STRUCTURAL AND EVENT BASED

MULTIMODAL VIDEO DATA MODELING

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCE
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

HAKAN ÖZTARAK

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

DECEMBER 2005

Approval of the Graduate School of Natural and Applied Science

_____

Prof. Dr. Canan Özgen

Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

_____

Prof. Dr. Ayşe Kiper

Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

_____

Prof. Dr. Adnan Yazıcı

Supervisor

Examining Committee Members

Prof. Dr. Özgür Ulusoy (Bilkent Üniv., CENG) _____

Prof. Dr. Adnan Yazıcı (METU, CENG) _____

Assoc. Prof. Dr. Nihan Kesim Çiçekli (METU, CENG) _____

Dr. Cengiz Erbaş (Aselsan) _____

Yük. Müh. Yakup Yıldırım (Havelsan) _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name : Hakan ÖZTARAK

Signature :

ABSTRACT

STRUCTURAL and EVENT BASED
MULTIMODAL VIDEO DATA MODELING

Öztarak, Hakan
M.S., Department of Computer Engineering
Supervisor: Prof Dr. Adnan Yazıcı
December 2005, 119 Pages

Investments on multimedia technology enable us to store many more reflections of the real world in digital world as videos. By recording videos about real world entities, we carry a lot of information to the digital world directly. In order to store and efficiently query this information, a video database system (VDBS) is necessary. In this thesis work, we propose a structural, event based and multimodal (SEBM) video data model for VDBSs. SEBM video data model supports three different modalities that are visual, auditory and textual modalities and we propose that we can dissolve these three modalities with a single SEBM video data model. This proposal is supported by the interpretation of the video data by human. Hence we can answer the content based, spatio-temporal and fuzzy queries of the user more easily, since we store the video data as the way that s/he interprets the real world data. We follow divide and conquer technique when answering very complicated queries. We have implemented the SEBM video data model in a Java based system that uses XML for representing the SEBM data model and Berkeley XML DBMS for storing the data based on the SEBM prototype system.

Keywords – Multimodal Video Modeling and Retrieval, Audio, Visual, Textual, Spatial, Temporal, Fuzzy, Berkeley XML DBMS, Java, XML.

ÖZ

YAPISAL ve OLAY TEMELLİ ÇOK BİÇİMLİ VİDEO VERİ MODELLEMESİ

Öztarak, Hakan
Yüksek Lisans, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi: Prof Dr. Adnan Yazıcı
Aralık 2005, 119 Sayfa

Çoklu ortam teknolojileri üzerine yapılan yatırımlar, gerçek hayatın yansımalarını dijital hayatta videolar şeklinde daha fazla saklayabilmemize olanak sağlamaktadır. Gerçek hayat varlıkları hakkında videolar kaydederek, birçok bilgiyi doğrudan dijital dünyaya taşımaktayız. Bu bilgilerin saklanması ve verimli biçimde sorgulanması için bir video veri tabanı (VVT) gereklidir. Bu tez çalışmasında, VVT'ler için yapısal, olay temelli ve çok biçimli bir video veri modeli (SEBM) sunulmaktadır. SEBM video veri modeli üç farklı biçimi ki bunlar görsel, işitsel ve yazısal biçimlerdir; desteklemektedir ve biz bu üç biçimin tek bir SEBM video veri modeli içerisinde entegrasyonunu önermekteyiz. Bu öneri insanın video verisini algılayış biçimi ile de desteklenmektedir. Böylelikle, biz video verisini kullanıcının gerçek dünya verisini algılayış biçimiyle sakladığımızdan dolayı, içeriğe dayalı, uzaysal-zamansal ve bulanık kullanıcı sorgulamalarını daha kolay cevaplayabilmekteyiz. Çok karışık sorgulamalara yanıt ararken parçala ve yönet tekniğini kullanıyoruz. XML'i SEBM veri modelini tasvir etmekte, Berkeley XML DBMS'i SEBM veri modelini saklamakta kullanan Java tabanlı bir SEBM prototip sistem gerçekleştirdik.

Anahtar Kelimeler - Çok Biçimli Video Modellemesi ve Erişimi, İşitsel, Görsel, Uzaysal, Zamansal, Bulanık, Berkeley XML DBMS, Java, XML.

To My Parents…

ACKNOWLEDGEMENTS

I would like to thank to my open-minded supervisor, Prof. Dr. Adnan Yazıcı for his guidance, advice, criticism, encouragements and insight throughout the research.

I am thankful to my company ASELSAN Inc. for letting and supporting of my thesis.

I am very grateful to my family for their believing to me.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS / ABBREVIATIONS

AVI	:	Audio Video Interleave
AVIS	:	Advanced Video Information System
BDB XML	:	Berkeley DB XML
DCT	:	Discrete Cosine Transform
HMM	:	Hidden Markov Model
HTML	:	Hypertext Mark-up Language
JMF	:	Java Media Framework
MBR	:	Minimum Bounding Rectangle
OCR	:	Optional Character Recognition
OVID	:	Object- oriented Video Information Database
SEBM	:	Structural, Event Based and Multimodal
STRG	:	Spatio-Temporal Region Graph
STRG-QL	:	Spatio-Temporal Region Graph Query Language
TV	:	Television
VDBS	:	Video Database System
XML	:	eXtended Mark-up Language

CHAPTER 1

INTRODUCTION

Since multimodality of the video data comes from the nature of the video, it is one of the important research topics for the database community. Videos consist of visual, auditory and textual channels [1]. These channels bring the concept of multimodality. Modeling and storing multimodal data of a video is a problem, because users want to query these channels from stored data in a video database system (VDBS) efficiently and effectively. In this thesis work, we propose a **s**tructural, **e**vent **b**ased and **m**ultimodal (SEBM) video data model for VDBSs. SEBM video data model supports three different modalities that are visual, auditory and textual modalities and we propose that we can dissolve these three modalities with a single SEBM video data model.

Definition of multimodality is given by Snoek et. al [1] as:

Definition 1 (Multimodality): The capacity of an author of the video document to express a predefined semantic idea, by combining a layout with a specific content, using at least two information channels.

Like in [2] and [1] we use three kinds of modalities:

- *Visual modality*: contains everything, either naturally or artificially created, that can be seen in the video document;
- *Auditory modality*: contains the speech, music, and environmental sounds that can be heard in the video document;

- *Textual modality*: contains textual resources that can be used to describe the content of the video document.

When we see something happening in video, we are interested in visual part of the video. Most probably, one or more entities in the video are acting. When we hear some sound, we are interested in auditory part of the video. Most probably, some sound is created by some entities either saying or making some sound. When we see something written either on the screen or on some entity, we are interested in textual part of the video. All of these visual, audial and textual domains should be considered in a multimodal video data model. Information extracted from three different domains is embedded into our SEBM video data model. In the visual domain, we extract visual events, visual objects, and spatial and temporal characteristics. In the auditory domain, we extract auditory events and auditory objects with their temporal characteristics. In textual domain, we consider visible texts.

Single video is composed of sequential frames, which are individual images. Each frame has individual image properties like color or shape. Every image can contain objects, positioned on the image. However, when we arrange these images sequentially, we can see that these objects are a part of some events. These objects may move or stop even though this stopping event, in general, is not considered as an action. As a result, every object can be thought as doing something. We developed a SEBM video data model by considering this fact, which is being a part of an event. The SEBM model is based on telling about what is going on in the video. We are interested in the structure of what is happening. We follow an event-based approach for the model, because human interprets the real world like that. In order to find the answer for the question of how the human interprets the video data, we use the following example: If we want someone to sit down and watch a particular video and then while watching, ask her/him what s/he sees on the screen or what is happening in the video; s/he may describe it for us by some sentences such as:

- There is a party happening in the house.
- In the party there is a dinner.
- There is a melody playing in the background.
- John is sitting at upper left corner of the dinner table.
- Kylie is standing near John.
- I understand that John is the Kylie's brother.
- John is saying "Bon appetite" to Kylie.
- Kylie is giving hamburger and cola to John.
- John is wearing an orange T-shirt.
- There is happy birthday writing on the screen.

As it can be seen, there are some patterns while telling what is going on in a specific video. These patterns give us some clues about what s/he might query. S/he is interested in about parts, entities, actions and relations between them in the video. Moreover, s/he uses patterns that show us the structure of what is happening, like telling who is doing what. By considering these interpretations, we introduce the SEBM video data model, which is mainly a combination of five different sub-models:

1. Video Sequence Sub-Model
2. Video Shot Sub-Model
3. Video Object Sub-Model
4. Video Object Characteristics Sub-Model
5. Video Event Sub-Model

First two of these sub-models are about temporally segmenting video into smaller parts, the video segmentation part. Third and fourth sub-models are about extracting the entities, the video entities part. Last one is about extracting the

actions in the video, the video actions part. Figure 1.1 shows the hierarchical structural of these five sub-models.

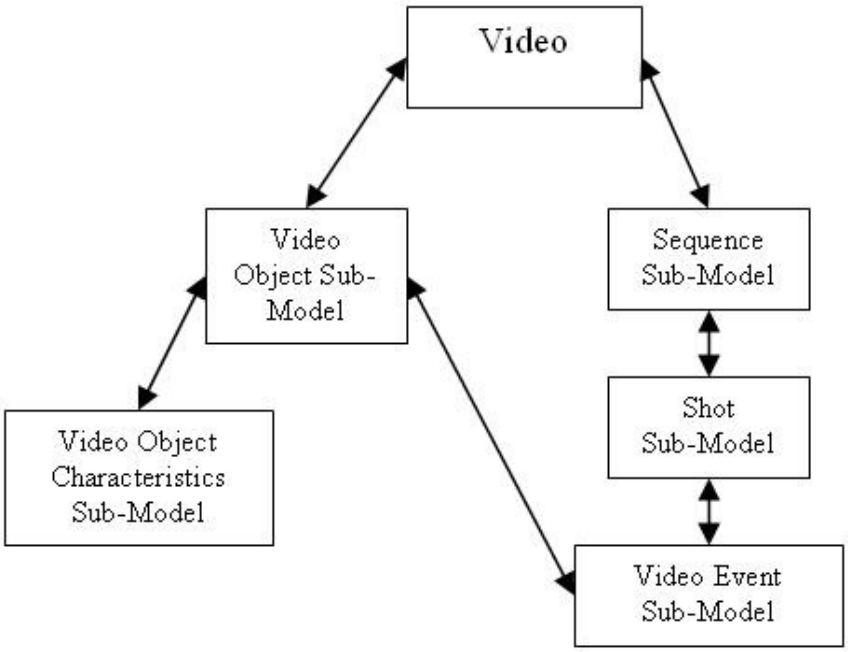


Figure 1.1 - Hierarchical Structure of Video Sub-Models

Nowadays researches in modelling video data are concentrating on efficient storing and effective querying the multimodal data and integrating it with temporal and spatial relationships. While modelling such data, both automatic annotation and effective querying are important aspects too, because modelling is an intermediate step between them. Snoek et. al. give the definition of multimodality and focus on similarities and differences between modalities, [1]. Babaguchi et. al investigate multimodal streams and try to extract semantic information from the video, [27]. They define collaborations between streams when extracting the semantic from the video. Oomoto et. al don't work on multimodality but investigates the video object concept which is a base for spatio-temporal works [7]. Day et. al extend the spatio-temporal semantic of video objects [5]. Ekin et. al introduce object characteristics, and actors in events [4]. Köprülü et. al. propose a model that defines spatial and temporal relationships of the object in visual domain which includes fuzziness, [3]. Durak in [2], extends the model proposed in [3]. She introduces a multimodal

extension of the model and gives two different structures for visual, auditory and textual modalities. Huang et. al. introduce a layered approach for multimodality like in networking concept, [37]. BilVideo is a good example for a Video Database Management System which uses semi-automatic annotation of the video considering spatio-temporal concepts, [8].

Main contributions of our work, shown in Figure 1.2, can be summarized as follows:

1. In this study we introduce a new multimodal video data model, SEBM, to handle three different modalities, visual, auditory and textual. The proposed multimodal video data model is structural and event-based and it is based on human interpretation of these three modalities. This interpretation is like telling what is happening in videos. In this study we think of videos as reflections of the real world in digital world. If one can express information in digital world as human does in real world, then we think that all of the queries coming from a user can be handled more accurately and effectively. So we can bypass the problem of handling the models in different data structures as in [2].
2. In this study, we introduce actor entities in video events in multimodal domains which are only defined for visual domains [4]. These entities give us the ability to express the structure of events. Moreover we introduce object characteristics that involve a particular feature of an object or relation of an object with other objects in multimodal domains different from [7].
3. We follow divide and conquer approach in query processing to answer complex, very long, nested and conjunctive spatial, temporal, content-based and possibly fuzzy video queries. This approach gives us the ability to deal with much more complex queries different than ones in [2] and [3].
4. We implement a prototype system that uses our model. This system uses automatization while segmenting the video temporally and has the ability to work on multimodal data.

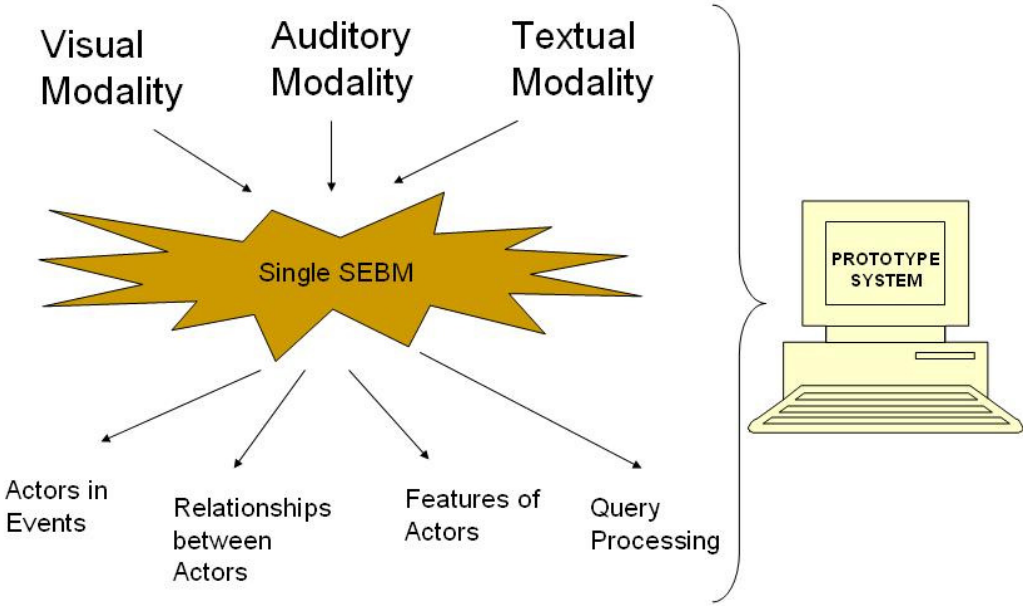


Figure 1.2 - SEBM Data Model Features

The rest of the thesis is organized as follows: Section 2 presents the researches done at the preparation phase of SEBM video data model and SEBM prototype system. Section 3 presents the SEBM video data model. All the sub-models are explained at this section. In Section 4, we describe how the querying is done on SEBM video data model. Query types are explained in details and query examples are given in each section. Implementation of a prototype system is discussed in Section 5. Every technology we used is investigated and why we choose the corresponding technology is discussed. The usage of the technology is explained by giving screenshots from the SEBM prototype system. The parts are related with sub-models. Some discussions about SEBM video data model and SEBM prototype system are addressed at Section 6. The last section provides conclusion with some future extensions of our model.

CHAPTER 2

RELATED STUDIES

Great amount of researches have been done on video databases in the world. In order to develop a complete video database system, which considers multimodality, automatic annotation, effective and efficient querying; some of them, which are related to these subjects, are inspected. The related researches are classified according to their major subjects.

2.1. Visual Modality

A method for detecting the structure of the video documents visually is given in [29]. In this work Javed et. al investigate talk and game shows on TV. They propose a method to detect guests and hosts shots. Their approach is as follows: The basic observation used is that in most talk shows, the same person is host for the duration of the program but guests keep on changing. Also host shots are typically shorter since only the host asks questions. For a given show, the key frames of the N shortest shots containing one detected face are correlated in time to find the shot most often repeated. The key host frame is then compared against all key frames to detect all similar host shots, and guest shots.

In [14] Chen et. al. explain the methods of automatic object identification containing overlapping situations. Pan et. al. work on slow-motion replays to extract information from video in [33]. They propose a method that localizes events by detecting slow motion replays. The slow motion replays are modelled and detected by an HMM (Hidden Markov Model).

Chen et. al. investigate broadcast news videos in order to find a specific person by using image processing techniques in [53]. They use clues like text, time scene content and face.

2.2. Auditory Modality

Rui et.al. investigate audio-track of the videos to extract the highlights of baseball programs, [23]. They propose that they don't need to compute on visual part of the video, since it is quite expensive to work on the visual part. Their purpose in this work is to reduce the cost of investigation. They need reducing the cost, because they want to make the computation in user's side which is a local set-top box on TV's, their target delivery vehicle.

Patel classifies videos using audio in [52]. Wold et. al work on classifying sounds like environmental sound or crowd sound in [32] and use auditory features such as loudness, pitch, bandwidth, harmony etc. for this purpose. Hence they try to achieve automatic annotation of the video through investigating audio tracks of the video.

Chang et. al. propose audio analysis as an alternative tool against visual analysis for parsing the videos, especially sports videos in [46]. Their goal is to detect football highlights. Energy of the sound is measured to find the events of

highlights. Audio analysis for video annotation is also investigated by Hauptmann et. al. in [50]. They work on Informedia digital video library system [19]. Tzenetakis et. al. emphasize the difficulties of audio analysis of videos in [54].

2.3. Textual Modality

Text extraction is another related research area for automatic annotation of video. Complex algorithms are used to solve this problem. Zhong et al. in [25] propose a method utilizing the DCT (Discrete Cosine Transform) coefficients of video which is compressed in MPEG format. They distinguish the text by segmenting the characters from the background. In [31] Li et. al. work on OCR (Optional Character Recognition) methods for detecting text in video, but their modules get slower when text number per frames gets bigger. In [59] Wu et. al. search for the ways of extracting the text and road signs for a video application that is used in driving assistant system. They use divide and conquer approach to detect the road signs and detection of other texts and use incremental approach as the texts are getting bigger and bigger as the vehicle approaches it.

2.4. Multimodality

Babaguchi et. al. investigate in [27] a collaborative analysis of the multimodal streams in order to extract semantic information from the video. They define intermodal collaboration as a strategy of collaborative processing, taking account of the multimodal streams semantically depending on each other. They propose that the computation for analyzing the visual stream is most costly and the use of other streams may be capable of reducing it. They work on three different collaborations: a) Visual and text streams, b) Visual, auditory and text streams and c) the graphics streams and external metadata. Tsekeridou et. al. integrate the auditory and visual modalities to detect speech, silence, speaker identities, no face

shot, face shot, and talking face shot using knowledge-based rules in [34]. First, talking people are detected by detecting faces in the camera shots, and then a knowledge-based measure is evaluated based on the amount of speech in the shot. Hauptmann et. al. investigate audio and visual information of video using speech recognition and image processing techniques in [45]. They classify queries according to scenes, textual information, peoples or camera operations. They offer methods for extracting textual information from videos. In order to answer textual queries more effectively Duygulu et. al. investigate on linking the textual data with visual data in [57, 58]. They use probability analysis to match the text with the visual frames.

2.5. Video Data Modeling

Oomoto et. al. propose in [7] an object oriented model, OVID, for the video databases. They introduce the video-object concept. Moreover, they propose interval inclusion inheritance between objects. Day et. al. model the spatio-temporal semantics of data in an object oriented way in [5]. Huang et. al. introduce a layered approach for multimedia content representation and storage in [37]. They follow multimodal approach that supports visual, auditory and text, for documents. They propose client/server architecture which uses content hierarchy that is content structure layer, logical event layer, physical event layer and raw data layer, for broadcast programs. Lee et. al. focus on querying the spatio-temporal video data in [40]. Firstly they develop STRG (Spatio-Temporal Region Graph) indexing mechanism on video data. STRG presented in [41] represents spatial and temporal visual information of video objects. Then they present graph based query language STRG-QL for this graph. Moreover they introduce a rule based query optimization mechanism.

Fan et. al. express the video in a multilevel way in [47]. They investigate two approaches to do the multi levelling. First one is the shot-based approach that

consists of cluster layer, sub-cluster layer, scene layer, shot layer, frame layer and region layer. Second one is the object based approach that consists of cluster layer, sub-cluster layer, scene layer, object layer, video object plane layer and region layer. By structuring the layer in a multilevel way, they propose that they guarantee high dimensional video indexing. This indexing mechanism speeds up querying process. In [9], Hacid et. al. define abstraction levels to model the video data, from low-level perceptual clues to semantic aspects of content.

2.6. Video Segmentation

Günsel et. al work on video segmentation in [26]. They propose a method for localization of news segments by channel logo detection and tracking and unsupervised syntactic scene change detection. Moreover they work on rule based news unit formation method based on the news episode structures. Pan et. al work on video segmentation in [33]. They propose shot motion characterization scheme which is a decision tree classifier built through a process of supervised learning. Another work on video segmentation is done in [28] by Ide et. al. They use image processing techniques to segment the video into shots. Then they apply semantic analysis on the shots. They define five different shot classes. They use face detection and lip movement detection methods to find *speech shots*. Then they use topic boundary detection to extract *anchor shots*. They separate *walking shots* by investigating oscillation of the camera. They classify *gathering shots* by detecting more than two similar sized facial regions in the same frame. Lastly they extract *computer graphics shots* by investigating the total duration of motionless frames. Hauptman et. al also works on Video Segmentation in [48]. Informedia Digital Video Library is their work area and they demonstrated that a fully automatic system can successfully extract story boundaries using available audio, video and closed captioning cues. In [49] Reb et. al. investigate a method of automatically determining the number of scenes in a given video sequence, and therefore automating the video segmentation process. Their methodology is based on the use

of fuzzy clustering of frames that are indexed using texture features. Zhang et. al. mention about difficulties while segmenting the video in [10]. They offer that video segmentation is necessary to extract high-level semantic meaning. Zhong et. al. present a segmentation framework to find unusual activities in videos through their work [56]. They divide the video into equal length of segments and classify the extracted features into prototypes, from which a prototype–segment co-occurrence matrix is computed.

2.7. Video Data Indexing

Another work that is done on video indexing is [30]. Cascia et. al. combine textual and visual information of video data in a single index in this work. They work on content based image retrieval from the web. They use latent semantic analysis to extract the textual information of the image from the HTML documents. DeMenthon et. al. develop a method for video indexing, retrieval and for action recognition in [42]. In this method, collections of videos are analyzed to produce spatio-temporal descriptors that summarize the location, color and dynamics of independently moving regions with only a small number of bytes. The similarities of sequences are defined using these descriptors.

2.8. Video Data Querying

In [35], Bolle et. al. investigate the research direction for video querying. They investigate how the user may query the video data. Such as the user may have once seen a particular piece of video and wants to retrieve it for viewing or reuse; or the user may be looking for a specific video but has never actually seen it before or the user may have only some vague idea of what it is that she or he is actually looking for. They explore how navigating and searching through video data is done. Snoek et. al work on multimodal queries in [36]. They propose a framework for

multimodal video data storage, but only the semantic queries and some simple temporal queries are supported. Furthermore, their visualized query interfaces are not powerful to perform all the possible queries. Yang et. al. explore efficient text querying in [60]. They work on locating named people from video transcripts and OCR text.

2.9. Video Semantic

In order to semantically classify videos in real-time systems, Zhou et. al investigate the use of video and audio content analysis, feature extraction and clustering techniques in [38]. They implement a supervised rule-based video classification system in order to use automatic segmentation. In [43] Hammiche et. al. express semantic video queries in tree form. They use ontology in parallel to find the answer of the queries. Moreover they use tree-embedding algorithms in order to relate the domain ontology.

2.10. Video Database Systems

An early work about video data storage is done in [24] by Adali et. al. They develop data structures to store spatial and temporal information of video data. This work has been accepted as base in [2 and 3]. Adali et. al develop a prototype system called AVIS (Advanced Video Information System). They define video objects, activity types and events with roles and teams. Video objects are the entities in video frames. They may be characters or objects. Activity types describe the subjects of a given video sequence. Events are kind of instances of activity types. In events, objects may have roles. Team is a set of objects that have roles in events. Adali et. al. define association map which associates events and objects with video frames, frame segment tree which stores frames in a binary tree, object, event and activity arrays, which stores the information about objects, events and

activities respectively. Oria et. al. explain in [11] how an efficient handling of complex multimedia data can be done in a framework.

Wactlar investigates what will be the new directions of the researches about video information extraction and summarization in [51]. In [12], Aref et. al. propose an environment called VDBMS (Video Database Management System), which provides a foundation for research, development, and experimentation in new areas of video database technology. Some query processing researches are done on VDBMS TestBed in [13]. Ulusoy et. al propose in [8] a system called BilVideo for spatio-temporal querying of VDBSs (Video Database System). They extract the objects and events by a module called fact-extractor. Zhou et. al. develop a web-based Video database systems called VideoQ in [39]. VideoQ system does automatic video shot analysis. User query visual attributes of the objects on client-side. Client sends the query to the server side and the database answers the query on the server side. They use feature space metrics to annotate the video visually. Another interesting feature of the VideoQ system is that, user express his/her queries visually like drawing it to the supplied screen.

Another video database system is iView and proposed in [44]. iVIEW system allows content indexing, searching and retrieval of text, audio and video material in both English and Chinese. Multimodality is supported. Cristel et. al work on video databases interfaces in [55]. They introduce multi-segment story board interfaces for video database systems.

CHAPTER 3

SEBM VIDEO DATA MODEL

Video is the reflection of the real world in digital world. In order to define the video data, we should investigate the data at the real world. In real world, there are objects or actors which correspond to video entities in digital world. They have names, properties and relationships. For example, John who is our friend is an entity. The table in our kitchen is another entity. The dog of our neighbour which always barks us is another entity. This object abstraction is valid for video data too. John, the dog or the table may be seen in our birthday video.

Time always passes; everything in real world is always doing something. For example John is eating hamburger on the table and the dog is barking as always it does. A plane is passing above my house. We can define everything to be doing something. They are parts of some events in real world either eating or barking. This event abstraction is valid for video data too.

SEBM video data model assumes that a video has objects which are parts of events happening at some interval during the timeline. These objects may have features that can be valid during the lifetime of the video and may have relations with other objects in the video. SEBM holds object information directly under the video

information. This situation can be thought of movies which list the actors of the film at the beginning or at the end. SEBM model segments the video through timeline in order to define every part more precisely. SEBM video model holds events under the segments of the video, because events happen at specific time interval in those segments. Every event has some actors which have roles in it. Events also have relationships with objects in order to define these roles.

Videos have not only semantic part but also physical part. SEBM relates semantics with physical part through events, because every event can be told to happen at a specific time interval and events may be seen at a specific region in frames that is present at that time interval. SEBM models events with temporal and spatial information.

SEBM focuses on events, because every object is a part of some event in videos. We propose that everything in the video can be defined by assigning some events to it. Modalities can be related with events. Visual modality has events that can be seen, auditory modality has events that can be heard and textual modality has events that written text can be seen on video frames. Every event may have relations with objects which are related to them in a structural way. Every object may have some specific role in an event.

We reach this structural and event based approach by investigating the queries of the users. User query video information as they interpret the real world. Users annotate and query the video different than single image which is static and has no time dimension. They aim explaining what is happening in the video by using events. Users always query the video as they annotate them in their minds. Hence we propose that if we model videos with the approach of users, we can then answer their queries more efficiently and effectively.

In subsection 3.1 how the video is segmented in SEBM is explained with reasons. Then video entities defined in SEBM are explored in subsection 3.2. Subsection 3.2 also explains how features of video objects and relations between them are expressed in SEBM video data model. In subsection 3.3, Video Actions part, the core part of the SEBM, video events are explained.

3.1. Video Segmentation

In [6], it is proposed that successful content-based video data management systems depend on three most important components:

1. Key-segments extraction,
2. Content descriptions,
3. Video retrieval.

Most of the time, people split a big problem into smaller problems and solve them individually. For key-segments extraction, when we ask somebody to tell about the video, s/he will divide the video semantically meaningful intervals like “there is a party happening” and “In the party there is a dinner”. We can see that these splitting come from the differences of parts like the changes in background or meaning. In our model, we temporally segment the whole video into meaningful intervals according to the semantics; for example “party”, “in house conversation” and “escape from prison”. We call each of these meaningfully different segments as “scene” or “sequence”.

In [10] Zhang et. al. define video shot as a video sequence that consists of continuous video frames for one camera action. Camera changing or motions and special edit effects cause shot boundaries. In our model, we further temporally divide sequences into smaller segments called “shots” according to the physical changing in parts, like colour. For example, if the camera or background changes at

a particular point in a particular sequence, we split the sequence from that point. Since this splitting is done according to physical changing, automatization is much easier in that part than sequence segmentation. On the other hand, sequence segmentation requires some artificial intelligence techniques, since sequence segmentation requires meaning changes in the video data. In Figure 3.1, video segmentation hierarchy is shown, in this figure a video is segmented into two sequences and n shots. $Sh_{(i)}$ is used for "Shot" numbered i.

Figure 3.1 - Video Segmentation Hierarchy

Video segmentation can be done automatically or manually according to the application. If the video will be segmented according to physical changes through the timeline, image-processing techniques can be helpful. On the other hand, if the video will be semantically segmented, an artificial intelligence technique should be used or user must manually enter necessary information for segmentation. In our system, semi-automatic segmentation is done. Firstly, video is segmented according to the background changes automatically. These segments are called shots. Then, these shots are manually grouped into sequences according to their semantics. From shots, sequences are created. These sequences form the whole Video. Figure 3.2 shows our segmentation process:

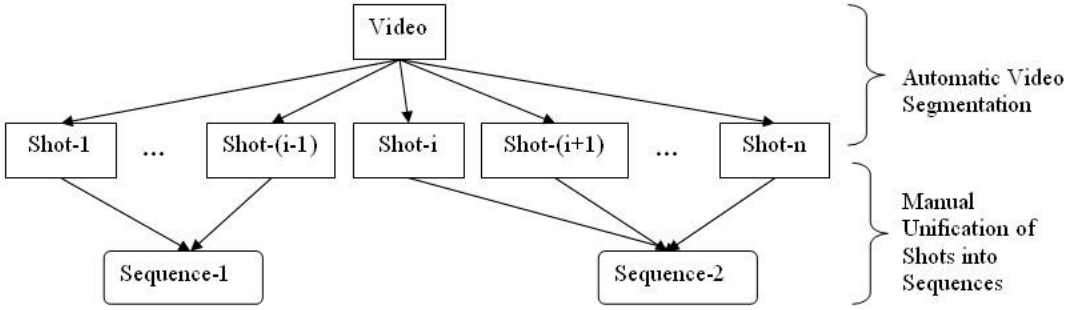


Figure 3.2 - Temporal Video Segmentation Process

Assume that video V is segmented into N different segments automatically. Then V= Sh(1)+Sh(2)+…+Sh(N) where N>0

1. $STime_V = STime_{Sh(1)}$ where STime=Start Time.

   Beginning point of the Sh(1) is equal to the beginning point of the Video.

2. $ETime_V = ETime_{Sh(N)}$ where ETime=End Time.

   Ending point of the Sh(N) is equal to the ending point of the Video.

3. $STime_{Sh(I)} = ETime_{Sh(i-1)}$ and

   $ETime_{Sh(i)} = STime_{Sh(i+1)}$

   Shots are sequenced without any gap in between.

Assume that from Sh(1) to Sh(i), Seq(1) is constructed. From Sh(i+1) to Sh(N) Seq(2) is constructed.

This means that Seq(1)=Sh(1)+…+Sh(i)

1. $STime_{Seq(1)} = STime_{Sh(1)}$

   Beginning point of the Seq(1) is equal to the beginning point of the Sh(1).

2. $ETime_{Seq(1)} = ETime_{Sh(i)}$

   Ending point of the Seq(1) is equal to the ending point of the Sh(i).

3. Since Video V is constructed from Sh(1) +…+ Sh(N), V is automatically extracted. Then V is;

V=Seq(1) (Sh(1)+…+Sh(i)) +

Seq(2) (Sh(i+1)+…+Sh(N)).

Shots of the video are grouped into the sequences manually according to the semantics of them. Every shot is a member of exactly one sequence. At the end of grouping, if video has N number of Shots and M number of sequence, N X M sequence-shot pairs are created. V is a set of Sequence-Shot Pairs.

3.1.1. Video Sequence (Scene) Sub-Model

Video is segmented according to the scene changes, where every scene has meaningful differences.

Seq=(ID, N, STime, ETime, SL) where Seq=Sequence.

1. ID=Unique Sequence ID (positive integer number)
2. N= Name of the sequence (String)
3. STime= Start time of the sequence in the video (positive double number)
4. ETime= End Time of the sequence in the video (positive double number bigger than STime)
5. SL (Shot List) = The list of shots that belong to the sequence. STime of the Sequence is equal to the Stime of the first shot according to the time in the list and Etime of the sequence is equal to the Etime of the last shot in the list.

In Figure 3.3, Video Sequence Sub-Model is shown with an example data in parenthesis.

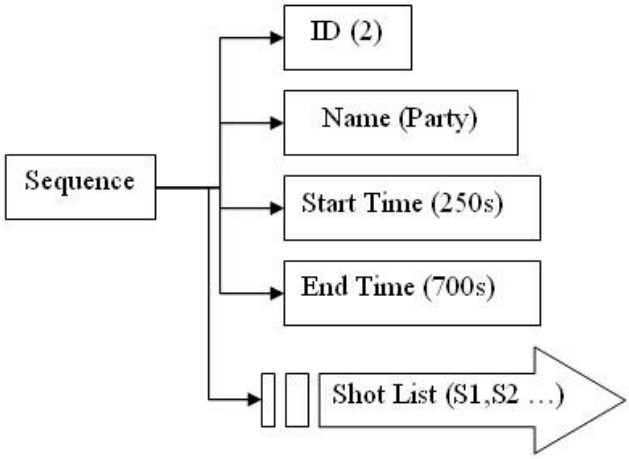


Figure 3.3 – Video Sequence Sub-Model

3.1.2. Video Shot Sub-Model

Every Sequence is further divided according to the camera or background changes that consist of the same kind of one or more events related to the same subject.

Shot=(ID, N, Stime, Etime, EL)

1. ID = Unique Shot ID (positive integer number)
2. N(Name) = Unique name in the same sequence
3. STime = Start time of the shot
4. ETime = End Time of the shot
5. Event List (EL) = The list that belongs to the shot. No same event belongs to more than one shot.

In Figure 3.4, Video Shot Sub-Model is shown with an example data in parenthesis.

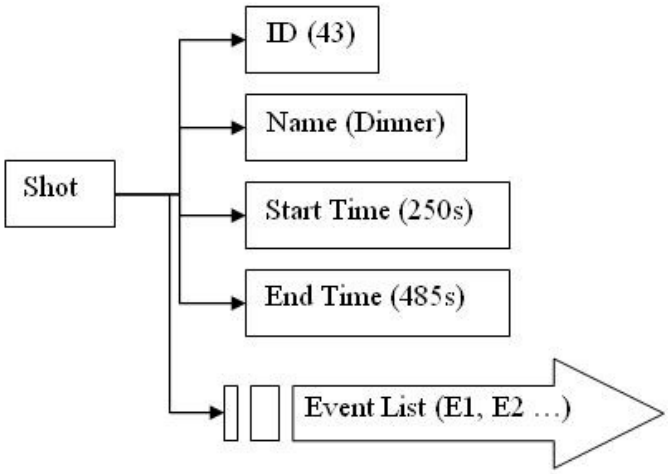


Figure 3.4 – Video Shot Sub-Model

3.2. Video Entities

Specific entities that are visible or tangible are called Video objects. How we can decide something is an object or not, is an issue. If we see or touch the entity in real world, we can declare it as a video object. Another way of telling what is happening in the video is to look where the entity is used in the sentence. We can declare the entity as a video object, if we use it as an object or subject (not a verb). For example; John, Kylie, T-Shirt, Hamburger etc. are Video objects. Ekin et. al. [4] call them as action units or interaction units. Objects may have features like "John is 25 years old", "Kylie has blue eyes" or may have relationships with other objects like "Kylie is the sister of Kylie", "John is a pilot of the plane".

3.2.1. Video Object Sub-Model

A video object refers to a semantically meaningful spatio-temporal entity. Formally, a video object is described in our model as VO={ID, N, CL, RL), where VO=Video Object.

1. ID = Unique ObjectID (positive integer number)
2. N = Name of the Object (String)
3. RL (Roller Event List) = The list of Event IDs where object VO has a role. By using this list, one can access the spatial or temporal information of the object VO directly through object itself.
4. CL (Characteristics List) = The list of characteristics. A particular video object may have a characteristic or not. If it has characteristics, these characteristics are expressed in Video Object Characteristics Sub-Model and inserted in CL.

In Figure 3.5, Video Object Sub-Model is shown with an example data in parenthesis.

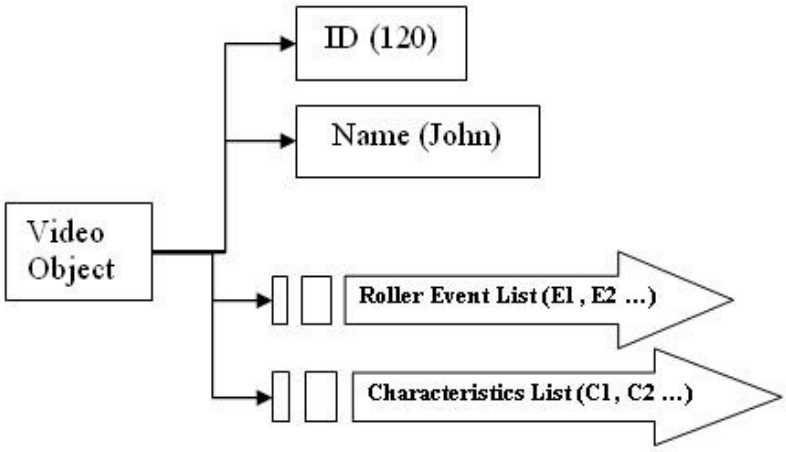


Figure 3.5 – Video Object Sub-Model

3.2.2. Video Object Characteristics Sub-Model

Characteristic is a feature of the Video object or a relation with other video objects. This sub-model is formally defined in our model as VOC={A, o2} where;

1. o2 = The subject of characteristic (Object Name or Feature)
2. A = Association between object that belongs to characteristic or the subject of characteristic, o2.

For example, object *John* may have a relation *brotherof* with *Kylie*. So, a video object named John will have a relation VOC={brotherof, Kylie}. Another example object *ball* may have a feature of *color blue.* Therefore, the object named *ball* will have a feature VOC={color, blue}. In Figure 3.6, Video Object Characteristics Sub-Model is shown with some example data in parenthesis.

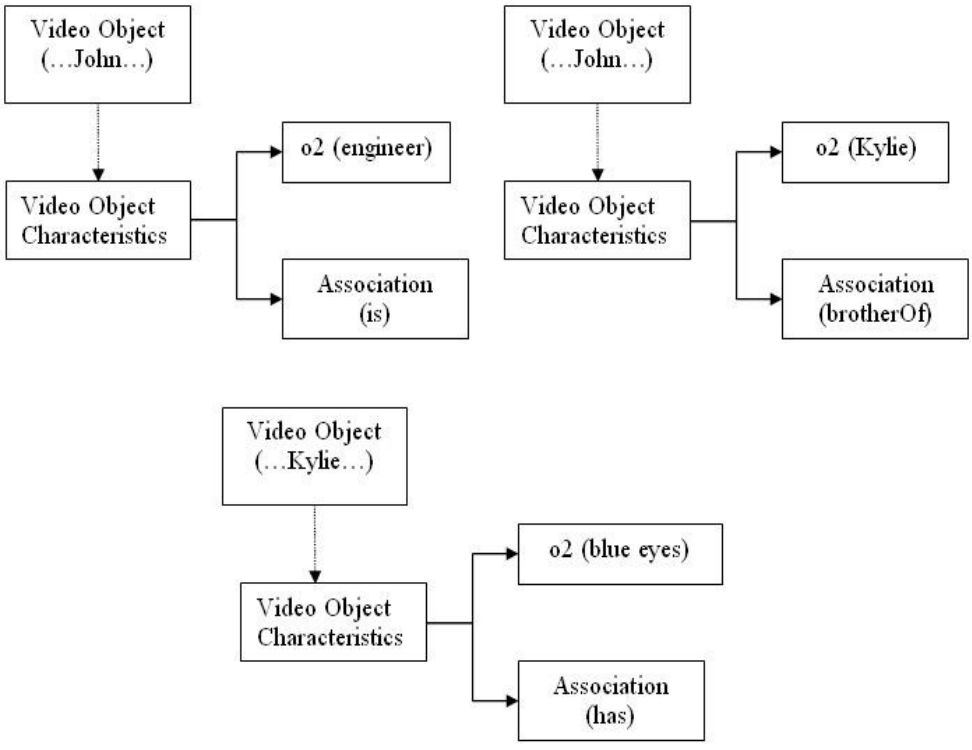


Figure 3.6 – Video Object Characteristics Sub-Model

3.3. Video Actions

Beside visual, audial and textual modalities, video has spatial and temporal aspects. In our sub-models these aspects are also considered. Spatial aspect is about position of an entity in a specific frame through the video. Spatial position in a specific frame can be given by two-dimensional coordinates. Temporal aspect is about time of a specific frame through the video. Hence a video element can be identified in a video with its frame position(s), X coordinate in frame(s), Y coordinate in frame(s). This identification is shown in Figure 3.7 where VE is for Video Event and VO is Video Object. Frame(i) is the ancestor Frame(i+1).

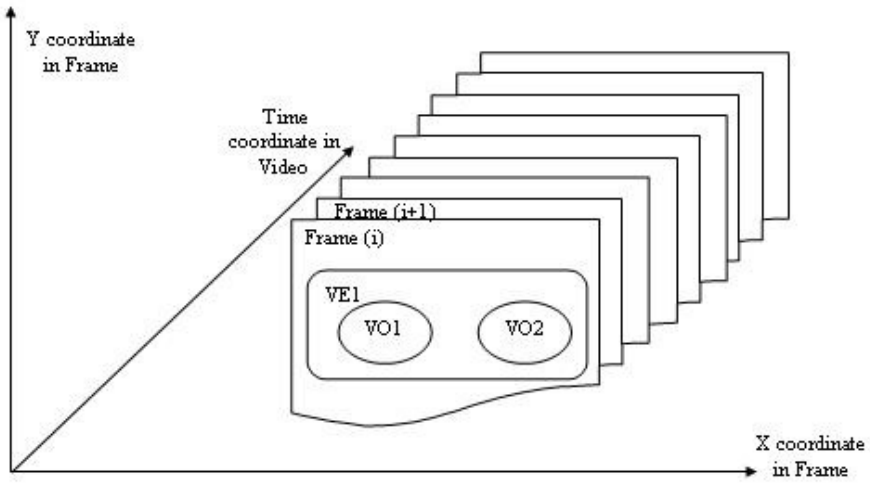


Figure 3.7 – Identification of a Video Element

Specific events that occur in a certain place during a particular interval of time are called Video Events. Video events occur in a particular shot of the video. As a result, particularly, every event belongs to directly to some specific shot and indirectly to some specific sequence. This situation can be seen in hierarchy of the sub-models, in Figure 1.1.

3.3.1. Video Event Sub-Model

Video Events are formally defined in our model as follows:

VE = {ID, N, ES, K, V, A, TSRL} where

1. ID= Unique Event ID in temporal segmentation
2. N= Name of the event. It's most probably a verb, like *drive*, *eat*, *walk* etc.
3. STime= Start time of the event.
4. ETime= End Time of the event.
5. ES= Event Structure consist of an object List of Object IDs defined in the video and labelled as

   {Who, Where, What, Whom, With}

   a. Who = Object who does the event or subject of the event. Such as John, Kylie, cat, dog, plane, tower.

   b. Where = The semantic place where the event occurs. Such as Park, School, House, Table. Non-visible places or events must not be considered here. For example in "John gave a present Kylie in the Party," *Party* is not a place.

   c. What = A thing, person, or matter to which thought or action is directed, object of the event or an object of investigation. For example, in a sentence "John gave Kylie a Hamburger", *Hamburger* is an object.

   d. Whom = Indirect object of the event, i.e, in "John gave Kylie a Hamburger", *Kylie* is an indirect object.

   e. With = Accompanied object or instrument of the event. In a sentence "John kills Kylie with knife", *Knife* is an accompanied object.

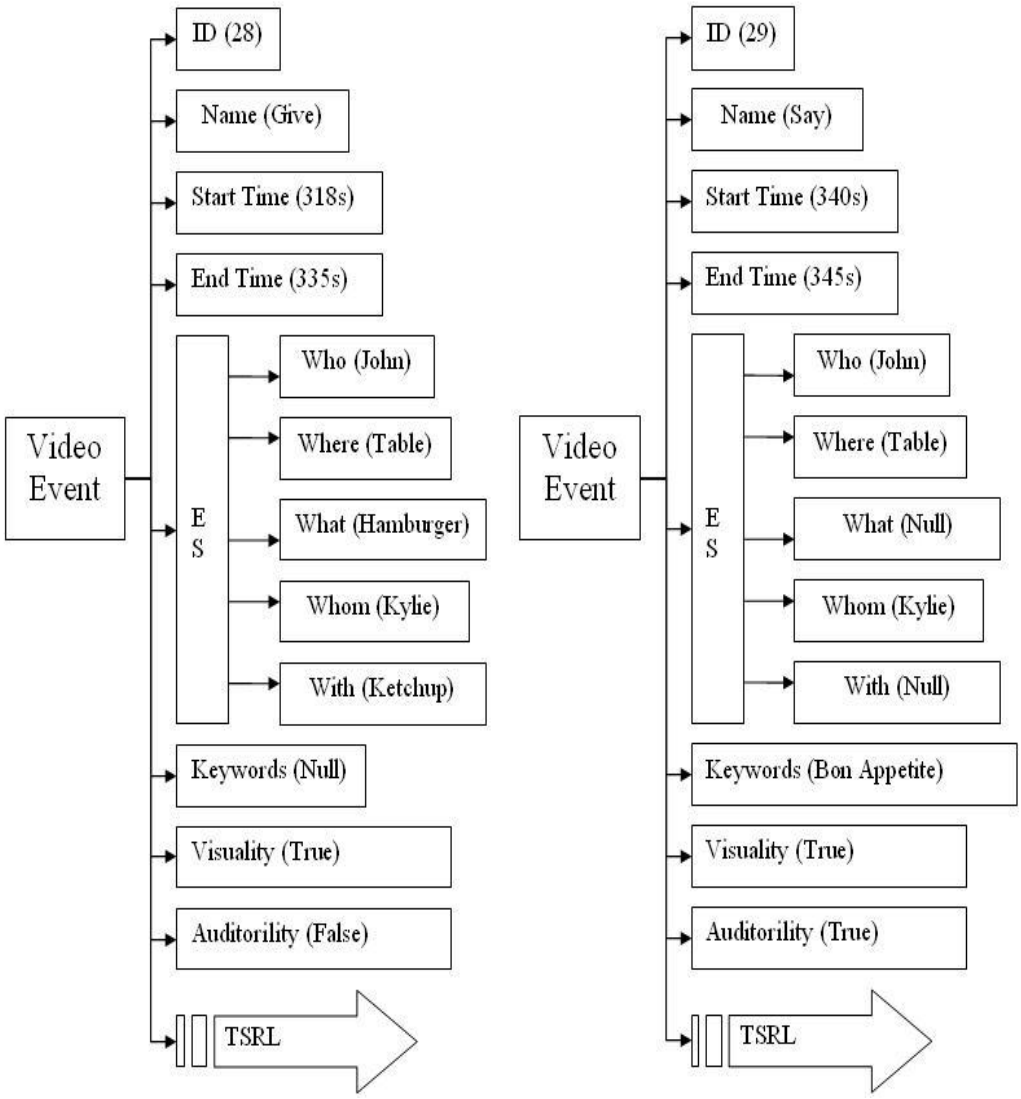
6. K = Keywords of the events, such as extra information or words that can be heard in the auditory event. Keywords are free texts. If some audio event must be defined and an event such as John said "Hello" must be declared, the word

"Hello" must be put in the keywords. If the text that is said is an object defined in the video, it must be put to the What part of the ES list.

7. V (Visuality) = The flag of visuality. If the event has a visual aspect or if the event has some visual scenes, it is set to true. If this is true, we understand that we will realize the event in the visual part.
8. A (Auditorility) = The flag of auditorility. If the event has an audial aspect or if the event has some hearable things, it is set to true or false. If this is true, we understand that we will realize the event in the auditory part. If visual and auditory parts of the event are both true, this means that event has both aspects. While watching the video, we should look at the screen and listen to the sound. If there is something heard in the video it is embedded to the model by making A (Auditorility) flag as true and writing what is said to the K parts of the model. If the noise heard is not a something that can be written, such as only a noise or explosion, the event name can be used to store this information.
9. TSRL (Temporal and spatial region list) = The member of the list is a region labelled by a specific time in a video. These regions are Minimum Bounding Rectangles defined in [15]. All the objects that belong to the ES must be seen in that rectangle.

Textual information in the video is embedded into the model as making a new event named "isWritten" and put the written text into the K field of the Video Event Sub-Model. The spatial and temporal information of the text is also included in the TSRL of the event. In Figure 3.8 and 3.9, Video Event Sub-Model is shown with some another example data in parenthesis.

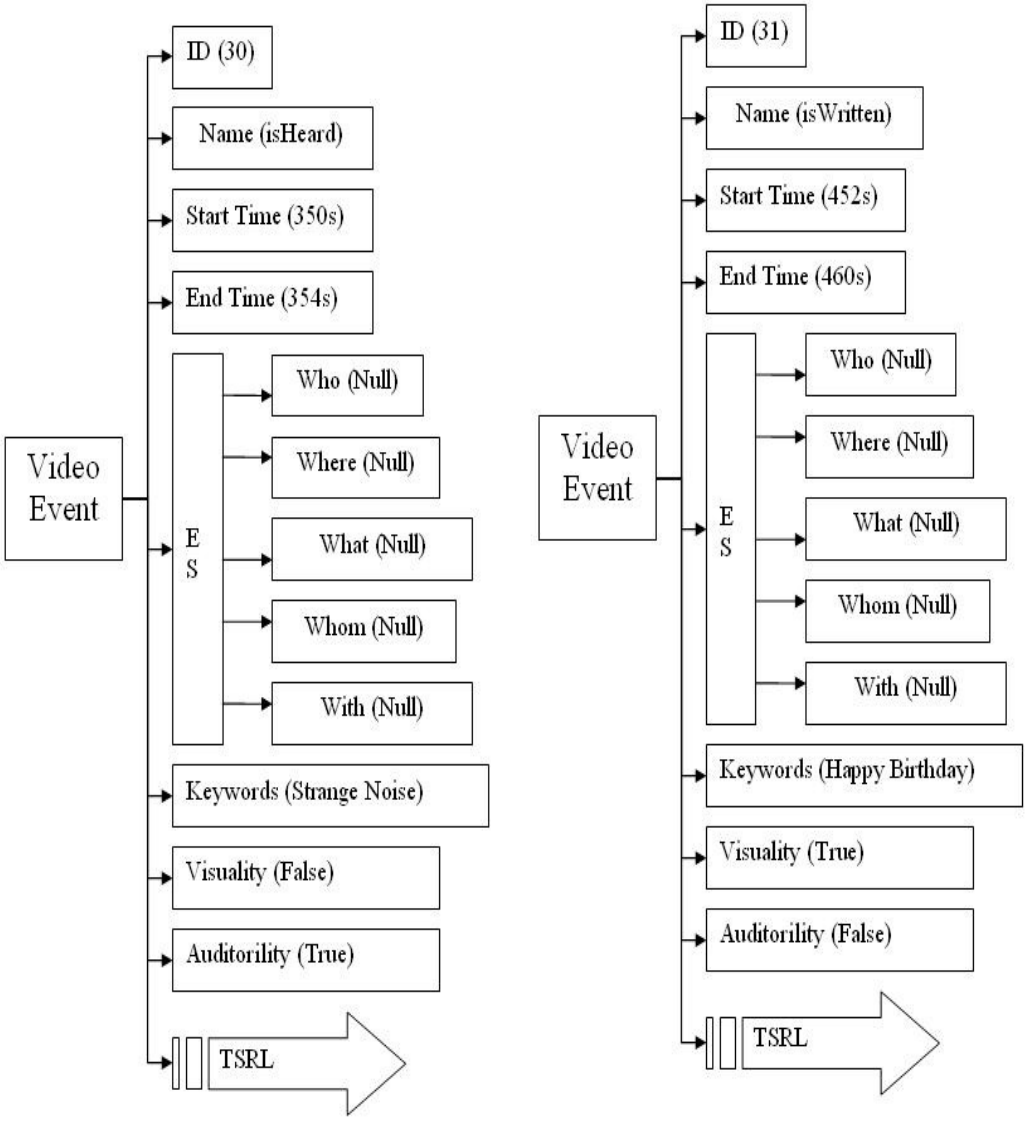
XSD Schema of SEBM Video Data Model is given in Appendix C. A complete example about SEBM video data model is given in Appendix C also.



(a) Example event has only visual modality

(b) Example event has both visual and auditory modality

Figure 3.8 – Video Event Sub-Model (1)



(a) Example event has only auditory modality

(b) Example event has textual modality

Figure 3.9 – Video Event Sub-Model (2)

CHAPTER 4

QUERYING THE SEBM VIDEO DATA MODEL

Extracted information from video(s) is stored in a video database and then queried and accessed. However, there are some issues while considering these processes. First of all, how information in video is extracted, manually or automatically, is an issue. Secondly, after information is extracted, how you store them in a database is a research problem to be solved. Lastly, when you store all information that you need about videos in a database, which types of queries are supported and how these queries are processed and how information in databases is efficiently accessed are also important issues.

Since there is no standard querying language or query models that you can use in video databases, querying the video database is a challenging problem. One possible solution is to develop a video data model that fits into the area of interest.

In this study, we introduce a model, in which we can represent the structural information about events or objects. Whereas, when we extract the necessary information from a specific video and represent them in our model, we can answer much more queries than we had originally expected. So, we can answer the queries

like; what is going on in the video, who are the people in the videos, what the relations between them are and what is happening when and where?

In our developed prototype system, the database can be queried on visual, auditory and textual contents. Temporal and spatial relations between events and objects can also be queried. Moreover, hierarchical and conjunctive contents can be queried. Besides these, we handle structural queries about objects and events. These query types can be put into five different groups:

1. Content based queries which include structural and hierarchical queries.
2. Spatial queries which include the queries related to regional and trajectory queries.
3. Temporal queries which include the queries that are related to the timestamps of video entities.
4. Fuzzy spatial queries
5. Compound queries

4.1. Content Based Queries

Content Based Queries are about the content of information that we extracted from the videos. With these queries, we can retrieve events, objects and their relationships in the video. These queries are either about only one of the sub-models, explained in previous sections or about combinations of more than one of these sub-models. We call the former as the Simple Queries (SQ) and the latter as the Complex Queries (CQ). Complex queries are derived from simple queries.

SQs are the queries related to information in our five sub-models.

1. Queries about Video Sequence Sub-Model (SQ1)
2. Queries about Video Shot Sub-Model (SQ2)
3. Queries about Video Event Sub-Model (SQ3)

4. Queries about Video Object Sub-Model (SQ4)
5. Queries about Video Object Characteristics Sub-Model (SQ5)

For example, “What information do we store in a specific model entity (sequence, shot, object, characteristic or event)?” or “which model entity has the specific information that we supply, like name or timestamps (start or end points)?” can be two example queries.

CQs are the queries about the relationships between sub-models.

1. Relations between Video Object and Characteristics Sub-Models, (CQ1).

   Which Object has characteristics X where X is the answer of SQ5?

   CQ1 type query examples:

   - Which kind of characteristics does the object named John have?
   - Which object has characteristics C, where C is brother of Kylie? In other words who is the brother of Kylie?
   - Who is the brother of sister of John?
   - Who is the brother of the person seen in the event of eating between timestamps [3.0, 30.0]?

2. Relations between Video Event and Object Sub-Models, or Video Event and Object Characteristics Sub-Models, (CQ2).

   Which event has Object Y having characteristics X where X is the answer of SQ5 and Y is the answer of SQ4 or CQ1?

   CQ2 type query examples:

- What is the Video Event that Object X is the subject of the event and Object Y is the object of the event between timestamps [t1, t2]? For example, what does John do to Jimmy between time [5.0, 20.0] Assume that John gives a book to Jimmy. Answer of this question is “give”.
- When does John give the book to the brother of Jimmy?
- When does John crash to the chair where Jimmy sits?
- When is John seen on the screen?
- When is the text “Happy Birthday” seen on the screen?
- When does John say “Hello” to the sister of Jimmy?

3. Relations between a Video Shot Sub-Model and other three Sub-models (Events, objects or characteristics), (CQ3).
    a. *Which shot has Event X where X is the answer of SQ3 or CQ2?*
    b. *Which shot has Object Y having characteristics X where X is the answer of SQ5 and Y is the answer of SQ4 or CQ1?*

CQ3 type query examples:

- In which shot Event X occurs where Object X having characteristic Y is subject of event? For example, in which shot John gives the Hamburger to the sister of Jimmy?

4. Relations between a Video Sequence Sub-Model and other four Sub-Models (Shot, Event, Object or Characteristic), (CQ4).
    a. *Which sequence has Shot X where X is the answer of SQ2 or CQ3?*
    b. *Which sequence has Event X where X is the answer of SQ3 or CQ2?*
    c. *Which shot has Object Y having characteristics X where X is the answer of SQ5 and Y is the answer of SQ4 or CQ1?*

CQ4 type query examples:

- Which sequence has Shot X having timestamps [t1, t2]? For example, which sequences have dinner shots?
- In which sequence does John drive the car?
- Give me the timestamps of the sequence where John fights with Kylie's brother.

These four types of CQs or five types of SQs can be used to compose much-complicated compound queries. Queries can be built to create conjunctive or disjunctive queries. For example, "When does John give Kylie's brother a hamburger while (and) Jimmy is saying hello?", "What are the sequences where the text *Happy birthday* appears on the screen or *strange noise* is heard at the background?",

Each of these four types of CQs is composed of SQs and possibly other CQs. SQs are processed first and then the results of these SQs are merged as the answer for CQs. That is, the divide and conquer technique is used here. The solving process is shown in Figure 4.1.

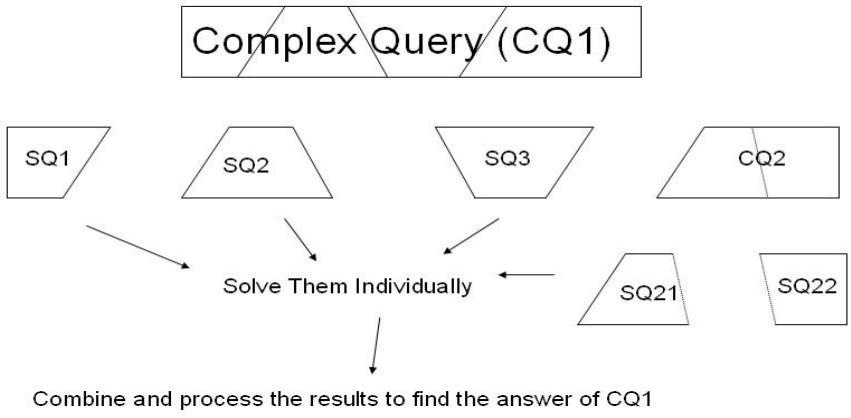


Figure 4.1 – Query Solving Process

For example, assume that we have a query like “When does John give Kylie’s brother a hamburger while (and) Jimmy is saying hello?” Firstly, we find Kylie’s brother, Tom. Next, we find the intervals where John gives Tom a hamburger for example [t1,t2]. Next, we find when Jimmy says hello, for example [t3, t4]. Lastly, we merge (intersect) both intervals and find the common time interval [t5, t6] which is the result. The intersection process is explained in part 4.4.

4.2. Hierarchical Queries

Video sequence-shot-event-object hierarchy is queried in hierarchical queries. While annotating the video, either manually or automatically it is divided firstly into sequences, then shots. This hierarchy of video parts is shown in Figure 4.2.

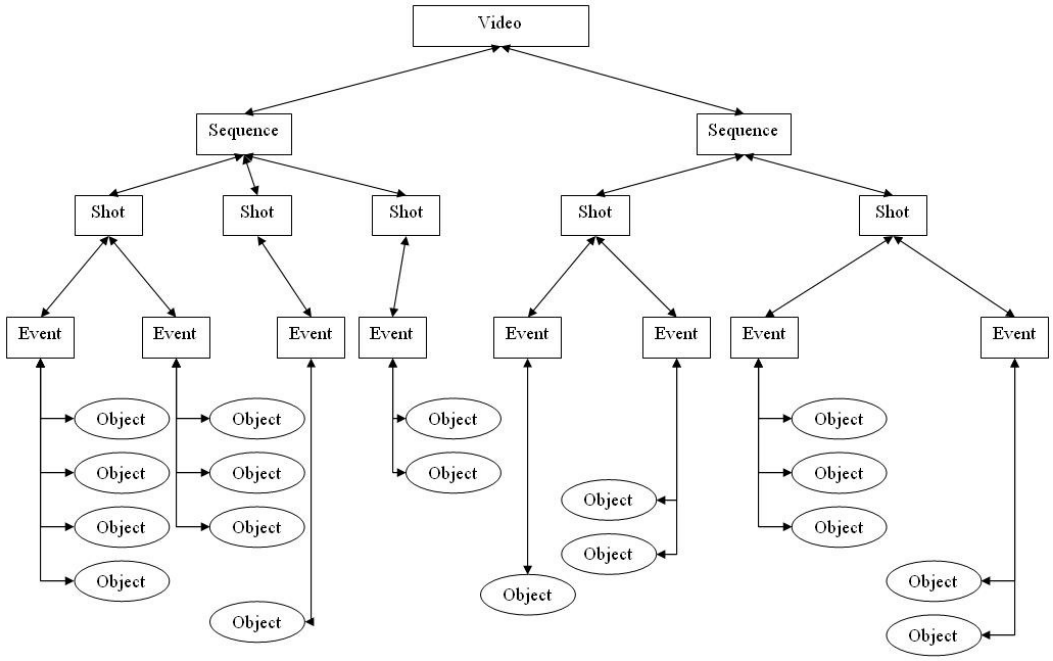


Figure 4.2 – Video Entities Hierarchy

From the users point of view, either sequences or shots can be seen as big video parts containing small video parts i.e. events. Hierarchical queries are some kind of content-based queries but containing the hierarchy of video entities. For example:

- What is happened in Party? (Assume "Party" is a sequence name and user wants to know which events take place in this sequence)
- Who or what are the actors of crashing event? (The objects of the event, named "Crash", is queried)
- Which songs are heard in song contest? (Assume that "song contest" is a sequence and every song time intervals are labelled as shot. In this example sequence-shot hierarchy is queried)

4.3. Spatial, Regional and Trajectory Queries

While annotating the video part, we put the spatial information to the model under Video Event Sub-Model, i.e. where a particular event occurs on the video screen. We get particular rectangular information (x coordinate of upper left point, y coordinate of upper left point, height and width). The rectangle is Minimum Bounding Rectangle. Minimum Bounding Rectangle is a minimum rectangle region that covers all parts of an event. This situation is shown in Figure 4.3 for an event that contains two objects.

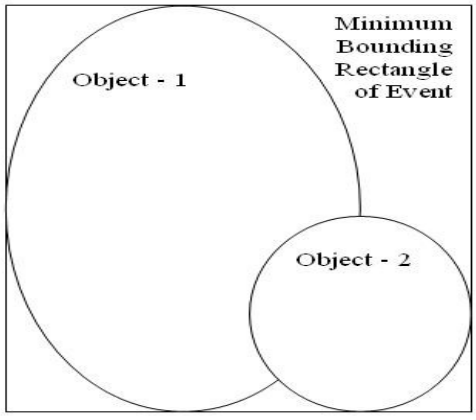


Figure 4.3 – Minimum Bounding Rectangle of an Event

Since Video events occur not only on a single frame but also on continues frames, we take more than one particular rectangular information where each rectangular includes the event and related objects. Adjacent two rectangular give us the trajectory information of the event in the screen at the corresponding frames. If there is a trajectory change, we add another rectangular information. For example assume that we have a video event “flying” of video object “plane” which has three region data, Region 1, Region 2 and Region 3; and another video event “walking” of video object “John” which has three region data, Region 4, Region 5 and Region 6. The trajectories of these events between frames A, B and C are shown in Figure 4.4.

Region information is discrete. But when we have queries like “Find the events that occur in a given rectangular on the screen” or “Give me the intervals where the plane passes the rectangular that I draw on the screen”, we take the rectangular area and look if it contains a rectangular that a particular event has.

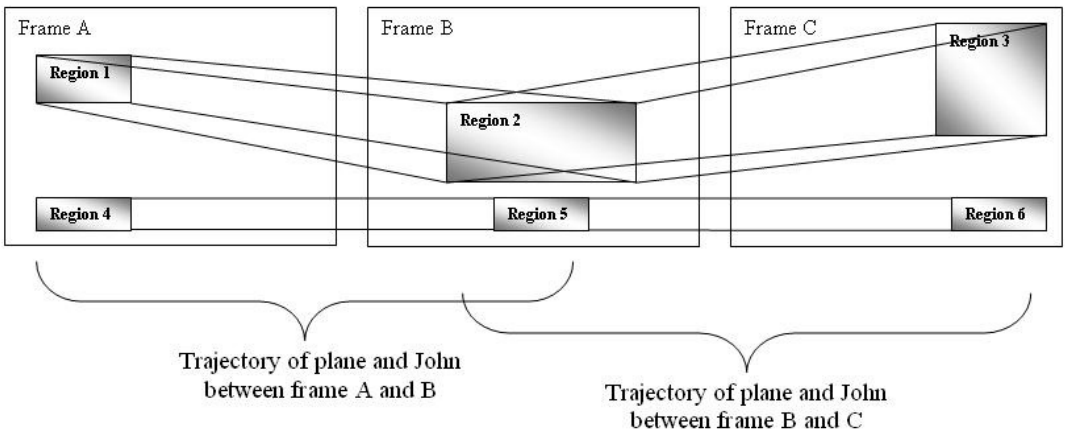


Figure 4.4 - Trajectory of an Event

Spatial relationships are classified into three categories in [2]:

1. Topological Relations that describe neighbourhood are shown in Figure 4.5.

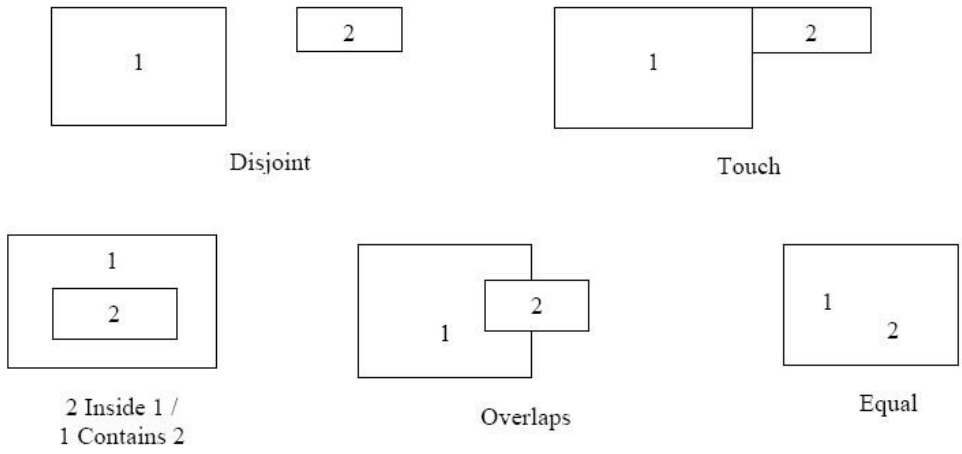


Figure 4.5 – Topological Relationships

2. Directional Relations that describe order in space are shown in Figure 4.6.

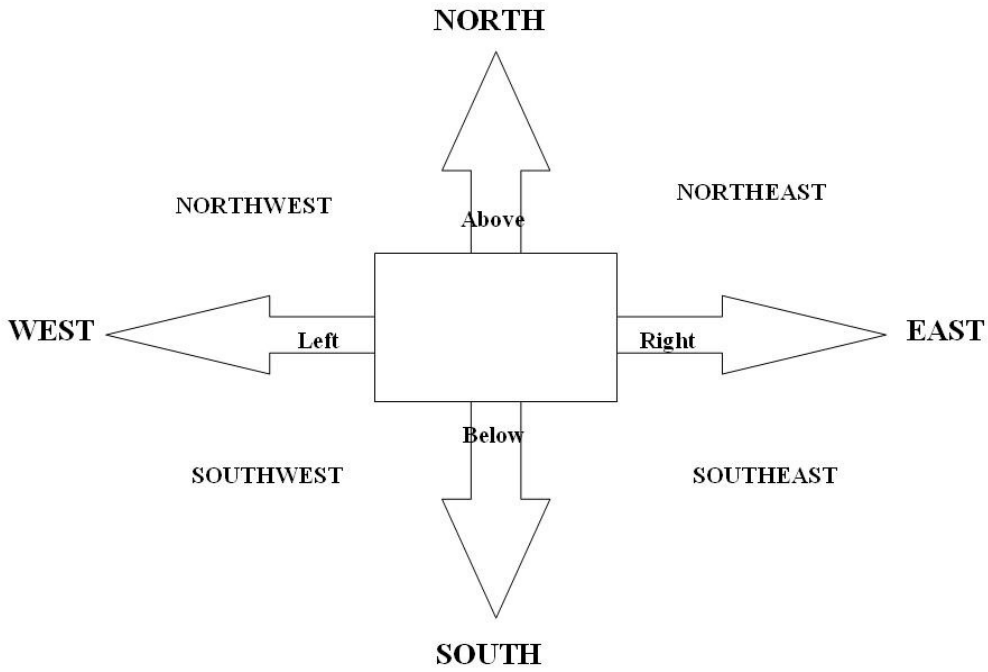


Figure 4.6 – Directional Relations

Köprülü et. al extended directional relations and constructed the definitions given in Table 4.1.

Table 4.1 - Extended Directional Relations

Relation	Definition
A BOTTOM B	A_x {b, bi, m, mi, o, oi, d, di, s, si, f, fi, e} $B_x \wedge A_y$ {b, m} B_y
A TOP B	A_x {b, bi, m, mi, o, oi, d, di, s, si, f, fi, e} $B_x \wedge A_y$ {bi, mi} B_y
A LEFT B	A_x {b, m} $B_x \wedge A_y$ { b, bi, m, mi, o, oi, d, di, s, si, f, fi, e} B_y
A RIGHT B	A_x {bi, mi} $B_x \wedge A_y$ { b, bi, m, mi, o, oi, d, di, s, si, f, fi, e} B_y
A TOP-LEFT B	(A_x {b,m} $B_x \wedge A_y$ {bi, mi, oi} B_y) $\vee$ (A_x {o} $B_x \wedge A_y$ {bi,mi} B_y)
A TOP-RIGHT B	(A_x {bi, mi} $B_x \wedge A_y$ {bi, mi, oi} B_y) $\vee$ (A_x {oi} $B_x \wedge A_y$ { bi,mi }B_y)
A BOTTOM-LEFT B	(A_x {b,m} $B_x \wedge A_y$ {b,m,o} B_y) $\vee$ (A_x {o} $B_x \wedge A_y$ {b,m} B_y)
A BOTTOM-RIGHT B	(A_x {b,m} $B_x \wedge A_y$ {b,m,o} B_y) $\vee$ (A_x {oi} $B_x \wedge A_y$ {b,m} B_y)
A OVERLAPS B	A_x {d, di, s, si, f, fi, o, oi, e} $B_x \wedge A_y$ { d, di, s, si, f, fi, o, oi, e } B_y
A EQUAL B	A_x {e} $B_x \wedge A_y$ {e} B_y
A INSIDE B	A_x {d} $B_x \wedge A_y$ {d} B_y
A CONTAIN B	A_x {di} $B_x \wedge A_y$ {di} B_y
A TOUCH B	(A_x {m, mi} $B_x \wedge A_y$ {d, di, s, si, f, fi, o, oi, m, mi, e} B_y) $\vee$ (A_x {d, di, s, si, f, fi, o, oi, m, mi, e} $B_x \wedge A_y$ {m, mi } B_y)
A DISJOINT B	A_x {b, bi} $B_x \vee A_y$ {b, bi} B_y

3. Distance Relations that describe space range between objects i.e. far, near etc.

4.4. Temporal Queries

Like spatial queries, temporal queries are for querying the comparison. But they are for comparison of timestamps between two video entities. Durak et. al. give temporal relationships as “before”, “meets”, “overlaps”, “during”, “starts”, “finishes”, “equal” [2]. Allen defined the meaning of these temporal relationships in [22] as shown in Table 4.2. In this table two video events called B and C are positioned in timeline that suits the relationship given in the first column.

In order to make comparison between timestamps, some operations on intervals must be done, for instance union or intersection. For example (comparison keywords are written in italic):

- Find the intervals in which John says “Hello” to Kylie *before* Kylie says “Hello” to John. (Two events named “say” are compared according to the timestamps with the keyword before)
- Find the intervals in which John kisses Kylie *while* dancing.
- Find the intervals in which both John *and* Jimmy run together.
- Find all objects appearing *between* 5th and 25th minutes. (The events between the given minutes are given and all the objects of found events are extracted)

Table 4.2 - Temporal Relationships

Relation	Symbol	Inverse	Meaning
B before C	*b*	*bi*	*BBBB CCCC*
B meets C	*M*	*mi*	*BBBBCCCC*
B overlaps C	*o*	*oi*	*BBBB* *CCCC*
B during C	*d*	*di*	*BBB* *CCCCCCCC*
B starts C	*s*	*si*	*BBBB* *CCCCCCCC*
B finishes C	*f*	*fi*	***BBBB*** *CCCCCCCC*
B equal C	*e*	*e*	*BBBB* *CCCC*

Pradhan et. al define interval operations in [61] as listed below:

1. ***Intersection:*** This operation takes two intervals I_1 and I_2 and produces intersected interval if

$$\max(\text{start}(I_1), \text{start}(I_2)) \leq \min(\text{end}(I_1), \text{end}(I_2)).$$

$I_{1\_2} = I[fs, fe]$ where

fs = max(start(I_1), start(I_2)) and

fe = min(end(I_1), end(I2)).

Examples:

- I_1[5, 15] and I_2[10, 30] then I_{1_2} [10, 15].
- I_1[10, 20] and I_2[30, 40], I_{1_2} does not produce any interval.

2. ***Union:*** This operation takes two intervals containing overlapping or adjacent intervals and produces a single non-overlapping contiguous interval. Union operation works as taking minimum starts of intervals and maximum ends of intervals.

   I_{1_2} = I[fs, fe] where

   fs = min(start(I_1), start(I_2)) and

   fe = max(end(I_1), end(I_2)).

   Example:

   - I_1 [25, 50] and I_2 [50, 150] then I_{1_2}[25, 150].

3. ***Extended Union:*** This operation can be considered interval concatenation. This operation takes two intervals as input and produces a single contiguous interval. The resulting interval will be contiguous no matter that the input intervals are adjacent, overlapping, non-overlapping and nonadjacent.

   I_{1_2} = I[fs, fe] where

   fs = min(start(I_1), start(I_2)) and

   fe = max(end(I_1), end(I_2)).

   Example

   - I_1[10, 20] and I_2[30, 40] then I_{1_2}[10,40].

4.5. Fuzzy Queries

The nature of human interpretation of real world is not always discrete. Because there is always some unknown spatial information, the video data querying system should consider possible fuzzy queries. Fuzzy queries are constructed by giving some threshold value and they may involve some fuzzy conditions as explained in [3]. For example:

- Show me the part of the video where the plane passes John just above his head. (Similar to spatial queries, the trajectory information of both plane and John is implicitly found as given in Table 4.3. Then the positions of both video objects in a particular time are compared according to given fuzzy value [3] to find the solution)
- Show me the part of the video where the text "Happy Birthday" is seen around upper left corner of the screen with a threshold value 0.5.
- Show me the part of the video where John is standing left of Kylie.

For the fuzzy spatial relationships Köprülü et. al. define membership functions in [3] by using membership value (μ). The membership value of the relationship is calculated using the angle between the line connecting the centres of rectangles and the x-axis. The membership functions are given in Table 4.3.

Table 4.3 - Fuzzy Spatial Relationships

Relation	Angle	Membership Value
Top	arctan(x/y)	1- (angle/90)
Left	arctan(y/x)	angle/90
Top-Left	arctan(x/y)	1 – (abs(angle-45) / 45)
Top-Right	arctan(y/x)	1 – ((angle – 45) / 45)

4.6. Compound Queries

All content based, hierarchical, spatial, regional, trajectory, temporal or fuzzy queries may be joined to form more complex queries. Even compound queries can form more complex compound queries. In fact, compound queries are either temporally or spatially related queries. By solving and combining the partial answers of compound queries, the final answer can be formed. This approach is similar to that one we follow while solving content based queries. For example:

- Show me the part of the video where the text “Happy Birthday” is seen in the upper left corner of the screen with a threshold value 0.5 and in which John kisses Kylie while dancing.

  (In this compound query there is two query sentence compounded temporally. First one is a fuzzy spatial query returning a temporal answer and second one is temporal a query. Answers of both queries are found and intersected)

- Show me the part of the video where John’s brother who is the friend of the person seen at the upper left of the screen between 30^{th} and 40^{th} seconds is standing near the car on a chair and Kylie is walking through the door.

  (Complex Content based and trajectory queries are combined. Answers of both queries are found and intersected)

CHAPTER 5

IMPLEMENTATION OF THE PROTOTYPE SYSTEM

We have implemented SEBM prototype system by using Java programming language. SEBM prototype system is a multimodal video annotation and retrieval system that uses SEBM to model the video data. In our implementation, we used XML (eXtended Mark-up Language) based technology [18] to design SEBM Sub-Models. Since SEBM Sub-Models forms a hierarchic tree structure, as shown in Figure 1.1, XML is suitable for expressing the sub-models.

In Figure 5.1, the architecture of SEBM prototype system implementation is given. Annotation of the video is done with a semi-automatic approach. For each video, one SEBM model is constructed and stored. Video Shot Sub-Model is created with the help of IBM MPEG-7 Annotation Tool [15]. Other Sub-Models are created manually. All Sub-Models are embedded in a single structural, event based and multimodal video data model in XML format. An example is given in Appendix C. The model is stored in Berkeley XML DBMS. Users query the video data via query interfaces either using the query interface supplied by the system or writing directly XQuery sentence to the manual query-solving interface.

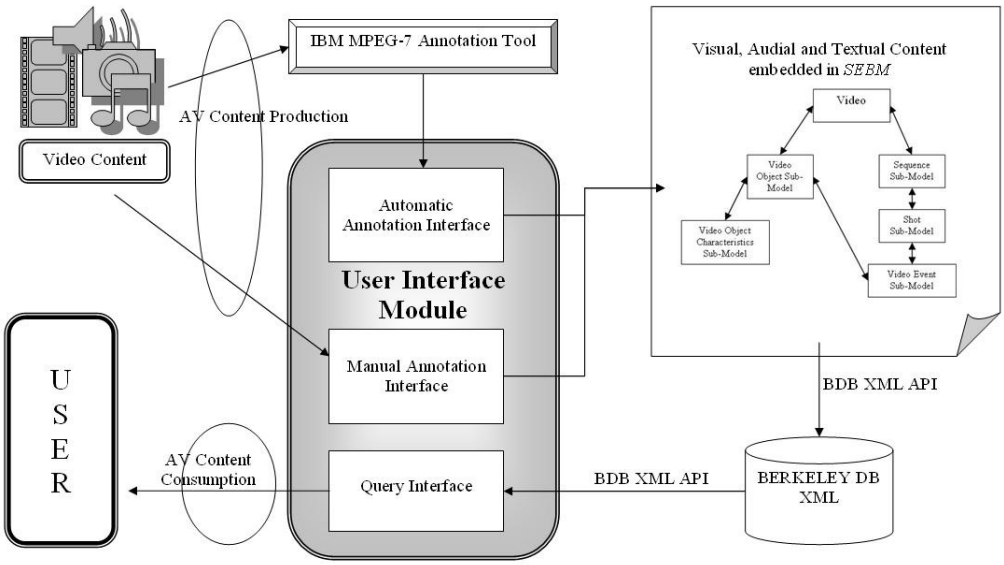


Figure 5.1 – System Architecture of SEBM Prototype System

For a particular event, an XML file containing the extracted information is stored in Berkeley DB XML (BDB XML) [16]. BDB XML is an embedded database specifically designed for the storage and retrieval of XML-formatted documents. BDB XML supports XQuery [17] to query the XML files stored in database. XQuery is a query language designed for the examination and retrieval of portions of XML documents. The application is implemented in Java programming language. Java Media Framework Application Programming Interface (JMF API) is used to handle the videos. JMF supplies control interface to control the video stream such as play, pause or stop. Moreover JMF supplies visual interface to show the video and draw rectangles on video. But visual interface has the ability to draw only on the videos in AVI format and coded Cinepac Codec. On the other hand IBM MPEG-7 Video Annotation Tool [15] extracts the shots of the videos in format MPEG. So the user has both formats of the same videos. A video converter can be used to achieve format bottleneck. I use STOIK Video Converter v1.1 freeware software. [64]

In Appendix B - Figure B.1, Manual Annotation Interface of our system is given. Result of automatic extraction is embedded to our system while annotating video manually. In Appendix B – Figure B.2, one of our Query interfaces of our system is

given. Content-based queries are done through this interface. SQs are directly queried from sub-models query regions. The answer for a CQ is formed by combining the results of SQs, corresponding to the CQ.

5.1. Modeling the SEBM Video Data Model

5.1.1. XML

XML (Extensible Markup Language) is a mark-up language for documents containing structured information [20]. The documents that have structure, obey some rules when carrying the information between applications, so the applications can easily resolve and use the information. In fact XML is created to be used on the Web [20]. XML data formats can be vector graphics, e-commerce transactions, object meta-data, server APIs etc.

The features of XML are listed below:

- XML uses tags to define specific elements within a document [21]. Tags define the meanings of the elements.
- XML does not specify the tags that are used in the document. The developer, confirming the meaning of the elements, selects the tags.
- XML documents are completely text-based that can be read by the humans. XML facilitates the creation of cross-platform tools and exchange of the files between applications.
- Flexibility of determining the structure of the document gives users the ability of expressing very complex structural or hierarchical information in a single text based document.
- Because an XML document is structured, platform-independent, and text-based, XML documents can be opened and operated on by a range of editing programs [21].

In prototype system implementation, we choose to use XML for the reasons explained below:

- XML gives us the opportunity to express the structure of SEBM in a hierarchical way in a single text based document.
- We do not have to convert the sub-models to the tables or rows in order to store them in a relational database.
- We can open, investigate or modify the intermediate SEBM files before storing it in a database by using simple text editor.

5.1.2. XML Usage of SEBM

Data is expressed with mark-ups in XML documents. Elements are the forms of mark-ups that we use, when expressing the SEBM model. The Figures 5.2 – 5.8 show the parts of the SEBM video data model expressed in XML language.

The whole video data begins with <Video> tag and ends with <\Video> tag. The "minus" and "plus" signs before the tags is not a part of XML document. They correspond if the element is in open form or close form and is put by the XML viewer program. For instance if the "VideoObject" tag is expanded, it will replace with the expressions similar to the one shown in Figure 5.5.

```
- <Video>
    <VideoName>film.avi</VideoName>
    <VideoPlace>file:\E:\film.avi</VideoPlace>
  + <VideoObjects>
  + <Sequences>
  </Video>
```

Figure 5.2 – Video Model

Figure 5.3 – Video Sequence Sub-Model

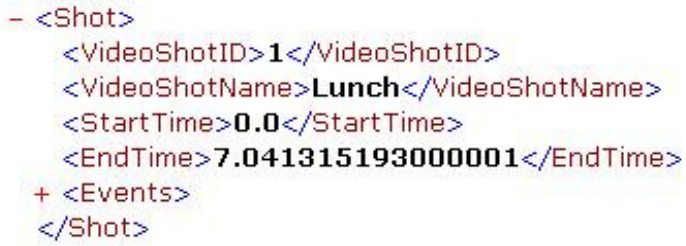


Figure 5.4 – Video Shot Sub-Model

Figure 5.5 - Video Object Sub-Model

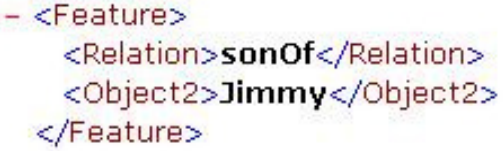


Figure 5.6 – Video Object Characteristic Sub-Model

Figure 5.7 – Video Event Sub-Model

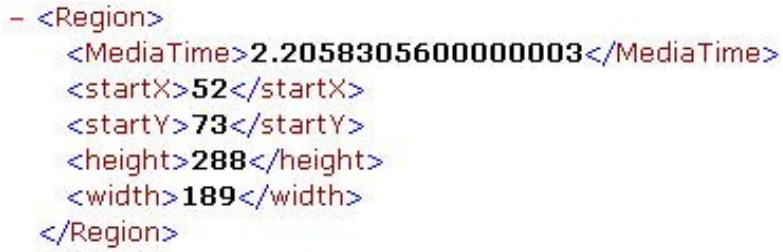


Figure 5.8 – Region Structure that is Part of Video Event Sub-Model

5.1.3. Annotation in Prototype System

5.1.3.1. Video Segmentation

SEBM prototype system has options for Video Segmentation, either automatic or manual. If the user chooses automatic video extraction, s/he uses IBM MPEG-7 annotation tool [15] to extract shots firstly. In Figure 5.9, a screen shot of the IBM MPEG-7 annotation tool, which is doing video segmentation for the part of the video of a Turkish film named GORA, [63] is given:

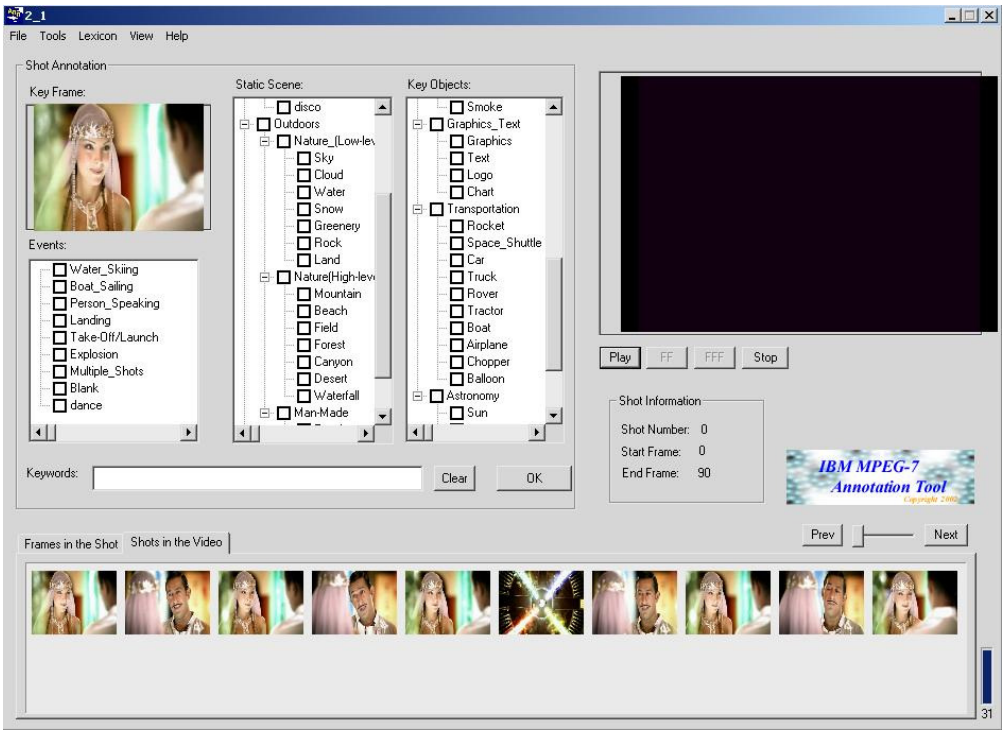


Figure 5.9 – IBM MPEG-7 Annotation Tool

IBM MPEG-7 Annotation Tool segments the video for annotation. The tool has manual annotation facilities but SEBM prototype system has its own manual annotation part to extract objects and events. In Figure 5.10 Video Shot key frames and Video Shot frame intervals are given. The tool produces an XML document that has information about shot list. An example of this output is given in Appendix A. Automatization has a drawback for our data model. For instance, if the video has very small changes in background from one frame to another frame, IBM MPEG-7 Annotation Tool differentiates it as two different shots that have length of smaller than 1 s. But in our model, we define the shot as video parts that consists same kind of one or more events. Usually they have longer intervals than 1 s. In our system user can overcome this drawback by doing some operations, like merging, on shot list that the tool produces.

Figure 5.10 – Video Shots Produced by IBM MPEG-7 Annotation Tool

In Figure 5.11 three different key-frames are given. IBM MPEG-7 Annotation Tool differentiates them as three different shots. When we create the video sequence sub-model from them, we see that the first two should be in the same sequence because of meaning (the same two people are exist and conversation is happening in the same place), but the last one should belong to another sequence (the people are changed and the subject is changed).

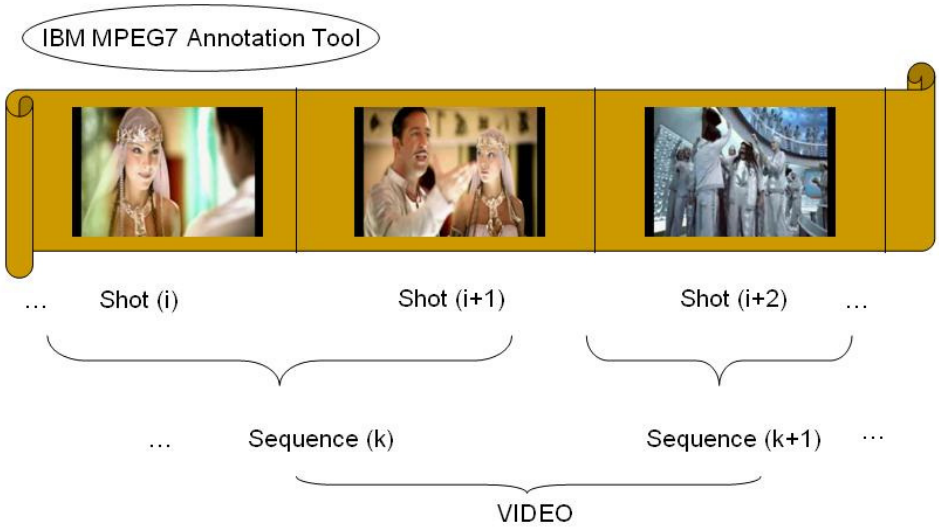


Figure 5.11 - Creation Process of Video Shot Sub-Models and Video Sequence Sub-Models

The output of the IBM MPEG-7 tool is consumed by SEBM Prototype system to develop Video Sequence and Video Shot Sub-Models. The consumption interface of SEBM prototype system is given in Figure 5.12.

The Interface has the ability to merge the shots, which have very small intervals, manually. Assume that the user wants to merge the shots from the first one to the fourth one as shown at the Figure 5.12. The system gets the start point of the first shot and gets the end point of the fourth shot, deletes all intermediate shots and creates one big shot from them.

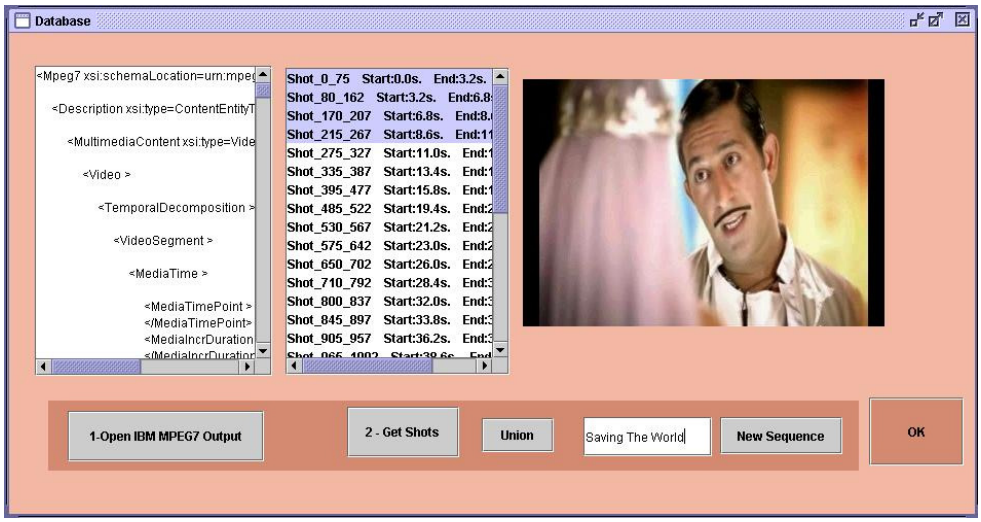


Figure 5.12 – Automatic Video Segmentation Interface

By using the interface, the user can see the key frames of each shot and by combining shots, s/he can construct sequences. Users create sequences manually, because they contain meanings rather than colour or camera changing like in shots.

When the user approves the segmentation of the video the system passes the created Video Sequence and Video Shot Sub-Models to the Annotation interface to continue with object and event creation. An example of passed XML is shown in Figure 5.13. If the user chooses the manual segmentation of video, s/he creates the sequences and the shots, by entering their names and getting their start and end times from the video window, which is opened at the beginning of the annotation. Shots are created by choosing a particular sequence created before. The part of the manual annotation screen is shown in Figure 5.14.

```
- <Sequences>
  - <Sequence>
      <VideoSequenceID>5</VideoSequenceID>
      <VideoSequenceName>Saving The World</VideoSequenceName>
      <StartTime>0.0</StartTime>
      <EndTime>8.6</EndTime>
    - <Shots>
      - <Shot>
          <VideoShotID>95</VideoShotID>
          <VideoShotName>Shot_0_75</VideoShotName>
          <StartTime>0.0</StartTime>
          <EndTime>3.2</EndTime>
          <Events />
        </Shot>
      - <Shot>
          <VideoShotID>96</VideoShotID>
          <VideoShotName>Shot_80_162</VideoShotName>
          <StartTime>3.2</StartTime>
          <EndTime>6.8</EndTime>
          <Events />
        </Shot>
      </Shots>
    </Sequence>
  - <Sequence>
      <VideoSequenceID>6</VideoSequenceID>
      <VideoSequenceName>Escape From Prison</VideoSequenceName>
      <StartTime>8.6</StartTime>
      <EndTime>13.4</EndTime>
    - <Shots>
      - <Shot>
          <VideoShotID>97</VideoShotID>
          <VideoShotName>Shot_170_207</VideoShotName>
          <StartTime>6.8</StartTime>
          <EndTime>8.6</EndTime>
          <Events />
        </Shot>
      </Shots>
    </Sequence>
  </Sequences>
```

Figure 5.13 – Video Sequence and Video Shots Sub-Models in SEBM

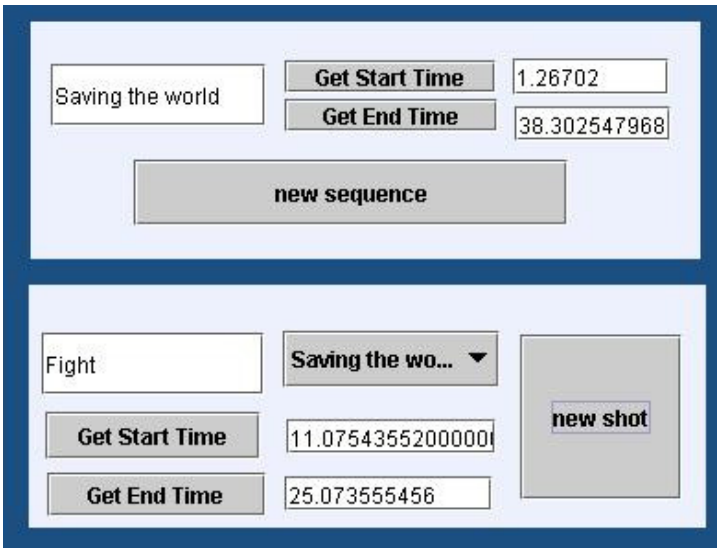


Figure 5.14 – User Interface for Creating Sequences and Shots Manually

5.1.3.2. Video Objects

Video Objects are created by assigning names to them in manual annotation part. This interface is shown in Figure 5.15.

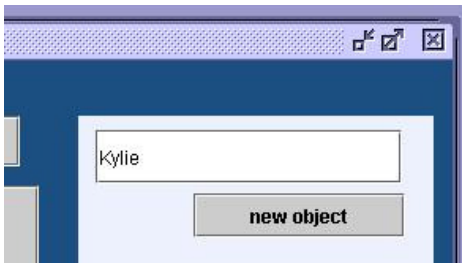


Figure 5.15 – Creating a Video Object

Then Video Object Feature or Video Object relations are created. These phases are shown in Figure 5.16 and 5.17. A feature is created by supplying feature identification i.e. “has” or “is” and a keyword i.e. “blue eyes” or “engineer”. A relation is created by supplying relation identification i.e. “BrotherOf” or “FriendOf” and defined object or undefined object.

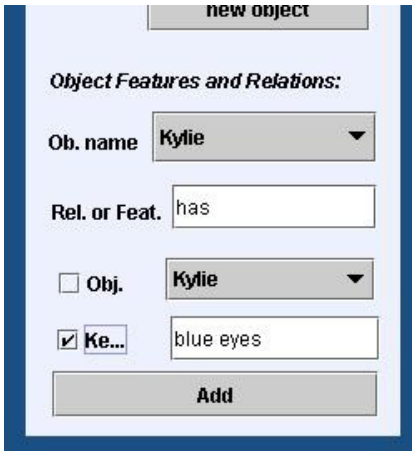


Figure 5.16 - Video Object Feature Creation (Kylie has blue eyes)

Figure 5.17 – Video Object Relation Creation (John is brother of Kylie)

After inserting the objects given above, the corresponding XML data that is inserted into SEBM is shown in Figure 5.18. In this figure, two objects named John and Kylie are defined. The objects are given a unique id number. In the video John has the relation with Kylie named *brotherOf*. This means John is a brother of Kylie. Kylie has the feature of having blue eyes. This is expressed in characteristics of Kylie with the feature named *has* and value as *blue eyes.* This means that Kylie has blue eyes.

Figure 5.18 expresses the relations between Video Object Sub-Model and Video Object Characteristics Sub-Model. This is a one to many relationships. One object may have many characteristics.

In a video, all the objects create the VideoObjects part of the SEBM video data model. The VideoObjects part directly stored under Video part as shown in Figure 1.1

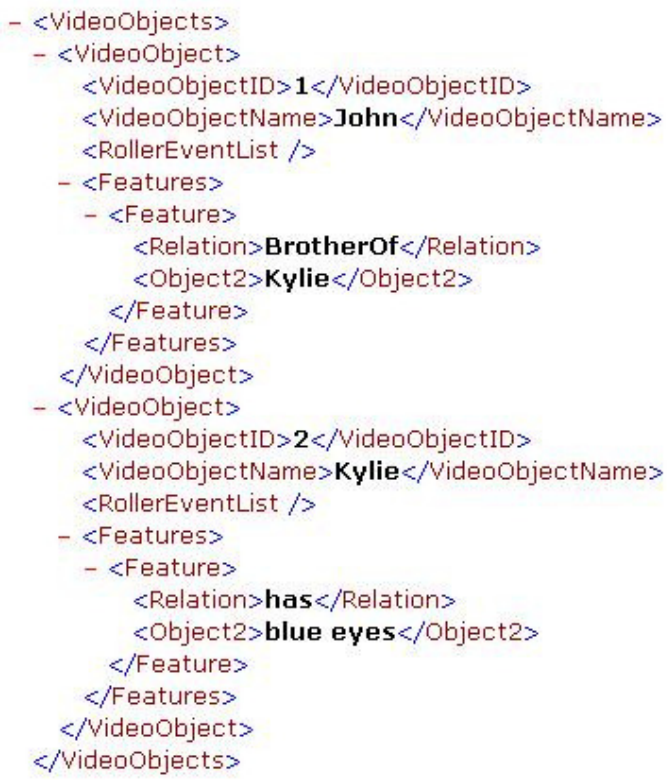


Figure 5.18 - Video Object and Video Object Characteristics Sub-Models in SEBM

5.1.3.3. Video Events

The user supplies Semantic of video actions by entering the information about them. The inputs are listed below:

- The Sequence Name that the event belongs to, (sequence must be defined before for the video),
- The Shot Name that the event belongs to (shot is chosen from the sequence list),
- The name of the event i.e. “Flying”, “Giving”, “isHeard”, “isWritten”, “Saying”, “Singing” etc.,

- The objects that has roles in the event such as who does the event, where the event takes place, what is the direct object of the event, what is the indirect object of the event, what is the accompanied object of the event,
- The flag of visuality,
- The flag of auditorility,
- The keywords of the event for example extra information or words that can be heard in the events.

The information listed above is embedded to the SEBM data model by using the user interface shown in Figure 5.19. The interaction between the semantic of video events and the physical layer of the video is supplied by TSRL (Temporal and Spatial Region List) of the event. User enters the TSRL information by using the video screen. A sample TSRL annotation is shown in Figure 5.20.

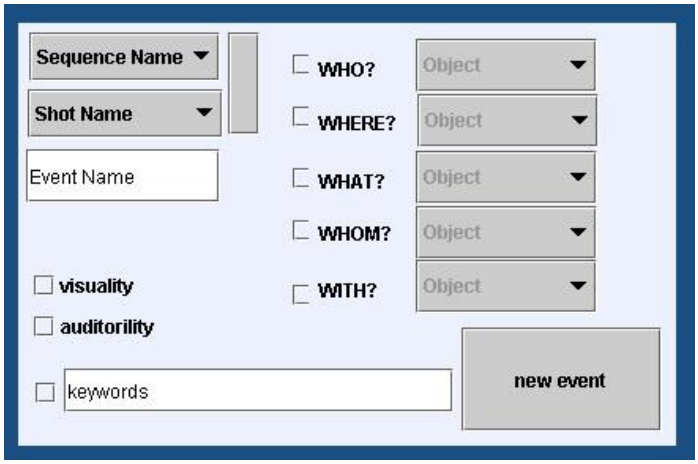
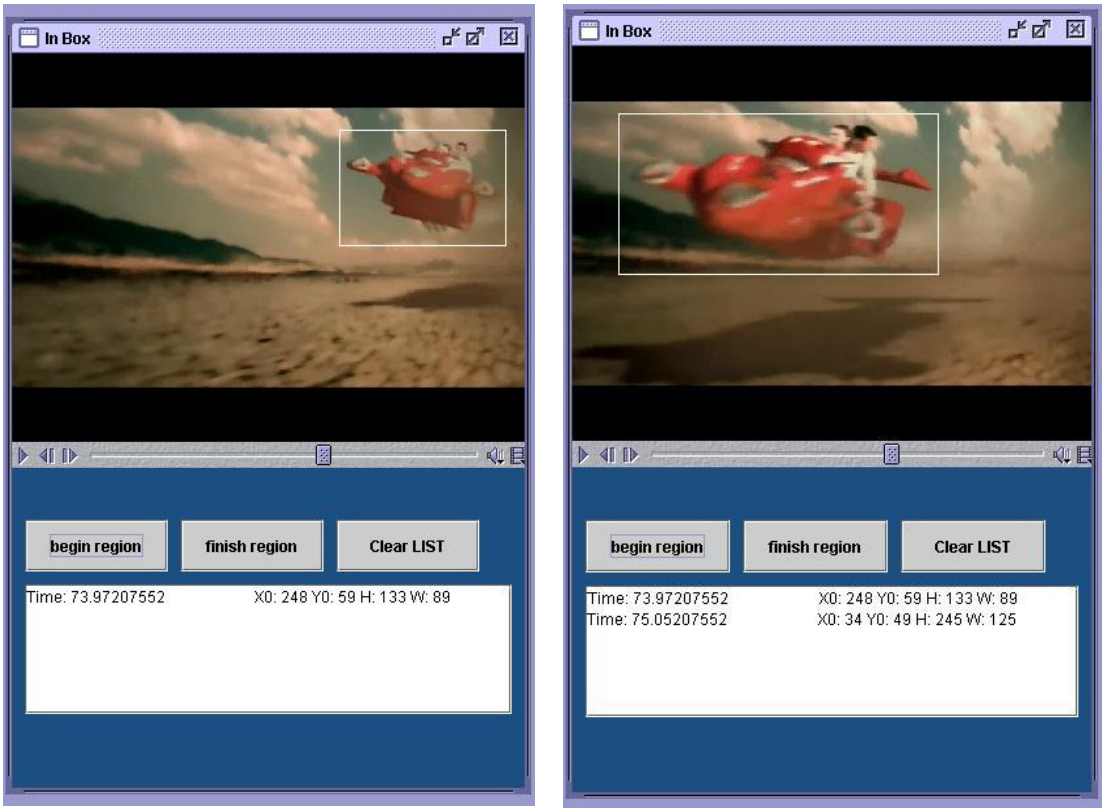


Figure 5.19 – User Interface for Video Events



(a) First Region

(b) Last Region

Figure 5.20 – Entering TSRL Information of an Event

When querying the event trajectory, the TSRL information between start and end TSRL information is filled automatically by the SEBM prototype system as explained in part 5.3.2.3

5.2. Storing the SEBM Video Data Model

5.2.1. Berkeley DB XML

Berkeley DB XML (BDB XML), developed by Sleepycat Software, is an embedded database specifically designed for the storage and retrieval of XML-formatted documents [16]. BDB XML is a programmer library that gives all the functionality that is used in an enterprise database management system. BDB XML is directly embedded into the application rather than communicating with the application in client-server architecture. BDB XML provides interfaces to manage, query and modify XML documents through a programming API. BDB XML provides data management completely hidden to the application. BDB XML conforms W3C standards for XML, XML Namespaces [16].

Important features [16] of BDB XML are listed below:

- ***In-process data access:*** BDB XML is compiled and embedded to the application in the same way as any library. It runs in the same process space of the application that it is embedded into. No extra database management system control is needed.
- ***Size Considerations:*** BDB XML has the ability to manage databases up to 256 terabytes in size.
- ***Database Environment Support:*** BDB XML environments support multiple databases, transactions, deadlock detection, lock and page control, and encryption.
- ***Atomic Operations – Transaction Support:*** When using BDB XML, complex sequences of "read and write access" can be grouped together into a single atomic operation using transaction support. Either all of the

read and write operations within a transaction succeed, or none of them succeed.

- ***Isolated Operations:*** Every transaction operating on a particular XML document isolated each other such that they work as if no other transaction is operating on the same document.
- ***Recoverability:*** In case of failure BDB XML guarantees the recoverability of the data. All committed data is available after a failure.
- ***Concurrent Access:*** Transaction and deadlock mechanism built in BDB XML enables multiple threads and processes to operate simultaneously securely and safely.

In prototype system implementation, we choose to use BDB XML for the reasons explained below:

- BDB XML is specifically designed for XML document storage. Querying XML documents is much easier than relational databases, since we have to convert XML data produced by our system to internal tables and rows.
- BDB XML supports XQuery that is specifically designed for XML document content querying.
- BDB XML is mush safer than file system, because it supports transaction protection.
- BDB XML is proposed to have much efficient data storage mechanism than simply storing the XML data in file system.
- BDB XML provides a programming API that has interfaces to query whole XML documents stored simultaneously.
- BDB XML is an open source database that can be downloaded through the Web with no charge.
- BDB XML usage requires no prior experience with other databases of Sleepycat Software.

- Since BDB XML is completely embedded in to the application, there is no DB administration exported to the user.
- BDB XML usage is easier such that only including the jar files that it is consist of into a java project like any other Java API.

5.2.2. BDB Usage in Prototype System

The databases in BDB XML live together in a place called *Environment*. Environment is an encapsulation of the data; log files and intermediate files of BDB XML infrastructure. The entire environment is stored in a file system directory. In an environment the XML documents are stored in *Container*. A container is a data file that contains XML documents, their metadata and indices. Containers are used when adding, deleting and modifying the XML documents. Containers are handled with *Managers*. Manager is a high-level class that is used to manage the containers, creating the input streams, creating the documents and query contexts, preparing and running the queries and create the transaction objects. This hierarchy is shown in Figure 5.21.

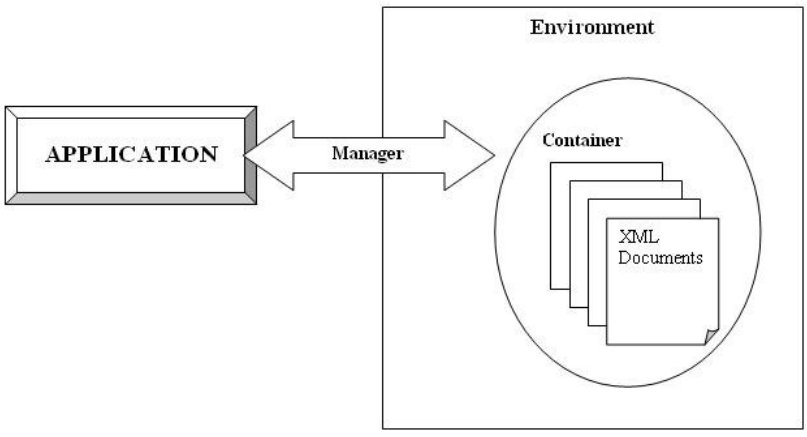


Figure 5.21 - BDB XML Database Environment

BDB XML supports *node* storage and *whole-doc* storage for document storing. First approach means that the documents are stored as individual nodes in the container. Each record in the XML document is stored in a single node. The node contains its value, its attributes and its attributes values if any. Second approach means that the documents are stored entirely. The database does not manipulate any line breaks or white spaces.

BDB XML uses XQuery to query the data stored in database. Querying the BDB XML requires two parts:

- Context of the query, such as namespaces or the result format
- The string expression of XQuery that will be used when retrieving the documents

The result is found by searching the entire database to find the document matching of the given query expression, in a given context.

5.3. Querying the SEBM Video Data Model

5.3.1. XQuery

XQuery is a powerful language designed for processing XML data. XQuery can be used for not only files in XML format but also other data including databases whose structure is similar to XML, such as nested or tree with attributes. XQuery can be used to:

- Query for a document. Note that queries can be formed against an individual document, or against multiple documents,
- Query for document subsections, including values found on individual document nodes,

- Manipulate and transform the results of a query.

XQuery is used when querying the database. XQuery views a particular document as collection of nodes. A subset of XQuery is XPath. XPath is used like traversing the nodes of the document. For example;

- "/" (Single slash) means the root of the XML document.
- /doc/chapter[5]/section[2] selects the second section of the fifth chapter of the doc
- node[1] selects the first node child of the context node
- //node selects all the *node* descendants of the document root and thus selects all *node* elements in the same document as the context node
- //node/item selects all the item elements in the same document as the context node that have an *node* parent

In the prototype system we are mostly interested in XPath, subset of XQuery. The features of XPath and XQuery are listed below:

- XQuery borrows path expressions from XPath. XQuery can be viewed as a generalization of XPath.
- XPath models an XML document as a tree of nodes.
- XPath fully supports XML namespaces.
- XQuery operates on the abstract, logical structure of an XML document, rather than its surface syntax. This logical structure is known as the data model.

In prototype system implementation, we choose to use XQuery for the reasons explained below:

- We implemented SEBM in a hierarchical structure and store it in BDB XML in an XML format. BDB XML supports XQuery for querying the information.
- XQuery is a powerful language that we can query the information in a way that we traverse the important elements of SEBM Sub-Models.
- The result of XQuery can be in XML format or text format, which we can handle easily in Java.

5.3.2. Querying in Prototype System

When querying the data from SEBM video data model stored in BDB XML, XQuery is used. XQuery sentence is sent to the embedded database and the result is computed. We support the query types given below:

1. Content Based Queries including Structural Queries
2. Hierarchical Queries
3. Spatial and Regional Queries including Trajectory Queries
4. Temporal Queries
5. Fuzzy Queries
6. Compound Queries

SEBM prototype system has a query interface, but no query language specific to SEBM. The user enters necessary information in the user interfaces and the system returns the results to the user. SEBM prototype system solves the content-based queries by matching the corresponding sub-models with queries. The user enters the necessary information for a specific sub-model. In fact, since sub-models correspond to the needs of the user, the only thing that the user does, is converting his or her question to the structure of information that can be entered to the interface. System gets the information and converts it to the

XQuery sentence and sends it to the BDB XML embedded system. The database returns the result to the system and the results are supplied to the user. This usage is shown in Figure 5.22.

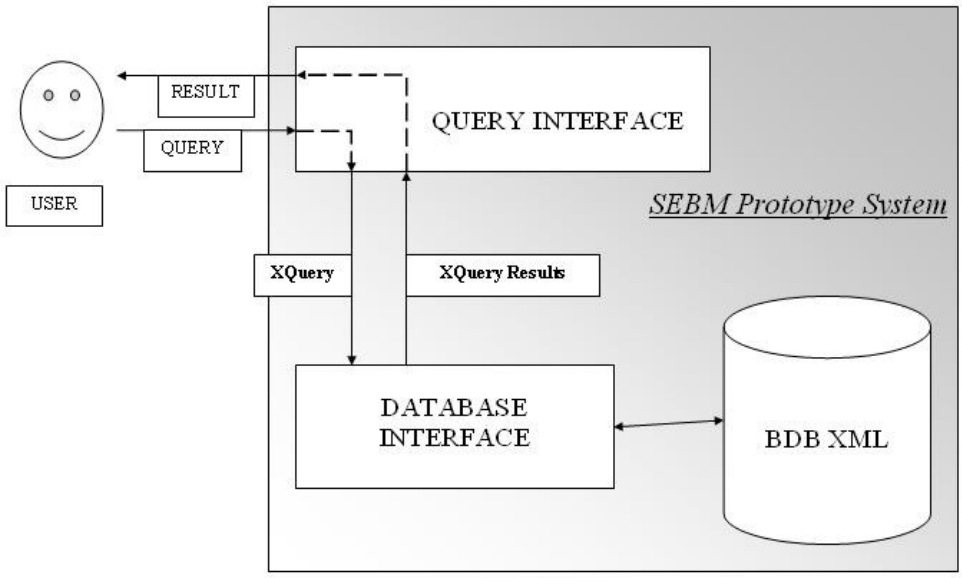


Figure 5.22 – SEBM Prototype System Querying Mechanism

5.3.2.1. Content Based Queries including Structural and Hierarchical Queries

Content Based Queries are about querying the sub-models. Visual, auditory and textual modalities can be queried. Structure of the events can be queried. The SEBM prototype system supplies four different sub-interfaces for five different sub-models. (Video Object Characteristics Sub-Model is queried through Video Object Sub-Model).

In Figure 5.23, the interface for querying the Video Sequence Sub-Model is shown. In this interface user can do the following queries:

1. Given a sequence name, play it. For example: “Play the scene called *saving the world”*.

2. Given a shot name, find the sequence of it. For example: “Find the scene where the *dinner* takes place”. (hierarchical query)
3. Given an event name, find the sequence of it. For example: “Find the scene where the *explosion* occurs”. (hierarchical query)
4. Given a sequence name, find the interval of it. For example: “Find the start and end time of the scene called *escape from prison*”.

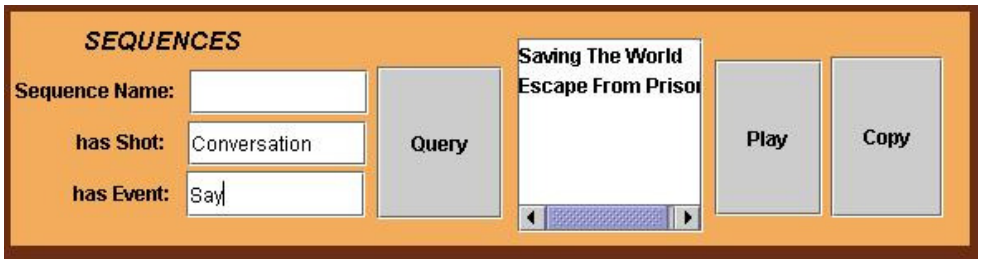


Figure 5.23 – Video Sequence Sub-Model Query Interface

The example XQuery expressions that are used when querying Video Sequence Sub-Model are listed below:

- Video Place of a particular scene named “Saving The World”:

  /Video/Sequences/Sequence[VideoSequenceName="Saving The World"]/../../VideoPlace

- Start Time of a particular scene named “Saving The World”:

  /Video/Sequences/Sequence[VideoSequenceName="Saving The World"]/StartTime

- End Time of a particular scene named “Saving The World”:

  /Video/Sequences/Sequence[VideoSequenceName="Saving The World"]/EndTime

- Sequence Name of a scene that has a particular shot named “Dinner”:

  /Video/Sequences/Sequence/Shots/Shot[VideoShotName="Dinner"]/..

/../VideoSequenceName

- Sequence Name of a scene that has a particular event named “Give”:

  /Video/Sequences/Sequence/Shots/Shot/Events/Event[

  VideoEventName ="Give"]/../../../../VideoSequenceName

- Start time of a particular scene named “Party” that has a particular event named “Give” under a particular shot named “Dinner”:

  /Video/Sequences/Sequence[VideoSequenceName =

  "Party"]/Shots/Shot[VideoShotName="Dinner"]/../../

  Shots/Shot/Events/Event[VideoEventName="Give"]/../../

  ../../StartTime

In Figure 5.24, the interface for querying the Video Shot Sub-Model is shown. In this interface user can do the following queries:

1. Given a shot name, play it. For example: “Play the shot called *dinner”*.
2. Given a sequence name, find the shots of it. For example: “Find the shots of scene called *Party*”. (Hierarchical query)
3. Given an event name, find the shot of it. For example: “Find the shot where the *explosion* occurs”. (Hierarchical query)
4. Given a shot name, find the interval of it. For example: “Find the start and end time of the shot called *dinner*”.

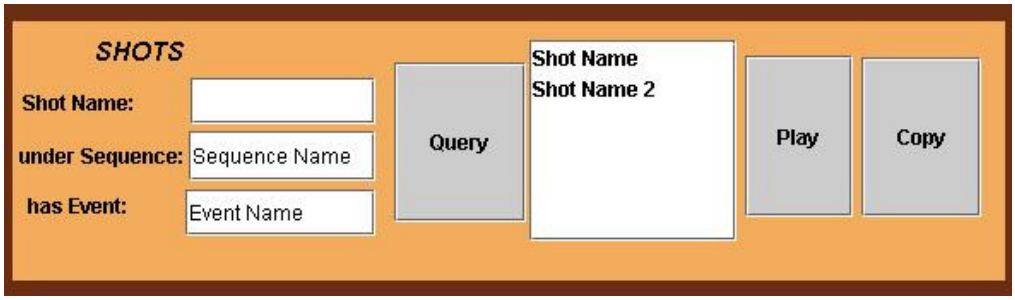


Figure 5.24 – Video Shot Sub-Model Query Interface

The example XQuery expressions that are used when querying the Video Sequence Sub-Model are listed below:

- Video Place of a particular shot named “Dinner”:

  /Video/Sequences/Sequence/Shots/Shot[VideoShotName="Dinner”]

  /../../../../VideoPlace

- Start Time of a particular shot named “Dinner”:

  /Video/Sequences/Sequence/Shots/Shot[VideoShotName="Dinner”]

  /StartTime

- End Time of a particular shot named “Dinner”:

  /Video/Sequences/Sequence/Shots/Shot[VideoShotName="Dinner”]

  /EndTime

- Shot Names of a particular sequence named “Party”:

  /Video/Sequences/Sequence[VideoSequenceName="Party”]

  /Shots/Shot/VideoShotName

- Shot Name of a shot that has a particular event named “Give”:

  /Video/Sequences/Sequence/Shots/Shot/Events/Event

  [VideoEventName="Give"]/../../VideoShotName

- StartTime of a particular scene named “Party” that has a particular event named “Give” under a particular shot named “Dinner”:

  /Video/Sequences/Sequence[VideoSequenceName="Party"]/

  Shots/Shot[VideoShotName="Dinner"]/Events/

  Event[VideoEventName="Give"]/../../StartTime

In Figure 5.25, the Interface for Querying the Video Object Sub-Model and Video Object Characteristics Sub-Model is shown. In this interface user can do the following queries:

1. Given an object Name, finds it through the videos. For example: “Is there *John* in my videos?”
2. Given an event, finds its actors. For example: “Find the actors of fight event.”
3. Given an object finds its characteristics. For example: “Find the characteristics of John.” The answer is “John is friend of Jimmy”.
4. Given a characteristic and/or associated object find the corresponding object. For example “Who is the friend of Jimmy?” or “”Find me the friends” or “Find me the person who has relationship with Kylie”.
5. Combination of first four queries. For example: “Find me the sister of Jimmy who has an actor of fighting event.”

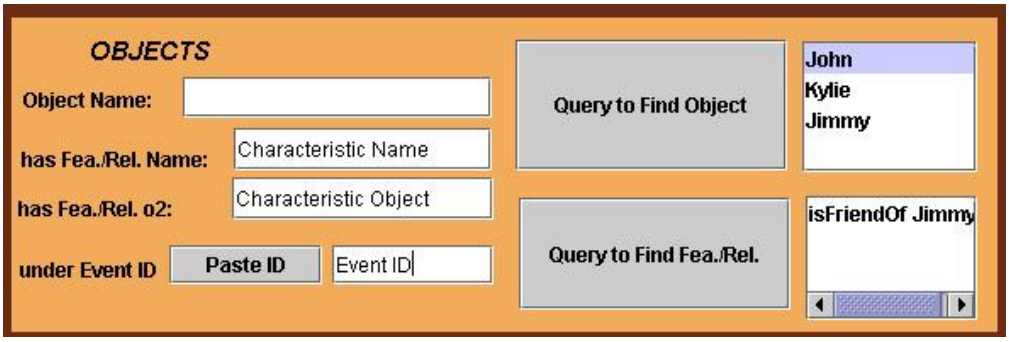


Figure 5.25 - Video Object and Video Object Characteristic Sub-Models Query Interface

The example XQuery expressions that are used when querying the Video Object and Video Object Characteristic Sub-Models are listed below:

- The query to find the person who has a friend:

  /Video/VideoObjects/VideoObject/Features/Feature

  [Relation="isFriendOf"]/../../VideoObjectName

- The query to find the sister of Jimmy:

  /Video/VideoObjects/VideoObject/Features/Feature

  [Relation="isSisterOf"]/../../Features/Feature

  [Object2="Jimmy"]/../../VideoObjectName

- The query to find the characteristics of an object named Kylie:

  /Video/VideoObjects/VideoObject[VideoObjectName="Kylie"]

  /Features/Feature

- The query to find the actors of an event that has an id of 4 (the event name is fight, the id is supplied from Video Event Sub-Model Query Interface)

  /Video/VideoObjects/VideoObject/RollerEventList

  [RollerEvent="4"]/../VideoObjectName

- The query to find the friend of Jimmy who has a role in event fight:

  /Video/VideoObjects/VideoObject/Features/Feature

  [Relation="isFriendOf"]/../../Features/Feature

  [Object2="Jimmy"]/../../RollerEventList[RollerEvent="4"]

  /../VideoObjectName

In Figure 5.26, the Interface for Querying the Video Event Sub-Model is shown. In this interface user can do the following queries:

1. Given an event name, finds the information about it, such as who are the actors and in what roles, what are the timestamps, which modalities it has, etc.
2. Structural Queries such as
   a. Who eat the cake between the minutes of 2 and 3?
   b. Find me the events where John is a subject of the event.

c. Where was John, when eating the cake?

d. Whom did John give the cake to?

e. Find me the events where John is the object of the event.

f. What did John cut the cake with?

g. What did John eat?

h. Did John eat something?

i. Was John is a part of an event called *eat*?

j. How did John eat the cake?

k. Find the events where John eats the cake angrily.

3. Find me, what did john say between the minutes of 2 and 3.
4. Play me the video part where John said “Bon appetite”.
5. Find me the visual events.
6. Did John make noise or did he say something, when eating the cake, i.e. did the event of eating have auditory modality?
7. Find me the events where John gives the salt to Kylie between minutes of 2 and 50.
8. Play me the event where John says “Bon appetite” to Kylie.
9. Play the video part where John singing a song and John is seen on the screen.

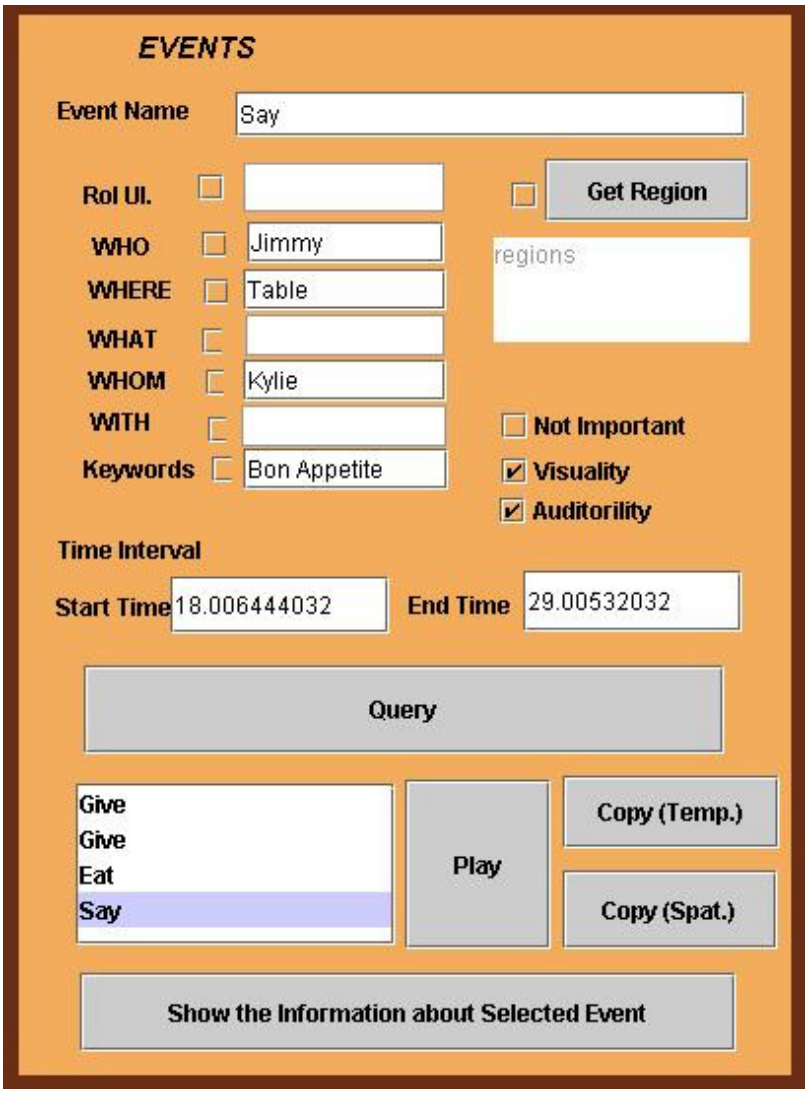


Figure 5.26 – Video Event Sub-Model Query Interface

The example XQuery expressions that are used when querying the Video Event Sub-Model are listed below:

- The query to find the whole events in videos:

  /Video/Sequences/Sequence/Shots/Shot/Events/

  Event/VideoEventName

- The query to find who eat the cake:

  /Video/Sequences/Sequence/Shots/Shot/Events/Event

  [VideoEventName="Eat"]/WHO

- The query to find the visual events where John is a subject of the event:

  /Video/Sequences/Sequence/Shots/Shot/Events/

  Event[WHOM="John"][Visuality="true"]

  [Auditorility="false"]/EventID

- The query to find the events where Jimmy says “Bon Appetite” to Kylie:

  /Video/Sequences/Sequence/Shots/Shot/Events/Event

  [VideoEventName="Say"][WHO="Jimmy"][WHOM="Kylie"]

  [Keywords="Bon Appetite"][Visuality="true"]

  [Auditorility="true"]/EventID

- Time intervals when John gives the salt to Kylie on the table:

  /Video/Sequences/Sequence/Shots/Shot/Events

  /Event[VideoEventName="Give"][WHO="John"]

  [WHERE="Table"][WHAT="Salt"][WHOM="Kylie"]

  [Visuality="true"][Auditorility="false"]/EventID

- Events that Jimmy has a role:

  /Video/VideoObjects/VideoObject[VideoObjectName="Jimmy"]

  /RollerEventList/RollerEvent

- Name of the auditory but not visual events:

  /Video/Sequences/Sequence/Shots/Shot/Events

  /Event[Visuality="false"][Auditorility="true"]/VideoEventName

- The query to find the events where Happy Birthday text seen:

  /Video/Sequences/Sequence/Shots/Shot/Events/

  Event[VideoEventName="isWritten"][Keywords="Happy Birthday"][Visuality="true"][Auditorility="false"]/EventID

Since the XQuery works on discrete values when comparing them, the queries that have expressions like “between minutes of 2 and 3” must be handled by the system. First the media times of the events are taken from the database and then they are looked for the candidate timestamps.

- The query to find the timestamp of the event named *Eat*:

  /Video/Sequences/Sequence/Shots/Shot/Events

  /Event[VideoEventName=”Eat”]/startTime

  and

  /Video/Sequences/Sequence/Shots/Shot/Events

  /Event[VideoEventName=”Eat”]/endTime

5.3.2.2. Temporal Queries

SEBM prototype system has temporal query interface shown in Figure 5.27. This interface is supported by interfaces that belong to the Video Sequence, Shot and Event Sub-Model query interfaces. These interfaces have its own interface to connect it with temporal query interface. These connections are done through “Copy” buttons near query result boxes. Copy buttons copies the temporal information (start and end time) with video name of selected sequence, shot or event. If the query has timestamps on its own, manually the start, end times and video names are entered in temporal query interface and *Copy Manually* button is pressed. Copying process takes the information supplied by the interfaces or entered manually.

The copied information can be handled in temporal query interface in two ways. It is either pasted as a new set or as a continuing set. In this way temporal sets are created. Temporal set identification number is given in

column D. (Column A – Video place, Column B – Start Time, Column C – End Time)

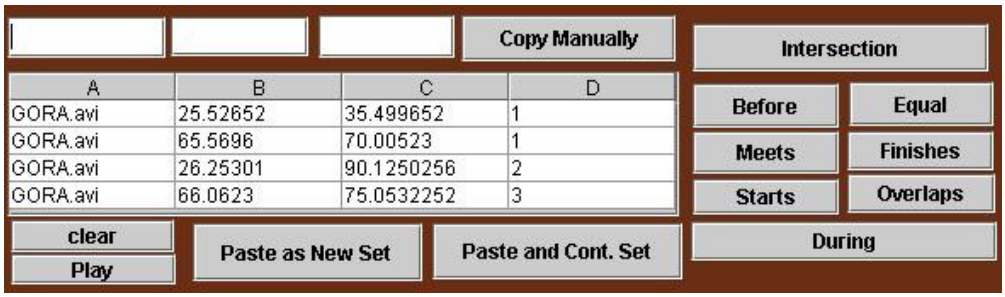


Figure 5.27 – Temporal Query Interface

Definition 2 (Temporal Set): The set of temporal information. Temporal information consists of video identification (video name or video place), start time and end time.

Assume that the user wants to solve the following query:

- Find me the intervals when John says “Hello” to his brother and Kylie is seen on the screen and “Happy Birthday” is written on the board.

Query can be divided into three sub-queries to solve them individually and after solving to combine the result.

- Find me the intervals when John says “Hello” to his brother.
- Find me the intervals when Kylie is seen on the screen.
- Find me the intervals when “Happy Birthday is written on the board.

Every query returns a temporal set as an answer. Assume that first query returns two answers, (these intervals are taken from Video Event Sub-Model Query interface), second query returns one answer and third query returns one answer. These answers are shown in Figure 5.27. Since all the queries are combined

with the temporal keyword “and”, all the temporal sets are intersected. The result is shown in Figure 5.28.

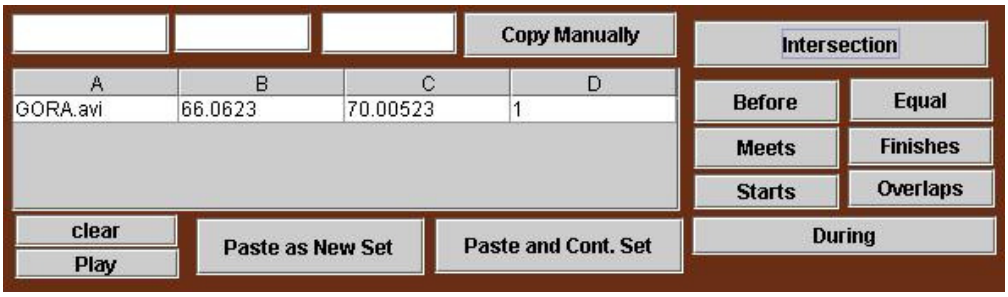
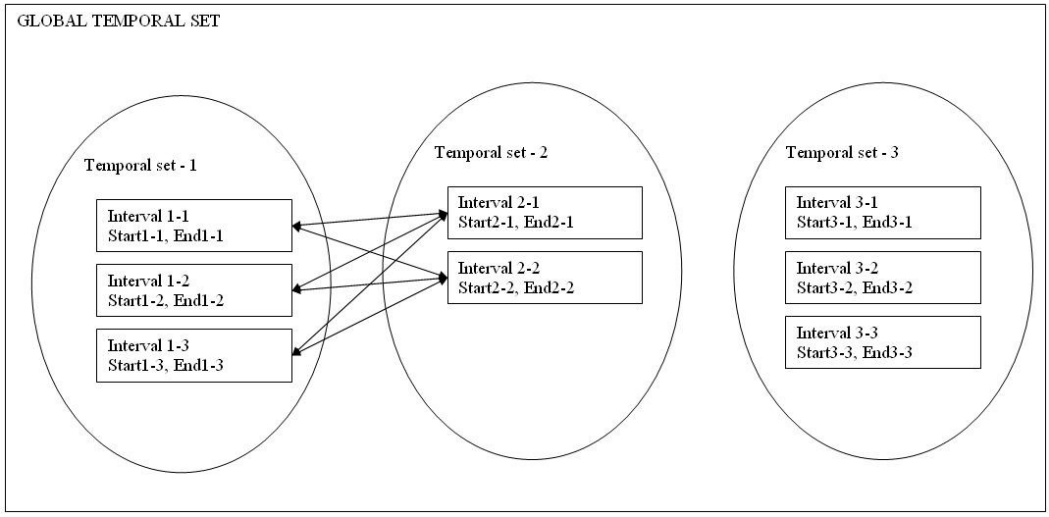


Figure 5.28 – Temporal Query Intersection Result

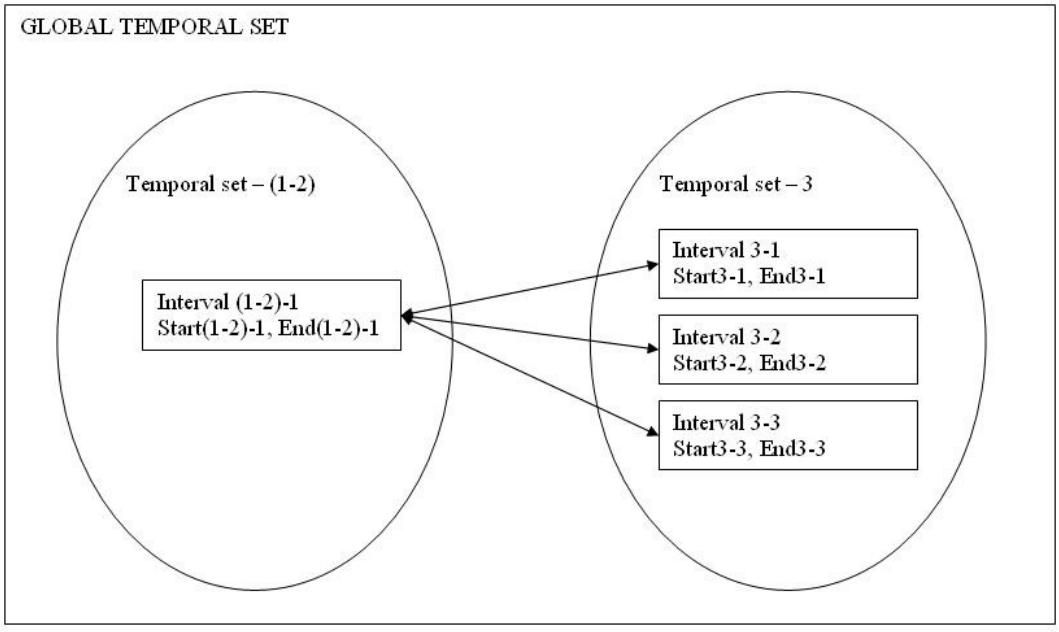
Temporal operators (intersection, before, equal, meets, finishes, starts, overlaps, during) are applied according to the given algorithm below:

1. Take the first two temporal sets. Construct a new set with the values explained in 1.a.
    a. Apply corresponding temporal relation explained in part 4.4 (Temporal Queries) to the every member of the first set with every member of the second set. If the relation returns a new temporal value and the intervals belong to the same video add them to the new set.
2. Add the new constructed set to the global temporal set. Apply the first step, if global temporal has more than one set.

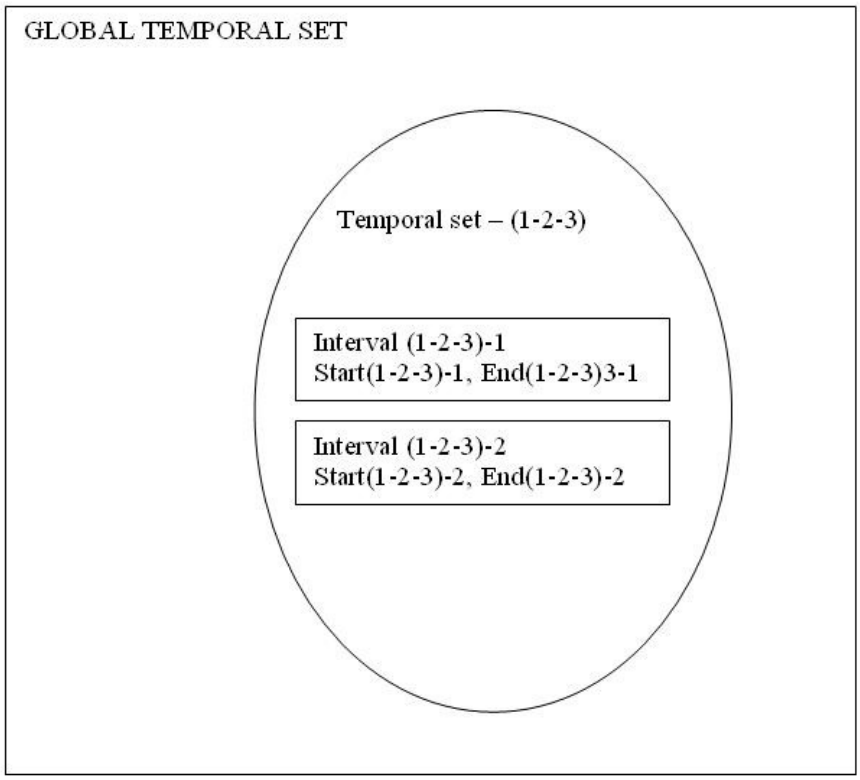
The algorithm run strategy is shown in Figure 5.29 step by step with the example of three set that have three, two and three temporal information respectively.



(a) The algorithm applied to the first two sets of Global set. Result set which has only one interval information is added to the Global set.



(b) The algorithm applied to the two sets of Global set. The result set has two interval data.



(c) Final result

Figure 5.29 – Temporal Operators Applying the Algorithm on Temporal Sets

The algorithm runs in $O(n^2)$. Because the algorithm runs on every set of global set that have O(n) members. Global set has also O(n) members.

When using the SEBM prototype system, the temporal relations given right of the temporal set list is applied. SEBM prototype system has the capability to clear the list for new querying and to play selected temporal information

5.3.2.3. Spatial and Regional Queries Including Trajectory Queries

When processing the Content Based Queries using Video Event Sub-Model query interface, SEBM prototype system can include regional or trajectory information to find the answer. Video Event Sub-Model query interface is used, because the TSRL information is stored directly under Video Event Sub-Model. Visual locations for an event, trajectory of a particular event or text locations can be queried in this way.

The query includes one or two region information. One region is for regional queries and two regions are for trajectory queries. In order to give a user a chance to enter this region information, Video Event Sub-Model query interface has a sub-interface opening a specific video shown in Figure 5.30. Since the stored videos can be in different size, user can use the videos original screens to enter the TSRL information.

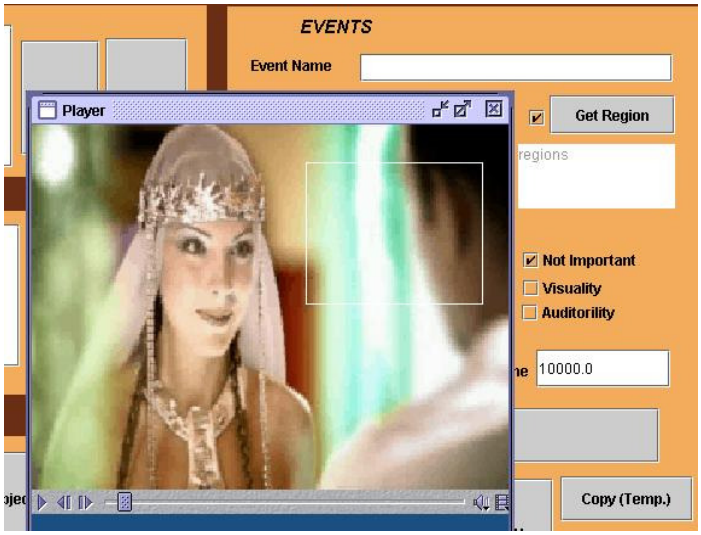


Figure 5.30 – Entering Region Information for Regional Querying

Example for a regional query can be "Find me the intervals where John is seen right of Kylie". When querying the "right of Kylie", the rectangle is drawn to the screen. Maybe Kylie is standing for a while and John is passing near him through the rectangle drawn.

SEBM prototype system runs the following algorithm for the regional queries.

1. For every event,
    a. For every region information of the event,
        i. If the region that user enters while creating the query is equal to or contains the region of the event as shown in Figure 5.31, add the event to the result list.

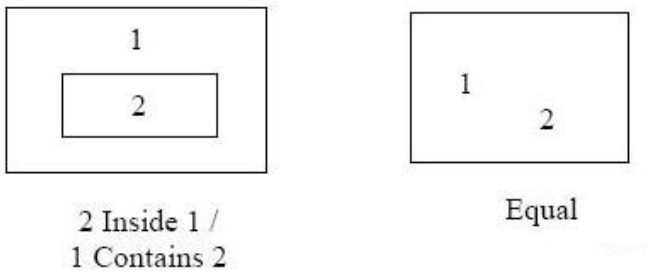


Figure 5.31 – Region Query Relationships

SEBM prototype system runs the following algorithm for the trajectory queries:

1. For every event,
    a. For every adjacent two region information of the event,
        i. If the first region that user enters while creating the query is equal to or contains the first region of the event and the second region that user enters while creating the query is equal to or contains the second region of the

event as shown in Figure 5.32, add the event to the result list.

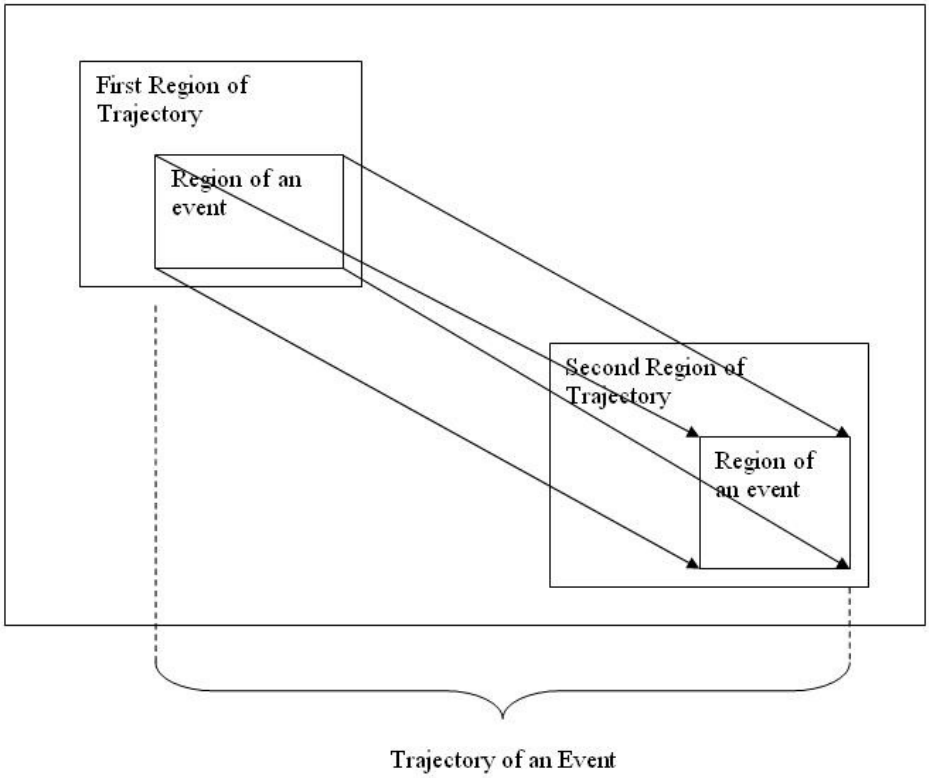


Figure 5.32 – Trajectory Query Relationship

Spatial queries consists keywords for comparing two events TSRL information such as left, top, top-left or top-right. Other relations such as right or below can be thought as the inverse of left or top. Spatial relations consider the timestamps of TSRL information different than regional and trajectory queries, because the events must happen at the same time in the same video. Since SEBM model stores the TSRL information not storing whole trajectory but only beginning, ending and direction changing locations, the whole trajectory of the events are created in run time. Trajectory creation process considers time with width and height changing ratios through time. After trajectory creation process, the TSRL information consist not only beginning and ending locations but intermediate locations. An example trajectory creation process is shown in Figure 5.33.

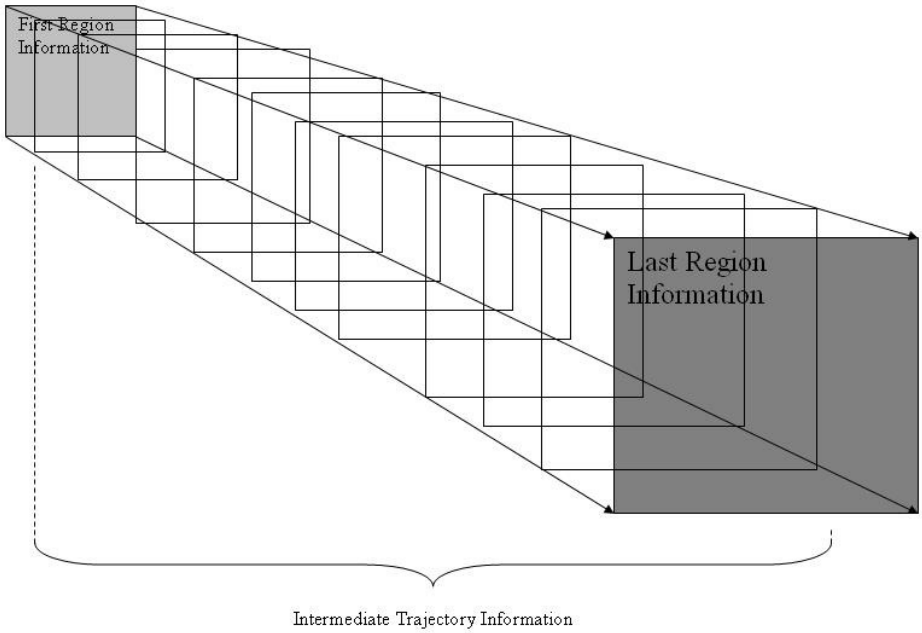


Figure 5.33 – Trajectory Creation Process for a Single Event

Every intermediate trajectory region has timestamp and belongs to some video in database. SEBM prototype system runs following algorithm for spatial queries by using the spatial query interface given in Figure 5.34:

1. For every two event,
    a. For every region of the first and second event that has same timestamp (or approximate timestamps)
        i. If the centers of regions provide the directional relations given in Section 3.3 then the timestamp is added to the result list.

In Spatial Query Interface the TSRL information of the events are copied through “Copy Spatial” button in Video Event Sub-Model Query Interface. (Threshold value is considered in Fuzzy queries)

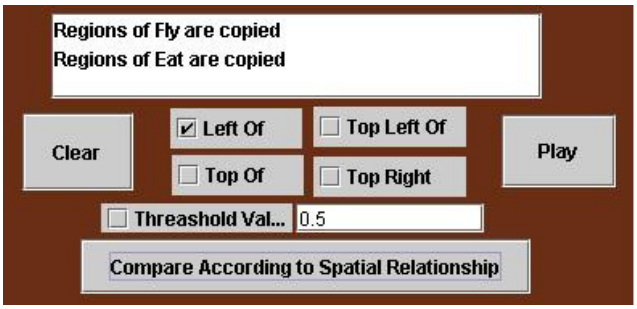


Figure 5.34 - Spatial Query Interface

Examples for a spatial query can be “Find me the intervals where Kylie stands left of John” or “Find me the intervals where the plane is flying just above the car, going on the road”. When querying, left of the table is drawn to the screen with a region.

5.3.2.4. Fuzzy Spatial Queries

Fuzzy spatial queries are created from the spatial queries by giving some threshold value. This threshold value is used to find the membership value. Since directional relations can have uncertainty, the threshold value is used when expressing this uncertainty. Threshold value is given between 0.1 and 1.0.

SEBM prototype system runs the following algorithm for fuzzy spatial queries by using the fuzzy spatial query interface given in Figure 5.35:

1. For every two event,
    a. For every two region pair of the first and second event that has same timestamp (or approximate timestamps)
        ii. If the centers of regions provide the directional relations given in Section 3.3 and the threshold value is smaller

than calculated membership value (using Table 3.3), then the timestamp is added to the result list.

Figure 5.35 – Fuzzy Spatial Query Interface Considering the Threshold Value

5.3.2.5. Compound Queries

Compound Queries are the queries compounded from other type of queries, i.e. content based, structural, hierarchical, spatial, regional trajectory or fuzzy. There is no depth or width restriction when creating this type of queries. Nesting can be used indefinitely.

In order to solve this type of queries, how they are created should be understood. For example beginning from a simple example how a complex compound query can be created is shown below:

1. Play me the interval of [2.5, 32.5] minutes in video A.

   Replace "of [2.5, 32.5] minutes in video A" with the query sub sentence, "when Jimmy gives a hamburger to Kylie"

2. Play me the interval when Jimmy gives a hamburger to Kylie.

   Replace Kylie with "John's sister".

3. Play me the interval when Jimmy gives a hamburger to John's sister.

Replace Jimmy with “The person who says “Bon Appetite” to George at [65.0, 88.6]”

4. Play me the interval when the person who says “Bon Appetite” to George at [65.0, 88.6], gives a hamburger to John’s sister.

Replace “at [65.0, 88.6]” with “when a strange noise is heard and Happy Birthday is written on the board”.

5. Play me the interval when the person who says “Bon Appetite” to George when a strange noise is heard and “Happy Birthday” is written on the board, gives a hamburger to John’s sister.

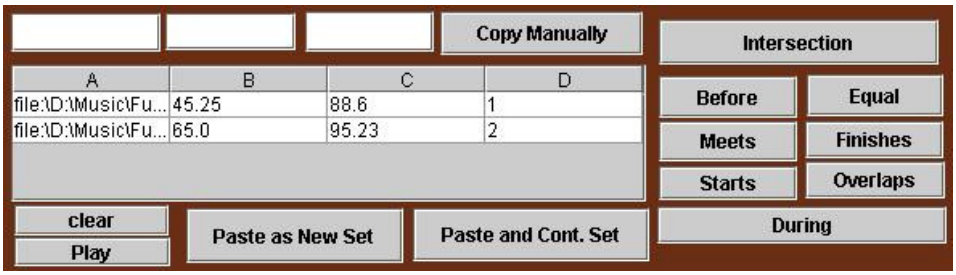
As we can see from the above example every replacement can be done with another query sentence. So if a user wants to solve the query like given in 5, s/he should follow these steps backward.

1. The event “when a strange noise is heard” should be found. By using Video Event Sub-Model Query Interface as shown in Figure 5.36-a. Then the temporal information of founded event should be copied to the temporal query Interface as a new set.
2. The event “when Happy Birthday is written on the board” should be found. By using Video Event Sub-Model Query Interface as shown in Figure 5.36-b. Then the temporal information of founded event should be copied to the temporal Query Interface as a new set
3. The two set should be intersected by using Temporal Query Interface as shown in Figure 5.36-c. In this example result is one interval data [65.0, 88.6] as shown in Figure 5.36-d. Assume that the events occur in the same video.
4. The event named say and consist George as an indirect object, keywords as “Bon Appetite” and timestamp [65.0, 88.6] is found as shown in Figure 5.36-e. Then the subject of the event is found by using Video Event Sub-Model Query Interface as shown Figure 5.36-f. Result is Jimmy.

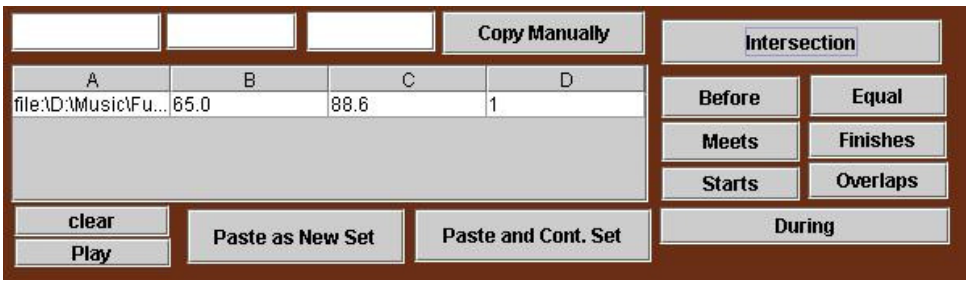
5. The event named “Give” and consist “Jimmy” as subject, hamburger as object and sister of John who is “Kylie” as indirect object (Figure 5.36-g), is found by using Video Event Sub-Model Query Interface as shown in Figure 5.36-h.
6. The event interval [2.5, 32.5] is played by using Play button near the query results as shown in Figure 5.36-i.

(a) The event when a strange noise is heard.

(b) The event when Happy Birthday is written on the board.

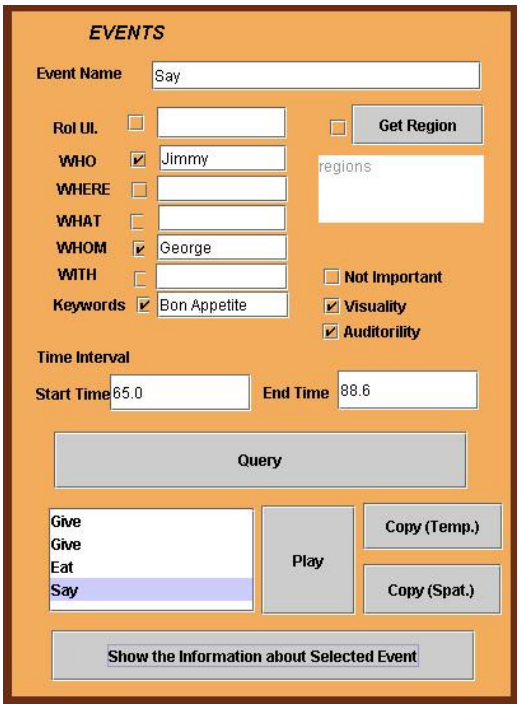


(c) Interval sets of two events

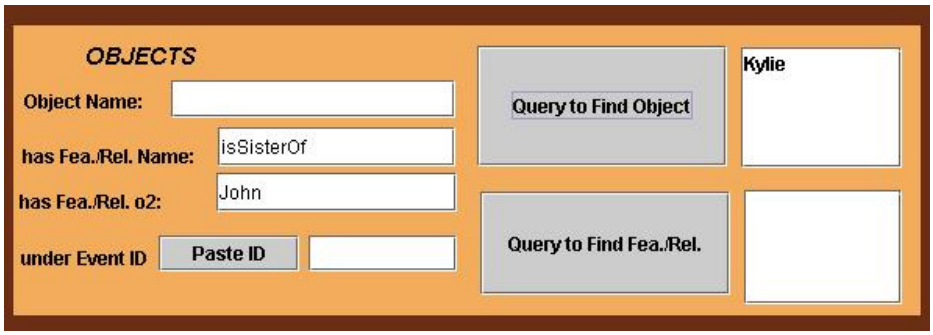


(d) Intersection of two sets

(e) The event called “Say”



(f) The subject of the event is found



(g) The sister of John is found

(h) The event called “Give”

(i) Playing the event “Give”

Figure 5.36 – Compound Query Solving Example

The SEBM prototype system also has an advantage for the users who want to write XQueries directly. In Figure 5.37 this interface is shown.

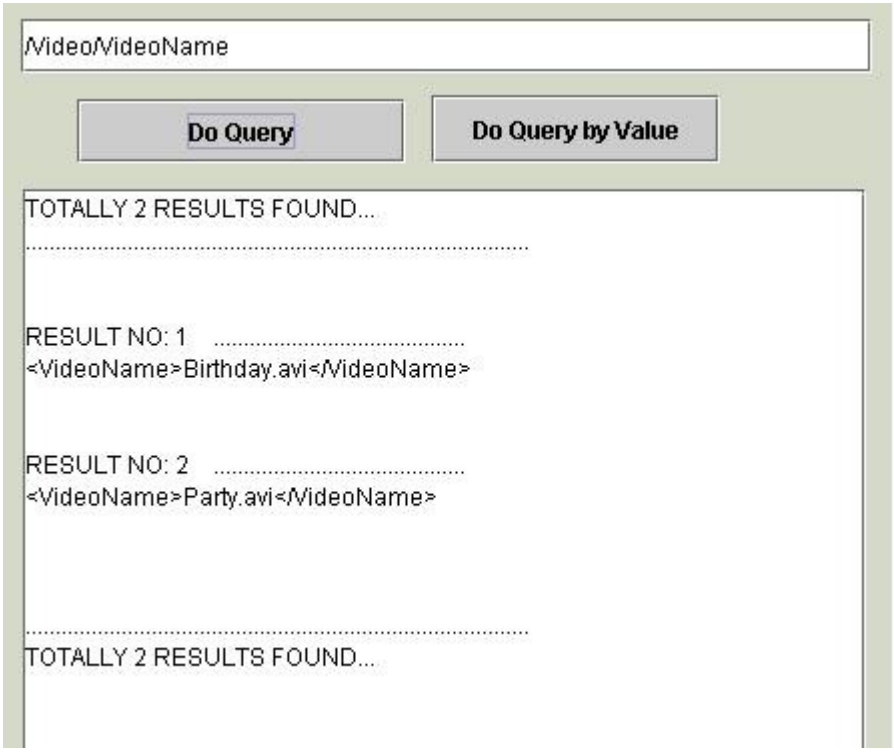


Figure 5.37 – XQuery Interface of SEBM Prototype System

Some queries that are run on this screen is listed below:

- //*/VideoObject/RollerEventList[RollerEvent=//*/Event[Auditorility="true"]/EventID]/../VideoObjectName

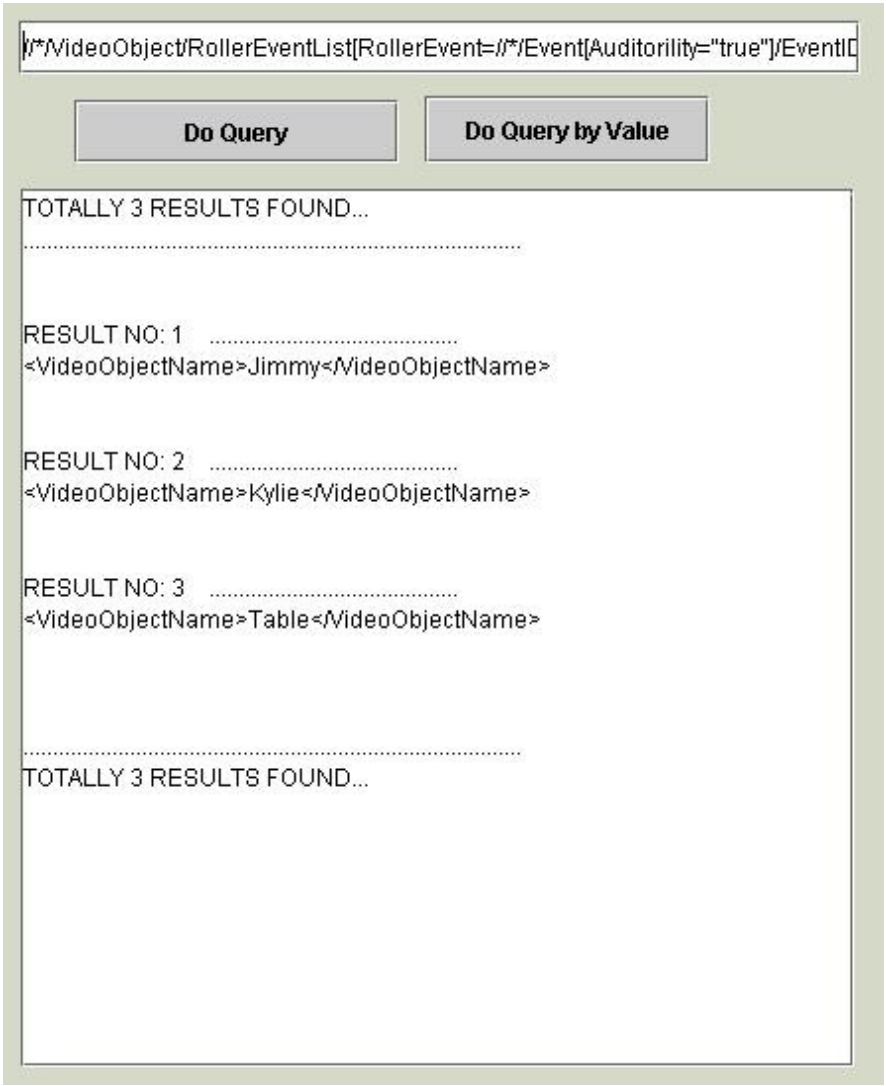


Figure 5.38 - Name of Auditory Objects

- //*/Event[Visuality="true"][Auditorility="true"]/VideoEventName

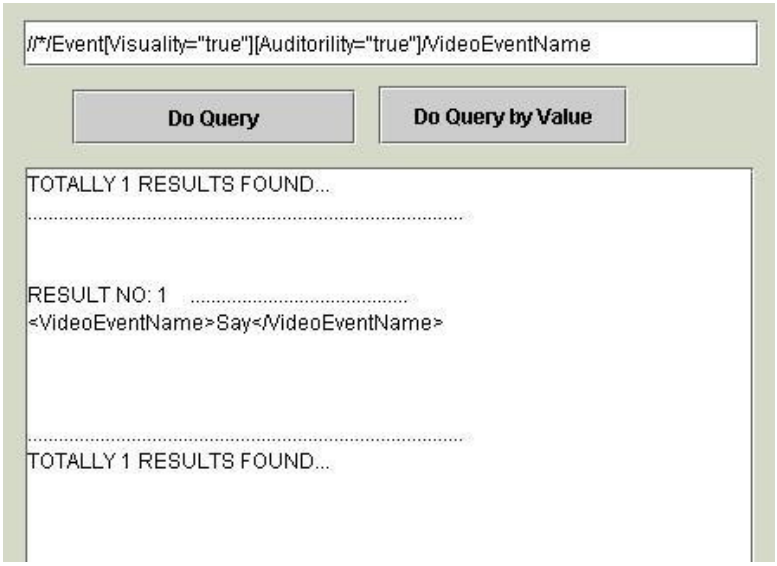


Figure 5.39 – Name of the Event that is Both Visual and Auditory

- //*/Feature[Object2=//*/Feature[Object2="engineer"]/../../VideoObject Name][Relation="isFriendOf"]/../../VideoObjectName

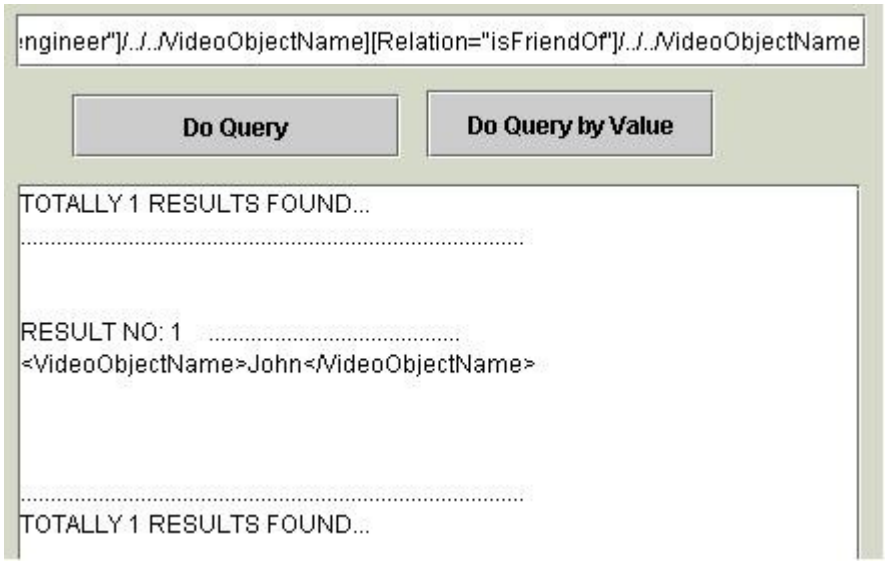


Figure 5.40 - Name of the Person who is the Friend of the Person who is an Engineer

CHAPTER 6

DISCUSSIONS

To evaluate the power of SEBM video data model, we have concentrated on two different aspects of video databases, annotation and querying aspects. Since we propose that SEBM lies on human interpretation of real world and we propose it as a general purpose model that can be applied to any area, we have worked on movies that can have subjects of any kind. The relationships between characters of movies give us the ability to study more on Video Object and Video Object Characteristics Sub-Models. Since there is an intense usage of three information channels in movies, we can concentrate on three modalities at the same time, which gives us the ability to work on elimination of modalities in Video Event Sub-Model.

We use the IBM MPEG-7 Annotation Tool while segmenting the movie into smaller parts and extracting the key frames of video segments, so that we can differentiate the advantages and disadvantages of using automatization tools. By using tools, we bring automatization to annotation, different from [2, 3 and 4]. We see that automatization can save a lot of time when segmenting, but if there is a lot of background changing; time is also consumed when merging the shots as explained in subsection 5.1.3.1. For example, for a movie clip that has length of 2:29 minutes IBM MPEG-7 annotation tool produces its segmentation output in 13 seconds with 31 different shots. But, in order to produce merged shots, we spent 40 seconds when reducing the shot group of 31 shots to 5 different shots. Totally 53 seconds are spent. If there is no tool usage, total production can be done, approximately, in 2 minutes. In that aspect,

the tool usage brings time advantage in our system. Since other parts different than video segmentation of SEBM video data model require artificial intelligence, when annotating the video, we do not prefer using automatization tools in those parts. We think that the tools can extract much or less information than we expect. For instance, for football videos, the eye color property of the players that we can store in Video Object Characteristics Sub-Model, can be important for doctors of the teams and they want to have it in their video database, but not important for football commentators. Hence, when using tools, optimizations must be done according to the selected area.

Another subject about automatization is using a tool that only considers visual part. We decided to segment the video by using visual features, because our main idea about multimodality is to embed the three channels into one single SEBM video data model, so by working on only one channel, we can segment the video according to other channels too. This idea is supported by the nature of videos also. The probability of shot changing of the video, where there is a huge changing in audial part but not in visual part, is low. For instance, if there is an explosion sound in the background, in general we expect some changing in visual scene, either an explosion image appears or people are seen escaping. In this way, we discard the redundant work on segmentation. This proposal is also parallel to [23 and 46] which propose that it is quite expensive to work on more than one information channel. But they work on audial part rather than visual part.

The model we propose gives us the ability to solve much more complex and nested queries different than [2 and 3]. We support this idea by implementing the SEBM prototype system. Since the user can form compound queries and answer them by using querying interface of SEBM prototype system shown in part 5.3.2.5, we have shown that SEBM can alter query intensity. SEBM prototype system shows the success of SEBM on compound querying rather than other systems like [43, 44 and 62].

CHAPTER 7

CONCLUSION

The SEBM video data model, which we propose in this study, makes it easy and effective to store the structural semantic information in a video database and then query the database. This video data model can be adapted to various domains, because it is based on understanding of the human about a particular video. Visual, auditory, and textual information in video are all considered in our model. Challenges on modelling the structure of the video are altered by using several sub-models. Very tightly coupled relations among these models result in more information embedded into the model than treating each independently. Query diversity is supported in our model. Content based, fuzzy, spatial and temporal queries are all supported. In our implementation, XML and Java are used to model the information in video and Berkeley XML DBMS is used to store and retrieve video information. Automatization in annotation of the video is part of our implementation. IBM MPEG-7 Annotation Tool is used for this purpose. For querying, XQuery facility of Berkeley XML DBMS is utilized.

Some interesting future works for our model are to develop an implicit characteristic inference mechanism and to include symmetry or transitivity into the model by using ontology. For example, if John is brother of Kylie, this implicitly means that video object named Kylie automatically will have the relation of VOC={sisterof, John} where VOC is Video Object Characteristics. This means that, a kind of inference

mechanism should be considered. If John has a characteristic VOC={brotherof, George}, this means that there should exist a characteristic of George that is VOC={brotherof, John}. This means that, symmetry relation should be considered. If John has a characteristic VOC={brotherof, George} and George has a characteristic VOC={brotherof, Bill} this means that there should exist a characteristic of John that is VOC={brotherof, Bill}. This means that, transitivity relation should also be considered.

Another future work is related to implicit features. Implicit features, come from Video Event Model and related to Video Object Characteristics, should implicitly be considered under that Video Object Characteristics model. For example, if there is an event drive which have WHO=John this means that John should have a characteristic that VOC={be, driver} automatically. But this inference requires ontology that has the information of driver, name of the person who drives.

In SEBM video data model, fuzziness is included only in spatial relations. However object features or relations may have some fuzzy aspects. For instance, the video object “John” may have not “yellow” or “red” but “orange” T-Shirt. His friendship with “Jimmy” may be stronger than his friendship with “George”. This fuzziness requires extra information in Video Object Characteristics Sub-Model. Hence fuzzy queries would be richer.

Natural language (i.e Turkish) interface is an important future extension to integrate with SEBM video data model, because we have developed the SEBM by using human interpretation of video data. This means that query sentences coming from the user in natural languages can be converted to the corresponding SEBM sub-model queries. Such as the verbs of the query sentence will be looked in Video Event Sub-Model. The organization of the objects in sentences i.e. subject, object, indirect object or accompanied object can give the structure of the Video Events. Video

characteristics can also easily be distinguished in query sentences, because they have keywords like "is", "has" etc.

By developing SEBM video data model we see that, we have developed a model for the real world also. Since video data is a reflection of the real world in digital world, this work can be adapted to the real world objects and events where video is not considered. For simulating real world objects and their movements in real world, the SEBM model can be useful.

On the other hand, we imagine if one implements the SEBM video data model that consists only the information that s/he needs, s/he should submit SEBM model with some visual key frames of objects from the server side and make it possible to regenerate the video on the client side. This work can reduce the size of the transmission as we think a single video in size 100 Mb and corresponding SEBM model 100 Kb and object key frames 1 Mb. But this work needs artificial intelligence and machine learning techniques to regenerate the events.

REFERENCES

[1] C. Snoek, M. Worring, "Multimodal Video Indexing: A review of the State-of-the-art", Multimedia Tools and Applications, 25, pp: 5-35, 2005.

[2] N. Durak, "Semantic Video Modeling And Retrieval With Visual, Auditory, Textual Sources", Ms Thesis, METU, 2004.

[3] M. Köprülü, N.K. Çiçekli, A.Yazıcı, "Spatio-temporal querying in video databases". *Inf. Sci. 160(1-4),* pp.131-152, 2004.

[4] A. Ekin, M. Tekalp, R. Mehrotra, "Integrated Semantic–Syntactic Video Modeling for Search and Browsing" in IEEE Transactions on Multimedia, Vol. 6, NO. 6, December 2004.

[5] Y. F. Day, S. Dağtaş, M. Iino, A. Khokhar, A. Ghafoor, "Object Oriented Conceptual Modeling of Video Data", Proc. Data Eng. (DE '95), pp. 401-408, 1995.

[6] D. Tjondronegoro, P. Chen, "Content-Based Indexing and Retrieval Using MPEG-7 and X-Query in Video Data Management Systems", World Wide Web: Internet and Web Information Systems, 5, 207–227, 2002.

[7] E. Oomoto, K. Tanaka, "OVID: Design and implementation of a video-object database system," IEEE Trans. Knowledge Data Eng.,vol.5, pp.629–643, Aug. 1993.

[8] Ö. Ulusoy, U. Güdükbay, M. Dönderler, Ş. Ediz, C. Alper, "BilVideo Database Management System", Proceedings of the 30th VLDB Conference, Toronto, Canada, 2004.

[9] M. Hacid, J. Kouloumdjian, "A database approach for Modeling and querying video data", IEEE Transactions on Knowledge and Data Engineering, vol. 12, no. 5, September /October 2000.

[10] C. Zhang, S. Chen, M. Shyu, "PixSO: A System for Video Shot Detection", Proceedings of the Fourth IEEE Pacific-Rim Conference On Multimedia, pp. 1-5, December 15-18, 2003, Singapore.

[11] V. Oria, P. Iglinski, T. Özsu, "A Framework for Multimedia Database Systems", 4th African Conference on Research in Computer Science, Dakar, Senegal, October 1998, pp. 293-304.

[12] W. Aref, A. Catlin, J. Fan, A. Elmagarmid, M. Hammad, I. Ilyas, M. Marzouk, X. Zhu, "A Video Database Management System for Advancing Video Database Research", International Workshop on Multimedia Information Systems, MIS 2002, Arizona, pp. 8-17.

[13] W. Aref, M. Hammad, A. Catlin, A. Elmagarmid, I. Ilyas, M. Marzouk, T. Ghanem, "Video Query Processing in the VDBMS Testbed for Video Database Research", The ACM International Workshop on Multimedia Databases ACM MMDB 2003, New Orleans, Louisiana, pp. 25-32.

[14] S. Chen, M. Shyu, C. Zhang, R. Kashyap, "Identifying Overlapped Objects for Video Indexing and Modeling in Multimedia Database Systems", International Journal on Artificial Intelligence Tools 10(4): 715-734, 2001.

[15] IBM MPEG-7 Annotation Tool Web site,

www.alphaworks.com/tech/videoannex, Last date accessed: September, 2005.

[16] Berkeley DB XML Web Site, www.sleepycat.com, Last date accessed: September, 2005.

[17] XQuery Web Site, www.w3.org/XML/Query, Last date accessed: September, 2005.

[18] XML Web Site, www.w3.org/XML, Last date accessed: September, 2005.

[19] Informedia-II Digital Video Library Web Site, www.informedia.cs.cmu.edu/, Last date accessed: October 2005

[20] XML.COM Web Site, "A Technical Introduction to XML", http://www.xml.com/pub/a/98/10/guide0.html?page=1, Last date accesses: November, 2005.

[21] MSDN Library – January 2005, XML in Action, Microsoft Corporation, 2003.

[22] J. Allen, "Maintaining Knowledge about Temporal Intervals", Communications of ACM, 26 (11), pp. 832-843, 1983.

[23] Y. Rui, A. Gupta, A. Acero, "Automatically extracting highlights for TV baseball programs," in ACM Multimedia 2000, Los Angeles, USA, 2000, pp. 105–115.

[24] S. Adali, K.S. Candan, S.S. Chen, K. Erol, V.S. Subrahmanian, "The advanced video information system: Data structures and query processing," Multimedia Systems, Vol. 4, No. 4, pp. 172–186, 1996.

[25] Y. Zhong, H.-J. Zhang, A.K. Jain, "Automatic caption localization in compressed video," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 22, No. 4, pp. 385–392, 2000.

[26] B. Günsel, A.M. Ferman, A.M. Tekalp, "Video indexing through integration of syntactic and semantic features," in Third IEEE Workshop on Applications of Computer Vision, Sarasota, USA, 1996.

[27] N. Babaguchi, Y. Kawai, T. Kitahashi, "Event based indexing of broadcasted sports video by intermodal collaboration," IEEE Transactions on Multimedia, Vol. 4, No. 1, pp. 68–75, 2002.

[28] I. Ide, K. Yamamoto, H. Tanaka, "Automatic video indexing based on shot classification," in First International Conference on Advanced Multimedia Content Processing, Vol. 1554 of Lecture Notes in Computer Science, Springer-Verlag: Osaka, Japan, 1999.

[29] O. Javed, Z. Rasheed, M. Shah, "A framework for segmentation of talk & game shows," in IEEE International Conference on Computer Vision, Vancouver, Canada, 2001.

[30] M. La Cascia, S. Sethi, S. Sclaroff, "Combining textual and visual cues for content-based image retrieval on the world wide web," in IEEE Workshop on Content-Based Access of Image and Video Libraries, 1998.

[31] H. Li, D. Doermann, O. Kia, "Automatic text detection and tracking in digital video," IEEE Transactions on Image Processing, Vol. 9, No. 1, pp. 147–156, 2000.

[32] E.Wold, T. Blum, D. Keislar, J. Wheaton, "Content-based classification, search, and retrieval of audio," IEEE Multimedia, Vol. 3, No. 3, pp. 27–36, 1996.

[33] H. Pan, P. Van Beek, M.I. Sezan, "Detection of slow-motion replay segments in sports video for highlights generation," in IEEE International Conference on Acoustic, Speech and Signal Processing, 2001.

[34] S. Tsekeridou, I. Pitas, "Content-based video parsing and indexing based on audio-visual interaction," IEEE Transactions on Circuits and Systems for Video Technology, Vol. 11, No. 4, pp. 522–535, 2001.

[35] R.M. Bolle, B.-L. Yeo, M.M. Yeung, "Video query: Research directions," IBM Journal of Research and Development, Vol. 42, No. 2, pp. 233–252, 1998.

[36] C.G.M. Snoek, M. Worring, "Multimedia event based video indexing using time intervals" Technical Report 2003-01, Intelligent Sensory Information Systems Group, University of Amsterdam, August 2003.

[37] Q. Huang, A. Purti, Z Lui, " Multimedia Search and Retrieval: New Concepts, System Implementation, and Application, IEEE Transactions on Circuits and Systems for Video Technology, Vol 10, No 5, August 2000.

[38] W. Zhou, S. Dao, J. Kuo, "On-Line Knowledge and Rule-Based Video Classification System for Video Indexing and Dissemination", Information Systems, 27, 559-586, 2002.

[39] S. Chang, W. Chen, H. Meng, H. Sundaram, D. Zhong, " A Fully Automated Content-Based Video Search Engine Supporting Spatiotemporal Queries", IEEE Transactions on Circuits and Systems for Video Technology, Vol. 8, No. 5, September 1998.

[40] J. Lee, J. Oh, "STRG-QL: Spatio-Temporal Region Graph Query Language for Very Large Video Databases", Proceedings of the 31st VLDB Conference, Trondheim, Norway, 2005.

[41] J. Lee, J. Oh, S. Hwang, "STRG Index: Spatio Temporal Region Graph Indexing for Large Video Databases", SIGMOD Conference 2005: 718-729.

[42] D. DeMenthon, D. Doerman, **"**Video Retrieval Using Spatio-Temporal Descriptors", ACM Multimedia '03, Berkeley, California, USA, November 2-8, 2003.

[43] S. Hammiche, S. Benbernou, M. Hacid, A. Vakali," Semantic Retrieval of Multimedia Data", MMDB'04, November 13, 2004, Washington, DC, USA.

[44] M.Lyu, E. Yau, S. Sze, "A Multilingual, Multimodal Digital Video Library System", JCDL'02, July 13-17, 2002, Portland, Oregon, USA.

[45] A. Hauptmann, R. Jin, T. Ng, "Video Retrieval Using Speech and Image Information, Electronic Imaging Conference (EI'03) Storage Retrieval for Multimedia Databases, Santa Clara, CA, January 20-24, 2003.

[46] Y. Chang, W. Zeng, I. Kamel, R. Alanso, "Integrated Image and Speech Analysis for Content-Based Video Indexing", IEEE Conf. on Multimedia Systems and Computing, 1996.

[47] J. Fan, W. Aref, A. Elmagarmid, M. Hacid, M. Marzouk, X Zhu, "MultiView: Multilevel Video Content Representation and Retrieval", Journal of Electronic Imaging / October 2001, Vol. 10(4), 895.

[48] A. Hauptmann, M. Witbrock, "Story Segmentation and Detection of Commercials in Broadcast News Video", Proceedings of the Advances in Digital Libraries Conference, 1998.

[49] W. Ren, M. Singh, S. Singh, "Automated video segmentation", Proc. 3rd International Conference on Information, Communications & Signal Processing, ICICS 2001, Singapore, October 2001.

[50] A. Hauptmann, A. Olligschlaeger, "Using location Information from Speech Recognition of Television News Broadcast", Proceedings of the ESCA ETRW Workshop on Accessing Information in Spoken Audio, Cambridge, England, April 19-20, 1999.

[51] H. Wactlar**, "**New Directions in Video Information Extraction and Summarization", DELOS Workshop, June, 1999.

[52] N.V. Patel, I.K. Sethi, "Audio characterization for video indexing," in Proceedings SPIE on Storage and Retrieval for Still Image and Video Databases, San Jose, USA, 1996, Vol. 2670, pp. 373–384.

[53] M. Chen, A. Hauptmann, "Searching for a Specific Person in Broadcast News Video", International Conference on Acoustics, Speech and Signal Processing, Montreal, Canada, May 17-21, 2004.

[54] G. Tzenetakis, M. Chen, **"**Building Audio Classifiers for Broadcast News Retrieval", 5th International Workshop on Image Analysis for Multimedia Interactive Services, Lisboa, Portugal, April 21-23, 2004.

[55] M. Christel, C. Huang, "Enhanced Access to Digital Video Through Visually Rich Interfaces", Proceedings of the IEEE International Conference on Multimedia and Expo, Baltimore, July 6-9, 2003.

[56] H. Zhong, J. Shi, M. Visontai, **"**Detecting Unusual Activity in Video", IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Washington, June 27- July 2, 2004.

[57] P. Duygulu, H. Wactlar, N. Papernick, D. Ng, "Linking Visual and Textual Data in Video", Workshop on Multimedia Contents in Digital Libraries, Greece, June 2-3, 2003.

[58] P. Duygulu, A. Hauptmann, "What's news, What's not? Associating News Videos with Words", The 3rd International Conference on Image and Video Retrieval, Dublin, Ireland, July 21-23, 2004.

[59] W. Wu, X. Chen, J. Yang, "Incremental Detection of Text on Road Sign from Video with Application to a Driving Assistant System", Proceedings of ACM Multimedia 2004, New York, October, 10-16, 2004.

[60] J. Yang, A. Hauptmann, "Video Grammar for Locating Named People", Joint Conference on Digital Libraries, Tucson, June 7-11, 2004.

[61] Pradhan S., Tajima K., Tanaka K., "A Query Model to Synthesize Answer Intervals from Indexed Video Units", IEEE Trans. on Knowledge and Data Eng. Vol.13, No.5, pp. 824-838, Sept./Oct. 2001.

[62] T. Kuo and A. Chen, "Content-Based Query Processing for Video Databases", IEEE Trans. on Multimedia. Vol.2, No.1, March 2000.

[63] BKM Film (Producer) and Ö. F. Sorak (Director), (2004), G.O.R.A. (Figures are courtesy of BKM Film).

[64] STOIK Imaging Web Site, www.stoik.com, Last date accessed: September, 2005.

APPENDICES

A. IBM MPEG–7 ANNOTATION TOOL OUTPUT EXAMPLE

```
<?xml version="1.0" encoding="iso-8859-1"?>
<Mpeg7 xmlns="urn:mpeg:mpeg7:schema:2001"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:mpeg7="urn:mpeg:mpeg7:schema:2001"
 xsi:schemaLocation="urn:mpeg:mpeg7:schema:2001 Mpeg7-
 2001.xsd">
  <Description xsi:type="ContentEntityType">
    <MultimediaContent xsi:type="VideoType">
      <Video>
        <TemporalDecomposition>
          <VideoSegment>
            <MediaTime>
              <MediaTimePoint>
                    T00:00:00:0F30000
               </MediaTimePoint>
              <MediaIncrDuration
                 mediaTimeUnit="PT1001N30000F">
                    91
               </MediaIncrDuration>
            </MediaTime>
            <TemporalDecomposition>
              <VideoSegment>
```

Part 1

```
            <MediaTime>
               <MediaTimePoint>
                 T00:00:01:15045F30000
               </MediaTimePoint>
             </MediaTime>
           </VideoSegment>
         </TemporalDecomposition>
       </VideoSegment>
       <VideoSegment>
         <MediaTime>
           <MediaTimePoint>
                 T00:00:03:5095F30000
            </MediaTimePoint>
           <MediaIncrDuration
             mediaTimeUnit="PT1001N30000F">
                 98
            </MediaIncrDuration>
         </MediaTime>
         <TemporalDecomposition>
           <VideoSegment>
             <MediaTime>
               <MediaTimePoint>
```

Part 2

```
                    T00:00:04:23143F30000
                </MediaTimePoint>
              </MediaTime>
            </VideoSegment>
          </TemporalDecomposition>
        </VideoSegment>
        <VideoSegment>
          <MediaTime>
            <MediaTimePoint>
               T00:00:06:20200F30000
             </MediaTimePoint>
            <MediaIncrDuration
              mediaTimeUnit="PT1001N30000F">
                 53
             </MediaIncrDuration>
          </MediaTime>
          <TemporalDecomposition>
            <VideoSegment>
              <MediaTime>
                <MediaTimePoint>
                       T00:00:07:16226F30000
                </MediaTimePoint>
```

Part 3

```
              </VideoSegment>
            </TemporalDecomposition>
          </VideoSegment>
        </TemporalDecomposition>
      </Video>
    </MultimediaContent>
  </Description>
</Mpeg7>
```

Part 4

B. ANNOTATION AND QUERY INTERFACES OF SEBM PROTOTYPE SYSTEM

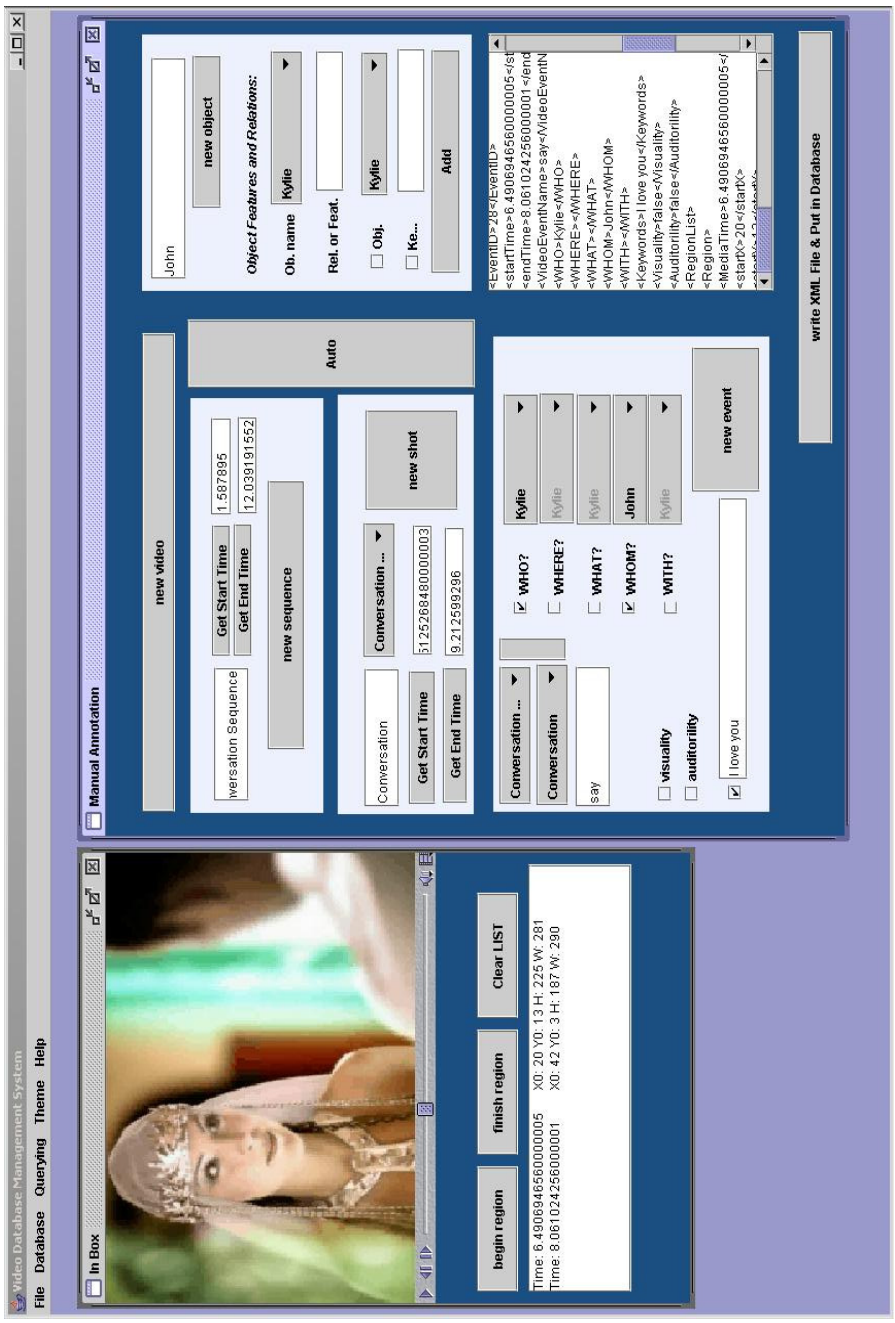


Figure B.1 - Screen Shot of the Annotation Interface

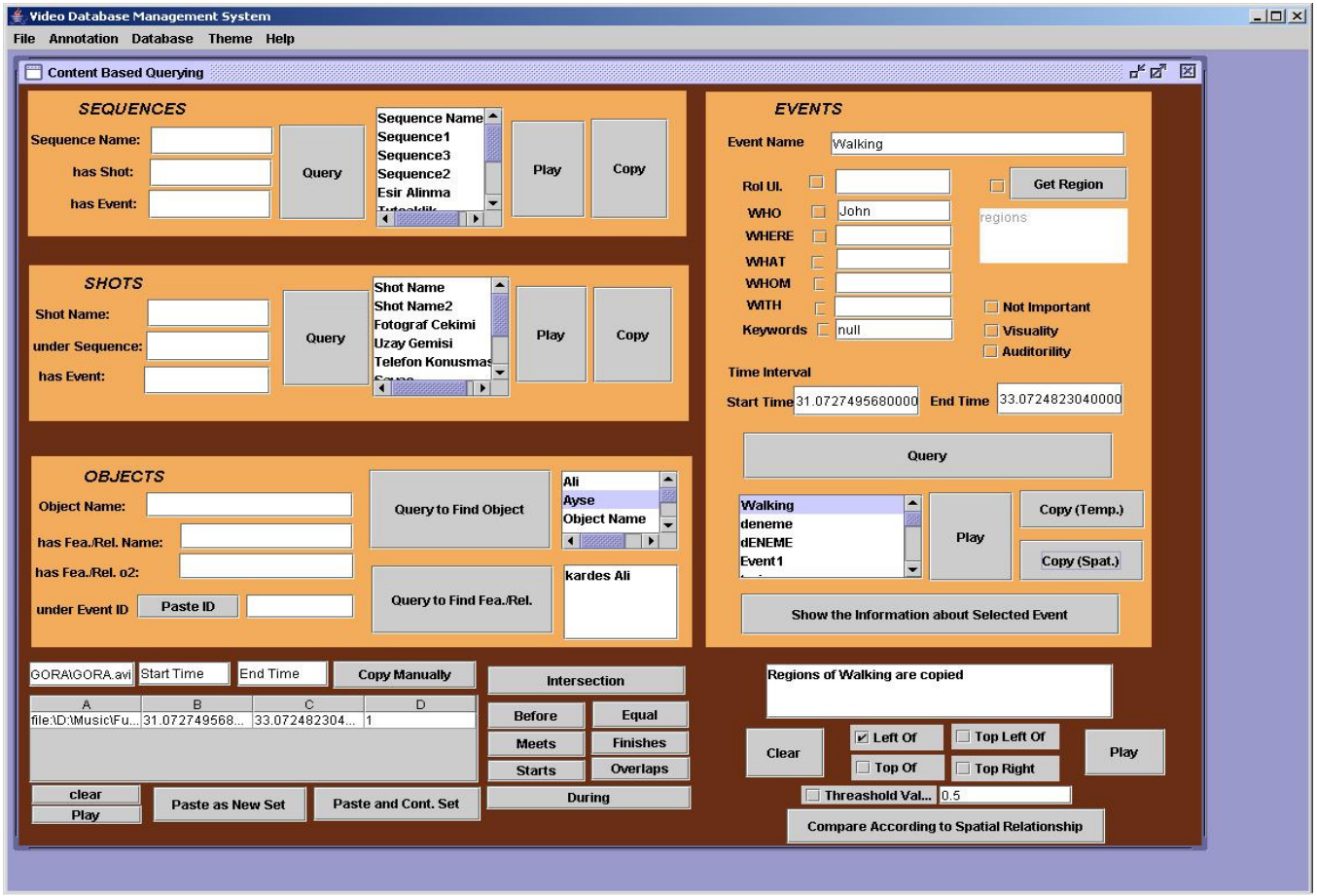


Figure B.2 - Screen Shot of the Querying Interface

C. TECHNICAL DESCRIPTION OF SEBM

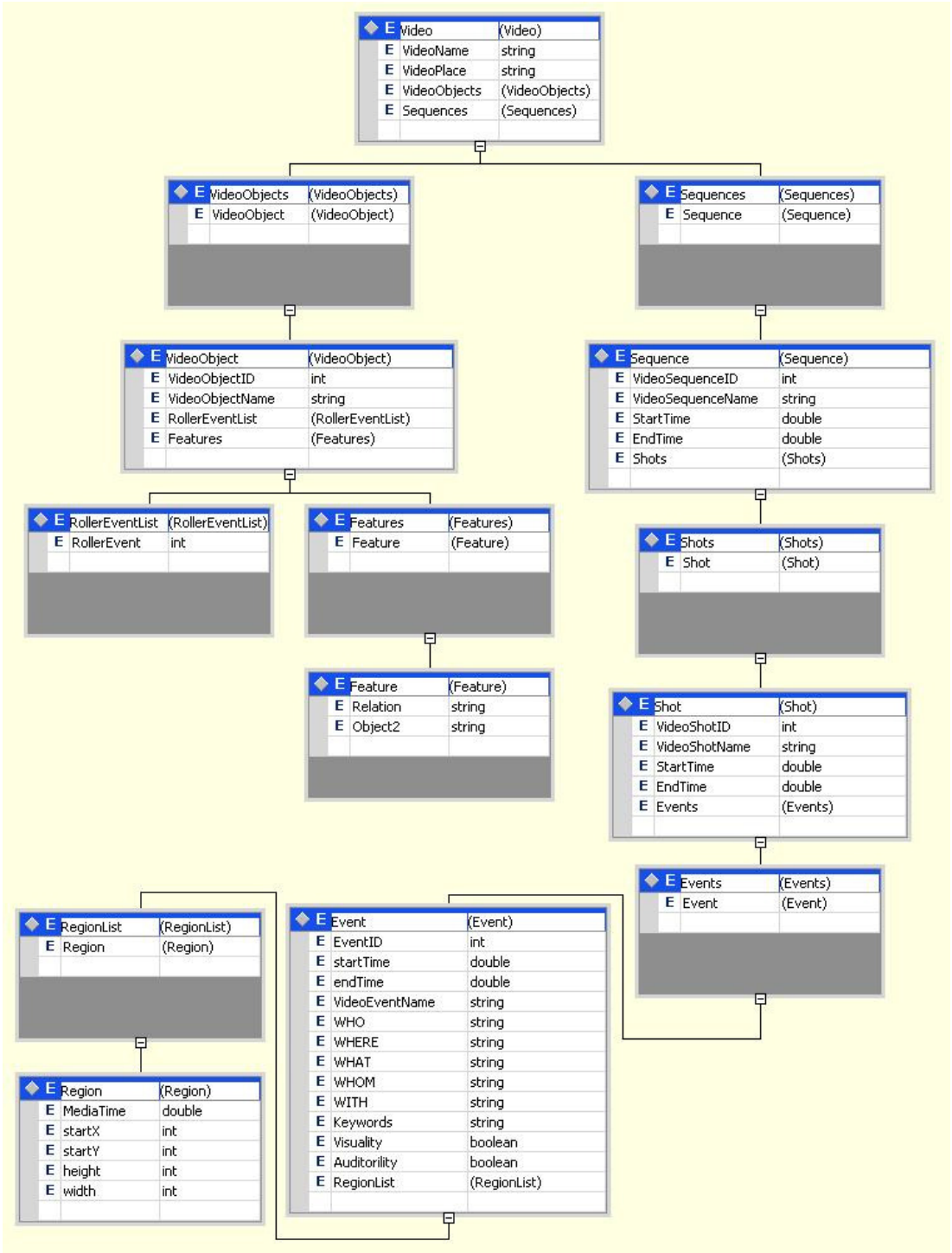


Figure C.1 – XSD Schema of SEBM Video Data Model

XSD Schema in XML format of SEBM Video Data Model is given below:

```
<?xml version="1.0" ?>
<xs:schema id="NewDataSet"
     targetNamespace="http://tempuri.org/16.avi.xsd"
     xmlns:mstns="http://tempuri.org/16.avi.xsd"
     xmlns="http://tempuri.org/16.avi.xsd"
     xmlns:xs="http://www.w3.org/2001/XMLSchema"
     xmlns:msdata="urn:schemas-microsoft-com:xml-msdata"
     attributeFormDefault="qualified"
     elementFormDefault="qualified">
     <xs:element name="Video">
      <xs:complexType>
        <xs:element name="VideoName" type="xs:string"/>
          <xs:element name="VideoPlace"
                      type="xs:string"/>
          <xs:element name="VideoObjects">
          <xs:complexType>
           <xs:sequence>
            <xs:element name="VideoObject" minOccurs="0"
                              maxOccurs="unbounded">
             <xs:complexType>
              <xs:sequence>
               <xs:element name="VideoObjectID"
                              type="xs:int"/>
               <xs:element name="VideoObjectName"
                              type="xs:string"/>
               <xs:element name="RollerEventList">
                 <xs:complexType>
                 <xs:sequence>
                  <xs:element name="RollerEvent"
                              minOccurs="0"
                              maxOccurs="unbounded"
                              type="xs:int">
                   </xs:element>
                  </xs:sequence>
                  </xs:complexType>
                </xs:element>
               <xs:element name="Features">
               <xs:complexType>
                <xs:sequence>
                 <xs:element name="Feature" minOccurs="0"
                    maxOccurs="unbounded">
                  <xs:complexType>
                   <xs:sequence>
                    <xs:element name="Relation"
                                type="xs:string"/>
                    <xs:element name="Object2"
                                type="xs:string"/>
                   </xs:sequence>
```

```
                </xs:complexType>
               </xs:element>
              </xs:sequence>
             </xs:complexType>
            </xs:element>
           </xs:sequence>
          </xs:complexType>
         </xs:element>
        </xs:sequence>
       </xs:complexType>
      </xs:element>
<xs:element name="Sequences">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="Sequence" minOccurs="0"
                          maxOccurs="unbounded">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="VideoSequenceID"
                type="xs:int"/>
      <xs:element name="VideoSequenceName"
                type="xs:string"/>
      <xs:element name="StartTime"
                type="xs:double"/>
      <xs:element name="EndTime" type="xs:double"/>
      <xs:element name="Shots">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="Shot" minOccurs="0"
                  maxOccurs="unbounded">
          <xs:complexType>
           <xs:sequence>
            <xs:element name="VideoShotID"
                    type="xs:int"/>
            <xs:element name="VideoShotName"
                    type="xs:string"/>
            <xs:element name="StartTime"
                    type="xs:double"/>
            <xs:element name="EndTime"
                    type="xs:double"/>
            <xs:element name="Events">
            <xs:complexType>
             <xs:sequence>
              <xs:element name="Event" minOccurs="0"
                        maxOccurs="unbounded">
               <xs:complexType>
                <xs:sequence>
                 <xs:element name="EventID"
                         type="xs:int"/>
                 <xs:element name="startTime"
                         type="xs:double"/>
                 <xs:element name="endTime"
                         type="xs:double"/>
                 <xs:element name="VideoEventName"
```

```
                    type ="xs:string"/>
                <xs:element name="WHO"
                    type="xs:string"/>
                <xs:element name="WHERE"
                    type="xs:string"/>
                <xs:element name="WHAT"
                    type="xs:string"/>
                <xs:element name="WHOM"
                    type="xs:string"/>
                <xs:element name="WITH"
                    type="xs:string"/>
                <xs:element name="Keywords"
                    type="xs:string"/>
                <xs:element name="Visuality"
                    type="xs:boolean"/>
                <xs:element name="Auditorility"
                    type="xs:boolean"/>
                <xs:element name="RegionList">
                 <xs:complexType>
                  <xs:sequence>
                   <xs:element name="Region">
                    <xs:complexType>
                     <xs:sequence>
                      <xs:element name="MediaTime"
                          type="xs:double"/>
                      <xs:element name="startX"
                          type="xs:int"/>
                      <xs:element name="startY"
                          type="xs:int"/>
                      <xs:element name="height"
                          type="xs:int"/>
                      <xs:element name="width"
                          type="xs:int"/>
                     </xs:sequence>
                    </xs:complexType>
                   </xs:element>
                  </xs:sequence>
                 </xs:complexType>
                </xs:element>
               </xs:sequence>
              </xs:complexType>
             </xs:element>
            </xs:sequence>
           </xs:complexType>
          </xs:element>
         </xs:sequence>
        </xs:complexType>
       </xs:element>
      </xs:sequence>
     </xs:complexType>
    </xs:element>
   </xs:sequence>
  </xs:complexType>
 </xs:element>
</xs:element>
```

```
  </xs:sequence>
 </xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
</xs:schema>
```

An example of SEBM video data model is given below in XML format.

```
<Video>
<VideoName>Birthday.avi</VideoName>
<VideoPlace>file:\D:\AVI\Birthday.avi</VideoPlace>
<VideoObjects>
     <VideoObject>
          <VideoObjectID>62</VideoObjectID>
          <VideoObjectName>John</VideoObjectName>
          <RollerEventList>
               <RollerEvent>33</RollerEvent>
               <RollerEvent>34</RollerEvent>
          </RollerEventList>
          <Features>
               <Feature>
                    <Relation>FriendOf</Relation>
                    <Object2>Jimmy</Object2>
               </Feature>
               <Feature>
                    <Relation>BrotherOf</Relation>
                    <Object2>Kylie</Object2>
               </Feature>
               <Feature>
                    <Relation>is</Relation>
                    <Object2>engineer</Object2>
               </Feature>
          </Features>
     </VideoObject>
     <VideoObject>
          <VideoObjectID>63</VideoObjectID>
          <VideoObjectName>Kylie</VideoObjectName>
          <RollerEventList>
               <RollerEvent>33</RollerEvent>
               <RollerEvent>34</RollerEvent>
          </RollerEventList>
          <Features>
               <Feature>
                    <Relation>SisterOf</Relation>
                    <Object2>John</Object2>
               </Feature>
               <Feature>
                    <Relation>has</Relation>
```

```
                    <Object2>blue eyes</Object2>
               </Feature>
          </Features>
     </VideoObject>
     <VideoObject>
          <VideoObjectID>64</VideoObjectID>
          <VideoObjectName>Jimmy</VideoObjectName>
          <RollerEventList>
          </RollerEventList>
          <Features>
               <Feature>
                    <Relation>FriendOf</Relation>
                    <Object2>John</Object2>
               </Feature>
          </Features>
     </VideoObject>
     <VideoObject>
          <VideoObjectID>65</VideoObjectID>
          <VideoObjectName>Hamburger</VideoObjectName>
          <RollerEventList>
               <RollerEvent>33</RollerEvent>
          </RollerEventList>
          <Features></Features>
     </VideoObject>
     <VideoObject>
          <VideoObjectID>66</VideoObjectID>
          <VideoObjectName>Ketchup</VideoObjectName>
          <RollerEventList>
               <RollerEvent>33</RollerEvent>
          </RollerEventList>
          <Features></Features>
     </VideoObject>
     <VideoObject>
          <VideoObjectID>67</VideoObjectID>
          <VideoObjectName>Table</VideoObjectName>
          <RollerEventList>
               <RollerEvent>33</RollerEvent>
               <RollerEvent>34</RollerEvent>
          </RollerEventList>
          <Features></Features>
     </VideoObject>
</VideoObjects>
<Sequences>
     <Sequence>
     <VideoSequenceID>25</VideoSequenceID>
     <VideoSequenceName>Party</VideoSequenceName>
     <StartTime>0.0</StartTime>
     <EndTime>31.197210624</EndTime>
     <Shots>
          <Shot>
             <VideoShotID>958</VideoShotID>
             <VideoShotName>Dinner</VideoShotName>
             <StartTime>7.6422696960000005</StartTime>
             <EndTime>19.158018048000002</EndTime>
```

```
<Events>
  <Event>
       <EventID>33</EventID>
       <startTime>9.840731136</startTime>
       <endTime>10.678240256</endTime>
       <VideoEventName>Give</VideoEventName>
       <WHO>John</WHO>
       <WHERE>Table</WHERE>
       <WHAT>Hamburger</WHAT>
       <WHOM>Kylie</WHOM>
       <WITH>Hamburger</WITH>
       <Keywords></Keywords>
       <Visuality>true</Visuality>
       <Auditorility>false</Auditorility>
       <RegionList>
         <Region>
          <MediaTime>9.840731136</MediaTime>
          <startX>73</startX>
          <startY>39</startY>
          <height>91</height>
          <width>100</width>
         </Region>
         <Region>
          <MediaTime>10.67820256</MediaTime>
          <startX>164</startX>
          <startY>209</startY>
          <height>99</height>
          <width>74</width>
         </Region>
       </RegionList>
  </Event>
  <Event>
       <EventID>34</EventID>
       <startTime>10.678240256</startTime>
       <endTime>11.20165601</endTime>
       <VideoEventName>Say</VideoEventName>
       <WHO>John</WHO>
       <WHERE>Table</WHERE>
       <WHAT></WHAT>
       <WHOM>Kylie</WHOM>
       <WITH></WITH>
       <Keywords>Bon Appetite</Keywords>
       <Visuality>true</Visuality>
       <Auditorility>true</Auditorility>
       <RegionList>
        <Region>
         <MediaTime>10.678240256</MediaTime>
         <startX>105</startX>
         <startY>49</startY>
         <height>222</height>
         <width>218</width>
        </Region>
        <Region>
         <MediaTime>11.20165601</MediaTime>
```

```
            <startX>100</startX>
            <startY>57</startY>
            <height>193</height>
            <width>217</width>
           </Region>
          </RegionList>
     </Event>
     <Event>
          <EventID>36</EventID>
          <startTime>11.829814272</startTime>
          <endTime>13.42720560</endTime>
          <VideoEventName>
               isHeard
          </VideoEventName>
          <WHO></WHO>
          <WHERE></WHERE>
          <WHAT></WHAT>
          <WHOM></WHOM>
          <WITH></WITH>
          <Keywords>Strange Noise</Keywords>
          <Visuality>false</Visuality>
          <Auditorility>true</Auditorility>
          <RegionList>
           <Region>
            <MediaTime>11.829814272</MediaTime>
            <startX>2</startX>
            <startY>6</startY>
            <height>379</height>
            <width>278</width>
           </Region>
           <Region>
            <MediaTime>13.42720560</MediaTime>
            <startX>2</startX>
            <startY>6</startY>
            <height>379</height>
            <width>278</width>
           </Region>
          </RegionList>
     </Event>
     <Event>
          <EventID>37</EventID>
          <startTime>15.179850752</startTime>
          <endTime>15.598605312</endTime>
          <VideoEventName>
               isWritten
          </VideoEventName>
          <WHO></WHO>
          <WHERE></WHERE>
          <WHAT></WHAT>
          <WHOM></WHOM>
          <WITH></WITH>
          <Keywords>Happy Birthday</Keywords>
          <Visuality>true</Visuality>
          <Auditorility>false</Auditorility>
```

```
                    <RegionList>
                     <Region>
                      <MediaTime>15.179850752</MediaTime>
                      <startX>48</startX>
                      <startY>47</startY>
                      <height>324</height>
                      <width>98</width>
                     </Region>
                     <Region>
                      <MediaTime>15.598605312</MediaTime>
                      <startX>56</startX>
                      <startY>61</startY>
                      <height>302</height>
                      <width>87</width>
                     </Region>
                    </RegionList>
                  </Event>
                </Events>
              </Shot>
     </Shots>
     </Sequence>
</Sequences>
</Video>
```