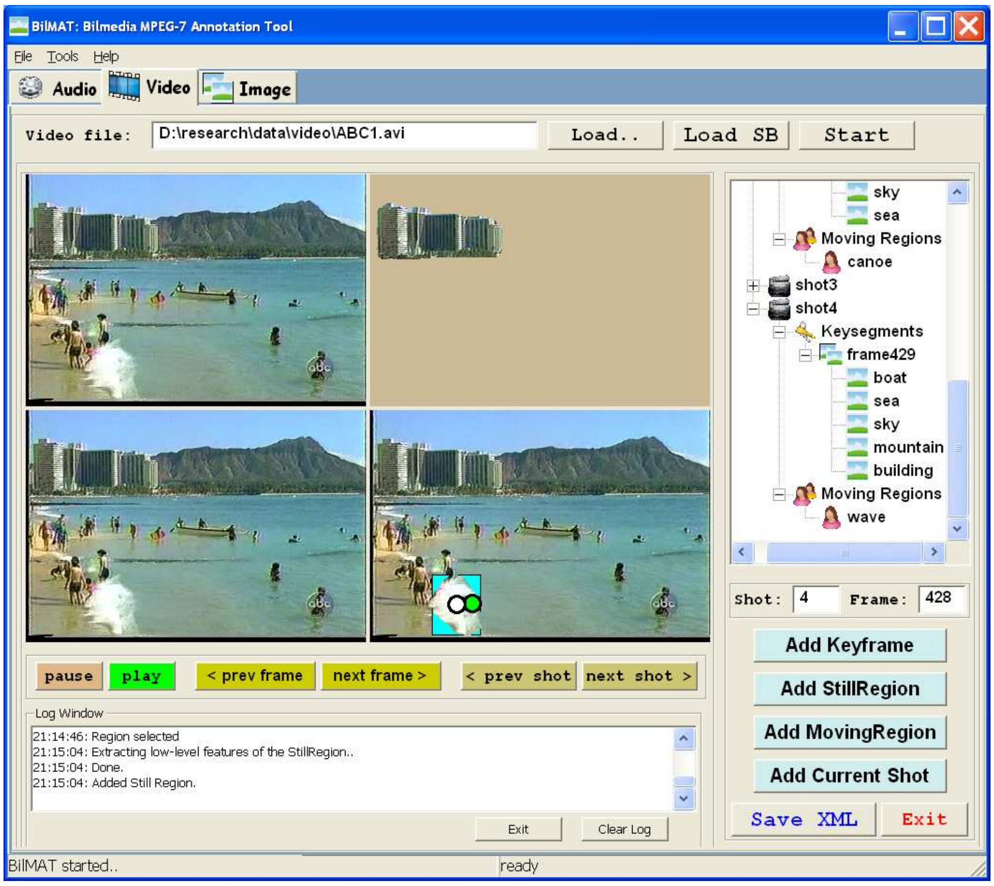
6.2.1 BilMAT for Semi-automatic Video Processing

BilMAT is short for **Bil**media **M**PEG-7 **A**nnotation **T**ool. It is designed to be a generic multimedia feature extraction and annotation tool to support image, audio and video data. In this work, we focus on video data only. The required manual work to parse a video consists of selecting the video segments and annotating them with a set of high-level semantic concepts.

Figure 6.1 shows a snapshot of BilMAT while processing a video. In the figure, 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 shown at the bottom right. Selected video segments along with their annotations are shown on the right in a hierarchical tree view reflecting the structure and showing the contents of the video.

The user loads a video along with its shot boundary information, i.e., the start and end times of each Shot; selects which descriptors to be used to represent each type of video segment, and then processes the video on a shot-by-shot basis. The MPEG-7 visual descriptors (color, texture, shape, motion, localization) for the selected video segments are computed by the tool, using the MPEG-7 feature extraction library of BilVideo-7, adapted from MPEG-7 XM Reference Software [33]. Some parts of the feature extraction library, along with executables, are publicly available at the BilVideo-7 website [9].

- *Shot Processing.* Since shot boundaries are already loaded, the only manual work is to enter the annotations for the Shot. The tool computes the low-level descriptors selected by the user, such as Group-of-Frame and Motion Activity descriptors, and adds the current Shot to the list of processed Shots and displays it in the tree view



Video file: D:\research\data\video\ABC1.avi

Shot: 4 Frame: 428

```
21:14:46: Region selected
21:15:04: Extracting low-level features of the StillRegion..
21:15:04: Done.
21:15:04: Added Still Region.
```

BilMAT started.. ready

Figure 6.1: BilMAT: **Bil**media **M**PEG-7 **A**nnotation **T**ool.

on the right. Then, the user may proceed to further process the Shot by adding Keyframes, Still and Moving Regions that reside in the Shot.

- *Keyframe Processing.* The user may select one or more Keyframes from the Shot and annotate them with high-level descriptors. The tool computes the user-specified low-level descriptors, such as Color Structure and Homogeneous Texture descriptors, and adds the Keyframes to the Shot and displays them on the right.

- *Still Region Processing.* The user may select, annotate and add a set of Still Regions for each Keyframe. The user-specified low-level descriptors, such as Dominant Color and Scalable Color descriptors, are extracted from the selected regions.

- *Moving Region Processing.* Finally, the user may also select, annotate and add a set of Moving Regions, i.e., salient objects, for each Shot. The visual appearance, position and MBR of the Moving Regions may change throughout the Shot, therefore, when a change occurs in these attributes they should be updated by the user. The user-specified low-level descriptors, such as Region Shape and Color Structure descriptors, are extracted from the selected regions as the visual appearance is updated. Faces in the Shot are also represented by Moving Regions with Face Recognition Descriptor. The tool has the capability to detect and track all the faces in the Shot, but the high-level annotations should still be provided by the user.

When processing the video is completed, the user may annotate the whole video (provide annotations related to the content of the whole video) and save the MPEG-7 representation into an XML file. This manual processing, though tedious, provides an accurate representation of the video, which is crucial for the effective retrieval of the video content.

6.3 Towards Automatic Video Parsing for MPEG-7 Representation

It is not practical to use a manual tool to parse and annotate large amounts of video. Moreover, due to human subjectivity, the video representations obtained by different people may be quite different, leading to unpredictability during retrieval. Consequently, there is a need for automatic tools for video parsing and annotation. In the following sections, we propose methods in an effort to automatize the whole process as much as possible.

The proposed methods address the automatic decomposition of the videos, and leave the automatic annotation of the video at several granularity levels (providing Free Text, Keyword and Structured annotations for Video, Shot, Keyframe, Still/Moving Region) as a future work. This is still an open research problem, and to the best of our knowledge, the problem of video annotation at several granularity levels, as described, has not yet been addressed in the literature.

6.3.1 Temporal Video Segmentation

Temporal video segmentation aims to decompose a video temporally into its constituent parts. In our case, these constituent parts are Shots and Keyframes.

6.3.1.1 Shot Boundary Detection

Shot boundary detection (SBD) is the process of automatically detecting the boundaries between Shots in a video, and the research in this field can be considered to be mature. Several methods have been proposed for Shot boundary and scene change detection [69, 70, 71]. SBD was one of the tracks of activity within the annual TRECVid benchmarking between 2001 and 2007. The wide range of techniques used by the participants and performance comparisons are presented in [71]. These methods use various features to measure the similarity between frames; for example, color histograms, edge information, motion information, keypoint matching, or a combination of these features. In [72], a local keypoint matching algorithm is presented to detect the Shot changes using a so-called color context histogram (CCH) [73] feature computed around Harris corner points [74].

Abrupt shot transitions are usually easy to detect since the distance between consecutive frames has a high peak value at the transition (Figure 6.2-(a)), and this can be detected by a simple thresholding approach. On the other hand, gradual transitions are harder to detect, and it is not enough to measure just the distance between the consecutive frames since they take longer, the variation of inter-frame distances is more smooth and the peak value is much lower. Therefore, it is common to use two different thresholds to detect both the abrupt and gradual transitions.

Considering the characteristics of abrupt and gradual shot transitions, we developed a two-pass, graph-based shot boundary detection algorithm, inspired from the graph-based image segmentation algorithm proposed by Felzenszwalb and Huttenlocher. In this image segmentation algorithm, an undirected graph $G = (V, E)$ is constructed from the input image; each pixel is a vertex $v \in V$ and is connected to the neighboring pixels with edges having edge weights $w(v_i, v_j)$. Edge weights are the distances between the pixels.

As shown in Algorithm 6.1, the edges are first sorted into nondecreasing weight order. Then, edges are processed in this sorted order; two components are merged if the edge weight is less than a threshold, which is updated after each merge operation based on an input parameter k and the weight of the edge connecting the recently merged components. Hence, the algorithm merges the most similar components first (greedy decision), i.e., the components that are connected by edges with the smallest weights, and the merge operation continues as long as the weights are smaller than the dynamically updated thresholds. The output is a disjoint set forest, in which each disjoint set corresponds to a connected component in the image. The algorithm is fast and it uses union-by-rank and path compression heuristics for disjoint set operations to further improve the running time.

We adapted our shot boundary detection algorithm from this segmentation algorithm by constructing an undirected graph $G = (V,E)$ from the video frames; each frame F is a vertex $v \in V$ and edge weights are computed as the distances between the frames, $w(v_i,v_j) = d(F_i,F_j)$. Figure 6.2-(a) shows the variation of inter-frame CSD distances (edge weights) throughout a video containing two abrupt and three gradual shot transitions. As discussed above, abrupt shot transitions are usually easy to detect, while gradual transitions need special treatment. Therefore, we apply a two-pass segmentation with different k values and different distance measures for the computation of the edge weights. In the first pass, we detect the abrupt transitions having large peak values at the transitions. We perform the the second pass over all the segments obtained in the first pass, to detect the gradual transitions.

In the first pass, we construct an undirected graph $G = (V,E)$ from all the video frames; the edge weights are the distances between the consecutive video frames, $w(v_i,v_{i+1}) = d(F_i,F_{i+1})$. Then, we segment the graph G using Algorithm 6.1 with a large k value (e.g., 10). The output of the first pass is a set of video segments delineated by abrupt transitions. These segments may contain gradual transitions, which are detected in the second pass of our algorithm. For each of these segments, we construct a new undirected graph $G' = (V',E')$ whose vertices consist of the frames in one of the segments and edge weights are computed as a weighted average of distances between

Algorithm 6.1 EGBS(G, k, c): Efficient graph-based segmentation

```
G: input graph to be segmented, $G = (V, E)$
k: input parameter, larger $k$ results in larger components
c: minimum size of a component to be output
/* initialize $|V|$ sets sets of size 1 */
for each $v \in V$ do
    MAKE-SET($v$)
end for
/* initialize threshold values */
for each $v \in V$ do
    threshold[$v$] $\leftarrow k$
end for
sort edges $E$ in non-decreasing weight order
/* go over the edges in sorted order */
for each $edge(v_i, v_j) \in E$ do
    $u_1 \leftarrow$ FIND-SET($v_i$)
    $u_2 \leftarrow$ FIND-SET($v_j$)
    if $u_1 \neq u_2$ and $w(v_i, v_j) \leq$ threshold[$u_1$] and $w(v_i, v_j) \leq$ threshold[$u_2$] then
        UNION($u_1$, $u_2$)
        $u \leftarrow$ FIND-SET($u_1$)
        /* $|C|$: size of the new set after the union operation */
        threshold[$u$] $\leftarrow w(v_i, v_j) + k/|C|$
    end if
end for
/* postprocessing:  eliminate components smaller than $c$ */
/* go over the edges in sorted order */
for each $edge(v_i, v_j) \in E$ do
    $u_1 \leftarrow$ FIND-SET($v_i$)
    $u_2 \leftarrow$ FIND-SET($v_j$)
    if $u_1 \neq u_2$ and (size($u_1$) $\leq c$ or size($u_2$) $\leq c$) then
        UNION($u_1$, $u_2$)
    end if
end for
```

frames on a neighborhood W:

$$d(F_i, F_{i+1}) = \frac{\sum_{j=-W}^{j=+W} (\frac{1}{j}) d(F_i, F_{i+j})}{\sum_{j=-W}^{j=+W} \frac{1}{j}}, \quad j \neq 0, \quad 1 \leq i + j \leq N$$

where, N is the number of frames in the segment, and the distance between frames F_i and F_{i+1} is computed by the weighted average of the distances considering W frames preceding and succeeding frame F_i. Figure 6.2-(b) shows the variation of CSD distances throughout a video (the same video as in Figure 6.2-(a)) using this weighted average distance approach for $W = 3$ and $W = 5$. Note that this graph is obtained using the whole video sequence in the distance computation rather than the individual segments obtained in the first pass; this is shown in Figure 6.2-(c); therefore, the graph in Figure 6.2-(c) does not contain the abrupt transitions.

We see that, using this type of distance measurement amplifies the distances at the gradual transitions (second, third and fourth transitions in the figure) while it smoothens the abrupt transitions (first and fifth transitions in the figure). This is why we detect the abrupt transitions in the first pass using only the inter-frame distance between the consecutive frames, since, this way we can determine the transition points more accurately.

Finally, we segment the graph G' for each segment using again Algorithm 6.1 with a small k value (e.g., 0.5). The output of this second pass is a set of video segments (Shots) delineated by gradual transitions. As a result, we obtain both the abrupt and gradual transitions using a two-pass algorithm.

6.3.1.2 Keyframe Selection

The aim of Keyframe selection is to obtain a set of frames that covers all aspects of a video sequence as much as possible. It is common to represent a Shot with a single Keyframe, the frame in the middle of the Shot. This causes considerable information loss for Shots containing strong camera motion and scene activity, which is why multiple Keyframes are usually needed for each Shot. Another straightforward approach is to uniformly sample the video sequence with a certain frame rate [75], but this may lead to redundancy.

The most common approach to Keyframe selection is to cluster the frames from the Shots based on their low level features such as color and texture, and choose the frames closest to cluster centroids as Keyframes [75, 76, 77, 78].

In a video indexing and retrieval setting, the aim is to store just enough number of Keyframes to represent a Shot. Storing fewer Keyframes adversely affects the retrieval performance, but is good for retrieval speed, since fewer Keyframes will be considered during query processing time. Storing many Keyframes per Shot has the opposite effect. Therefore, we take a clustering approach to strike a balance between representation detail and retrieval speed.

We cluster the frames in a Shot with respect to their low-level MPEG-7 descriptors (CSD, HTD) to account for the variations in the visual appearance and also keep the number of selected Keyframes as low as possible. We employ an incremental K-means algorithm for clustering [79, 80], as summarized in Algorithm 6.2. We process each frame in its temporal order; if its distance to the current cluster is below a threshold, it is added to the cluster and the cluster centroid is updated. Otherwise, a new cluster is formed. At the end of processing a Shot, we obtain a set of clusters with non-overlapping frames. The clusters are taken as the Keysegments and the centroid frame of each cluster is selected as the Keyframe (Figure 6.3-(a)).

Figure 6.3-(b) shows Keyframe samples selected by the incremental k-means algorithm described above. Color (CSD), texture (HTD) and a combination of color and texture (CSD + HTD) are used as low-level descriptors to represent the frames. The distances are normalized to [0.0, 1.0] as described before, and a threshold value of 0.1 is used in the examples given. The value of the threshold can be used to adjust the number of Keyframes selected; a smaller threshold for a larger number of Keyframes and vice versa. Using CSD and HTD together, with a threshold value of 0.1, results in perceptually good Keyframes, as also demonstrated by the selected Keyframes in the figure.

6.3.2 Spatial Segmentation for Still Regions

In BilVideo-7 data model, Still Regions are intended to represent the background regions, which usually constitute a large portion of the scenes. For example, regions corresponding to sky, greenery, grass, sea and forest are all good candidates for Still Regions. In contrast to Moving Regions, we do not need a very accurate boundary/shape information for such regions, since queries related to background regions will usually be related to color, texture and possibly a coarse position information. As a result, after selecting a set of Keyframes from a Shot, a coarse segmentation of each Keyframe will

Algorithm 6.2 Keyframe Selection(VS, T)

```
$VS$: input video sequence (Shot) having $N$ frames, $VS = \{F_1, F_2, \ldots, F_N\}$
$T$: threshold for starting a new cluster, $0 < T \leq 1.0$
/* initialize */
$KS \leftarrow \emptyset$ /* Keysegments */
$KF \leftarrow \emptyset$ /* Keyframes */
$C \leftarrow F_1$ /* Current cluster */
/* go over all the frames $F_i \ldots F_N$ in their temporal order */
for $i = 1$ to $N$ do
  /* compute the distance to the centroid of the current cluster */
  $d \leftarrow \textbf{distance}(F_i, \textbf{centroid}(C))$
  if $(d < T)$ then
    $C \leftarrow C \cup F_i$ /* Add this frame to the current cluster */
    update the centroid of cluster $C$
  else /* New cluster */
    $KS \leftarrow KS \cup C$ /* Update Keysegments */
    $KF \leftarrow KF \cup \textbf{centroid}(C)$ /* Update Keyframes */
    $C \leftarrow F_i$ /* Start a new cluster */
  end if
end for
```

be sufficient for our purposes. Then, the largest regions in the segmentation, possibly with an area above a certain threshold (e.g., 20% of the frame area), can be chosen as the representative Still Regions. Next, we review some of the well-known segmentation algorithms that can be utilized for this task.

Object segmentation is used to identify regions of interest in a scene and is one of the most challenging tasks in image/video processing. It serves as the key technique in many applications, including content-based indexing and retrieval, compression, recognition, event analysis, understanding, video surveillance, intelligent transportation systems, and so on. The problem of unsupervised image/video object segmentation is ill-defined because semantic objects do not usually correspond to homogeneous spatio-temporal regions in color, texture, or motion. Therefore, the segmented objects are often not consistent with human visual perception. Consequently, practical application of these algorithms is normally limited to region segmentation rather than object segmentation [81].

There is a large literature on spatial image segmentation ranging from graph-based methods, region merging techniques and graph cuts to spectral methods. In

Blobworld [82], segmentation is obtained by clustering pixels in a joint color-texture-position feature space using Expectation Maximization (EM). In [83], the authors construct an edge flow graph based on detected edges, and use the graph to find objects in the scene. Normalized Cuts [84] algorithm constructs a graph from the image; each node (pixel) has an edge to all other nodes (pixels). The segmentation is obtained by finding the normalized cuts of the graph. It is one of the most successful image segmentation algorithms in literature but it is computationally costly.

In JSEG algorithm [85], images are first quantized to several representative classes. Then, each pixel is replaced by its representative class label. By applying a “good” segmentation criterion to local windows, a “J-image” is produced. Finally, a region growing approach is used to segment the image based on multi-scale J-images. It is also applied to video sequences with an additional region tracking scheme and shown to be robust on real images and video.

An efficient graph-based segmentation (EGBS) is proposed in [86]. It runs in time linearly with the number of graph edges and is much faster than the Normalized Cuts algorithm. It constructs a graph, in which each pixel is a vertex and is connected to the neighboring pixels. It is a greedy algorithm and works by first sorting the edges in increasing order of weight and then processing the edges in this order in the segmentation of the graph. Finally, a disjoint set forest (DSF) is obtained; each set corresponds to one component in the image.

Nock and Nielsen proposed a fast segmentation algorithm, called statistical region merging (SRM), based on statistical properties of color images [87]. The approach takes into account expected homogeneity and separability properties of image objects to obtain the final segmentation through region merging. It is unsupervised and well suited to noisy images, while the method presented in [88] requires some user assistance. The algorithm has an input parameter $Q \in \{1, 2, ..., 255\}$ for the statistical complexity of the scene to segment. The smaller the value of Q , the coarser the segmentation. It is hence possible to control the coarseness of the segmentation and build a hierarchy of coarse-to-fine (multiscale) segmentations of an image.

Among the good segmentation algorithms reviewed above, we selected the JSEG [85], EGBS [86] and SRM [87] and assembled them into a segmentation library using OpenCV [89]. Using this library we developed a segmentation tool, BilSEG (Figure 6.4), to explore the effects of preprocessing/filtering, color space and segmentation parameters on these algorithms. Using the tool, a cascade of various filters (median, Gaussian, mean shift, bilateral, etc.) can be applied to the input image prior to segmentation, and various color spaces (RGB, HSV, CIE Lab, CIE LUV, YCbCr) can be used for the input image (JSEG uses LUV color space).

We observed that JSEG and SRM are good candidates for Still Region segmentation, while EGBS tends to produce oversegmentation (splitting the image into many small regions), especially on textured images. In terms of speed, JSEG is 5-6 times slower than SRM and EGBS, and EGBS is a bit faster than SRM. Figure 6.5 shows example segmentations using JSEG (region merging threshold 0.6) and SRM (Q value 10). We favor SRM for Still Region segmentation for its speed and better control of hierarchy of coarse-to-fine segmentation by tuning the value of Q. A Q value of around 10 is appropriate for our purposes.

In some cases, large regions may belong to salient objects (Moving Regions) in the scene, e.g., the flower in Figure 6.5. To avoid redundancy (storing the same region as both Still and Moving Region), the output of Moving Region segmentation can be used to exclude such regions.

6.3.3 Segmentation for Moving Regions

In BilVideo-7 data model, Moving Regions are intended to represent the salient objects in the Shots. Salient objects are the most prominent objects that the users might want to perform more detailed queries about, compared to other less important objects in the scene. In the following, we first review the literature on saliency, and then focus on how to employ saliency analysis within the context of video indexing and retrieval.

6.3.3.1 Saliency

In the literature, salient objects are defined as the visually distinguishable, conspicuous image components that attract our attention at the first glance, as in Figure 6.6. These are usually high contrast regions, or regions with significantly different appearance compared to their surroundings. Detection of salient regions is also referred to as *image attention analysis*.

The first remarkable work on saliency is by Itti *et al.* [90]. It combines multiscale image features into a single topographical saliency map. Using this map and a dynamic neural network, the attended image locations are selected in order of decreasing saliency. In [91], a saliency map is generated based on local contrast analysis, then a fuzzy growing method is used to extract attended areas or objects from the saliency map by simulating human perception.

In [92], the authors propose a salient object extraction method by a contrast map using three features (luminance, color and orientation), and salient points for object-based image retrieval. The work in [93] investigates empirically to what extent pure bottom-up attention can extract useful information about the location, size and shape of objects from images and demonstrates how this information can be utilized to enable unsupervised learning of objects from unlabeled images. In [94], image segmentation is formulated as the identification of single perceptually most salient structure in the image.

In [95], the authors try to obtain OOI (Object-of-Interest) segmentation of natural images into background and a salient foreground by region merging within a selected attention window based on saliency maps and saliency points from the image. In [96], the log spectrum of each image is analyzed to obtain the spectral residual, which is transformed into spatial domain to obtain the saliency map which in turn indicates the positions of proto-objects. In [97], salient object detection is formulated as an image segmentation problem, in which the salient object is separated from the image background. A set of novel features are proposed: multi-scale contrast, center-surround histogram, and color spatial distribution to describe a salient object locally, regionally, and globally. A Conditional Random Field (CRF) is learned using a human labeled set of training images to effectively combine these features for salient object detection.

There has been little work on salient object detection in video, taking into account the valuable motion information. The model proposed in [98] predicts the saliency of a spatio-temporal event based on the information it contains. The joint spatial and temporal conditional probability distributions of spatio-temporal events are modeled and their spatio-temporal saliencies are computed in an integrated way. Motion channels are added to intensity-based saliency maps in [99]. The authors argue that addition of motion information, as they described, did not improve the performance. In [100], spatial and temporal saliency maps are fused to compute a spatio-temporal saliency map. The spatio-temporal saliency framework described in [101] combines spatial feature detection, feature tracking and motion prediction in order to generate a spatio-temporal saliency map to differentiate predictable and unpredictable motions in video.

6.3.3.2 Saliency for Moving Regions

For a video/image object, the notion of being salient or not is a subjective matter; different people may select different objects from the same content. Elazary and Itti claim that selecting interesting objects in a scene is largely constrained by low-level visual properties rather than solely determined by high-level object recognition or cognitive processes [102]. The authors support this claim by analyzing the selected objects in LabelMe image database to evaluate how often interesting objects are among the few most salient locations predicted by a computational model of bottom-up attention. From this work, we can conclude that low-level visual features can be employed to determine the salient objects, at least to some degree.

In contrast to pixel-based saliency map approaches reviewed above, we take a segmentation-based approach to determine the salient objects/regions with the aim of obtaining the boundaries of objects better. Next, we list the characteristics of video objects that make them perceived as salient by a human observer [103, 104, 105]. We also suggest possible features that may be used to discriminate such objects from the others or from the background.

1. In videos, objects in camera focus are usually important (e.g., a speaking head in the middle). Objects in camera focus have higher contrast and sharper edges compared to the background. This can be measured using region variance, entropy, and edge strength on the region boundary.

2. Visually conspicuous regions are salient. This is indicated by how different the region is from its surrounding and from the rest of the scene, and hence can be measured by inter-regional contrast on specific features (e.g., color, texture, motion).

3. Moving objects may be important (e.g., walking person, sailing boat); hence velocity is an important clue.

4. Too large, too small, too long/thin regions are usually not important. For example, large regions are mostly background. This suggests using area and shape features.

5. Salient objects should be consistent; they should appear in most of the frames within a Shot (e.g., at least 10% of the frames in the Shot).

Using these characteristics, we compute the following features for each region and obtain a feature vector of length 18. We obtain the segmentations of each frame in a Shot using the JSEG algorithm (see Section 6.3.2). These features are easy to compute once the segmentation of a frame is available.

1. Regional color, shape, texture and motion features

   - Region color variance (maximum of 3 RGB channels) and entropy (from grayscale image)

     Given a region R and a descriptor D that takes on values $\{d_1, \ldots, d_r\}$ (e.g., in an 8-bit grayscale image, D is from 0 to 255), the regional entropy is defined as

$$H_{D,R} = -\sum_{\mathbf{i}} P_{D,R}(d_i) \log_2 P_{D,R}(d_i) \tag{6.1}$$

where $P_{D,R}(d_i)$ is the probability of descriptor D taking the value d_i in the region R as described in [106].

- Average region velocities in X and Y directions computed by optical flow between successive frames
- Region area & shape properties: ratio of region area to frame area, aspect ratio, ratio of region area to MBR area (compactness)

2. Inter-regional features

   - Local & global contrast: sum of difference of mean color, variance, entropy, velocity of a region from its neighbors, and from all other regions, weighted by region areas (Figure 6.7).

     For a region A, the contrast features (C_A) are computed as,

     $$C_A = \sum_{\mathbf{X}} w_X \left| F_A - F_X \right| \tag{6.2}$$

     where F is a color, texture or motion feature, w is a weight reflecting the effect of how large the region X is. For local contrast, X is any region neighboring region A; for global contrast, X represents all the remaining regions.

   - Boundary edge strength.

6.3.3.3 Classification and Tracking of Moving Regions

We collected 300+ positive/negative salient region examples, computed the above-mentioned features, normalized them to zero mean and unit variance, and trained a Support Vector Machine (SVM) [107, 108, 109]. Using this SVM, we classify each region as being salient or not. For each salient region, the distance to the separating hyperplane returned by the SVM is assigned as the saliency score. The higher the score, the more salient the region. We rank the regions according to this score and select the first N regions. This parameter can be used to tune the detection precision & recall of the system. The number of salient regions as detected by SVM can be zero or more, hence our system can say that there is no salient region in a frame.

We need to track each salient region throughout the Shot for consistency check and also for trajectory information, which is stored in the database. The literature on tracking is broad [110, 111, 112, 113]. Tracking algorithms assume that the object to be tracked is given as input, which is not a valid assumption if the system is fully automatic and should both detect and track the objects without any user assistance. Therefore, we take an approach similar to the saliency-based discriminant tracking approach [114] in the sense that the target is detected in each frame using saliency analysis and hence tracked.

We keep a list of tracked salient regions within each Shot. In each frame, we try to find a match for each tracked region by first imposing position and shape constraints and then checking color histogram distance between the regions. At the end of processing a Shot, if a region appeared less than a threshold (10% of the frames in a Shot), it does not qualify as a salient region. This threshold can also be used to tune the detection precision & recall.

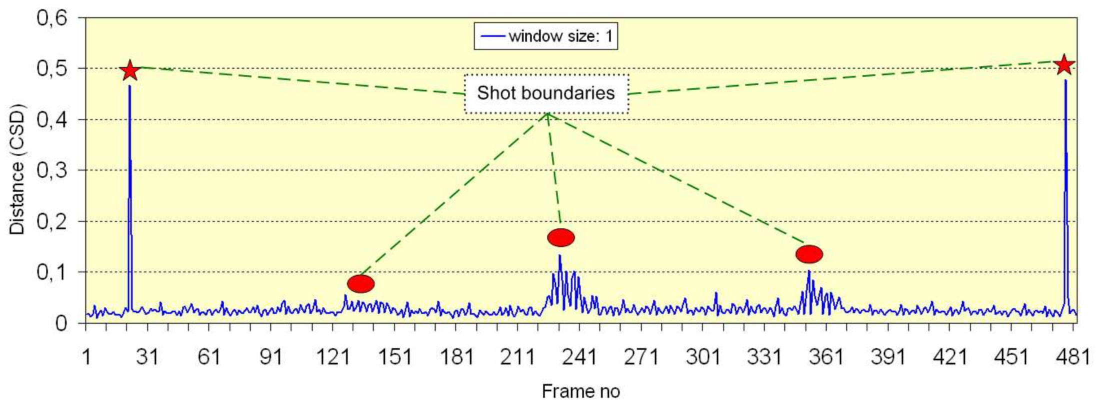
6.3.3.4 Results and Discussion

We tested our system on several video sequences with length hundreds of frames each. Figure 6.8 shows example detections of varying quality. If the frames are easy to segment, so that the segmentation quality is satisfactory, the resulting detections are good. In an example opposite case, as shown in the first row, second image of Figure 6.8, the walking person could not be correctly detected due to poor segmentation.

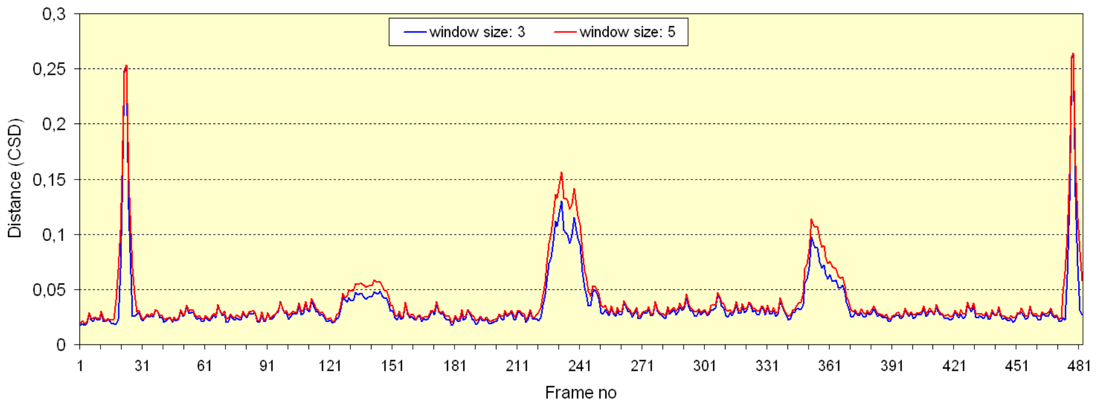
We compared the performance of our system with one of the leading saliency model (SM) [90] approaches, whose MATLAB implementation is freely available at [115]. The SM approach is developed for images; therefore, we extracted Keyframes from each Shot and run the MATLAB implementation on the Keyframes. Figure 6.9 shows detection examples by the two methods. We limited N to 5 in the experiments. In most cases, our approach performs much better in terms of human visual perception and in terms of our definition of saliency. We also computed the precision-recall values of the two systems on 2 test video sequences with a total of 668 frames and evaluated the first 5 detections as correct/wrong/missed. The precision-recall graph in

Figure 6.10 indicates that our system is better in detecting the salient regions.

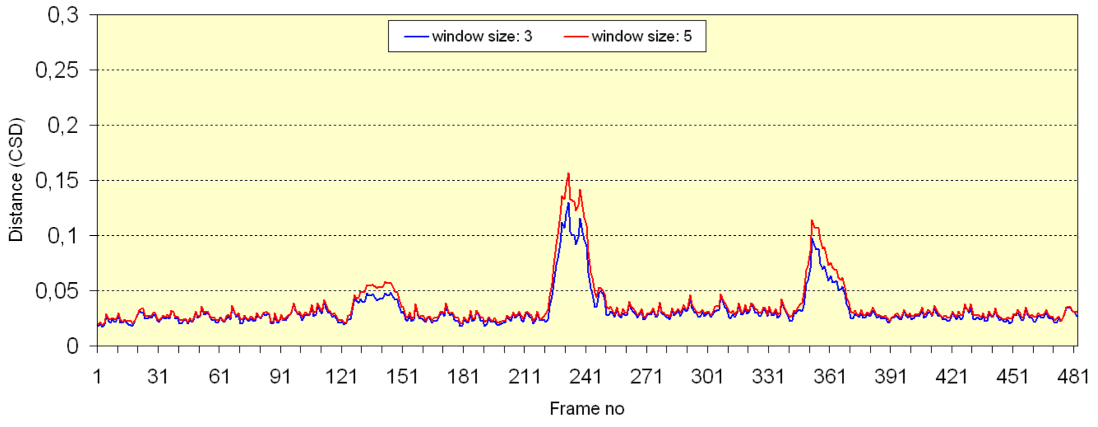
Our approach achieves good detection at region level, but when the objects are not homogeneous in color/texture, it fails to capture the objects as a whole (e.g., golfer in Figure 6.8, second row, second column), since the unsupervised segmentation algorithm cannot handle such cases. This is still an open research problem. Our current work focuses on using color, texture, motion and saliency cues synergetically to recover the salient objects as a whole. A recent work by Alexe *et al.* [116] is a good step forward in this direction.



(a)

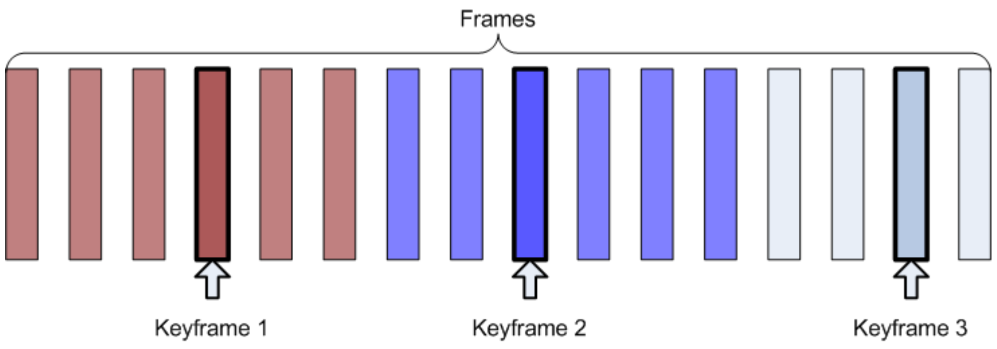


(b)

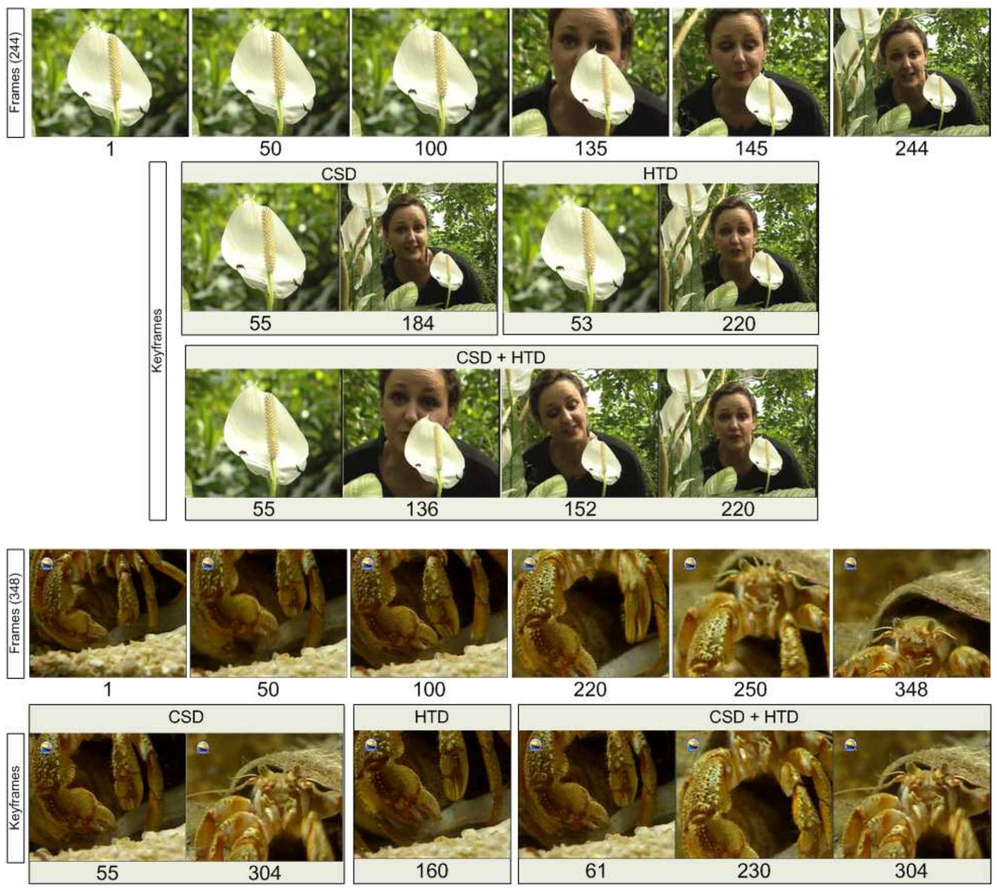


(c)

Figure 6.2: Shot boundary detection, CSD distance for a video sequence having two abrupt and three gradual shot transitions. (a) Inter-frame distances between consecutive frames and shot boundaries, (b) weighted inter-frame distance on a neighborhood, (c) the same distance as in (b) for the three segments delineated by abrupt transitions.

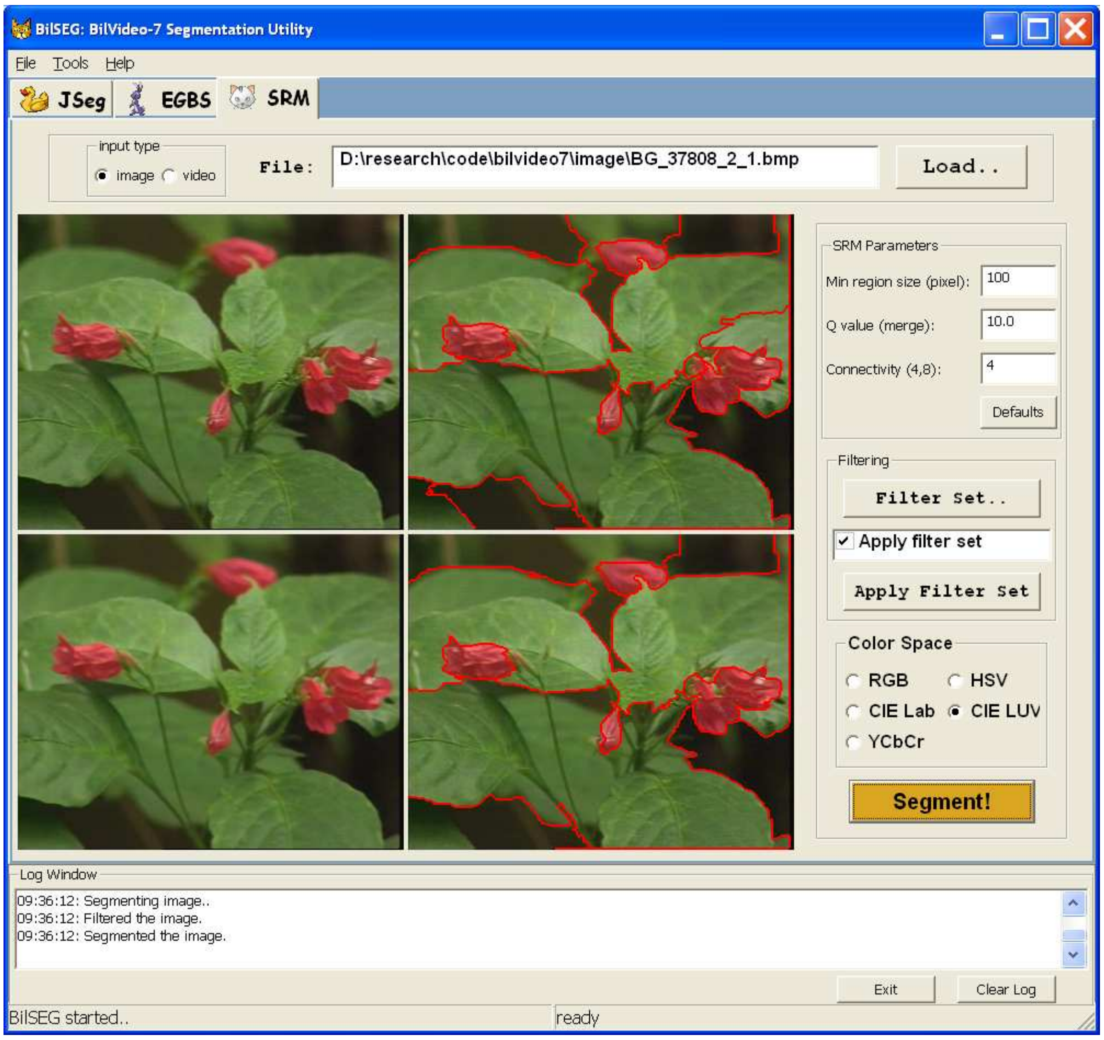


(a)



(b)

Figure 6.3: (a) Keyframe selection, (b) Keyframe examples. Frame numbers according to the start of the Shot are shown at the bottom of each image. CSD, HTD, CSD + HTD are the low-level descriptors used to select each group of Keyframes.



input type: [✓ image] video
File: D:\research\code\bilvideo7\image\BG_37808_2_1.bmp

SRM Parameters
Min region size (pixel): 100
Q value (merge): 10.0
Connectivity (4,8): 4

Filtering
[✓] Apply filter set

Color Space: RGB HSV CIE Lab [✓ CIE LUV] YCbCr

```
09:36:12: Segmenting image..
09:36:12: Filtered the image.
09:36:12: Segmented the image.
```

BilSEG started.. ready

Figure 6.4: BilSEG: BilVideo-7 Segmentation Utility. Input image is shown at the top left, filtered image at the bottom left, the last two segmentations on the right.

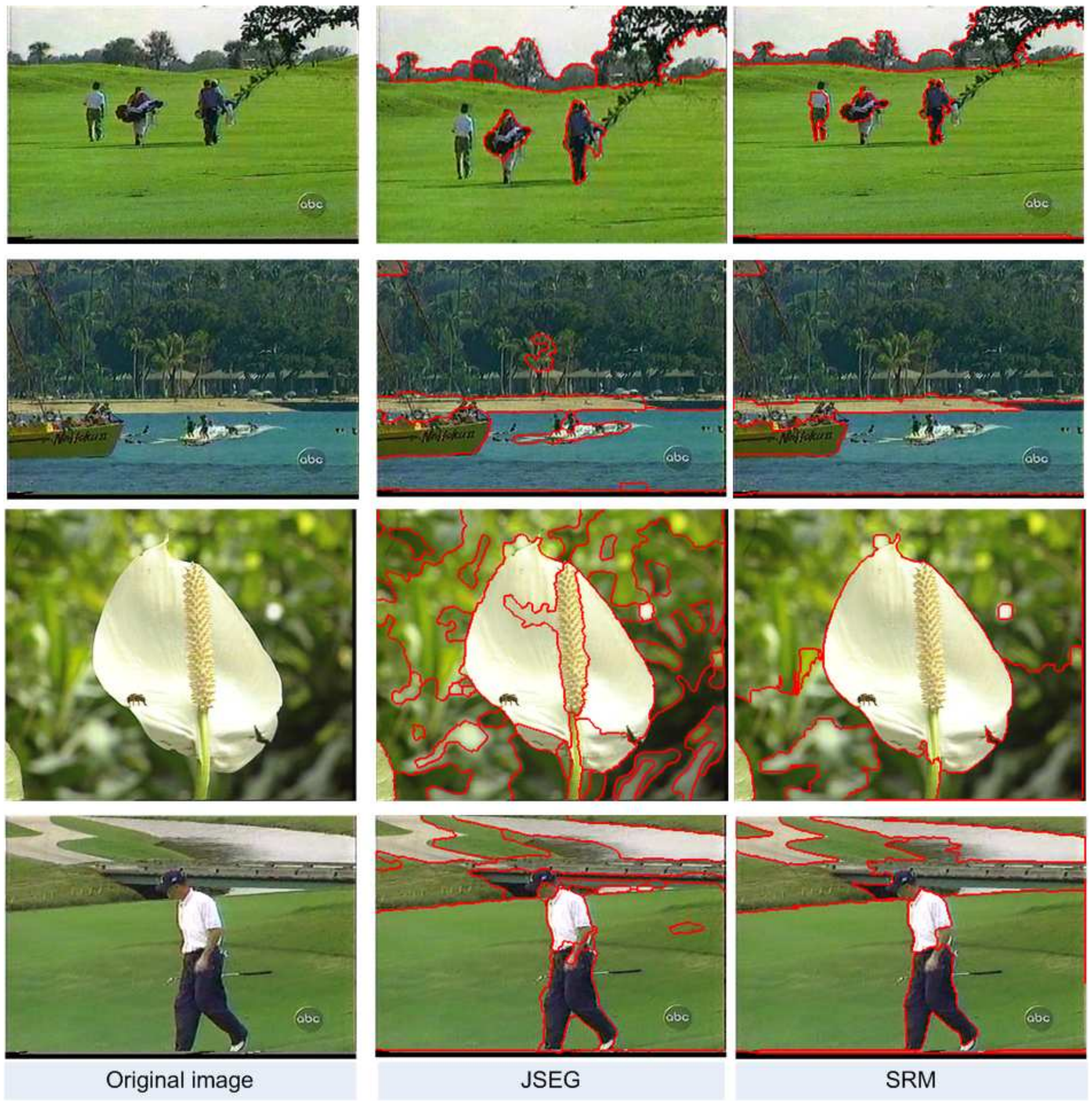


Figure 6.5: Segmentation examples using JSEG [85] and SRM [87].

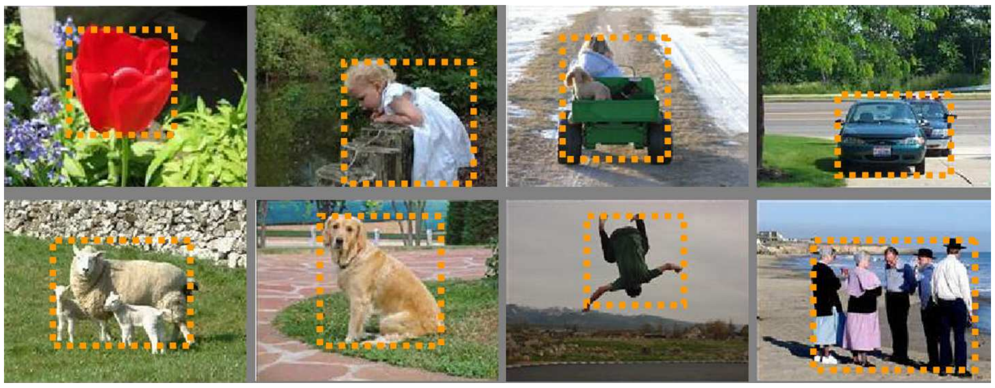


Figure 6.6: Salient objects are visually distinguishable, conspicuous image components that attract our attention at the first glance.

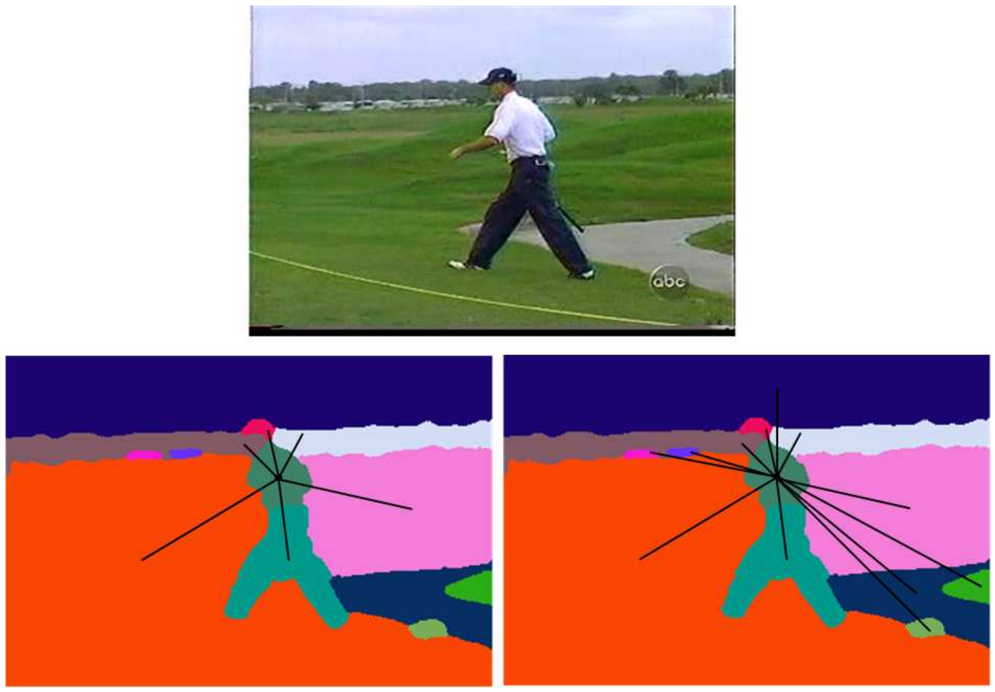


Figure 6.7: Computing the contrast features for a region. Top: original video frame, left: local contrast measuring how different a region is from its surrounding, right: global contrast, measuring how different a region is from the rest of the frame.

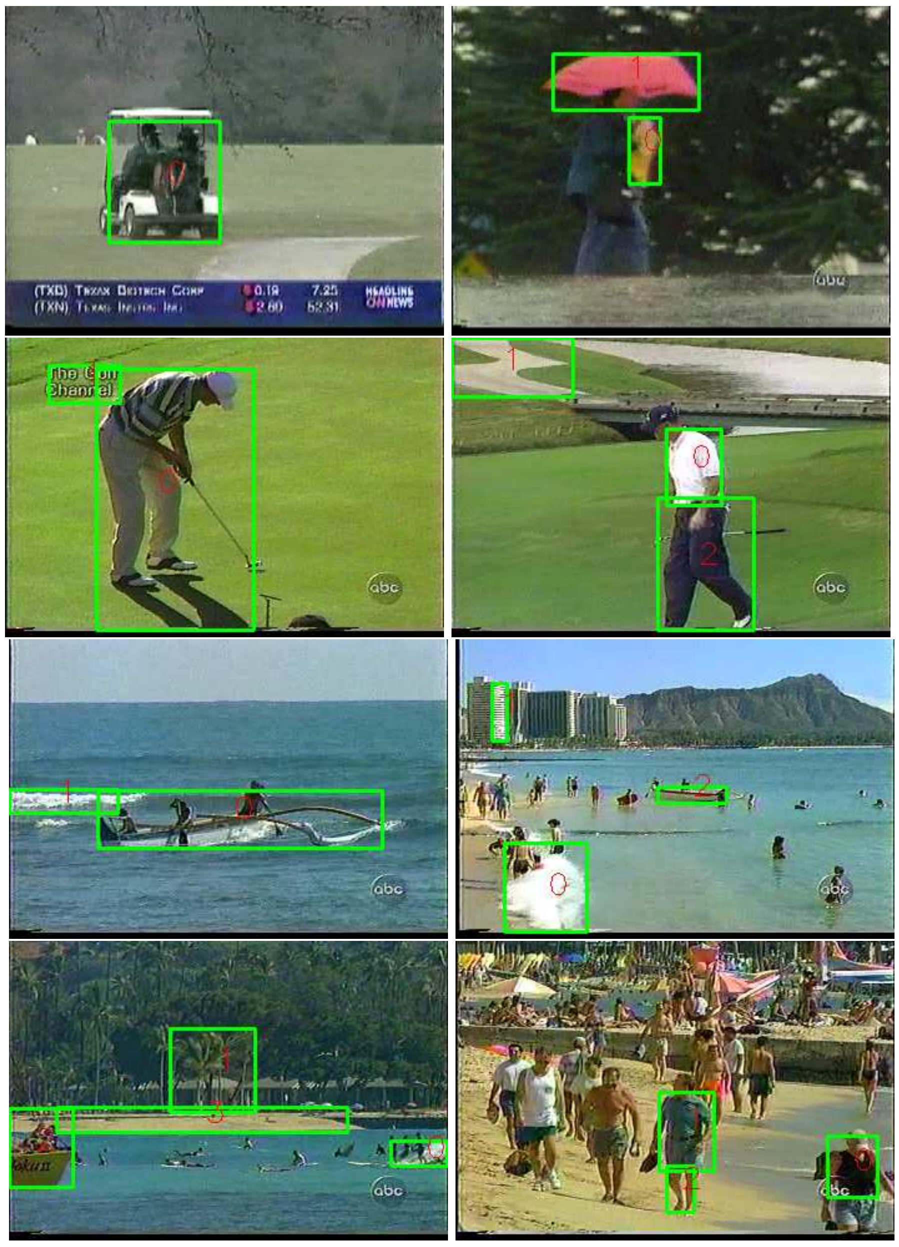


Figure 6.8: Example salient region detections. Numbers within rectangles show the rank of saliency for the enclosed region.

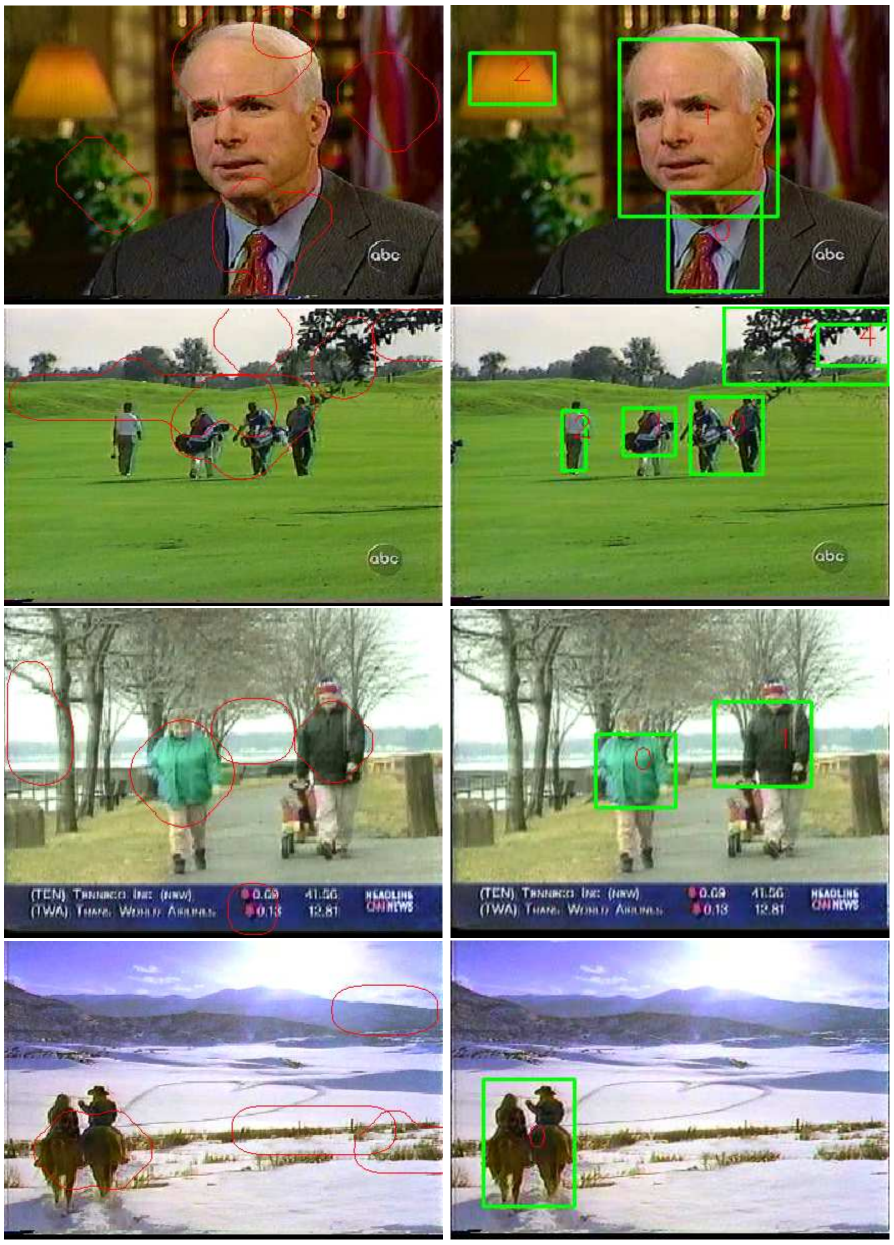


Figure 6.9: Visual comparison of first 5 detections. Left: SM, right: our approach.

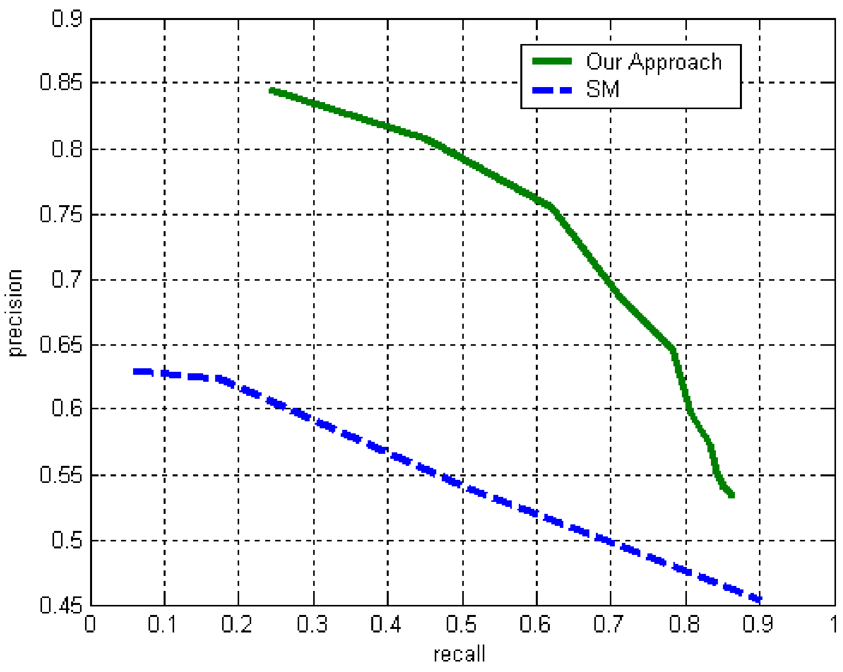


Figure 6.10: Precision-recall graph for the detection of first 5 salient regions, comparing our approach with SM [90].

Chapter 7

Experiments

7.1 Implementation Details

The system is implemented in C++. Graphical user interfaces are created with open-source, cross-platform C++ GUI library wxWidgets [117]. Open Source Computer Vision Library (OpenCV) [89] and FFmpeg [118] 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 (partly available online at [9]) that we adapted from the MPEG-7 XM Reference Software [33]. XML data is handled with open-source Xerces-C++ XML Parser library [119]. Finally, Tamino [53] 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.

7.2 Data Set

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 [120], consisting of news, documentary, educational and archiving program videos. We obtained the MPEG-7 representations of the videos manually with our MPEG-7 compatible video feature extraction and annotation tool, BilMAT (Section 6.2.1).

7.3 Sample Queries

Three spatial queries are shown in Figure 7.1. The first query at the top searches for the video segments in which a golfer is *above* a golf cart, formulated as a text-based spatial relation query, “golfer *above* golf cart”. 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 in the middle, “Clinton *left* Blair”, is sketch-based. 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 third query at the bottom is a sketch-based spatial query containing three objects. This query is also handled successfully. There is no limit on the number of objects in the sketch-based query interface.

Several low-level queries are shown in Figure 7.2. In the image-based query (a), query image is represented by CSD and DCD descriptors and searched in Keyframes. Three region-based low-level queries are shown in Figure 7.2-(b). The first query searches for an anchorman by providing the region shown on the left, specifying CSD as the descriptor and searching in Moving Regions (salient objects). The last two region-based queries searches for a face by providing a face region and specifying CSD + RSD and FRD as the descriptors respectively. Figure 7.2-(c) shows a video sequence based query using the GoF descriptor. All query results are satisfactory considering the input video segments and the types of descriptors used.

Figure 7.3 shows three motion trajectory queries. The first query searches for objects moving from left to right, by plotting a set of trajectory points with the mouse. The second query elaborates on the first query by providing the color information of the moving query object as well (by inputting a pink region which is selected from an image); this results in promoting the video segment having the specified color information to a higher rank. The last query searches for video segments containing two objects moving to each other from the left and right of the scene to meet in the middle. As illustrated by these examples, the queries may contain any number of objects with any number of suitable descriptors, which demonstrates the querying power of the system.

Figure 7.4 shows various composite queries, in which high-level semantics in the form of keyword annotations and low-level descriptors (DCD, CSD, EHD, RSD, MTr, etc.) are used together to describe the query video segments. In the first composite query, the Keyframe is represented with DCD and *golf green*; the Moving Region is represented with CSD, RSD and *golfer*. The second query is similar to the first one; the Keyframe is represented with CSD and the Moving Region is represented with CSD, RSD and MTr. Hence, the inclusion of motion trajectory information in the query specification is reflected in the query result.

In the third composite query, two Still Regions at the top and at the bottom are represented with CSD and EHD. The Moving Region in the middle is represented with semantic concepts *airplane or boat or helicopter*. Finally, the composite query at the bottom searches for a scene, in which there is one Moving Region represented with CSD and *horse*, and one Still Region represented with only *green or grass*. Again, the queries are handled successfully and the result rankings are in agreement with our expectations.

Using such composite queries, the user can access video segments having any specific composition described in the query. The number and type of video segments 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.

Table 7.1: Query execution 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.

Query type	*Description (Segments and descriptors)*	*Execution time (sec)*
Textual query	Keyframe (keyword)	0.125
Textual query	Moving Region (keyword)	0.125
Textual 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 Figure 7.4, first	Keyframe (DCD+keyword), Moving Region (CSD+RSD+keyword)	0.438
Composite query Figure 7.4, third	2 Still Regions (CSD+EHD), Moving Region (keyword)	0.391

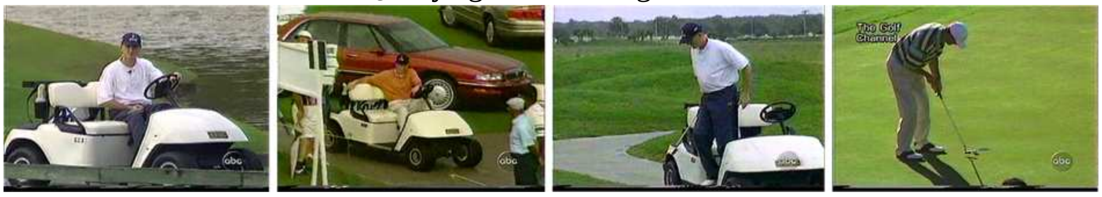
7.4 Running Time

Table 7.1 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 size of the descriptors and the complexity of the matching algorithm (distance measure). Note that query execution is based on exhaustive search, i.e., all the relevant video segments in the database are processed to obtain a subquery result.

As shown in the table, queries involving low-level descriptors take longer to execute compared to text-based queries since the distance computation between the low-level descriptors is computationally more expensive. Spatial relation queries are fast, although the spatial relation similarities are computed at query execution time, for flexibility and accuracy of matching. Another observation is that queries involving Moving Regions takes longer than, for instance, Still Regions. This is expected, since multiple instances and hence multiple descriptors are stored for Moving Regions to account for the variation in their visual appearances and locations.

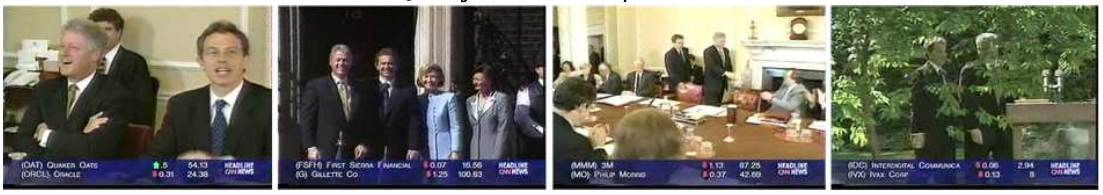
The multi-threaded query processing architecture provides some degree of parallelism and shortens the query execution times when the subqueries are executed in separate threads. For instance, a Keyframe query with CSD takes 0.141 seconds and a Keyframe query with HTD takes 0.125 seconds to execute, while 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 last two composite queries in the table.

Query: golfer *above* golf cart

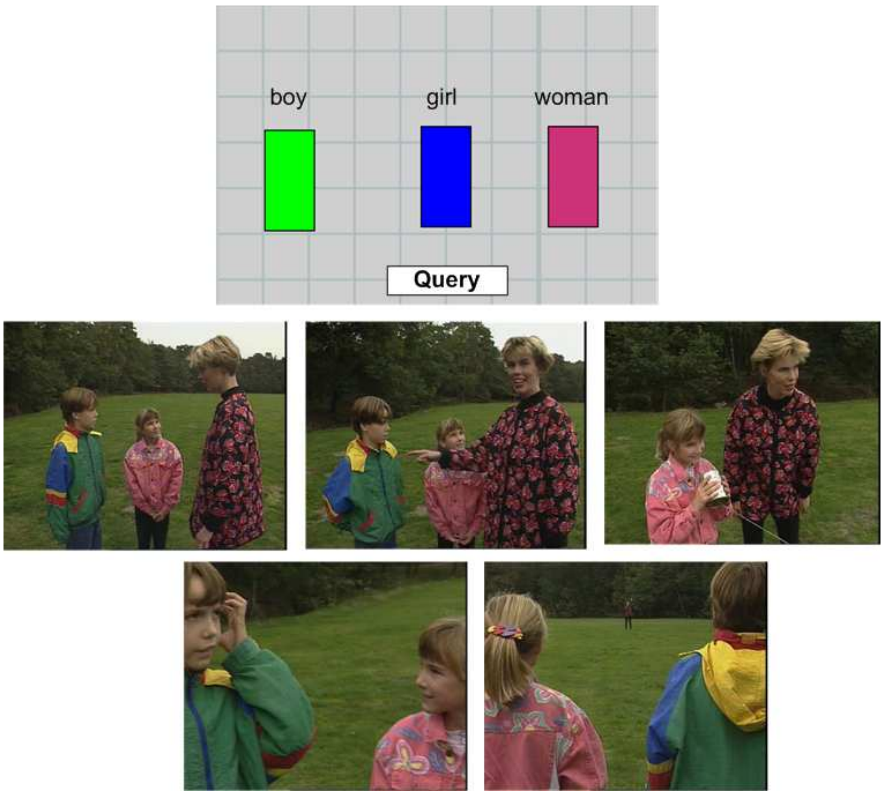


(a)

Query: Clinton *left* Blair

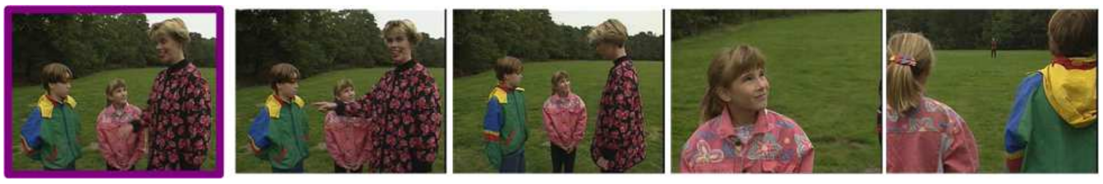


(b)

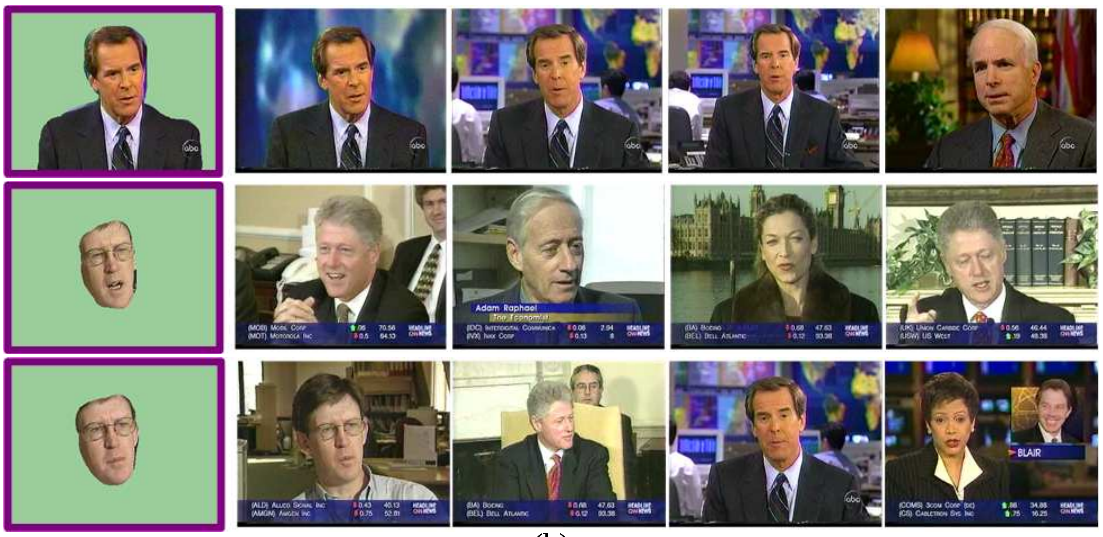


(c)

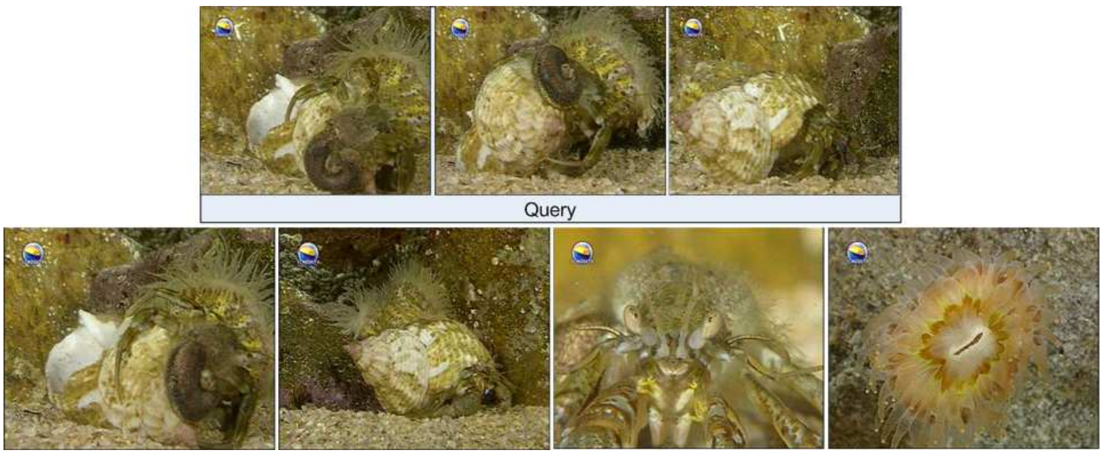
Figure 7.1: Spatial queries. (a) Text-based spatial relation query, "golfer *above* golf cart". (b) Sketch-based spatial relation query, "Clinton *left* Blair", formulated by drawing two rectangles and labeling them as *Clinton* and *Blair*. (c) Sketch-based spatial relation query containing 3 objects.



(a)



(b)



(c)

Figure 7.2: Low-level queries. (a) Image-based query, descriptors: CSD + DCD. (b) Region-based queries, descriptors: CSD (first), CSD + RSD (second), FRD (third). (c) Video sequence based query, descriptor: GoF.

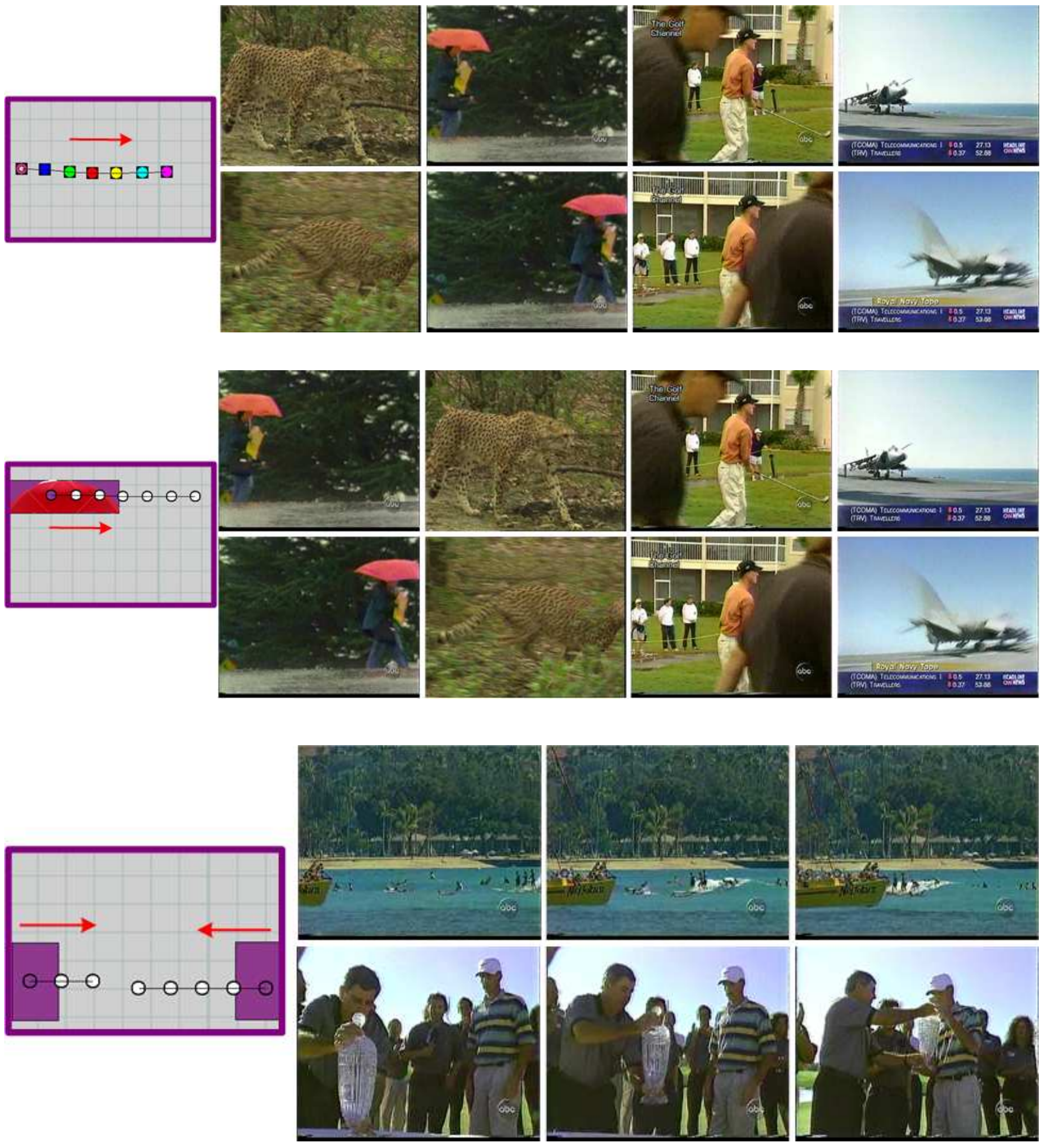


Figure 7.3: Trajectory query examples.

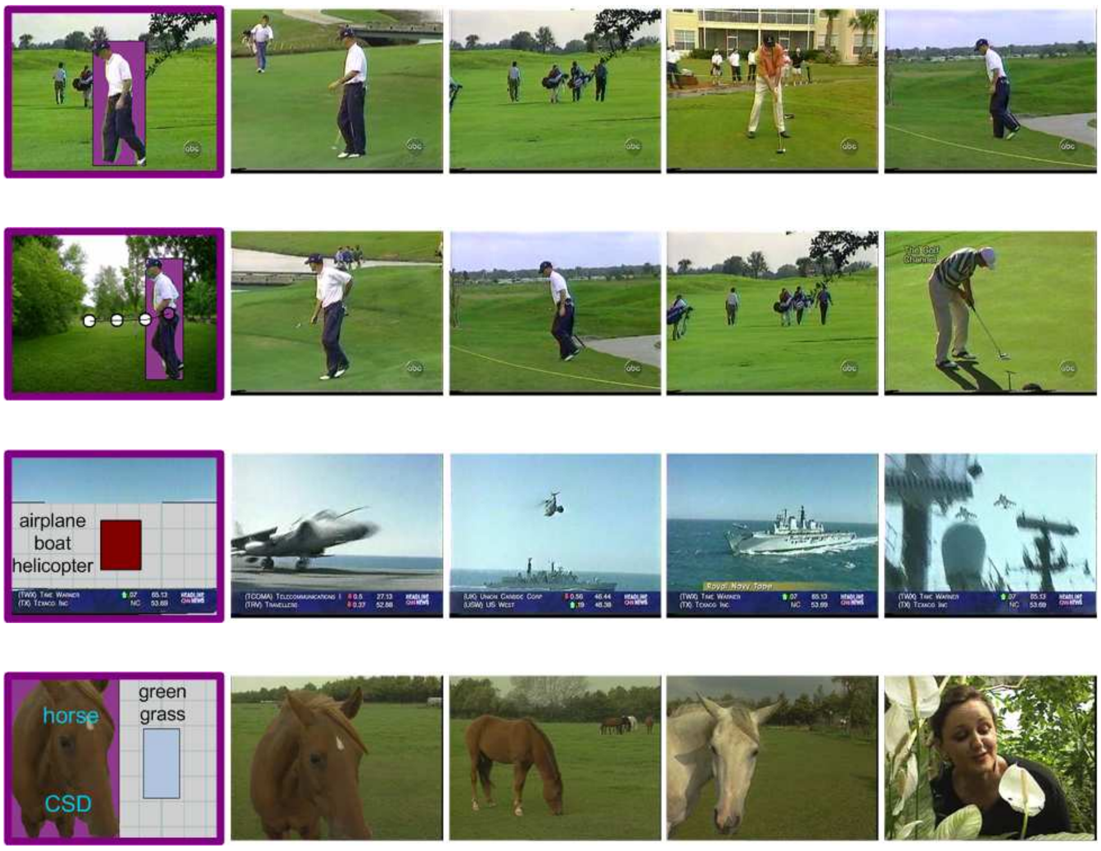


Figure 7.4: Various composite query examples. Queries are on the left.

Chapter 8

Conclusions and Future Work

We described our prototype MPEG-7 compatible video database system, BilVideo-7, that supports different types of multimodal queries seamlessly. 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.

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, the MPEG-7 compatible video feature extraction and annotation tool should be equipped with automatic processing capabilities to reduce manual processing time, error and human subjectivity during region selection and annotation.

Finally, an MPEG-7 compatible multimedia database system, which would also support the representation and querying of audio and image data, can easily be built based on the architecture of BilVideo-7. Images can be considered to be a special case of Keyframes which are decomposed into Still Regions; therefore, an image database system can be considered to be a subset of BilVideo-7. Audio data can be represented similar to video, decomposing into audio shots and audio subshots and extracting the low-level MPEG-7 audio descriptors, and query processing will be exactly the same.

Bibliography

[1] N. Morsillo, G. Mann, and C. Pal, “YouTube Scale, Large Vocabulary Video Annotation,” *Video Search and Mining*, vol. 287, pp. 357–386, 2010.

[2] S. Baluja, R. Seth, D. Sivakumar, Y. Jing, J. Yagnik, S. Kumar, D. Ravichandran, and M. Aly, “Video Suggestion and Discovery for YouTube: Taking Random Walks Through the View Graph,” in *Proceedings of International Conference on World Wide Web*, pp. 895–904, 2008.

[3] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, , and P. Yanker, “Query by Image and Video Content: The QBIC system,” *IEEE Computer*, vol. 28, no. 9, pp. 23–32, 1995.

[4] S. F. Chang, W. Chen, H. J. Meng, H. Sundaram, and D. Zhong, “VideoQ: An Automated Content Based Video Search System Using Visual Cues,” in *Proceedings of ACM International Conference on Multimedia*, pp. 313–324, 1997.

[5] J. R. Smith and S. F. Chang, “VisualSEEk: A Fully Automated Content-Based Image Query System,” in *Proceedings of ACM International Conference on Multimedia*, pp. 87–98, 1997.

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

[7] M. Baştan, H. Çam, U. Güdükbay, and Özgür Ulusoy, “BilVideo-7: An MPEG-7-Compatible Video Indexing and Retrieval System,” *IEEE MultiMedia*, vol. 17, no. 3, pp. 62–73, 2009.

[8] M. Baştan, H. Çam, U. Güdükbay, and Ö. Ulusoy, "An MPEG-7 Compatible Video Retrieval System with Integrated Support for Complex Multimodal Queries," Bilkent University, Tech. Rep., 2009.
Online: http://www.cs.bilkent.edu.tr/tech-reports/2009/BU-CE-0905.pdf

[9] "BilVideo-7 Web Site," 2010.
Online: http://www.cs.bilkent.edu.tr/~bilmdg/bilvideo-7

[10] C. Faloutsos, W. Equitz, M. Flickner, W.Niblack, D. Petkovic, and R. Barber, "Efficient and Effective Querying by Image Content," *Journal of Intelligent Information Systems*, vol. 3, no. 3-4, pp. 231–262, 1994.

[11] T. T. Gevers and A. W. M. Smeulders, "PicToSeek: Combining Color and Shape Invariant Features for Image Retrieval," *IEEE Transactions on Image Processing*, vol. 9, no. 1, pp. 102–119, 2000.

[12] J. L. James Z. Wang and G. Wiederhold, "SIMPLIcity: Semantics-Sensitive Integrated Matching for Picture Libraries," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 9, pp. 947–963, 2001.

[13] "ALIPR: Automatic Photo Tagging and Visual Image Search," 2010.
Online: http://alipr.com

[14] A. Pentland, R. W. Picard, and S. Sclaroff, "Photobook: Content-Based Manipulation of Image Databases," *International Journal of Computer Vision*, vol. 18, no. 3, pp. 233–254, 1996.

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

[16] M. E. Dönderler, E. Saykol, Ö. Ulusoy, and U. Güdükbay, "BilVideo: A Video Database Management System," *IEEE Multimedia*, vol. 10, no. 1, pp. 66–70, 2003.

[17] 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.

[18] O. Küçüktunç, U. Güdükbay, and Ö. Ulusoy, “A Natural Language-Based Interface for Querying a Video Database,” *IEEE Multimedia*, vol. 14, no. 1, pp. 83–89, 2007.

[19] N. B. Özgür, M. Koyuncu, and A. Yazıcı, “An Intelligent Fuzzy Object-Oriented Database Framework for Video Database Applications,” *Fuzzy Sets and Systems*, vol. 160, no. 15, pp. 2253–2274, 2009.

[20] J. Sivic and A. Zisserman, “Video Google: A Text Retrieval Approach to Object Matching in Videos,” in *Proceedings of IEEE International Conference on Computer Vision*, pp. 1–8, 2003.

[21] J. Sivic and A. Zisserman, “Video Google: Efficient Visual Search of Videos,” *Toward Category-Level Object Recognition*, pp. 127–144, 2006.

[22] A. Zisserman and J. Sivic, “Efficient Visual Search for Objects in Videos,” *Proceedings of the IEEE*, vol. 96, no. 4, pp. 548–566, 2008.

[23] C. Diou, G. Stephanopoulos, N. Dimitriou, P. Panagiotopoulos, C. Papachristou, A. Delopoulos, H. Rode, T. Tsikrika, A. P. de Vries, D. Schneider, J. Schwenninger, M. L. Viaud, A. Saulnier, P. Altendorf, B. Schroter, M. Elser, A. Rego, A. Rodriguez, C. Martínez, I. Etxaniz, G. Dupont, B. Grilheres, N. Martin, N. Boujemaa, A. Joly, R. Enficiaud, A. Verroust, S. Selmi, and M. Khadhraoui, “VITALAS at TRECVID-2009,” in *Proceedings of TREC Video Retrieval Evaluation Workshop*, Gaithersburg, USA, 2009.

[24] C. G. M. Snoek, K. E. A. van de Sande, O. de Rooij, B. Huurnink, J. R. R. Uijlings, M. van Liempt, M. de Rijke, J. M. Geusebroek, T. Gevers, M. Worring, A. W. M. Smeulders, D. C. Koelma, M. Bugalho, I. Trancoso, F. Yan, M. A. Tahir, K. Mikolajczyk, and J. Kittler, “The MediaMill TRECVID 2009 Semantic Video Search Engine,” in *Proceedings of TREC Video Retrieval Evaluation Workshop*, Gaithersburg, USA, 2009.

[25] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, “Content-Based Image Retrieval at the End of the Early Years,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 12, pp. 1349–1380, 2000.

[26] R. C. Veltkamp, M. Tanase, and D. Sent, “Content-Based Image Retrieval Systems: A Survey,” Department of Computing Science, Utrecht University, Tech. Rep., 2000.
Online: http://aa-lab.cs.uu.nl/cbirsurvey/cbir-survey/cbir-survey.html

[27] M. S. Lew, N. S. C. Djeraba, and R. Jain, “Content-based Multimedia Information Retrieval: State of the Art and Challenges,” *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 2, no. 1, pp. 1–19, 2006.

[28] Y. Liu, D. Zhang, G. Lu, and W.-Y. Ma, “A Survey of Content-based Image Retrieval with High-level Semantics,” *Pattern Recognition*, vol. 40, no. 1, pp. 262–282, 2007.

[29] R. Datta, D. Joshi, J. Li, and J. Z. Wang, “Image Retrieval: Ideas, Influences, and Trends of the New Age,” *ACM Computing Surveys*, vol. 40, no. 2, pp. 1–60, 2008.

[30] W. Ren, S. Singh, M. Singh, and Y. Zhu, “State-of-the-art on Spatio-temporal Information-based Video Retrieval,” *Pattern Recognition*, vol. 42, no. 2, pp. 267–282, 2009.

[31] “ISO/IEC 15938-6: Information Technology – Multimedia Content Description Interface – Part 10 & 11: MPEG-7 Schema Files,” 2010. Online: http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-7_schema_files

[32] “XML Schema 1.1,” 2010. Online: http://www.w3.org/XML/Schema

[33] “ISO/IEC 15938-6:2003: Information Technology – Multimedia Content Description Interface – Part 6: Reference software,” 2010.
Online: http://standards.iso.org/ittf/PubliclyAvailableStandards/index.html

[34] H. Eidenberger, “How Good are the Visual MPEG-7 Features?” in *Proceedings of IEEE Visual Communications and Image Processing Conference*, pp. 476–488, 2003.

[35] 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, 2001.

[36] “ISO/IEC FCD 15938-12:2007: Information Technology – Multimedia Content Description Interface – Part 12: Query Format,” 2007. Online: http://www.chiariglione.org/mpeg/working_documents/mpeg-07/mp7qf/mp7qf-fcd.zip

[37] W. Bailer and P. Schallauer, “Detailed Audiovisual Profile: Enabling Interoperability Between MPEG-7 Based Systems,” in *Proceedings of International Multi-Media Modelling Conference*, pp. 217–224, 2006.

[38] 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.

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

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

[41] 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.

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

[43] A. Yazıcı, O. Yavuz, and R. George, “An MPEG-7 Based Video Database Management System,” *Spatio-Temporal Databases: Flexible Querying and Reasoning*, pp. 181–210, 2004.

[44] S. Arslan and A. Yazıcı, “An Efficient Image Retrieval System Using Ordered Weighted Aggregation,” *Granular Computing: At the Junction of Rough Sets and Fuzzy Sets*, vol. 224, pp. 43–54, 2008.

[45] “IBM VideoAnnEx Annotation Tool,” 2010.
Online: http://www.research.ibm.com/VideoAnnEx

[46] 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 *Lecture Notes in Computer Science*, vol. 4253, pp. 633–640, 2006.

[47] J. Löffler, K. Biatov, C. Eckes, and J. Köhler, “IFINDER: An MPEG-7-Based Retrieval System for Distributed Multimedia Content,” in *Proceedings of ACM International Conference on Multimedia*, pp. 431–435, 2002.

[48] A. C. Gkoritsas and M. Angelides, “COSMOS-7: A Video Content Modeling Framework for MPEG-7,” in *Proceedings of International Multimedia Modelling Conference*, pp. 123–130, 2005.

[49] 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 Symposium on Information and Communication Technologies*, pp. 1–6, 2004.

[50] M. Lux, J. Becker, and H. Krottmaier, “Caliph&Emir: Semantic Annotation and Retrieval in Personal Digital Photo Libraries,” in *Proceedings of Conference on Advanced Information Systems Engineering*, pp. 85–89, 2003.

[51] “Going Native: Making the Case for XML Databases,” 2010.
Online: http://www.xml.com/pub/a/2005/03/30/native.html

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

[53] “Tamino: Software AG XML Data Management Product,” 2010.
Online: http://www.softwareag.com/Corporate/products/wm/tamino/default.asp

[54] “XQuery 1.0: An XML Query Language,” 2010.
Online: http://www.w3.org/TR/xquery

[55] J. F. Allen, “Maintaining Knowledge About Temporal Intervals,” *Communications of ACM*, vol. 26, no. 11, pp. 832–843, 1983.

[56] H. Eidenberger, “Distance Measures for MPEG-7-based Retrieval,” in *Proceedings of ACM SIGMM International Workshop on Multimedia Information Retrieval*, pp. 130–137, 2003.

[57] C. G. M. Snoek, M. Worring, and A. W. M. Smeulders, “Early versus Late Fusion in Semantic Video Analysis,” in *Proceedings of ACM International Conference on Multimedia*, pp. 399–402, 2005.

[58] J. Kittler, “Combining Classifiers: A Theoretical Framework,” *Pattern Analysis and Applications*, vol. 1, no. 1, pp. 18–27, 1998.

[59] L. A. Alexandre, A. C. Campilho, and M. Kamel, “On Combining Classifiers Using Sum and Product Rules,” *Pattern Recognition Letters*, vol. 22, no. 12, pp. 1283–1289, 2001.

[60] Z. Tu, X. Chen, A. L. Yuille, and S.-C. Zhu, “Image Parsing: Unifying Segmentation, Detection, and Recognition,” *International Journal of Computer Vision*, vol. 63, no. 2, pp. 113–140, 2005.

[61] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman, “LabelMe: A Database and Web-Based Tool for Image Annotation,” *International Journal of Computer Vision*, vol. 77, no. 1, pp. 157–173, 2008.

[62] “LabelMe: The Open Annotation Tool,” 2010.
Online: http://labelme.csail.mit.edu

[63] J. Yuen, B. Russell, C. Liu, and A. Torralba, “LabelMe Video: Building a Video Database with Human Annotations,” in *Proceedings of IEEE International Conference on Computer Vision*, vol. 2, pp. 1–8, 2009.

[64] “LabelMe video: The Open Video Annotation Tool,” 2010.
Online: http://labelme.csail.mit.edu/VideoLabelMe

[65] X. G. i Nieto, N. Camps, and F. Marques, “GAT: A Graphical Annotation Tool for Semantic Regions,” *Multimedia Tools and Applications*, vol. 46, no. 2, pp. 155–174, 2010.

[66] A. Sorokin and D. Forsyth, “Utility Data Annotation with Amazon Mechanical Turk,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 1–8, 2008.

[67] “Amazon Mechanical Turk,” 2010. Online: http://www.mturk.com

[68] Z. Theodosiou, A. Kounoudes, N. Tsapatsoulis, and M. Milis, “MuLVAT: A Video Annotation Tool Based on XML-Dictionaries and Shot Clustering,” in *Proceedings of International Conference on Artificial Neural Networks*, pp. 913–922, 2009.

[69] J. Yuan, H. Wang, L. Xiao, W. Zheng, J. Li, F. Lin, and B. Zhang, “A Formal Study of Shot Boundary Detection,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 2, pp. 168–186, 2007.

[70] V. T. Chasanis, A. C. Likas, and N. P. Galatsanos, “Scene Detection in Videos Using Shot Clustering and Sequence Alignment,” *IEEE Transactions on Multimedia*, vol. 11, no. 1, pp. 89–100, 2009.

[71] A. F. Smeaton, P. Over, and A. R. Doherty, “Video Shot Boundary Detection: Seven Years of TRECVid Activity,” *Computer Vision Image Understanding*, vol. 114, no. 4, pp. 411–418, 2010.

[72] C. R. Huang, H. P. Lee, and C. S. Chen, “Shot Change Detection via Local Keypoint Matching,” *IEEE Transactions on Multimedia*, vol. 10, no. 6, pp. 1097–1108, 2008.

[73] C. Huang, C. Chen, and P. Chung, “Contrast Context Histogram–An Efficient Discriminating Local Descriptor for Object Recognition and Image Matching,” *Pattern Recognition*, vol. 41, no. 10, pp. 3071–3077, 2008.

[74] C. Harris and M. Stephens, “A Combined Corner and Edge Detector,” in *Proceedings of the Fourth Alvey Vision Conference*, pp. 147–151, 1988.

[75] B. T. Truong and S. Venkatesh, “Video Abstraction: A Systematic Review and Classification,” *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 3, no. 1, pp. 1–37, 2007.

[76] A. M. Ferman and A. M. Tekalp, “Efficient Filtering and Clustering Methods for Temporal Video Segmentation and Visual Summarization,” *Journal of Visual Communication and Image Representation*, vol. 9, no. 4, pp. 336–351, 1998.

[77] C. Taskiran, J. Y. Chen, A. Albiol, L. Torres, C. A. Bouman, and E. J. Delp, “ViBE: A compressed Video Database Structured for Active Browsing and Search,” *IEEE Transactions on Multimedia*, vol. 6, no. 1, pp. 103–118, 2004.

[78] D. Besiris, A. Makedonas, G. Economou, and S. Fotopoulos, “Combining Graph Connectivity & Dominant Set Clustering for Video Summarization,” *Multimedia Tools and Applications*, vol. 44, no. 2, pp. 161–186, 2009.

[79] A. K. Jain, M. N. Murty, and P. J. Flynn, “Data Clustering: A Review,” *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323, 1999.

[80] A. K. Jain, “Data Clustering: 50 Years Beyond K-means,” *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666, 2010.

[81] F. Porikli and Y. Wang, “Automatic Video Object Segmentation Using Volume Growing and Hierarchical Clustering,” *EURASIP Journal on Applied Signal Processing*, vol. 2004, no. 1, pp. 814–832, 2004.

[82] C. Carson, S. Belongie, H. Greenspan, and J. Malik, “Blobworld: Image Segmentation Using Expectation-Maximization and Its Application to Image Querying,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 8, pp. 1026–1038, 2002.

[83] W. Y. Ma and B. S. Manjunath, “EdgeFlow: A Technique for Boundary Detection and Segmentation,” *IEEE Transactions on Image Processing*, vol. 9, no. 8, pp. 1375–1388, 2000.

[84] J. Shi and J. Malik, “Normalized Cuts and Image Segmentation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[85] Y. Deng and B. S. Manjunath, "Unsupervised Segmentation of Color-Texture Regions in Images and Video," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 8, pp. 800–810, 2001.

[86] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient Graph-Based Image Segmentation," *International Journal of Computer Vision*, vol. 59, no. 2, pp. 167–181, 2004.

[87] R. Nock and F. Nielsen, "Statistical Region Merging," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 11, pp. 1452–1458, 2004.

[88] R. Nock and F. Nielsen, "Semi-supervised Statistical Region Refinement for Color Image Segmentation," *Pattern Recognition*, vol. 38, no. 6, pp. 460–465, 2005.

[89] "OpenCV: Open Source Computer Vision Library," 2010.
Online: http://opencvlibrary.sourceforge.net

[90] L. Itti, C. Koch, and E. Niebur, "A Model of Saliency-Based Visual Attention for Rapid Scene Analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 11, pp. 1254–1259, 1998.

[91] Y. F. Ma and H. J. Zhang, "Contrast-based Image Attention Analysis by Using Fuzzy Growing," in *Proceedings of ACM International Conference on Multimedia*, pp. 374–381, 2003.

[92] S. Kwak, B. Ko, and H. Byun, "Automatic Salient-Object Extraction Using the Contrast Map and Salient Points," in *Lecture Notes in Computer Science*, vol. 3332, pp. 138–145, 2004.

[93] U. Rutishauser, D. Walther, C. Koch, and P. Perona, "Is Bottom-up Attention Useful for Object Recognition?" in *Proceedings of International Conference on Computer Vision Pattern Recognition*, vol. 2, pp. 37–44, 2004,.

[94] F. Ge, S. Wang, and T. Liu, "Image-Segmentation Evaluation From the Perspective of Salient Object Extraction," in *Proceedings of IEEE Computer Society*

Conference on Computer Vision and Pattern Recognition, vol. I, pp. 1146–1153, 2006.

[95] B. C. Ko and J.-Y. Nam, "Automatic Object-of-Interest Segmentation from Natural Images," in *Proceedings of International Conference on Pattern Recognition*, pp. 45–48, 2006.

[96] X. Hou and L. Zhang, "Saliency Detection: A Spectral Residual Approach," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2007.

[97] T. Liu, J. Sun, N. N. Zheng, X. Tang, and H. Y. Shum, "Learning to Detect a Salient Object," in *Proceedings of International Conference on Computer Vision Pattern Recognition*, pp. 1–8, 2007.

[98] G. Qiu, X. Gu, Z. Chen, Q. Chen, and C. Wang, "An Information Theoretic Model of Spatiotemporal Visual Saliency," in *Proceedings of IEEE International Conference on Multimedia and Expo*, pp. 1806–1809, 2007.

[99] T. J. Williams and B. A. Draper, "An Evaluation of Motion in Artificial Selective Attention," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops*, vol. 3, p. 85, 2005.

[100] O. L. Meur, D. Thoreau, P. L. Callet, and D. Barba, "A Spatio-temporal Model of the Selective Human Visual Attention," in *Proceedings of IEEE International Conference on Image Processing*, vol. 3, pp. 1188–1191, 2005.

[101] Y. Liu, C.-S. Bouganis, and P. Y. K. Cheung, "A Spatiotemporal Saliency Framework," in *Proceedings of IEEE International Conference on Image Processing*, pp. 437–440, 2006.

[102] L. Elazary and L. Itti, "Interesting Objects are Visually Salient," *Journal of Vision*, vol. 8, no. 3, article no. 3, 15 pages, 2008.

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

[104] T. Sevilmiş, M. Baştan, U. Güdükbay, and Özgür Ulusoy, “Automatic Detection of Salient Objects and Spatial Relations in Videos for a Video Database System,” *Image and Vision Computing*, vol. 26, no. 10, pp. 1384–1396, 2008.

[105] S. Goferman, L. Zelnik-Manor, and A. Tal, “Context-Aware Saliency Detection,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2010.

[106] T. Kadir and M. Brady, “Saliency, Scale and Image Description,” *International Journal of Computer Vision*, vol. 45, no. 2, pp. 83–105, 2001.

[107] K. Crammer and Y. Singer, “On the Algorithmic Implementation of Multiclass Kernel-based Vector Machines,” *Journal of Machine Learning Research*, vol. 2, pp. 265–292, 2002.

[108] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun, “Support Vector Machine Learning for Interdependent and Structured Output Spaces,” in *Proceedings of International Conference on Machine Learning*, pp. 104–112, 2004.

[109] T. Joachims, “Multi-Class Support Vector Machine,” 2010.
Online: http://svmlight.joachims.org/svm_multiclass.html

[110] D. Comaniciu, V. Ramesh, and P. Meer, “Kernel-based Object Tracking,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, pp. 564–577, 2003.

[111] A. Yilmaz, O. Javed, and M. Shah, “Object Tracking: A Survey,” *ACM Computing Surveys*, vol. 38, no. 4, article no. 13, 45 pages, 2006.

[112] I. Leichter, M. Lindenbaum, and E. Rivlin, “Bittracker–A Bitmap Tracker for Visual Tracking under Very General Conditions,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 9, pp. 1572–1588, 2008.

[113] M. Yang and Y. W. G. Hua, “Context-Aware Visual Tracking,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 7, pp. 1195–1209, 2009.

[114] V. Mahadevan and N. Vasconcelos, “Saliency-based Discriminant Tracking,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1007–1013, 2009.

[115] “Saliency Toolbox.” Online: http://www.saliencytoolbox.net

[116] B. Alexe, T. Deselaers, and V. Ferrari, “What is an Object?” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2010.

[117] “wxWidgets: Open Source Cross-Platform GUI Library,” 2010. Online: http://www.wxwidgets.org

[118] “FFmpeg: Cross-platform Audio Video Library,” 2010. Online: http://ffmpeg.mplayerhq.hu

[119] “Xerces-C++ XML Parser,” 2010. Online: http://xerces.apache.org/xerces-c

[120] A. F. Smeaton, P. Over, and W. Kraaij, “Evaluation Campaigns and TRECVid,” in *Proceedings of ACM International Workshop on Multimedia Information Retrieval*, pp. 321–330, 2006.