ONTOLOGY-BASED SPATIO-TEMPORAL VIDEO MANAGEMENT SYSTEM

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

BY

ATAKAN ŞİMŞEK

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

SEPTEMBER 2009

Approval of the thesis:

ONTOLOGY-BASED SPATIO-TEMPORAL VIDEO MANAGEMENT SYSTEM

submitted by **ATAKAN ŞİMŞEK** in partial fulfillment of the requirements for the degree of **Master of Science in Computer Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen _____
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Müslim Bozyiğit _____
Head of Department, **Computer Engineering**

Assoc.Prof.Dr. Nihan Kesim Çiçekli _____
Supervisor, **Computer Engineering Department, METU**

Examining Committee Members:

Assoc. Prof. Dr. Ferda Nur Alpaslan _____
Computer Engineering Dept., METU

Assoc. Prof. Dr. Nihan Kesim Çiçekli _____
Computer Engineering Dept., METU

Assoc. Prof. Dr. Ahmet Coşar _____
Computer Engineering Dept., METU

Asst. Prof. Dr. İlkay Ulusoy _____
Electrical and Electronics Engineering Dept., METU

Dr. Ayşenur Birtürk _____
Computer Engineering Dept., METU

Date: **09.09.2009**

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: ATAKAN ŞİMŞEK

Signature :

ABSTRACT

ONTOLOGY-BASED SPATIO-TEMPORAL VIDEO MANAGEMENT SYSTEM

Şimşek, Atakan
M.S, Department of Computer Engineering
Supervisor : Assoc.Prof.Dr. Nihan Kesim Çiçekli

September 2009, 91 pages

In this thesis, a system, called Ontology-Based Spatio-Temporal Video Management System (OntoVMS) is developed in order to supply a framework which can be used for semantic data modeling and querying in video files. OntoVMS supports semantic data modeling which can be divided into concept modeling, spatio-temporal relation and trajectory data modeling. The system uses Rhizomik MPEG-7 Ontology as the core ontology. Moreover ontology expression capability is extended by automatically attaching domain ontologies. OntoVMS supports querying of all spatial relations such as directional relations (north, south ...), mixed directional relations (northeast, southwest ...), distance relations (near, far), positional relations (above, below ...) and topological relations (inside, touch ...); temporal relations such as starts, equal, precedes; and trajectories of objects of interest. In order to enhance querying capability, compound queries are added to the framework so that the user can combine simple queries by using "(", ")", "AND" and "OR" operators. Finally, the use of the system is demonstrated with a semi-automatic face annotation tool.

Keywords: Spatio-Temporal Querying, MPEG-7 Ontology, Domain Ontology

ÖZ

ONTOLOJİ TABANLI UZAY-ZAMANSAL VİDEO BİLGİ YÖNETİM SİSTEMİ

Şimşek, Atakan
Yüksek Lisans, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi : Doç.Dr. Nihan Kesim Çiçekli

Eylül 2009, 91 sayfa

Bu tez kapsamında, anlamsal veriyi modelleyebilmek ve video dosyalarında sorgulama yapabilmek için "Ontoloji Tabanlı Uzay-Zamansal Video Bilgi Yönetim Sistemi (OntoVMS)" adında bir altyapı geliştirilmiştir. Bu altyapı; konsept modelleme, uzay-zamansal ilişki çıkarım ve hareket yolu verisinden oluşan anlamsal veriyi modelleme imkanı sunmaktadır. Sistem Rhizomik MPEG-7 ontolojisini temel ontoloji olarak kullanmaktadır ve Alan ontolojilerini bu temel ontolojiye otomatik bir şekilde ekleyerek, sorgulama kabiliyetini geliştirmektedir. OntoVMS; yönsel ilişkileri (kuzey, güney...), ara yön ilişkilerini (kuzeydoğu, güneybatı...), mesafe ilişkilerini (yakın, uzak), pozisyona bağlı ilişkileri (yukarıda, aşağıda...) ve topolojik ilişkileri (içinde, dokunuyor) sorgulama imkanı vermektedir. Sistem bunlara ek olarak, "aynı anda", "daha önce" gibi zamansal ilişkileri ve ilgilenilen nesnelerin hareket yollarını sorgulama kabiliyetlerini kullanıcıya sunmaktadır. Sorgulama kabiliyetini artırmak için; kullanıya "(", ")", "AND" ve "OR" operatörlerini kullanarak birleşik sorgu tanımlamasına imkan vermektedir. Son olarak, geliştirilen sistemin kullanımı yarı-otomatik yüz etiketleme sistemi ile tanıtılmaktadır.

Anahtar Kelimeler: Uzay-Zamansal Sorgulama, MPEG-7 Ontolojisi, Alan Ontolojisi

To my faithful wife

ACKNOWLEDGMENTS

I would like to express my appreciation and indebtedness to my supervisor Assoc. Prof. Dr. Nihan Kesim Çiçekli for her supportive, encouraging and constructive approach throughout my master study and her efforts during supervision of the thesis.

I would like to thank to my thesis jury members Assoc. Prof. Dr. Ferda Nur Alpaslan, Assoc. Prof. Dr. Ahmet Coşar, Asst. Prof. Dr. İlkay Ulusoy and Dr. Ayşenur Birtürk for reviewing and evaluating my thesis.

I also want to thank to Scientific and Technical Research Council of Turkey (TÜBİTAK) for its support to the project with grant number EEEAG 107E234.

I would like to thank to TÜBİTAK - BİDEB for their support with grant number 2228.

I would also like to thank my supervisors and my colleagues from TÜBİTAK UEKAE - İLTAREN for their understanding and support during my academic studies.

I am also grateful to Hilal Tarakçı and Mehmet Yılmaztürk for their valuable support during the implementation phase of this thesis.

Finally, I would like to express my special thanks to my wife and my family, for their endless patience, encouragement and support during my master study.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVATIONS

API	Application Programming Interface
GUI	Graphical User Interface
MBR	Minimum Bounding Rectangle
SPARQL	Simple Protocol and RDF Query Language
OntoVMS	Ontology-Based Spatio-Temporal Video Management System

CHAPTER 1

INTRODUCTION

1.1 Overview

In the last decade, there is an increasing demand on video usage in order to store multimedia contents. This is the result of digital technology improvements. Each day dozens of films are published by film producers and hundreds of news videos can be seen in TV channels. Moreover video sharing web sites have become very popular according to user demands. Also the acceptance of 3G contract in mobile operators gives opportunity to video transfer in mobile phones. According to these improvements, video files are used in many aspects of our daily lives. Academic environments are interested in these sharp increases in video usage, because these demands create new research areas. Semantic content representation is the most popular research area among others, because videos are used in order to store their semantic contents. For instance a personal video is important because it represents a retrospective memory. A news video is used for displaying important events.

Semantic video storage and manipulation brings some extra issues for databases and video management tools, because all videos can be represented in different standards and there is no common method to reach the semantic data in a specific video. In order to solve this data storage problem, Moving Picture Expert Group developed a standard, namely MPEG-7, to describe multimedia contents [20].

MPEG-7 standard is improved for supplying descriptors and tools about representing multimedia data contents. Its descriptors support image, video, audio and audiovisual multimedia types [17]. In spite of the fact that, this standard is developed for gathering different video representation formats in a one standard representation, its XML based definition structure

prevents semantic interoperability between different multimedia management systems [28]. Mainly MPEG-7 standard has developed with two handicaps: it does not support reasoning capability and in MPEG-7 standardization the same semantic event can be represented in different ways. In order to reach a standard representation and supply reasoning capability, there are a lot of attempts to move MPEG-7 to Semantic Web [13].

The most popular MPEG-7 Ontologies can be listed as; Jane Hunter Ontology [16], Tsinaraki Ontology [30], Rhizomik Ontology [13] and COMM Ontology [29]. In this thesis, MPEG-7 Rhizomik ontology is used as the core ontology and Tsinaraki's approach is used as automatic ontology combination methodology, because this ontology and this methodology support automatic data transformation and domain ontology attachment capabilities. In this way different domain ontologies can be attached to MPEG-7 ontology in order to enhance system capabilities.

A video is a sequence of images, called frames. The content of an image can be described by the actors and objects appearing in the image. Actors and objects are parts of events. For instance a player and a goal keeper are the actors, and the ball is the object of the goal event. Therefore it can be concluded that events are described according to the relations between objects in video files. These object relations can be defined according to objects' positions, spatio- temporal relations or these relations can be defined by their appearance order in a video file, i.e. temporal relations. To summarize, a semantic event is described as the interaction of media objects in space and time [32].

Spatio-temporal semantic information can be used in a variety of fields. For instance, it can be used in surveillance, health and medical systems [26]. In medical area spatio-temporal image and video retrieval can be used for tumor recognition. In surveillance videos spatio-temporal querying can be used to understand the reasons of a car crash. In a football match it can be used to find offside positions.

In order to propose a solution to semantic data retrieval problem in video systems we develop a framework; **Onto**logy-Based Spatio-temporal **V**ideo **M**anagement **S**ystem (OntoVMS). OntoVMS is developed in order to supply a framework which can be used for data modeling and querying in video files. OntoVMS supports semantic data modeling which can be divided into concept modeling, spatio-temporal relation and trajectory data modeling. In video systems spatial relations are composed of directional relations (north, south ...), mixed directional

relations (northeast, southwest ...), distance relations (near, far), positional relations (above, below ...) and topological relations (inside, touch ...) [26]. Temporal relations can be defined by Allen's temporal algebra relations (starts, equal, precedes ...) [32]. OntoVMS supports all of these relation types; therefore the user can pose positional and temporal queries in this system. To enhance querying capability, compound queries are added to our framework. The user can combine simple queries by using "(", ")" ,"AND" and "OR" operators in order to construct a compound query. OntoVMS uses Dijakstra Shunting algorithm [27] to divide and interpret user compound queries.

Another important capability of OntoVMS is modeling the trajectory data. With the help of this, framework users can watch the motion path of the interested objects in a video. This capability is commonly used in surveillance videos. For instance the motion path of a thief is a very important data in a burglary in order to improve the security policy.

The main difference between OntoVMS and other spatio-temporal retrieval systems is its ontology support. OntoVMS uses both MPEG-7 Rhizomik ontology and domain ontologies for modeling and querying the video semantic content, including spatio-temporal and trajectory data. OntoVMS query engine is connected to the JENA ontology API in order to get ontological data from domain ontology. SPARQL queries or JENA specific queries are prepared to get the necessary ontological data. This feature supplies detailed querying capability to the users, by which the user can make ontological querying. For instance it is possible that the user is interested in carnivore animals only in a documentary. Or, the user may be interested in only the positions of home team players when a goal event occurs in a football video.

OntoVMS is developed as a web application in order to serve a public usage. On account of all types of user profiles, our framework serves as a user friendly web interface. All types of client profiles can easily use annotation and query interface because web applets consist of combo boxes. Another property of OntoVMS is the optimization of time and space sources. It extracts spatio-temporal relations during annotation of videos in order to decrease query response time [11]. On the other hand, the storage of spatio-temporal relations is optimized by using a knowledge-base structure [6]. In this structure, only basic relations are stored in annotation part. In querying part if the user is interested in non-stored relations, inference rules are used in order to extract necessary relations [9]. It means system uses reasoning process in order to answer user queries. In the knowledge-base JPL prolog API is used,

therefore user queries are transformed into SWI-Prolog codes. Prolog result set is transformed into user understandable query results.

Lastly, OntoVMS is the main framework for an ontological multimedia management system therefore some parts of this framework will improve in the following years. The infrastructure is designed and implemented in a flexible way so that it can support full automatic or semi-automatic annotation when it is available. The current version of the OntoVMS uses manual annotation except face tracking. In face tracking a semi-automatic tool is integrated and this capability is illustrated in the case study part of this thesis.

1.2 Contribution of the Thesis

This thesis is conducted as a part of a research project partially supported by TUBITAK (TUBITAK-EEEAG 107E234). According to the project, a framework will be developed to support the management of personal videos. The development of a personal multimedia information management system is the core part of the project. This system should provide the user with capabilities for semantic content modeling and querying of multimedia systems. This thesis work is one of the main infrastructures of this project.

In order to develop this system some studies were already done. An ontological semantic querying system was proposed in previous year [28]. That was an initial system which mainly dealt with the combination of MPEG-7 ontology and domain ontologies. Therefore, it has a very limited querying capability. It can be used for concept querying in videos only. According to project specifications, in order to construct a full-functional personal multimedia management system, some extra capabilities must be added.

After a literature survey it is decided that spatio-temporal and trajectory querying capabilities should be added into the project because these querying capabilities meet nearly all user expectations from a video management system. In this thesis, pure spatio-temporal querying capability is enhanced with the use of domain ontologies. Thus we achieved a framework which can use domain ontology knowledge in semantic and spatio-temporal querying. Moreover in order to optimize system space and time requirements a knowledge-base structure is used [6].

The contributions of this thesis can be listed as follows:

- The main contribution of this thesis is adding spatio-temporal and trajectory data modeling and querying capabilities into a MPEG-7 ontology based multimedia retrieval system. In this way, spatio-temporal querying capability can be used with MPEG-7 ontologies.
- The proposed solution can use any type of domain ontology, because the system is developed as a generic system. It means spatio-temporal querying capability can be used with any type of domain ontology.
- Web Ontology Language (OWL) and MPEG-7 Rhizomik ontology is used for storing spatio-temporal object storage, therefore interoperability problem is overcome.
- In this thesis, knowledge-base spatio-temporal annotation and querying is used. Thus knowledge-base structure is used with domain and MPEG-7 ontologies. Namely, Jena ontology API and Prolog API are used at the same time.
- In this thesis work, trajectory data modeling and ontological querying concepts are used together, because OntoVMS supports ontological trajectory querying capability.
- The proposed solution provides compound ontological spatio-temporal querying capability through which domain independent compound querying feature is supported.

1.3 Organization

The rest of the thesis is organized as follows:

- Chapter 2 gives brief information about the main issues, such as MPEG-7 and ontologies, which are used in this thesis.
- Chapter 3 investigates the systems which are related to our work. The main differences between these systems and our work are discussed.
- Chapter 4 explains the details of the proposed framework for Spatio-Temporal and Trajectory data modeling. The features of the knowledge-base system and the annotation part of the system are presented.

- Chapter 5 gives the details of OntoVMS querying phase. Ontology based querying is explained. Some demonstrative queries are processed and lastly OntoVMS user interfaces are explained.
- Chapter 6 presents a case-study about the usage of face-tracking module and OntoVMS.
- Chapter 7 concludes the thesis and discusses some possible future extensions.

CHAPTER 2

BACKGROUND

Due to the recent developments in video technology, lots of different video description formats are developed. As a result of this situation all formats use their own way to represent semantic data. Therefore the same data can be represented in a dozen of different ways. This situation prevents semantic interoperability between different multimedia management systems. To solve this problem MPEG-7 standard is proposed. In this chapter MPEG-7 standard is explained with its handicaps. Then MPEG-7 Ontology proposals are summarized.

2.1 MPEG-7

MPEG-7 is developed by MPEG (Moving Picture Expert Group), as an ISO/IEC standard in order to describe multimedia contents [20]. MPEG-1, MPEG-2 and MPEG-4 standards are used to represent the actual data of the multimedia contents [22]; MPEG-7 standard differs from these standards since it represents the information about the semantic contents.

The goal of MPEG-7 standardization is to supply a rich set of tools to describe multimedia contents. In this way the standard supplies interoperability and data exchange between applications that deal with multimedia contents. MPEG-7 standard consists of mainly the following elements [20]:

- **Descriptors (D)** : Elements that can be used for defining syntax and semantics of the features.
- **Description Schemes (DS)** : Describe the structure and semantics of the relationships between components, i.e. Descriptors and Description Schemes

- **Description Definition Language (DDL)** : It is a language that defines the syntax of the MPEG-7 Description Tools.

MPEG-7 descriptors support image, video, audio and audiovisual multimedia types (Figure 2.1 [17]). Due to the complexity of the standard there are several ways to represent the same multimedia content using different syntax. For that reason data exchange between different multimedia systems becomes useless, because the same data can be described differently in each system. For example a video which includes an event "Zinedine Zidane scoring against England" can be represented at least in three different ways [21]:

1. **Using Free Text Annotation:**

   <FreeTextAnnotation xml:lang="en"> Zinedine Zidane scoring against England.< /FreeTextAnnotation>

2. **Using the keyword annotation:**

   < KeywordAnnotation xml:lang="en">
   <Keyword>Zinedine< /Keyword>
   <Keyword>Zidan< /Keyword>
   <Keyword>scoring< /Keyword>
   <Keyword>England< /Keyword>
   <Keyword>goal< /Keyword>
   < /KeywordAnnotation>

3. **Using a structured annotation with labels:**

   < StructuredAnnotation>
   < Who>
   <Name xml:lang="en"> Zinedine Zidane < /Name>
   < / Who>
   < WhatAction>
   <Name xml:lang="en"> Zinedine Zidane scoring against England. < /Name>
   < / WhatAction>
   < /StructuredAnnotation>

Thus, MPEG-7 does not have formal semantics and it is inadequate to support formal reasoning. Therefore implicit data cannot be obtained in MPEG-7 standard. In order to solve interoperability problem, there are several attempts to move MPEG-7 to Semantic Web [13].

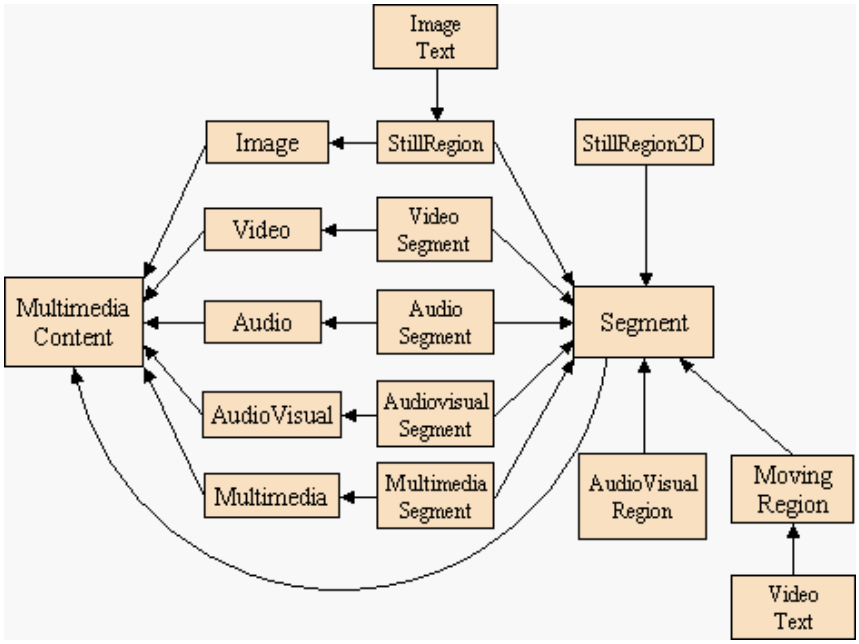


Figure 2.1: MPEG-7 Multimedia Hierarchy

2.2 MPEG-7 Ontology

MPEG-7 definition flexibility allows describing any multimedia segment, with any level of granularity [29] using any level of abstraction. It causes an ambiguity, MPEG-7 XML Schemas define 1182 elements, 417 attributes, and 377 complex types that make it very hard to manage and provide interoperability [29]. In order to enhance interoperability, different MPEG-7 ontologies are developed. Moreover MPEG-7 definitions can be combined with other domain ontologies by the help of these MPEG-7 ontologies. The proposed MPEG-7 ontologies are described below [21] and also summarized in Table 2.1 [29]:

Table 2.1: Summary of MPEG-7 Based Multimedia Ontologies

	Hunter	**DS-MIRF**	**Rhizomik**	**COMM**
Foundations	ABC	none	none	DOLCE
Complexity	OWL-Full	OWL-DL	OWL-DL	OWL-DL
Coverage	MDS+Visual	MDS+CS	All	MDS+Visual
Applications	Digital Libraries, e-Research	Digital Libraries, e-Learning	Digital Rights Management, e-Business	Multimedia Analysis and Annotations

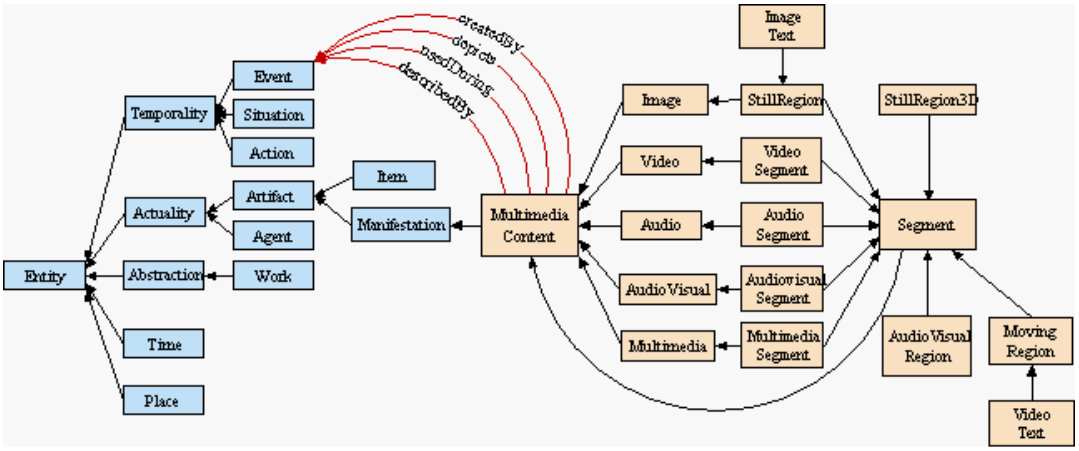


Figure 2.2: Extending ABC Ontology with MPEG-7

1. **Jane Hunter Ontology:**

   Hunter believes that there is a need to represent semantics of MPEG-7 descriptors, in a machine understandable way in order to supply interoperability [16]. XML Schema supplies little support in this issue, and it should be improved; therefore Resource Description Framework (RDF) is chosen. However RDF has some limitations, it is illegal in RDF to specify multiple range constraints on a single property. Therefore, Hunter extends descriptions by using DARPA Agent Markup Language (DAML+OIL) [16].

   Hunter wants to create a super ontology which can be merged with RDF schemas from other domains in order to provide a data exchange platform, called MetaNet ontology [16]. ABC ontology is used as the core ontology in order to combine MPEG-7 and domain ontologies [21]. *ABC ontology: Manifestation class* supplies an attachment point to *MPEG-7: Multimedia content class* (Figure 2.2 [17]).

2. **Tsinaraki (DS-MIRF) Ontology:**

   In 2004, Tsinaraki et al. proposed the DS-MIRF ontology, which fully covers OWL-DL semantics of the MPEG-7 MDS [29]. The ontology is integrated with OWL ontologies for the domain of football and Formula 1 to demonstrate how domain knowledge can be systematically combined with the general structure of MPEG-7 [29]. Tsinaraki proposed a methodology that allows transformation from MPEG-7 XML Schema to ontology [30]. Tsinaraki's approach has led to the development of a transformation model which captures the semantics of any XML schema into OWL DL ontology.

3. **Rhizomik Ontology:**

   In 2005, Garcia and Celma proposed a new approach to transform MPEG-7 schema to OWL. Their approach provides a complete and automatic mapping framework to translate MPEG-7 to standard OWL [13]. It is a generic solution for XML Schema to OWL mapping.

   In Rhizomik ontology there are 2372 classes and 975 properties and other domain ontologies can be linked into this ontology's element. Thus, they have generated an upper-ontology for multimedia data where other domain ontologies can be added [13]. The Rhizomik ontology originally was expressed in OWL Full, because 23 properties have both data type and object type ranges [29]. This problem has been solved by creating two different properties *owl:DatatypeProperty* and *owl:ObjectProperty*, and OWL DL version has been produced.

   Main contribution of Rhizomik study is proposing an automatic transformation from XML Schema into Semantic Web. This transformation can be detailed in two steps [13].

   (a) XSD2OWL Mapping:

   This step is responsible for capturing implicit schema semantics. After automatic OWL transformation, name collusions should be resolved manually because OWL uses unique name domains but XML has independent name domains. Transformation details can be seen in Table 2.2 [13].

   (b) XML2RDF Mapping:

   This is XML to RDF mapping step and this is used for instantiating the OWL ontologies by using XML metadata [13]. Structure-mapping approach is selected because the aim is to produce RDF metadata as transparently as possible.

4. **COMM Ontology:**

   In 2007, Arndt et al. proposed a solution COMM (the Core Ontology of Multimedia for annotation) [29]. This ontology is designed manually in order to support MPEG-7 intended semantics using re-engineering. It is OWL DL ontology and consistency has been validated using FACT++ [29].

Table 2.2: Rhizomik XSD2OWL Translation Constructs

XML Schema	OWL	Shared informal semantics
element — attribute	rdf:Property owl:DatatypeProperty owl:ObjectProperty	Named relation between nodes or nodes and values
element @substitutionGroup	rdfs:subPropertyOf	Relation can appear in place of a more general one
element@type	rdfs:range	The relation range kind
complexType|group |attributeGroup	owl:Class	Relations and contextual restrictions package
complexType//element	owl:Restriction	Contextualised restriction of a relation
extension@base | restriction@base	rdfs:subClassOf	Package concretises the base package
@maxOccurs | @minOccurs	owl:maxCardinality owl:minCardinality	Restrict the number of occurrences of a relation
sequence | choice	owl:intersectionOf owl:unionOf	Combination of relations in a context

Among all of these MPEG-7 ontologies, Rhizomik MPEG-7 ontology is selected as the core ontology in this thesis work, because of its expressing capability.

CHAPTER 3

RELATED WORK

There has been considerable amount of work on semantic data modeling and data representation for multimedia systems. This chapter presents some of these studies that are most relevant to the work in this thesis. We compare the proposed framework with the existing works with respect to semantic querying capabilities and spatio-temporal issues.

3.1 Extended-AVIS

In [18], a video model is proposed for spatio-temporal querying of objects in video databases. It is also possible to query trajectories of objects in this model. The proposed model was an extension to the model proposed in [1], namely AVIS. AVIS is a flexible video model since it can be used to model any video data. In [18] some extra features were added to AVIS in order to provide ability to model both semantic content and spatial properties of objects; query spatio-temporal relationships between objects; query moving objects in a video (trajectory querying).

In AVIS, information about entities and objects are stored as a special index called association maps. Using association maps, a frame-segment tree is constructed and in querying part this frame-segment tree is used (Figure 3.1 [18]).

In Extended-Avis region information is added to the association maps and the frame segment tree, in this way spatio-temporal querying ability is added.

The main differences between this thesis and Extended-AVIS can be summarized as follows:

- Extended AVIS do not use any ontological metadata. However, it is possible to make

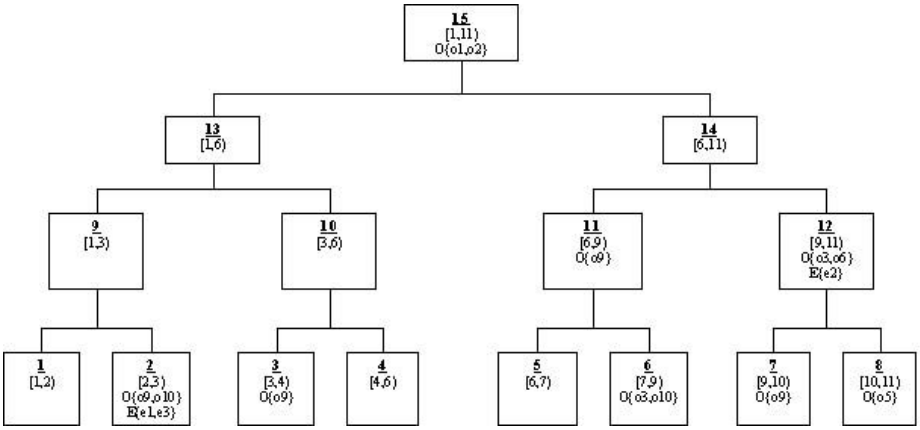


Figure 3.1: AVIS Frame Segment Tree

ontological querying in this work.

- This thesis uses MPEG-7 metadata in its querying part.
- Extended AVIS do not support compound queries. Such querying abilities are provided in this thesis.

3.2 An Ontology-Based Multimedia Information Management System

An ontology based information management system [28] is developed for proposing a solution to domain independent data annotation problem in multimedia systems. MPEG-7 ontology is used as the core model because MPEG-7 has some interoperability problems. According to this solution, Rhizomik MPEG-7 ontology can be used as the core model and any other domain ontologies can be attached to this ontology. In this way, it proposes a domain independent generic solution for multimedia annotation systems. The users can annotate their videos and pose necessary queries using different domain ontologies.

It is a web application which consisting of a client and a server side. The most important feature of the system is its user friendly client side which supplies an automatic engine to import domain ontology into the MPEG-7 Rhizomik ontology. After attaching the domain ontology, the user can make semantic annotation on the loaded videos.

The work mentioned above is chosen as the basis for this thesis. In this work we have added new features to the system and extend its abilities with the following issues:

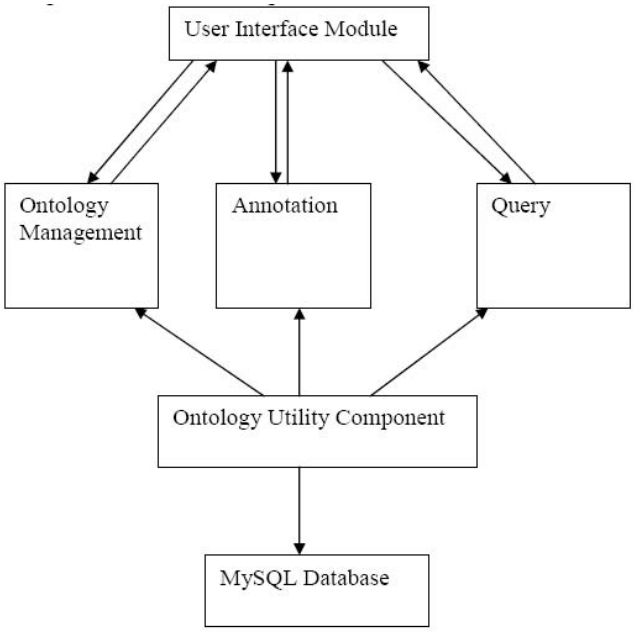


Figure 3.2: Main Architecture of the Ontology Based Multimedia System

- Spatio - Temporal querying
- Trajectory querying capabilities
- Compound querying capabilities
- Ontology based data retrieval capabilities.

3.3 BilVideo

BilVideo [11, 7, 31, 9] is a video database management system which provides spatio-temporal querying capabilities on video files. Spatio-Temporal querying includes directional, topological, 3D relational and trajectory projection query types.

As it can be seen in Figure 3.3 [11] that, the system mainly consists of knowledge - base structure. In this structure, Fact Extractor module is used for extracting facts in a semi-automatic way. After extracting all facts in a video file, some facts are pruned by the help of the spatial relations inference mechanism (this algorithm is used in this thesis too, thus the details of the algorithm are explained in the following sections). With the use of the inference mechanism, only necessary facts are stored in the database. This algorithm is used

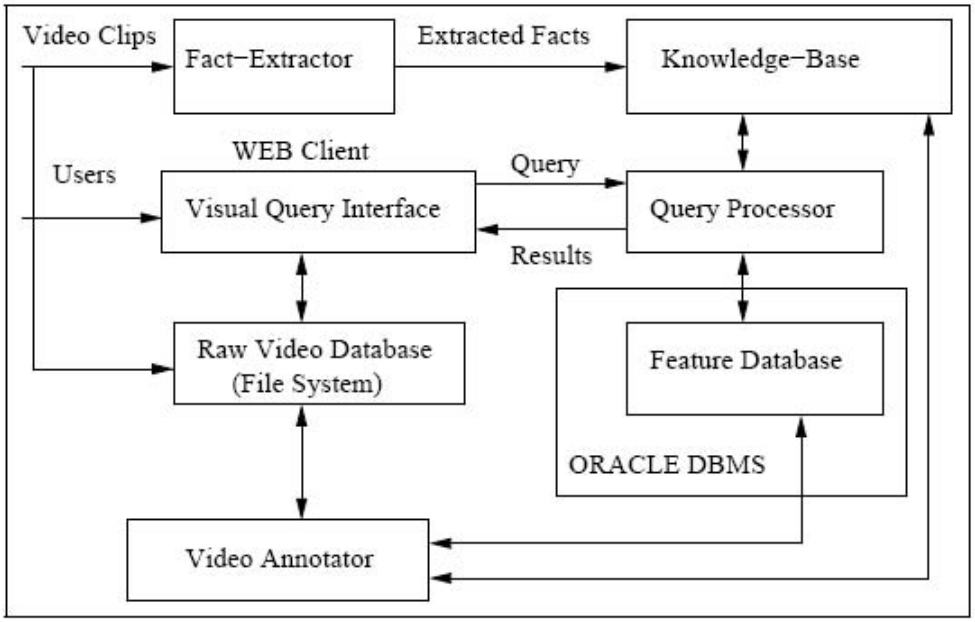


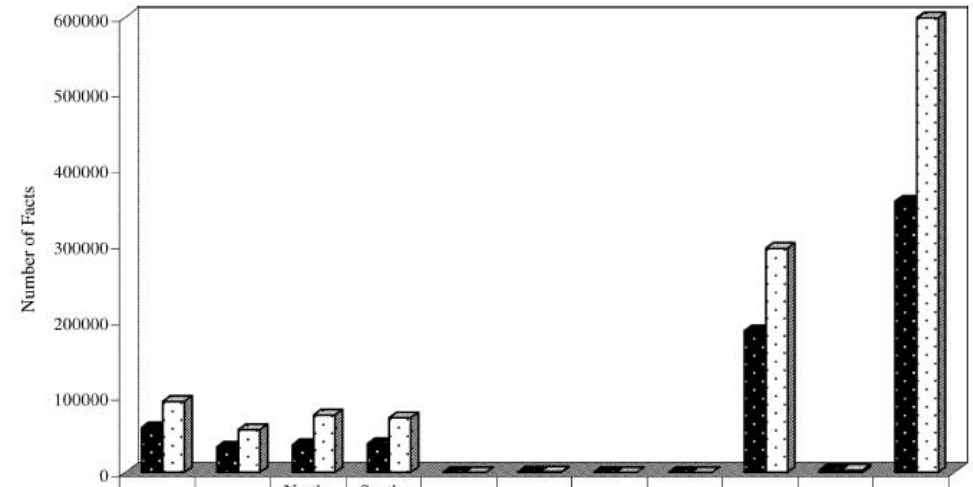
Figure 3.3: Main Architecture of the BilVideo System

for optimizing the system space as it can be seen in Figure 3.4 [9]. In this Figure there are two fact bases, one of them is constructed after pruning some relations; the other is constructed to hold all relations of the video data. Fact-base 1 answers all queries by the help of storing 357129 relations on the other hand Fact-base 2 stores 599400 facts in order to answer user queries. There is nearly 45 percent space efficiency. After the pruning process Kowledge - Base stores only the necessary facts.

In querying phase, reverse inference is used for extracting pruning facts and all facts are used to answer user queries.

This thesis differs from BilVideo in the following way:

- In new versions of BilVideo they will use MPEG-7 standard. However MPEG-7 standard has interoperability problems as mentioned before. On the other hand, MPEG-7 Rhizomik ontology is used instead of pure MPEG-7 standard in this thesis to deal with interoperability.
- In this thesis any domain ontology can be attached into MPEG-7 ontology, therefore MPEG-7 capability is extended using domain ontologies.
- Ontology based reasoning capability is another feature of this thesis. The user can make



	West	South	North-west	South-west	Equal	Cover	Inside	Touch	Disjoint	Overlap	All
■ Facts-base 1	58765	33528	36022	37987	0	592	274	188	186538	3235	357129
□ Facts-base 2	93620	56442	75875	71763	0	596	274	188	295392	3250	599400

Figure 3.4: Space Efficiency of Inferencing

domain specific querying but BilVideo makes only spatial or temporal inference.

- One important deficiency of my thesis according to BilVideo is semi-automatic fact extractor module. Their fact extractor module extracts all objects by using semi automatic annotation; on the other hand in our solution, manual annotation is used except human face extraction. In our solution face tracker module is integrated for semi-automatic face annotation.

- BilVideo supports querying 3D and 2D spatial relations, but in this thesis 3D relations are not supported. This ability can be added in future works.

3.4 Spatio-Temporal Query Formalism

Wattamwar and Ghosh [32] propose a solution to formulate a schema to formally specify arbitrary spatio-temporal relations between objects in multimedia databases. Allen's algebra and Papadias (Figure 3.5 [24]) binary string encoding schemas are used in the formalism. According to Papadias string encoding schema, temporal relations between two events can be defined using binary 5-tuples ($R_{tuvwx} : t, u, v, w, x \in \{0, 1\}$) where 0 indicates empty intersection and 1 indicates nonempty. According to these schemas, the 13 relations of Allen's

algebra are defined in their work. For instance ***precedes***: R_{10000} , ***meets*** : R_{11000} , ***overlap*** : R_{11100} ,***finished by*** : R_{11110} and so on [32].

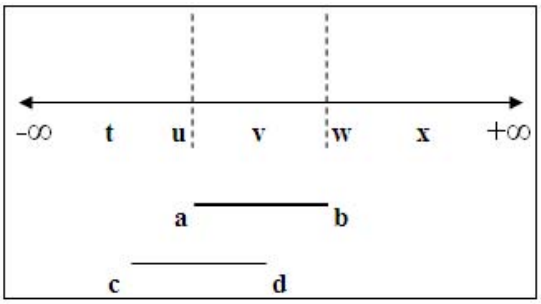


Figure 3.5: Five Regions of Interest

Moreover fuzziness and 3D formalisms are added to this work to enhance query capability and nine region of interest schema is used. The differences between their work and this thesis are:

- Nine region of interest is used in their work, but a rule based modeling is used in this thesis.
- They use MPEG-7 standard in multimedia querying whereas MPEG-7 ontology is used in this thesis.
- Fuzziness and 3D relations are added to their work. This can be added in future works to this thesis.

3.5 Thematic, Spatial and Temporal Queries over Semantic Web Data

Perry et al. [25] proposed a framework for analyzing RDF metadata model for thematic spatial and temporal relationships between named entities. By using this framework, they analyze complex thematic relationships between different people, places and events. National security domain is used to demonstrate their works. To define basic classes and relationships, an upper level ontology is used and domain ontology is attached to this upper level ontology. Deeper domain ontologies are integrated using *rdfs:subClassOf* and *rdfs:subPropertyOf* statements [25].

Framework supports *touch, overlap, equal, inside, covered by, contains, covers, any interact, within distance* spatial relationships and *before, after, during, overlap, during inv, overlap inv, any interact* temporal relationships.

Main differences between this framework and this thesis are:

- Working in different domains; this framework is used in GIS domain but this thesis considers multimedia systems.
- In this framework ontological reasoning is used but in this thesis both ontological reasoning and spatio-temporal reasoning is used.
- This framework uses standard upper domain ontology and integrates domain ontology to this upper ontology. In this thesis upper domain ontology is selected as MPEG-7 Rhizomik ontology and any domain ontology can be attached to this ontology.

3.6 BilVMS

BilVMS [12] is an MPEG-7 based multimedia management system, which can analyze video and audio files. This system can divide video into frames due to its semantic contents and these frames can be grouped according to their semantics. Another important feature of the system is video summarization by analyzing audio information and image semantics together to get better results. Moreover, skin color filtering and automatic face recognition capabilities increase its indexing ability. Main differences between BilVMS and this thesis are:

- BilVMS uses MPEG-7 standard but my dissertation uses MPEG-7 Rhizomik ontology.
- BilVMS doesn't support spatio-temporal querying or domain ontology; on the other hand it can analyze audio files.

3.7 Ontology-Supported Video Database Model

Ontology-Supported Video Database Model (OVDAM) is proposed in order to serve an automatic semantic content extraction from videos [33]. This model provides automatic object, event and concept extraction [34]. N-Cut image segmentation algorithm and the genetic

algorithm-based classifiers are used to determine segments in video keyframes and to make classification of segments to objects [34]. In this ontological model, various fuzzy spatio-temporal relations are defined between ontological classes [33]. Moreover rules are used with domain ontologies in order to decrease spatial computation cost and to define complex situations. This solution is implemented as a generic system therefore it can be used in any domain [33].

In this solution RDF is used to represent metadata and OWL is used to represent ontology [34]. MPEG-7 descriptors are used to support exchanging information between other MPEG-7 compliant applications. Moreover this solution supports spatial relations between objects and temporal relations between events [33].

The main differences between OVDAM and our solution are:

- OVDAM is developed by using MPEG-7 descriptors on the other hand our solution used MPEG-7 Rhizomik ontology in order to enhance semantic interoperability problem.
- In OVDAM, objects and events are extracted semi-automatically on the other hand in our solution only human faces are extracted semi-automatically by using face tracker module.
- OVDAM uses fuzzy modeling and querying capability but our system does not support this feature. It can be added as a future work.
- Our solution uses knowledge-base structure and spatio-temporal inferencing in order to decrease system storage necessity and increase system query performance.

CHAPTER 4

ONTOVMS SPATIO-TEMPORAL INFORMATION MODELING

Video entities mainly consist of actors, events and objects. To generalize, actors can be thought as specific object types; therefore video data management problem can be reduced into modeling of relationships between events and objects.

Video processing mainly starts with separating video file into shots (frames). After this step, spatial relationships between objects are modeled in each frame [26]. Lastly, video frames are combined and temporal relationships are calculated. In real world, relationships between objects are important especially when events are occurring. For example in surveillance videos, positions and relationships between objects are very important when a crime is happening. Therefore querying capability of spatio-temporal data modeling can be improved by using semantic data.

In this thesis, a system, called Ontology-Based Spatio-Temporal Video Management System (OntoVMS) is developed, which uses MPEG-7 ontologies and can manipulate video data by extending spatio-temporal relations with domain ontologies.

In section 4.1 the general architecture of the system is explained. The integration of MPEG-7 and domain ontologies is explained in section 4.2. Section 4.3 explains the supported spatial and temporal relations. Section 4.4 discusses why knowledge-base structure is selected as the proposed solution. And finally trajectory data modeling is examined in section 4.5.

4.1 General Properties of OntoVMS

OntoVMS is a video data management system, which uses Rhizomik MPEG-7 Ontology as the core ontology. Moreover ontology expression capability is extended by automatically attaching domain ontologies. Details of the automatic attachment procedure can be found in [28]. OntoVMS can support domain specific spatio-temporal querying capability. OntoVMS uses Allen's temporal algebra and spatial logic in the formulation of spatio-temporal relations. This thesis provides a generic solution therefore it can be used with any domain ontology and in any field type of video. OntoVMS supports:

- **Spatio-Temporal Querying:** Strict Directional Relations, Mixed Directional Relations, Topological Relations, Distance Relations, Positional Relations, Appearance Relations and Temporal Relations.
- **Trajectory Querying:** Directional Trajection, Source Destination Trajection, Specific Object Trajection, and Ontological Object Trajection.
- **Semantic Querying:** Concept querying.
- **Compound Querying:** Combination of the query types.

4.1.1 System Specifications

OntoVMS is a standard web application and Eclipse Europe is used in system development. System is composed from two parts:

- Server Side:
  - Java programming language
  - Jena Ontology API
  - SPARQL query language
  - Rhizomik MPEG-7 ontology
  - JMF (for multimedia player applet)
  - SWI-Prolog

  - JPL API (for knowledge base query)

- Client Side:
  - Facelets
  - Richfaces and Myfaces components
- Apache Tomcat as web server
- Mozilla Firefox as web browser
- MySQL database

4.1.2 System Architecture

OntoVMS is implemented according to client-server architecture. In this architecture web client can add new individuals to the system and the user can make spatio-temporal and trajectory querying by using web browsers. Moreover, in order to supply graphical components to the user, java applets are used. The general architecture of the system is presented in Figure 4.1. Main modules of the system are:

- MySQL database: This database is completely used by Jena Ontology API, in order to store all ontological data. Besides MPEG-7 and domain ontology, annotated semantic concepts and objects' spatial positions are stored here, because MySQL is the main database of the system and save/load operations are handled by using this database.
- Jena Ontology API : This API is used for managing all ontological procedures. It is the public interface of the MySQL databases; therefore all database operations use this component.
- System Initializer: This module is used for initializing the system to provide annotation and query interfaces. In initializing procedure MPEG-7 ontology is loaded to the database and the system requests domain ontology from the user. The selected domain ontology is attached to the MPEG-7 ontology by using Jena API.
- Semantic Annotator: This module is used for making semantic annotations. Details of the module can be found in [28].

- JPL Prolog API: This is the interface module of the Knowledge-Base storage. After extracting necessary facts, they are stored in the Knowledge-Base. When the query engine executes a query, the answer of this query is returned by using SWI-Prolog in JPL prolog API.

- Spatio-Temporal Annotator and Query Engine modules are explained at Sections 4.4.4 and 5.1 in detailed.

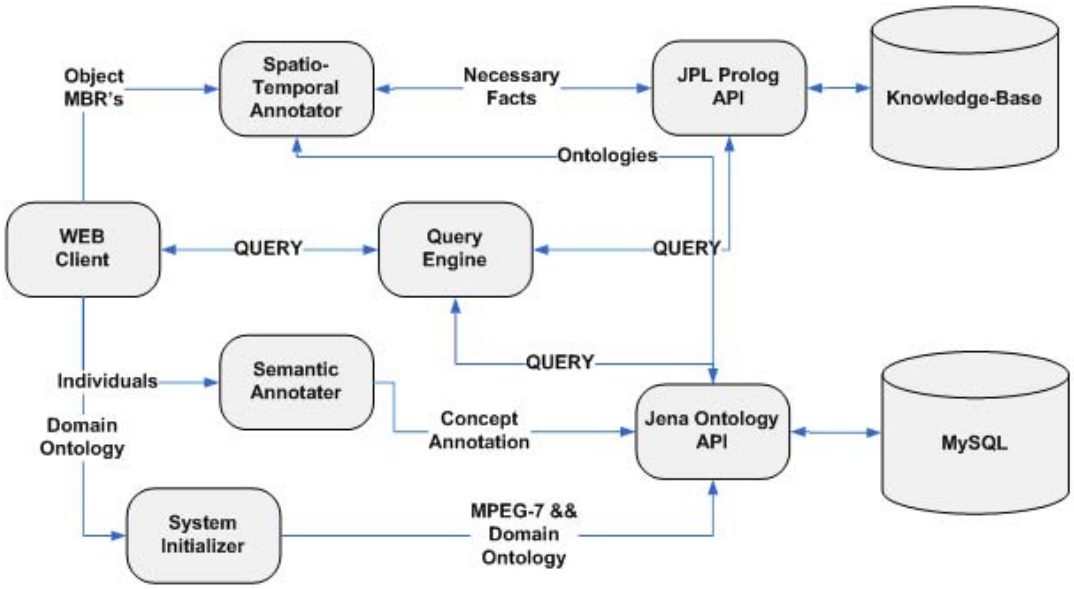


Figure 4.1: System Architecture

4.2 Ontology Based Annotation

In OntoVMS, all semantic and spatio-temporal annotations are performed with the use of domain ontologies. In other words, annotator modules are aware of the attached domain ontology, therefore annotation concepts are stored as individuals of the related domain class. For example if the user wants to define the position of a football player in a soccer video, the spatio-temporal annotator module looks for all defined individuals from Jena ontology API. Due to ontology awareness, if the user wants to define a new object in the video, the user should define this new individual. After defining it, all annotator modules give the opportunity for annotation by using this object.

Ontology based annotation extends the querying capability with ontological reasoning. For

example, assume the client uses “human ontology” which defines “human” as the superclass of the subclasses “male” and “female”. When the user defines the object John, which is a member of the “male” class, it is automatically inferred that John is an individual of “human” class. Therefore this individual can be used in both “human” and “male” class type querying.

4.3 Spatio-Temporal Relations

In order to investigate spatial or temporal relations of objects, firstly the positions should be recorded in the video system. In this thesis, minimum bounding rectangle (MBR) formalization is used to define the positions of objects. Although regions of objects are recorded manually, the architecture of the system is suitable for attaching a semi-automatic annotation module easily in the future. Positions of all objects defined by a rectangle (Figure 4.2) , which is, by definition, the minimum sized rectangle that contains the object of interest.

Figure 4.2: MBR Samples of Objects

As it can be seen in Figure 4.3, an object’s MBR is represented with its four corner points. These points have some pixel values which are calculated as their distance to the origin of the coordinate axis. There is no need to store all these four corner points, because by using only the first and the fourth point’s coordinates all other corner points can be inferred. In OntoVMS implementation, only two corner points are stored and all spatio-temporal calculations are made by using these points.

Spatio-Temporal Relations can be divided into subgroups according to relationship defini-

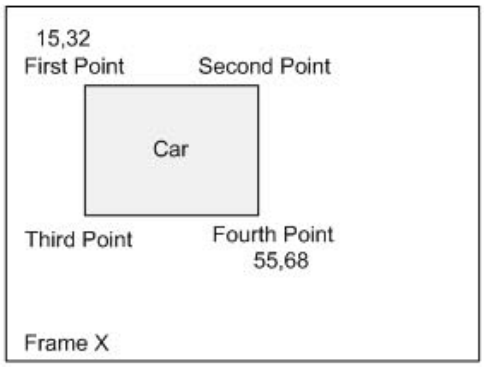


Figure 4.3: MBR Corner Points

tions. Spatial relations describe the relative positions of two objects, whereas temporal relations describe the frame appearance of two objects relative to each other [10]. Below relations are supported by OntoVMS spatio-temporal annotator module:

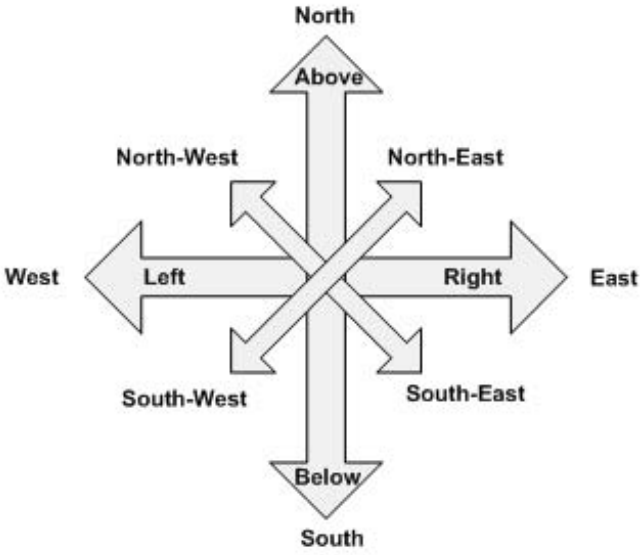


Figure 4.4: Directional Relations

- **Spatial Relations:** Divided into sub relations; strict directional, mixed directional, positional relations and topological relations.
  - **Strict Directional:** There are four strict directional relations; North, South, East and West (Figure 4.4). These relations represent the basic directions of a compass. In order to determine which directional relation is held between objects,

center points of the MBRs are used. If two objects have the same center points, there is no directional relation between these objects. Whereas if they have different center points, the relative positions of the center points according to eight directional coordinate axes represent their directional relation.

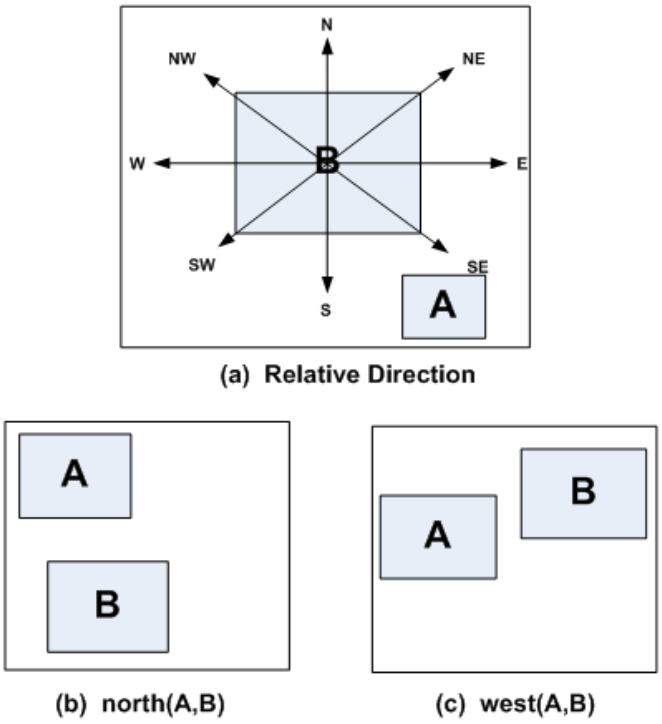


Figure 4.5: Strict Directional Relations

– **Mixed Directional:** There are four mixed directional relations; Northeast, Northwest, Southeast and Southwest. These are intercardinal directions of a compass (Figure 4.4).

– **Positional Relations:** There are four positional relations; Left, Right, Above and Below. Although they are equal to directional relations, in daily life these relations are more common than directional relations; therefore these relations are added to OntoVMS's supported relation list. Due to fact that, there are no mixed-directional equivalence in positional relations, positional relations cover mixed-direction. For instance (Figure 4.6)

  $Above(A, B) \equiv North(A, B) \cup Northeast(A, B) \cup Northwest(A, B)$

  $Below(A, B) \equiv South(A, B) \cup Southeast(A, B) \cup Southwest(A, B)$

– **Distance Relations:** There are only two distance relations in literature; near and

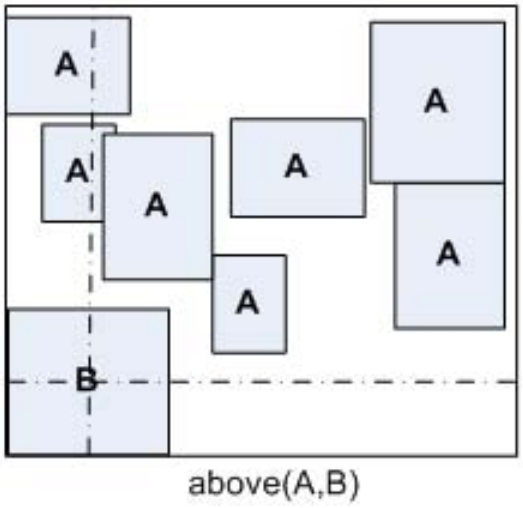


Figure 4.6: Above Relation Sample

far. There is no custom mathematical definition of nearness, therefore nearness and farness relation can be differently implemented in spatio-temporal frameworks. In OntoVMS distance relation is formulated as:

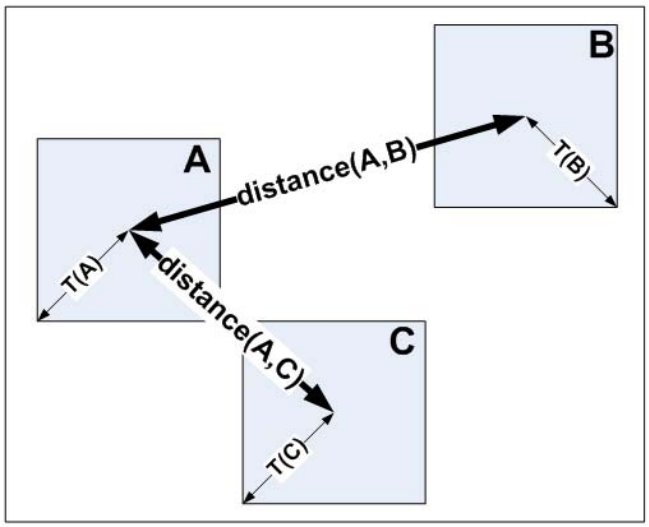


Figure 4.7: OntoVMS Distance Formula

*Definition 1: If difference between center points of two objects is less than 1.4 * sum of their half-diagonal distance, these objects holds near relation.*

For instance, in Figure 4.7 object A and object C holds Near(A,C) relation because;

$distance(A, C) < 1.4 * (T(A) + T(C))$

*Definition 2: If difference between center points of two objects is greater than 1.4 * sum of their half-diagonal distance, these objects holds far relation.*

For instance, in Figure 4.7 object A and object B holds Far(A,B) relation because;

$distance(A, B) > 1.4 * (T(A) + T(B))$

– **Topological Relations:** These relationships are used for representing the relations of the objects boundaries. Therefore in MBR representation, MBR boundaries are used in relationship calculation. There are eight types of topological relationships; Overlaps, Equal, Inside, Contain, Cover, Covered-By, Touch and Disjoint. In OntoVMS two operators are defined for investigating and formulating topological relationships [14].

  * In(A): This operator represents the interior part of the related object MBR.
  * Bo(A): This operator represents the boundary lines of the related object MBR.

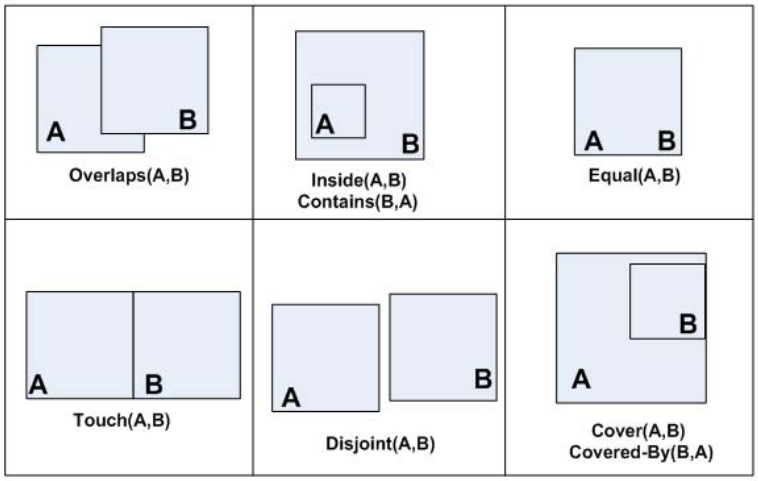


Figure 4.8: Topological Relations

Table 4.1 ([14]) summarize the formalization of topological relations by using "In" and "Bo" operators.

- **Temporal Relations:** Temporal relations mainly deal with object appearance frame sequence relative to the frame sequence of other objects. In order to formalize temporal relations Allen ([2]) proposes temporal interval algebra (Figure 4.10 [32]). Owing to that, temporal reasoning can be applicable. Temporal relations consist of thirteen relations: precedes, meets, overlaps, finished by, contains, starts, equal, started by, during,

Table 4.1: Formulation of Topological Relations

	$Bo(A) \cap Bo(B)$	$Bo(A) \cap In(B)$	$In(A) \cap Bo(B)$	$In(A) \cap In(B)$
Overlaps	$\neq \varnothing$	$\neq \varnothing$	$\neq \varnothing$	$\neq \varnothing$
Equal	$\neq \varnothing$	$\varnothing$	$\varnothing$	$\neq \varnothing$
Inside	$\varnothing$	$\neq \varnothing$	$\varnothing$	$\neq \varnothing$
Contain	$\varnothing$	$\varnothing$	$\neq \varnothing$	$\neq \varnothing$
Touch	$\neq \varnothing$	$\varnothing$	$\varnothing$	$\varnothing$
Disjoint	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$
Cover	$\neq \varnothing$	$\varnothing$	$\neq \varnothing$	$\neq \varnothing$
Covered-By	$\neq \varnothing$	$\neq \varnothing$	$\varnothing$	$\neq \varnothing$

finishes, overlapped by, met by and preceded by. Except equal, the first six relations are inverse relations of the last six relations.

In OntoVMS, temporal relations are calculated according to the appearance order of objects. For instance Figure 4.9 displays a sample temporal ordering between objects A and B. According to the definition of "overlap" relation:

$$A_{start} < B_{start} \text{ AND } A_{end} < B_{end} \text{ AND } A_{end} > B_{start}$$

Therefore, the sample situation in Figure 4.9 satisfies the "overlap" relation.

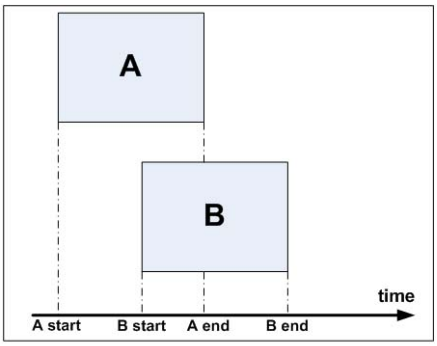


Figure 4.9: Overlap Relation in OntoVMS

In OntoVMS, when the user annotates an object on a frame, this frame is automatically added to the frame appearance list of the object. There is no extra annotation work done by the users in order to make a temporal querying. It means spatial annotation is sufficient for our framework. Details of the temporal querying are explained in query processing section (Section 5.5).

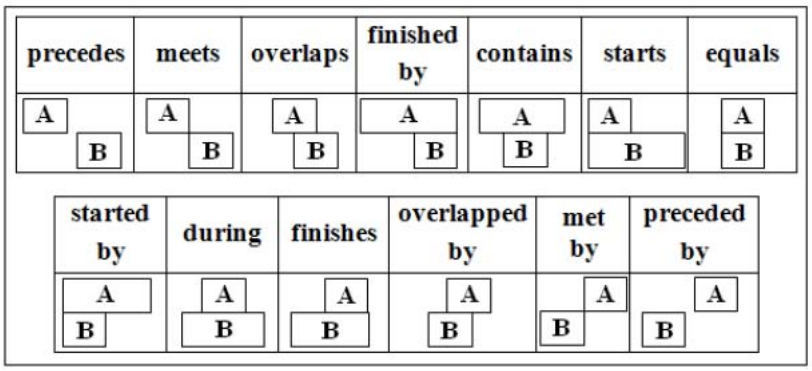


Figure 4.10: Allen Temporal Relations

- **Appearance Relations:**

  OntoVMS supports two additional temporal predicates: “Appear” and “NotAppear”. These predicates are used for querying the frame intervals in which an object of interest appears. For instance the user can make queries such as; “show all frames where Mehmet is displayed”, “show all humans (ontologic class) appearing in the last five minutes of the video” or “show all time intervals where the police officer is NOT displayed when the burglary event is occurred”. These relations are calculated by using the appearance lists of the objects, thus relative ordering is not important.

4.4 Knowledge Base Structure

Since there is no limitation in video size and interested objects, there can be millions of spatio-temporal relations for a single video. For instance, a normal size personal video duration is nearly 30 minutes, and on the average there are four objects of interest in each video frame. As a quick calculation, 1 minutes has 60 seconds, one second has 10 frames for a simple video and 4 objects holds nearly 50 spatial or temporal relations; 30 * 60 * 10 * 50 = 900.000 spatio-temporal entry should be stored for a normal size video. It should be pointed that we assumed each second has 10 frames, but we know that today video technology is many times more than our assumption. Therefore storing all relationships and making a look-up table solution is very inefficient for personal applications, because personal databases have limited size. Another solution is calculating these relationships in query phase without storing them, but this solution needs a lot of query processing time, therefore this solution is not efficient

either.

Optimized solution of this problem is the combination of necessary calculations and necessary storage. In this way only necessary database storage is used and queries are answered in reasonable time. It is time to point that OntoVMS is a web application therefore reasonable amount of calculation in query time makes no noticeable difference for users, because users can not differentiate milliseconds unit of measurements. For these reasons, knowledge-base structure and rule based modeling is the reasonable solution [6], [8], [9].

In rule base modeling, annotated information is stored as knowledge facts, in permanent storage. Rule based modeling in OntoVMS has two types of rule sets; basic facts and inference rules. Basic facts represent only necessary rules which are annotated by semantic annotator module and inference rules represent reasoning rules among spatio-temporal relationships which are already defined [6]. This modeling schema has some advantages; it is easy-to-understand and easy-to-manage, because stored rules (basic facts) are simple tuples therefore they are easy to understand. Moreover inference rules are formalized by Allen algebra and spatial logic; therefore they are easy-to-manage [6].

In OntoVMS knowledge-base, SWI Prolog engine is run. In order to increase system speed, frame based indexing is used to store spatial relations (basic facts). For instance, if there is a relation *"in frame 12, car object is to the west of the house object"* the spatial relation is stored as:

$$west(12, car, house)$$

With this representation, frame based and interval based querying is possible. Frame based queries are answered by Prolog engine, on the other hand interval based queries can be answered by a java code block by interpreting prolog result set.

4.4.1 Spatial Inference

OntoVMS stores only necessary relations and unnecessary relations are pruned. Non-stored relations are extracted in query phase, if user makes some queries about them. In order to prune relations; inference rules are used in spatial-annotator module. Spatial logic gives opportunity to define some inference rules between spatial operators (all rule sets can be found

in Appendix A). In order to clarify inference and prune operations, a sample situation is investigated.

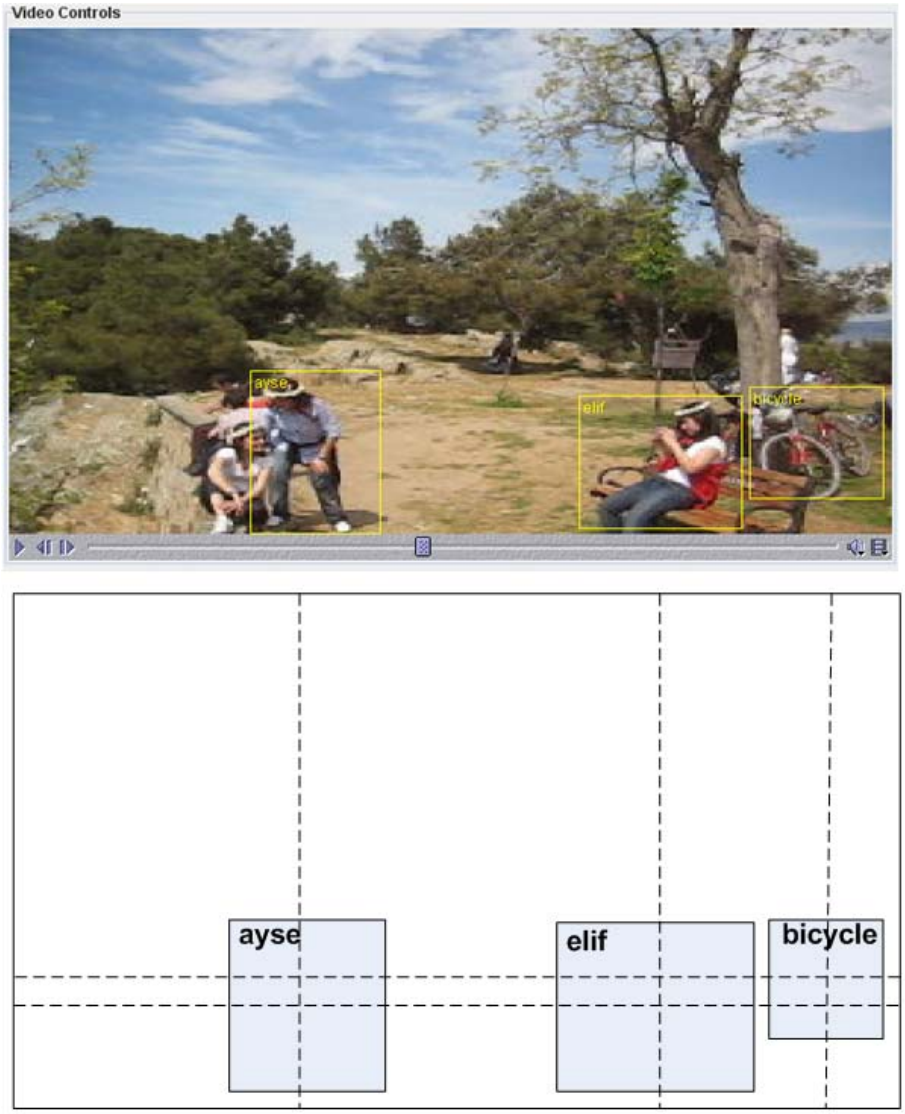


Figure 4.11: Inference Sample

In Figure 4.11, the user is interested in three objects; bicycle, elif and ayse. This figure is one frame of the sample video, therefore only spatial relations can be investigated in this example. By definition (Appendix A [8]):

Rule Set 1.1 *(Inverse property) The relations west, north, north-west, north-east, right and above are inverses of east, south, south-east, south-west, left and below, respectively.*

$(a) \quad west(A, B) \Leftrightarrow east(B, A)$

(b) $north(A, B) \Leftrightarrow south(B, A)$

(c) $north - west(A, B) \Leftrightarrow south - east(B, A)$

(d) $north - east(A, B) \Leftrightarrow south - west(B, A)$

(e) $right(A, B) \Leftrightarrow left(B, A)$

(f) $above(A, B) \Leftrightarrow below(B, A)$

Rule Set 1.2 *(Transivity) If* $ß \in S$*, where S is the set of directional relations, then*

$ß(A, B) \wedge ß(B, C) \Rightarrow ß(A, C)$

Rule Set 1.3 *The relations right, left, above and below can be expressed by other directional relations.*

(a) $east(A, B) \vee north - east(A, B) \vee south - east(A, B) \Leftrightarrow right(A, B)$

(b) $west(A, B) \vee north - west(A, B) \vee south - west(A, B) \Leftrightarrow left(A, B)$

(c) $north(A, B) \vee north - east(A, B) \vee north - west(A, B) \Leftrightarrow above(A, B)$

(d) $south(A, B) \vee south - east(A, B) \vee south - west(A, B) \Leftrightarrow below(A, B)$

If we look center points of the each object we can easily calculate directional relations among these objects.

(1) $ayse\ \ left\ \ elif$ (7) $elif\ \ right\ \ ayse$

(2) $ayse\ \ left\ \ bicycle$ (8) $ayse\ \ right\ \ bicycle$

(3) $elif\ \ left\ \ bicycle$ (9) $bicycle\ \ right\ \ elif$

(4) $ayse\ \ west\ \ elif$ (10) $elif\ \ east\ \ ayse$

(5) $ayse\ \ west\ \ bicycle$ (11) $bicycle\ \ east\ \ ayse$

(6) $elif\ \ west\ \ bicyle$ (12) $bicyle\ \ east\ \ elif$

It can be easily seen that there are 12 directional relations between these three objects, and the user can ask any of them. In traditional way all 12 directional relations should be stored.

On the other hand if reverse-inference rules are used in pruning phase and inference rules are used in querying phase some relations can be discarded.

Using rules (a) and (e) in rule set 1.1 (inverse property), relation 7, 8, 9, 10, 11 and 12 can be discarded. Thus, in a trivial operation half of the relations are pruned.

Using rule (b) in rule set 1.3 relations 1, 2 and 3 can be pruned. Therefore until now storing only three relations is sufficient for extracting all facts.

Lastly using rule set 1.2 (transivity propery), relation 5 can be inferred from using relation 4 and 6:

$$west(ayse, elif) \wedge west(elif, bicycle) \Rightarrow west(ayse, bicycle)$$

It means that relation 5 can be pruned with the help of relation 4 and relation 6. In summary two relations are sufficient to store all data instead of storing twelve relations in above sample.

4.4.2 Temporal Inference

Like spatial relationships, some temporal relations can be inferred by other temporal relations using Allen's temporal algebra. In OntoVMS temporal relations are pruned in annotation step and unnecessary relations are discarded. In order to clarify temporal inference a sample situation is investigated.

In Figure 4.12 there are three objects, but these objects appear in different key frames. Object A appears in frame K, K+1 and K+2, object B appears in frame K+3,K+4, and K+5 and lastly object C appears in frame K+6, K+7 and K+8. According to their appearance order following temporal relations can be calculated.

(1) $A \ \ precedes \ \ B$ (4) $B \ \ preceded-by \ \ A$

(2) $A \ \ precedes \ \ C$ (5) $C \ \ preceded-by \ \ A$

(3) $B \ \ precedes \ \ C$ (6) $C \ \ preceded-by \ \ B$

According to Allen algebra following temporal inference rules can be defined.

Rule 1 : *(Inverse property) The relations precedes, meets, overlaps, finished-by contains and*

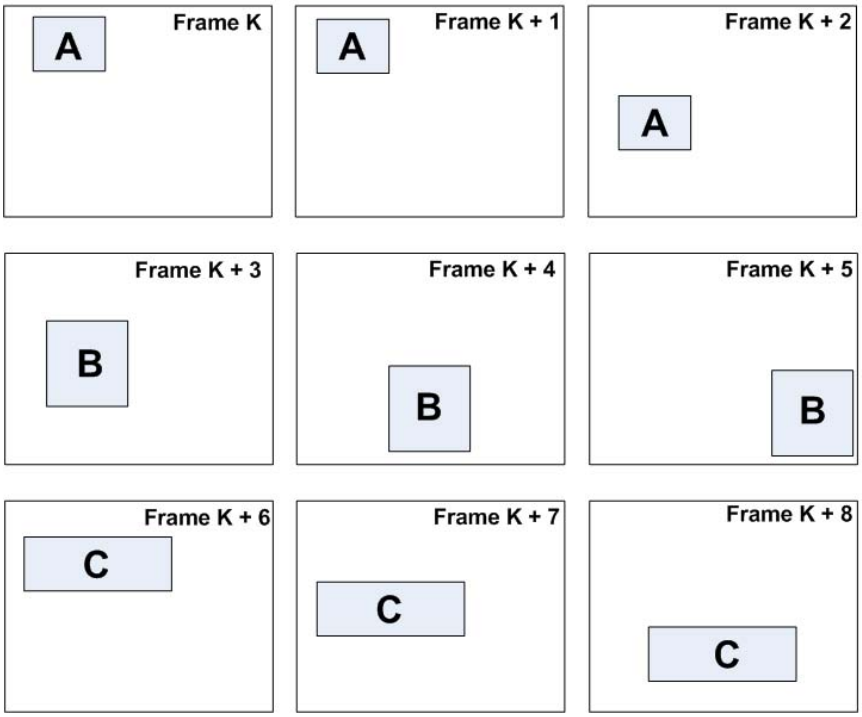


Figure 4.12: Temporal Inference Sample

starts are inverses of preceded-by, met-by, overlapped-by, finishes, during and started-by respectively.

(a) $precedes(A, B) \Leftrightarrow preceded - by(B, A)$

(b) $meets(A, B) \Leftrightarrow met - by(B, A)$

(c) $overlaps(A, B) \Leftrightarrow overlapped - by(B, A)$

(d) $finished - by(A, B) \Leftrightarrow finishes(B, A)$

(e) $contains(A, B) \Leftrightarrow during(B, A)$

(f) $starts(A, B) \Leftrightarrow started - by(B, A)$

Rule 2 *(Transivity) If* $ß \in S$*, where S is the set of precedes, preceded-by, finishes, finished-by, during, contains, starts, started-by and equal, then*

$ß(A, B) \wedge ß(B, C) \Rightarrow ß(A, C)$

By using (a) rule of the Rule 1 set, it can be easily realized that relations 4, 5 and 6 can be

discarded. This trivial rule eliminates half of the temporal rules. Moreover using Rule 2, relation 1 and relation 3 can infer relation 2.

$precedes(A, B) \wedge precedes(B, C) \Rightarrow precedes(A, C)$

In conclusion, storing only two relations (1 and 3) is sufficient for preserving all temporal data in this sample situation.

4.4.3 Fact Pruning Algorithm

As it was explained in Section 4.4.1 that, there is no need to store all spatio-temporal relations, storing only some of them is sufficient for answering all spatio-temporal queries. In this situation, the main problem is finding which relations should be stored, which relations can be ignored. For answering this question a heuristic algorithm is improved [8]. Algorithm does not claim that all unnecessary relations can be pruned, but it claims all pruned relations can be reconstructed by others. It means that algorithm does not propose the best solution; instead it proposes completely safe solution to this problem.

For pruning unnecessary relations, algorithm steps adapted from [8], are:

1. Step: Objects in each frame K are sorted according to its MBR center point x coordinate value. For instance in Figure 4.13 objects are sorted according to their center x coordinate value; sorted sequence is A, B, C, D and E.

2. Step: According to Step 1, objects get a value from 0 to n. Leftmost object gets 0 value and rightmost object gets n value. For instance in Figure 4.13

   A object gets = 0 value

   B object = 1

   C object = 2

   D object = 3

   E object = 4

3. Step: Take an initially empty array *m*;

4. Step: Starting from label 0 (in our example Figure 4.13, object A), pair objects according to their label values. All pairs have a common property; second object of the

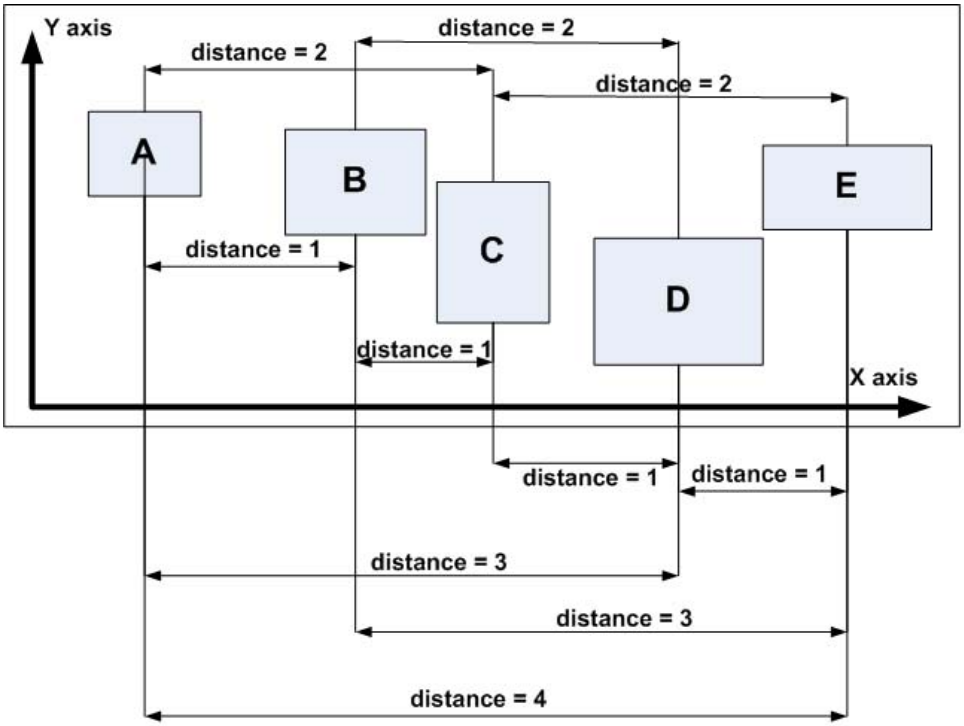


Figure 4.13: Fact Pruning Algorithm

pair has bigger label value than the first object of the pair, and pairs are constructed according to their label difference values. Start with label difference value = 1

5. Step: Starting from the leftmost pair, compute relations between objects in a pair.

   IF this relation can be infered from relations inside m

   discard this relation,

   ELSE

   add this relation into m

6. Step: Apply step 5 to all pairs

7. Step: Increase label difference value and return to the step 4, if label difference value is equal to object count, pass to step 8.

8. Return array *m* as core relations.

In order to demonstrate the algorithm, we should apply the steps of the algorithm to objects in Figure 4.13.

After sorting objects and assigning label values, starting from label distance value = 1 construct pairs.

Label distance = 1 pairs = AB, BC, CD, DE

distance = 2 pairs = AC, BD, CE

distance = 3 pairs = AD, BE

distance = 4 pairs = AE

Starting from label distance 1 to label distance 4, calculate relations between pair objects and if this relation can be inferred from other relations which are the members of m array, discard this relation, otherwise add this relation to array m.

Properties of the algorithm:

1. Algorithm guarantees that all discarded relations can be reconstructed by using stored relations.
2. System starts computing relations with the minimum number of label difference; it means that system starts with the neighbor objects, in real life users mainly interest in neighbor objects in their queries. Therefore most of times no reconstruction is necessary in query phase.
3. Complexity: $O(kn^2)$

   n : average number of objects in a frame

   k : number of frames

4.4.3.1 Space Efficiency:

In OntoVMS, the inference mechanism and knowledge-base are used to optimize system space efficiency. OntoVMS is a personal video management system therefore space specification of this system is limited with a personal computer space specifications. In order to decrease space requirements, inference rules are used.

Because of some trivial and non-trivial rules, the system has great space savings. According to Allen's algebra 12 out of 13 temporal relations have inverse property, and according to spatial

logic directional and mixed-directional rules have inverse property. Moreover in distance relations, near is the inverse of the far relation. Therefore nearly half of the relations can be pruned in the annotation module.

In addition, some non-trivial inference rules (Appendix A) are used in the elimination of unnecessary relationships. Despite the fact that, system performance is changed according to object distribution in video frames, nearly %60-70 space savings can be reached. For instance in 4.14, the chart of space efficiency according to different number of objects in a single frame is presented.

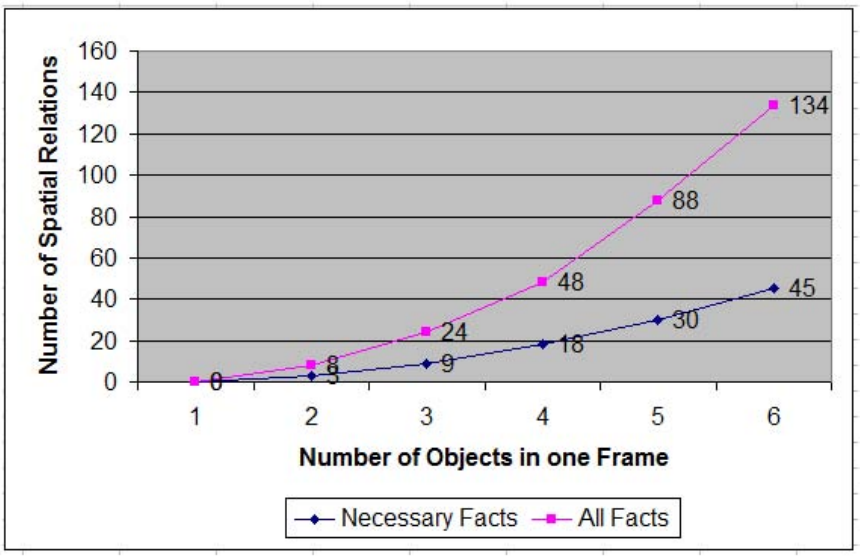


Figure 4.14: Space Efficiency

4.4.4 Spatio-Temporal Data Modeling In OntoVMS

One of the important differences between OntoVMS and other video management systems is ontology based annotation capability. In order to supply this feature, spatio-temporal annotator module of the OntoVMS dynamically communicates with Jena ontology API. It makes system more complicated (Figure 4.15). Therefore spatio-temporal annotation part of the OntoVMS implementation should be explained with more detailed. Spatio-Temporal modeling can be divided into the following steps:

- **User Annotation:**

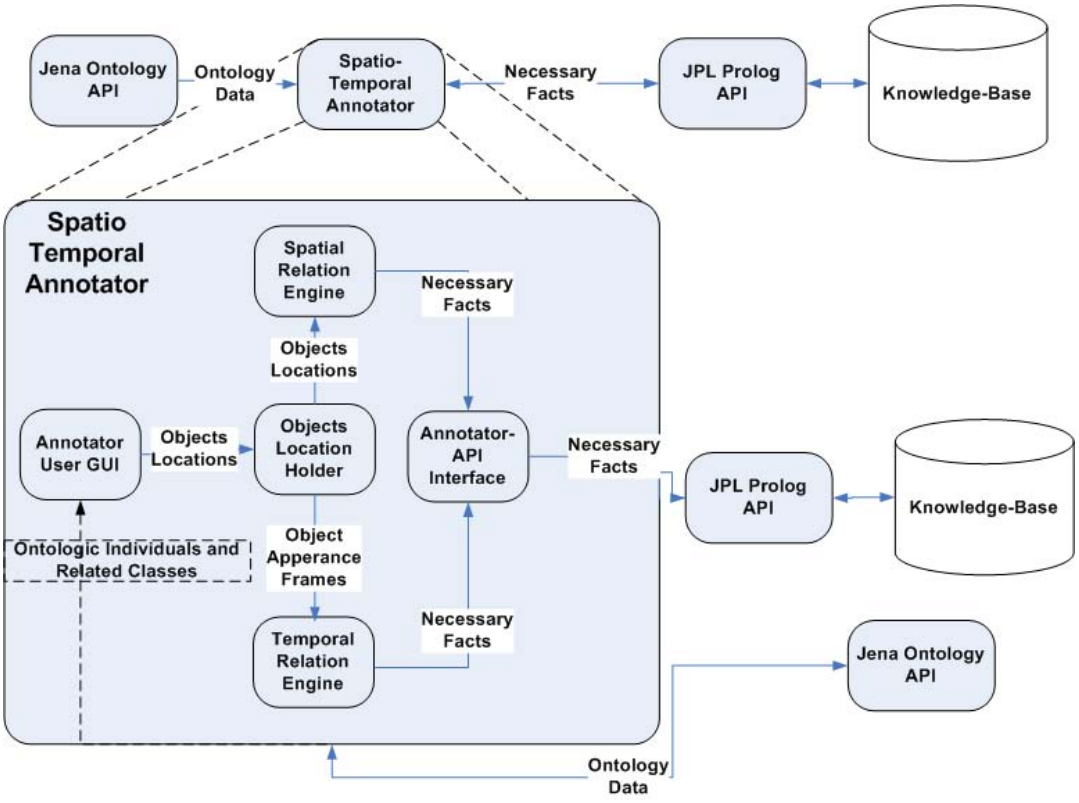


Figure 4.15: Detailed Architecture of the Spatio-Temporal Annotator

There are two different annotation environments. In the first environment, the GUI is used for concept annotation. The user can annotate an event with its start and end time points. Moreover the user can define new individuals by using this interface. These definitions are used for calculating spatio-temporal relationships in the other annotation environment. In the second environment, the GUI is used for pointing the locations of objects in specific video frames. As it can be seen in Figure 4.16, the object list at the user interface is filled with the individuals of the related domain ontology. The objects are listed in a specific format; for instance *Male_%_mehmet*. This notation represents that *mehmet* is an instance of the "male" class of the domain ontology. In summary, the user can select any object of interest by using this user friendly GUI.

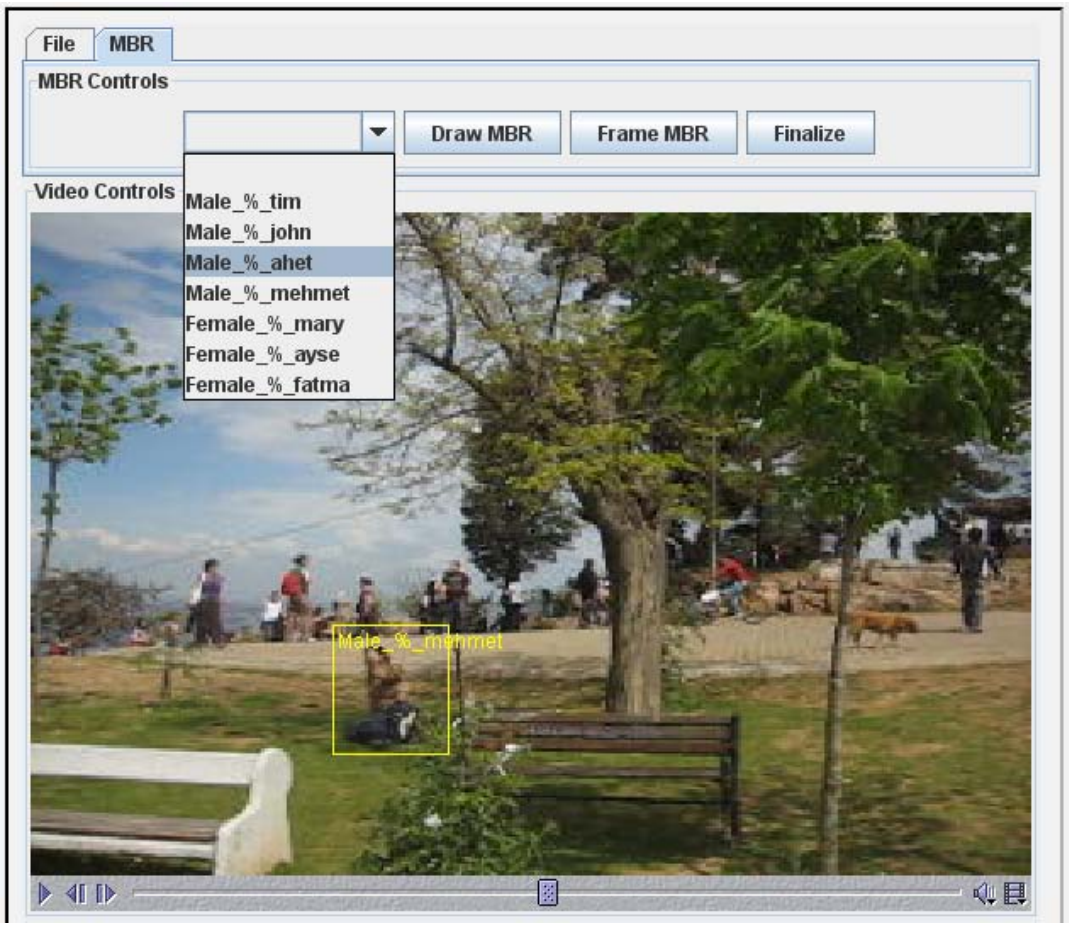


Figure 4.16: OntoVMS SpatioTemporal Annotation GUI

- **MBR storage:**

  After the annotations are finished, the object location holder module gets all spatial

data. This data is shared with other modules; spatial relation engine, temporal relation engine and Jena ontology API. The object location holder module prepares location data in order to add this information into the related class of the ontological model (Figure 4.17). Jena API stores the coordinates of all individuals because this model should be saved or loaded as an external file.

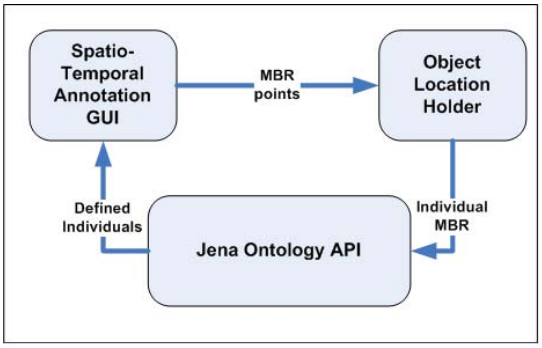


Figure 4.17: OntoVMS Annotation Jena Modeling Interaction

- **Necessary Spatial Relation Calculation:**

  After taking location information of all objects, Spatial relation engine starts to apply the heuristic algorithm which is explained in Section 4.4.3. According to this algorithm, spatial relations are calculated in an object pair. If it is a core (necessary) relation, then the relation is stored, otherwise the relation is discarded. In this way the system makes pruning operation dynamically. In other words, the system never holds lots of relations in its storage space. At the end of this operation, necessary relations are transferred to Annotator API interface (Figure 4.15).

- **Necessary Temporal Relation Calculation:**

  Temporal Relation Engine works a bit differ from the Spatial Relation engine, because there is no spatial information necessity in the Temporal Relation Engine. For temporal relations, appearance list is sufficient to calculate all relations. Therefore this engine gets only object appearance list from the Objects Location Holder module. Temporal inference rules are applied and necessary relations are calculated. Like Spatial Relation Engine, this engine transfers core relations to Annotator API interface (Figure 4.15).

- **Necessary Fact Storage on Knowledge-Base(prolog codes):**

  Knowledge-base is the spatial and temporal storage space of OntoVMS framework.

Core facts are stored as prolog codes in this part of the system. OntoVMS Annotator API interface has mainly two tasks in spatio-temporal data modeling. At initializing part, Annotator API interface module makes connection with JPL Prolog API. In this way JPL prolog API is initialized to accept inference rules. After that point, previously declared inference rules (Appendix A) are loaded into the system as prolog codes.

For instance spatial North relation rules such as;

$$North(A, B) \Rightarrow South(B, A)$$

$$North(A, B) \wedge North(B, C) \Rightarrow North(A, C)$$

is represented as:

northX(N,A,B) :- south(N,B,A).

northX(N,A,B) :- north(N,A,B).

northX(N,A,B) :- north(N,A,C) , northX(N,C,B).

On the other hand temporal Precedes relation rules such as;

$$Precedes(A, B) \Rightarrow Preceded - By(B, A)$$

$$Precedes(A, B) \wedge Precedes(B, C) \Rightarrow Precedes(A, C)$$

is represented as:

precedesX(A,B) :- precededBy(B,A).

precedesX(A,B) :- precedes(A,B).

precedesX(A,B) :- precedes(A,C) , precedesX(C,B).

The main difference between spatial *North* relation and temporal *Precedes* relation is that, spatial relations are defined for each frame of the videos, and therefore in prolog representation there is another field *N*, which represents the frame number of the each spatial relation. On the other hand temporal relations are defined for the whole video, independent of specific frames therefore frame index is not used.

Another important task of this module is storing necessary facts into the knowledge base at the end of the annotation parts. Spatial relations are stored as three variable

tuple like *north(12,A,B)*, which represents *at frame 12 object A is North of the object B*. On the other hand temporal relations are stored as two variable tuple format like *meets(A,B)*, which means *object A and object B meets in video according to their frame appearance list*. At the end of this operation, JPL API is now ready to answer user queries.

4.5 Trajectory Data Modeling

By definition, trajectory means the path of an object in space. In videos, the same definition can be used in order to model trajectory of an object. To simulate trajectory, positions of the objects in each frame should be considered (Figure 4.18). Therefore like spatial relations, object positions are important for each key frame in a video. Moreover, trajectory movements are made in a time interval, because of that appearance list is important in trajectory modeling, like temporal relations. For combining appearance and position list features, displacement list can be used ([5]).

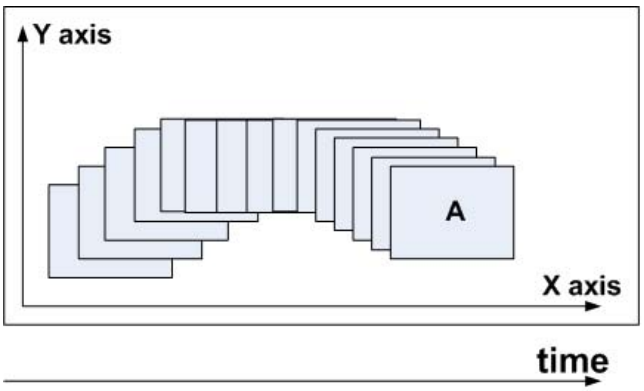


Figure 4.18: Trajectory of Object A

In OntoVMS, trajectory data is stored as in the following formalization.

(object_name, center_point_x, center_point_y,displacement_list)

Object_name : Name of the object

Center_point_x = MBR center point x coordinate

Center_point_y = MBR center point y coordinate

Displacement_list = A list which holds array of (frame_number, x_disp,y_disp)

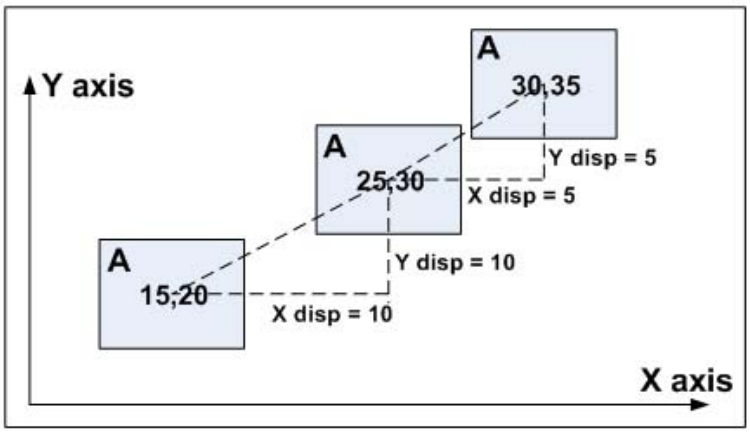


Figure 4.19: Trajectory Displacement List

In displacement list, one entry is held if object appears in a frame. The first element of the list represents the first key frame in which this object appears. For the next frame, x_displacement value is calculated as the difference between MBR center point x_coord in this frame and the previous position x_coordinate. y_displacement is calculated in the same way considering y_coordinates this time. For instance in Figure 4.19 there is an object. This object appears in three frames of a video. Center point of the object in

Frame 1 = 15,20

Frame 2 = 25,30

Frame 3 = 30,35

If we construct frame displacement list

First item = (1, 15 - 15, 20 - 20) = (1,0,0)

Second item = (2, 25 - 15, 30 - 20) = (2,10,10)

Third item = (3, 30 - 25, 35 - 30) = (3,5,5)

therefore formalization is

(A,15,20,[(1,0,0), (2,10,10), (3,5,5)])

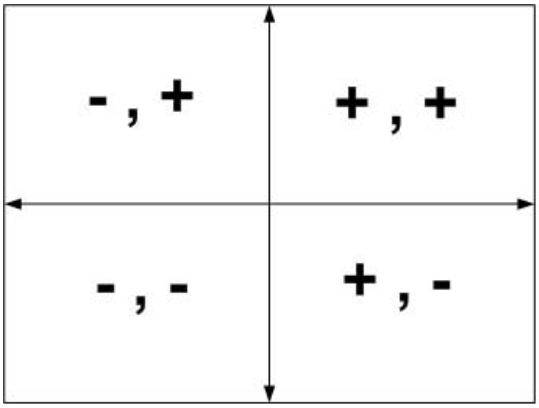


Figure 4.20: Sign Values according to Coordinate Axes

This formalization is selected in order to decrease query processing time; because in query phase without any calculation, displacement list shows object direction according to the previous point. Figure 4.20 shows that if displacement values are positive, object is at north-east direction according to the previous position. If displacement x value is near 0 and y value is positive, it can be conclude that object is in north direction according to its previous position.

CHAPTER 5

QUERY PROCESSING

In OntoVMS, querying capability is extended with ontological data. In our framework it is possible to pose a query such as “return all humans near to car, when a crash happens”. Such a query is more meaningful than “return all objects near to car, when a crash happens”. In the second query, the answer may contain some other objects, such as a tree or a waste basket, but the user may not be interested in these objects. OntoVMS query engine gives the user the opportunity to query only interested objects, for instance at our sample situation; human class and its members.

Query processing can be divided into two steps; user query collection and execution of them. In this thesis, a user friendly GUI is developed by combo boxes to simplify query interface. In the execution of queries, Jena ontology API and Prolog JPL APIs are used in order to enhance query processing capabilities.

This chapter is organized as follows: In Section 5.1 the details of OntoVMS query engine are explained, In Section 5.2 the use of knowledge-base in query processing is explained. In Section 5.3 query output types are explained. In Section 5.4 some general information is given about Spatio-Temporal query types. After that, the implementation details of the OntoVMS query engine are discussed in Section 5.5. In order to clarify querying capabilities of the OntoVMS framework, the query interface is presented in Section 5.6. Lastly some illustrative query examples are shown in Section 5.7.

5.1 OntoVMS Query Engine

OntoVMS Query Engine consists of the following modules:

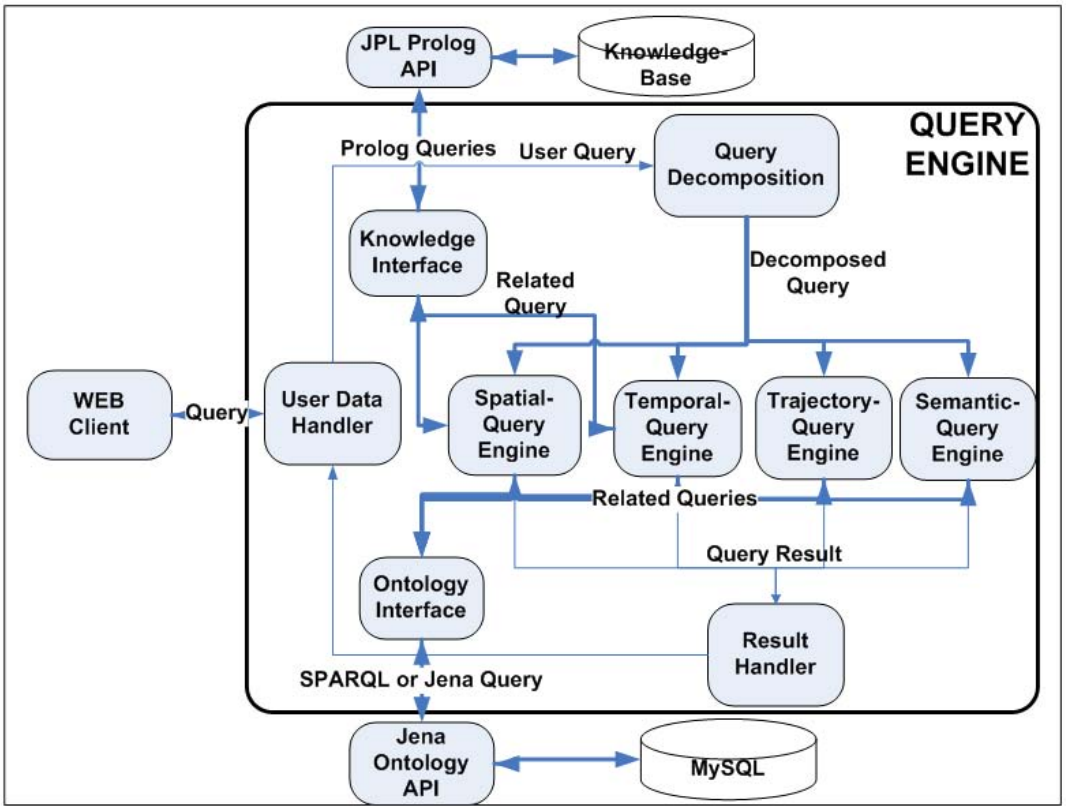


Figure 5.1: OntoVMS Query Engine

- **User Data Handler:**

  This module is implemented for collecting data from the user interfaces. The system has very user friendly web interfaces. In OntoVMS GUI Combo boxes are used to define a query. This module gets combo box values from java web applet in order to translate them into query sentences.

- **Query Decomposition:**

  This module gets query strings from User Data Handler module. It parses the query strings and divides queries into subqueries according to query types. In this way each subquery can be processed by a specific engine. According to system specifications, OntoVMS has five query types; spatial query, temporal query, appearance query, trajectory query and semantic type query, therefore this module decomposes input queries into five subquery types.

- **Spatial Query Engine:**

  It is the most complicated module of the query engine, because there are lots of spatial query types (directional, mixed-directional, topological ...) and this engine must communicate with both knowledge-base and ontology based databases according to query content. If there is no ontological content in the query string, only prolog query can be enough. On the other hand if there is an ontological content, it makes ontological spatial query by using SPARQL or JENA specific queries.

- **Temporal Query Engine:**

  Working principle of this engine is parallel with spatial query engine. This engine must communicate with both the knowledge base and the ontology based database. After the query string is taken from the Query Decomposition module, temporal query engine makes related temporal queries by using prolog inference mechanism, and it gets related individuals or classes from the ontology API.

- **Trajectory Query Engine:**

  This engine works differently from other query engines, because in this engine there is no connection with knowledge-base structure. Trajectory data has different type of modeling, therefore knowledge-base connection is unnecessary for this module. Since this module can make ontological trajectory querying, it communicates dynamically with the ontology API.

- **Semantic Query Engine:**

  The working principle of this module is similar to the trajectory query engine. Knowledge-base connection is not necessary in this module either. It dynamically communicates with the ontology API. Semantic query engine is used for extracting time intervals of events; therefore ontology connection is adequate for this type.

- **Knowledge Interface:**

  It is the communication interface of the prolog API. Spatial and Temporal Query Engines use this module to communicate with JPL prolog API, and prolog results are formatted in this API before sending to the related engine.

- **Ontology Interface:**

  This interface is used in ontological communications. All query engines use this module in order to communicate with Jena Ontology API. According to query engine type, SPARQL queries or Jena specific queries are used. For example in order to get all class names, SPARQL queries are used; on the other hand to get specific class instances Jena *getIndividual* method is used.

- **Result Handler:**

  It is used for combining query results from query engines in order to transform these results into the user understandable format. Results are collected in a web API structure and they are reflected as the items of the data table (Figure 5.2 shows the query result of *"return all humans which appears on the screen between 11.0 and 15.0 seconds"*).

Start Frame	End Frame	First Object	Relation	Second Object	SHOW Video
11.0	15.0	fatih	Appear	Appear	Play..
11.0	15.0	mehmet	Appear	Appear	Play..
11.0	15.0	atakan	Appear	Appear	Play..

Figure 5.2: Result Handler Output

5.2 Knowledge-Base Inference in Query Phase

As it was stated in earlier; in annotation phase, some spatial or temporal relations are pruned according to the inference rules by using a heuristic algorithm. On the other hand in querying phase, reasoning is used in order to answer queries about non-stored relations. In query processing SWI-Prolog is run as the knowledge-base inference engine. Therefore prolog rules are used for knowledge-base querying.

To show system stability (all pruned rules can be extracted from stored relations), the same example in Section 4.4.1 is used, and it is tried to extract all facts in the same example.

Stored Relations are:

(1) $ayse \;\; west \;\; elif$

(2) $elif \;\; west \;\; bicyle$

By using rule set 1.2 (transivity propery) (Appendix A), new relation can be inferred with the help of these two relations;

$ß(A, B) \wedge ß(B, C) \Rightarrow ß(A, C)$

$west(ayse, elif) \wedge west(elif, bicycle) \Rightarrow west(ayse, bicycle)$

Therefore this new relation is added to our relation list

(1) $ayse \;\; west \;\; elif$

(2) $elif \;\; west \;\; bicyle$

(3) $ayse \;\; west \;\; bicyle$

Using rule (b) in rule set 1.3 relations (Appendix A) three new relations can be inferred.

$ayse \;\; left \;\; elif$

$elif \;\; left \;\; bicycle$

$ayse \;\; left \;\; bicycle$

Finally using inverse property of the spatial relations other six relations can be inferred; rules

(a) and (e) in rule set 1.1 (Appendix A).

elif right ayse

ayse right bicycle

bicycle right elif

elif east ayse

bicycle east ayse

bicyle east elif

It means we can construct the same relation list which was pruned in section 4.4.1.

(1) *ayse left elif*

(2) *ayse left bicycle*

(3) *elif left bicycle*

(4) *ayse west elif*

(5) *ayse west bicycle*

(6) *elif west bicyle*

(7) *elif right ayse*

(8) *ayse right bicycle*

(9) *bicycle right elif*

(10) *elif east ayse*

(11) *bicycle east ayse*

(12) *bicyle east elif*

These reasoning operations prove that the system is stable and there isn't any data loss because of the fact-pruning operation.

5.3 Query Output Types

In this thesis work, three output types are supported by OntoVMS; it means that the user can get results of three different types of query outputs.

- Object: This type is used for showing all instances or all objects that satisfy user query.
- Object - Frame: This output type returns objects and their intervals according to the selected queries.

- Boolean: Used for answering a boolean type query, result list consist of yes or no boolean type operators.

5.4 Query Types

Since there isn't any universal video query language, all spatio-temporal systems decide their own query types and query languages. According to user needs; spatial, temporal, trajectory and concept queries are supported in OntoVMS, because these are the most common query types for users. Moreover compound queries are added to increase the querying capability.

1. **Concept Querying:**

   This query type can be used for selecting objects or events of interest. For example with the help of this query type the user can ask the system to "return all frames when prime minister appears on scene" or "return frame intervals where goal event occurs" type queries. This query type can be mostly used if the user interested in a specific object or event in a video.

2. **Spatial Querying:**

   Spatial Querying is dealing with the locations of objects in a video frame and positions of objects relative to each other. It allows users to query about directional relations, topological relations, positional relations and distance relations. For example in Figure 5.3 there is a goal position in a football video and the user wants to make some spatial queries, because objects positions and their relationships are important in this frame.

   If ball is in the goal basket area (topological type)

   If goal keeper is near to ball (distance type)

   If team_1 player is left of the team_2 player (directional type)

3. **Temporal Querying:**

   Temporal queries deal with objects appearances in videos relative to a time sequence. According to Allen's algebra there are thirteen different temporal relations, as it was explained in Section 4.3. Using this query type the client can make following queries:

   Return those objects that appear before the dog.

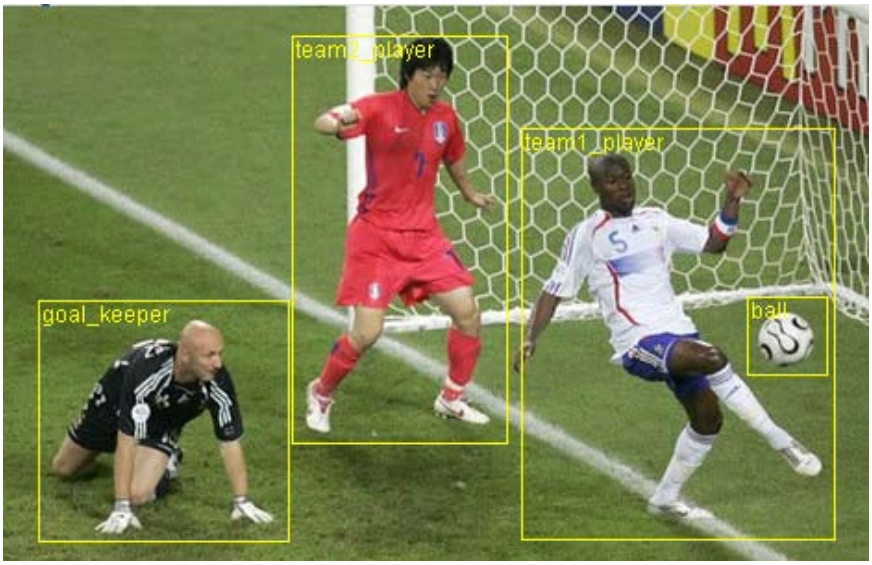


Figure 5.3: Goal Position in Football Domain

Return humans (ontology class) where humans and yellow car appears at the same time.

4. **Appearance Querying:**

   Appearance queries differ from standard temporal query types because relativity among objects is not used in this querying type. This type deals with the appearances of objects of interest in video frames. Using this query type, the user can ask the following queries:

   Return all frames where the lions appear in the video (documentary video).

   Return all intervals, where the director of the movie does NOT appear during the time interval between the 10th minute and 20th minute of the video.

5. **Trajectory Querying:**

   This type is used for querying the motion path of objects of interest. In OntoVMS, three types of trajectory queries are supported. Source-destination trajectory type, object trajectory and directional trajectory queries are provided to the users. Using source-destination trajectory type (Figure 5.4), the user can select the starting and ending regions of a trajectory. In this way the system finds objects that have trajectories starting and ending in the selected regions. The second type of trajectory query is object trajectory. In this type, the user can select a single object or a class member in the domain ontology. For example the user can ask the system to "return trajectory of all dogs".

The third type of supported trajectory query is directional trajectory. In this type, the user can query about objects which make directional motion, e.g. "return all objects which go to the east direction".

Figure 5.4: Source-Destination Type Trajectory Query

6. **Compound Queries:**

   In OntoVMS, user query capability is enhanced by using complex & compound querying. By definition complex query represents the usage of inverse and transitive properties of the simple spatial and temporal relations. In our system these capabilities are used in knowledge-base structure in order to answer user queries. On the other hand compound query represents the combination of simple query types, for instance combination of spatial and temporal user queries.

   OntoVMS compound querying supports combination of concept, spatial, temporal and appearance queries. In this way, the user can query complex or compound situations. For instance, *"return all males where they are standing to left of the safe-deposit box when burglary happens and they appear before the police officer."* is a legal query in our framework. This compound query can be divided into three different simple queries. First query is a spatial type query; *"Return all males where they are standing to left of the safe-deposit box"*. In this query, ontological male class objects are examined according to their positions. The second query is a temporal query; *"Return all males where they appear before the police officer"*. In this query, ontological male class

objects are examined according to their appearance order in the video file. For this query execution, all male objects are returned if these objects appear before the police officer. Last query is a concept type user query; *"when burglary happens"*. This query type is used for narrowing a video interval into user interested size. In this way, this query selects only a specific time interval from the whole video. After executing these three different types of queries, Union or Intersection operations are applied to their results in order to combine different query types.

5.5 OntoVMS Ontology Based Query Processing

OntoVMS query processor supports both spatio-temporal relationships and ontological querying capabilities. Therefore querying phase of the OntoVMS, dynamically communicate with both knowledge-base and ontological database in order to answer user query. Each query type has a specific query procedure therefore query phase follows different ways for different query types. This procedure is explained in detail in the following sub sections.

5.5.1 Query Parsing and Decomposition

OntoVMS ontology based query processor supports compound query processing; therefore users are free to make compound queries. In order to process a compound query, it must be divided into sub queries before executing any of them. OntoVMS compound queries are defined by using simple query types, AND operator, OR operator, open parenthesis and close parenthesis. For example:

((Simple_Query_1 OR Simple_Query_2) AND (Simple_Query_3 OR Simpe_Query_4))

Therefore before dividing into sub queries, compound query should be parsed and interpreted. If there is a structure error it should be found before query execution. Therefore query parser checks compound query string before processing. If the string is free of format errors, query decompositor divides big query string into sub queries, and distributes these sub queries to the related engines. In order to divide a big query string into sub queries, Dijakstra Shunting algorithm is used [27]. This algorithm gives the definition of a method which is used for parsing mathematical equations. The first step of the algorithm is; transforming equation

from infix notation to postfix notation according to operator precedences'.

According to algorithm details, postfix notation is used at evaluation step. In this step, operands are evaluated according to specific operator definitions and operator precedences'. OntoVMS uses its own operator definitions for Union and Intersection operators. These operators are defined separately for each query combinations.

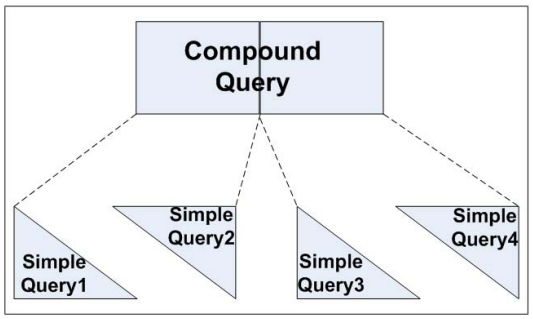


Figure 5.5: Decomposition of Compound Query

5.5.2 Query Execution

OntoVMS query execution process consists of two steps, query execution procedure in the related query engine and result preparation for user interface. Query execution step has different procedures for different query types therefore it is explained in the following sub sections separately. On the other hand result preparation step is a fixed step; which gets query results from the related engine and transforms these results into user interface structure in order to exhibit in web applet.

5.5.2.1 Concept Query Execution

According to concept modeling, individuals are stored in MySQL database and this database is controlled by Jena Ontology API. Therefore all concept queries are formulated by using the ontology API. Semantic querying is completed in four steps. In the first step a SPARQL query is prepared and all ontological classes are listed. The user can select a class name from the ontology. After that, another query is prepared and it gets all individuals which belong to the selected class. In the third step the user selects the object of interest. After that the engine prepares a Jena Ontology query in order to get the selected individuals. In the fourth step, the

ontology API gives the values of the selected individuals. Lastly, the engine transports the results to the Result Handler module in order to display them at the web interface.

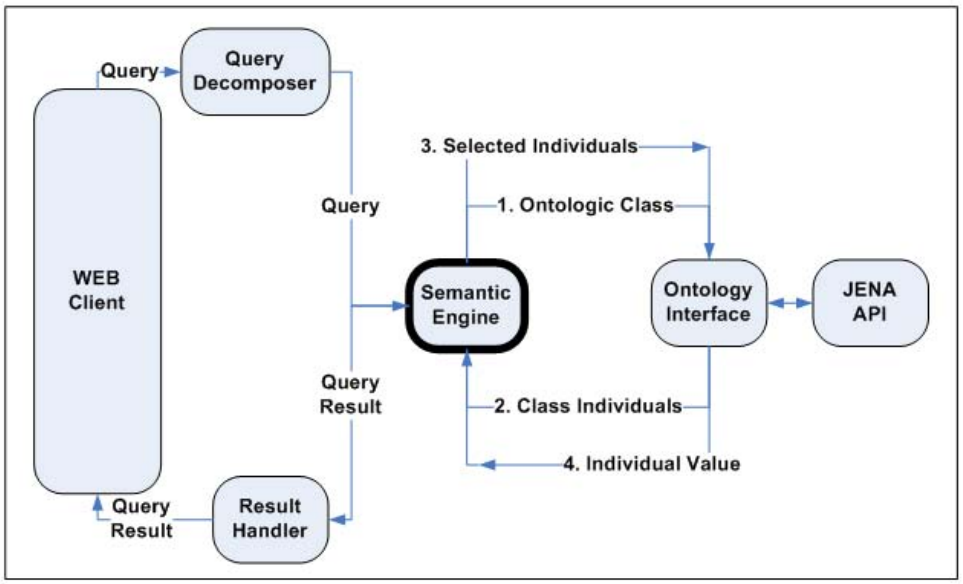


Figure 5.6: Semantic Query Engine

5.5.2.2 Spatial Query Execution

In OntoVMS, spatial relations are stored in knowledge-base as prolog facts. Since our system supports ontological querying, Jena API connection is necessary in spatial query execution. There are different types of spatial queries, but fortunately there are no differences in the execution procedures of these spatial query types. Spatial querying starts with a SPARQL query sentence which is used for filling the spatial query interface combo boxes. In order to serve as a user friendly web interface, the related combo boxes are filled at initialization part. Thus, all class names and all individuals are taken from Jena ontology API (Figure 5.7).

At that point there are lots of different query options that are provided to the user. If the user makes a Boolean type query, the query string is transformed into prolog code and prolog query is processed by JPL API. According to API result, yes or no boolean type query result is returned. For example if there is a situation like in Figure 5.8 and the user asks a:

Query:

Is Ali is west of the Car object at frame K

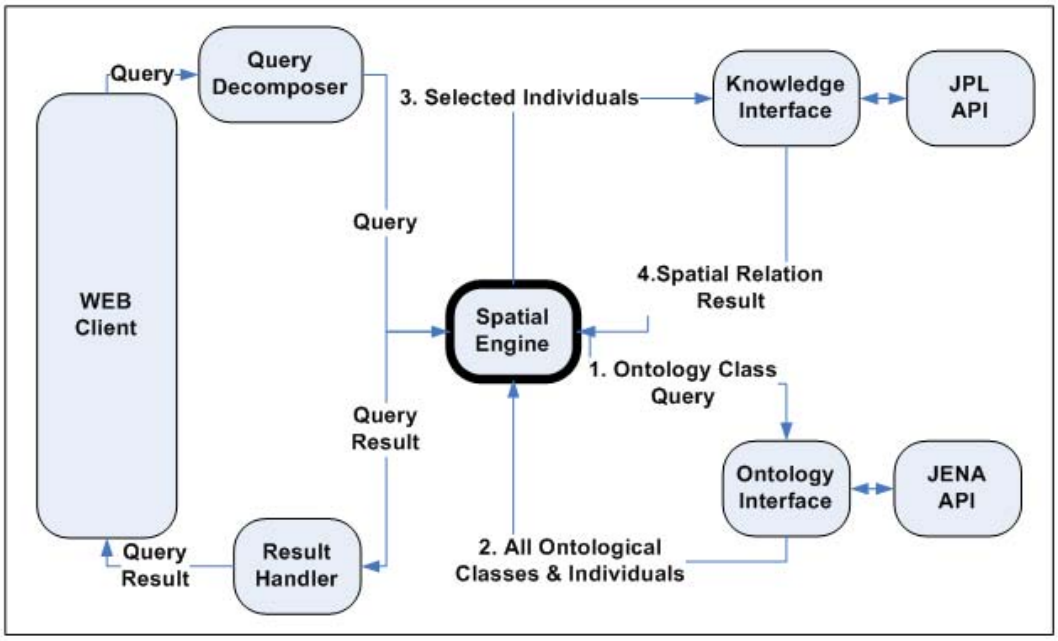


Figure 5.7: Spatial Query Engine

Result:

TRUE

If the user wants to list all objects (object output type query) in a specific frame, a prolog variable type query string is prepared. According to this variable query, Prolog API creates a result set which contains all objects. Spatial Engine transforms these objects to the Result Handler.

Query:

Return all objects which are left of the Car, at frame K.

Result

Bird, Ali, Ayse, Waste-basket

If the user wants to list all objects in a video segment, lots of prolog variable type queries are prepared. Each of these variable queries is processed, and the result set is combined by the Spatial Engine. Thus both frame interval and object querying are processed in this way.

Query:

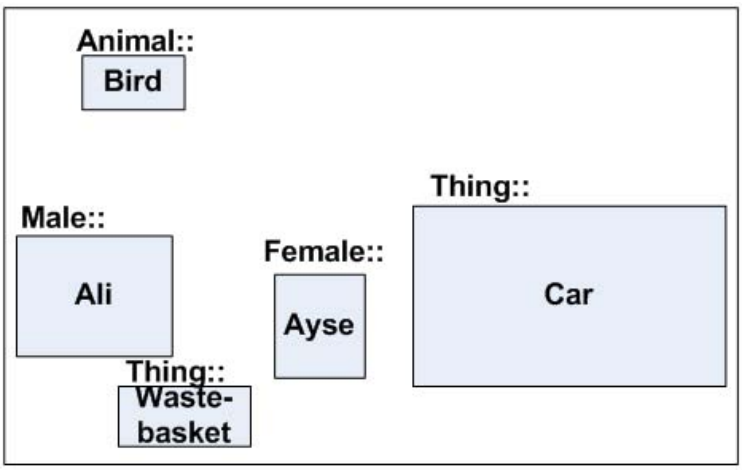


Figure 5.8: Ontological Spatial Query Example

Return all objects which are to the left of the Car between frame K and frame L.

Result

Frame K : Bird, Ali, Ayse, Waste-basket

Frame K +1 = Bird, Ayse

...

If the user wants to list all objects which belong to a specific ontological class in a specific frame, a prolog variable type query string is prepared. According to this variable query, Prolog API creates a result set which contains all objects. After taking this result set, engine filters this list according to ontological data. Only selected class individuals are picked from this list and these objects are sent to the Result Handler module.

Query:

Return all Male class objects which are left of the Car, at frame K.

Result

Ali

If the user wants to list all objects belonging to a specific ontological class in all frames, lots of prolog variable type queries are prepared. Each of these variable queries is processed, and

the result set is combined in Spatial Engine. At that point ontological filter is applied. Only selected class individuals are picked from this list and these objects are sent to the Result Handler module.

Query:

Return all Animal class objects which are to left of the Car between frame K and frame L.

Result

Frame K : Bird

Frame K +1 = Dog

...

5.5.2.3 Temporal Query Execution

In OntoVMS temporal query execution displays the same characteristics with spatial query execution. In this execution type, both Jena Ontology API and JPL Prolog API are used, because ontological data is taken from Jena and knowledge-base facts are taken from the Prolog APIs. Like Spatial Query execution, temporal query execution starts with the initialization of combo boxes in web interface. Related SPARQL query is processed in order to get class names and defined individuals. Then, according to the user's choice corresponding Prolog query string is created and knowledge-base data is taken from Prolog result set. If the user selects ontological querying, the result set is filtered according to ontological facts.

By definition temporal relations deal with appearance of objects in frame sets. Therefore, temporal relations are defined for a video, not for a specific frame or time interval.

5.5.2.4 Trajectory Query Execution

Trajectory query execution differs from the execution of other query types because as stated in Section 4.5, trajectory has a different data modeling. Knowledge-base connection is not necessary because knowledge-base stores only spatio-temporal relations. Since ontological classes are used in trajectory data modeling, Jena ontology API connection is used for trajec-

tory query processing.

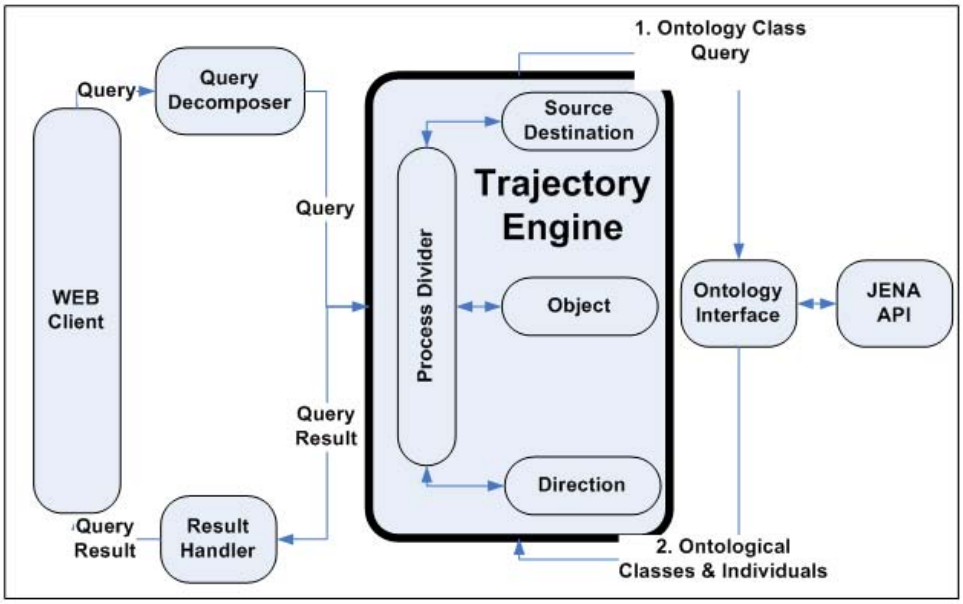


Figure 5.9: Trajectory Query Engine

Trajectory query processing starts with an ontological class query like spatial and temporal query executions. The prepared ontological query is processed in order to get class names and individuals from the domain ontology (Figure 5.9). The result is taken from the ontology interface and combo boxes are filled. According to combo box values, three different trajectory options are provided to the users.

The first choice is source-destination type querying. In this query type the user can select a source and a destination region by using the graphic component of the web interface. In this query type Process Divider module employs source-destination type trajectory query engine. This query processor calculates pixel values of source and destination regions. After calculating these areas, all objects are investigated to see if they have been in that area in their motion path. If some objects are matched with these coordinates, these objects and their frame intervals are returned. For example in a sample situation like Figure 5.10, B object is the answer of this query type.

The second choice is object type user querying. In this query type the user can examine a specific object motion path or motion paths of all individuals which belongs to a specific ontological class. In this query type, the displacement list is not used because the user wants

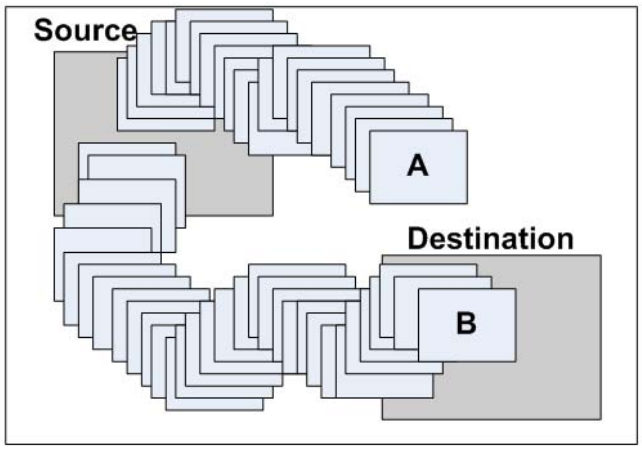


Figure 5.10: Source Destination Trajectory Sample

to see the motion of the object. Therefore only object appearance list is displayed. If the object makes a motion path and exits from the scene, and after some time it returns to the scene; a new motion path is created and two different motion intervals are displayed to the user. If the user chooses ontological classes, then the motion paths of the individuals of the specified class are displayed only.

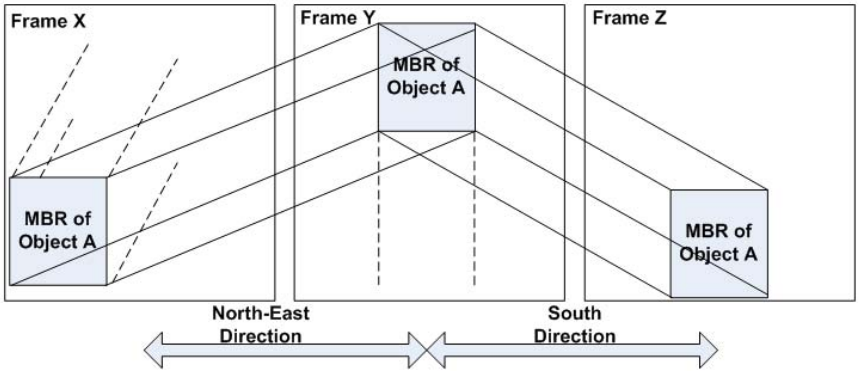


Figure 5.11: Directional Type of Trajectory Querying

The last trajectory query type is directional trajectory querying. In this query type directions of the motion paths are investigated. According to trajectory data modeling (Section 4.5), object's trajectories are stored in the following format:

(object name, center point x, center point y,displacement list)

In directional querying, object displacement list is examined. According to the displacement

list values, object trajectories can be found by using the coordinate plane. For instance, if displacement x value is increasing according to the previous value of the list and y value is constant it can be concluded that the object moves towards the east direction. If the object makes a directional trajectory and changes this direction in time, the object trajectory is divided into subparts and each part can be queried individually. For example in a sample situation like Figure 5.11, if the user poses a query to "return all objects which make south direction trajectory", the result will be "object A is the result between frame Y and Z".

5.5.2.5 Appearance Query Execution

Appearance query execution process is parallel with object type trajectory query execution; thus, separate engine was not implemented. It uses trajectory engine connections and its features. Like trajectory querying, Ontology API connection is enough for appearance query processing. This query type uses appearance lists of the objects in order to answer the user queries. Moreover semantic query engine output (event filter) and user defined time intervals can be used as a filtering mechanism in this type queries. For instance "return all objects and their frames which appear first 5 minutes of the video" is a time filtered appearance query sample.

5.5.2.6 Compound Query Execution

OntoVMS supports compound queries which are combinations of semantic, spatial, temporal and appearance query types in order to enhance querying capability. There is no direct connection between compound query engine and database APIs, because compound engine does not make any spatial or temporal calculations. It hires simple query engine types for calculating the result of the compound queries. Figure 5.12 shows that, Compound Query Engine gets compound query strings from the user interface and divides the whole process into related engines and finally it combines their result sets. For instance, the user can make a compound query like "*Return all Team1 defense football players where these players are near the goal-basket at the first goal and they appear at the same time with goal-keeper.*", in this query three different query types are combined in order to construct a compound query; spatial query, temporal query and concept queries. In such a situation; in order to answer the user query, compound query engine combines the results of these three query engines.

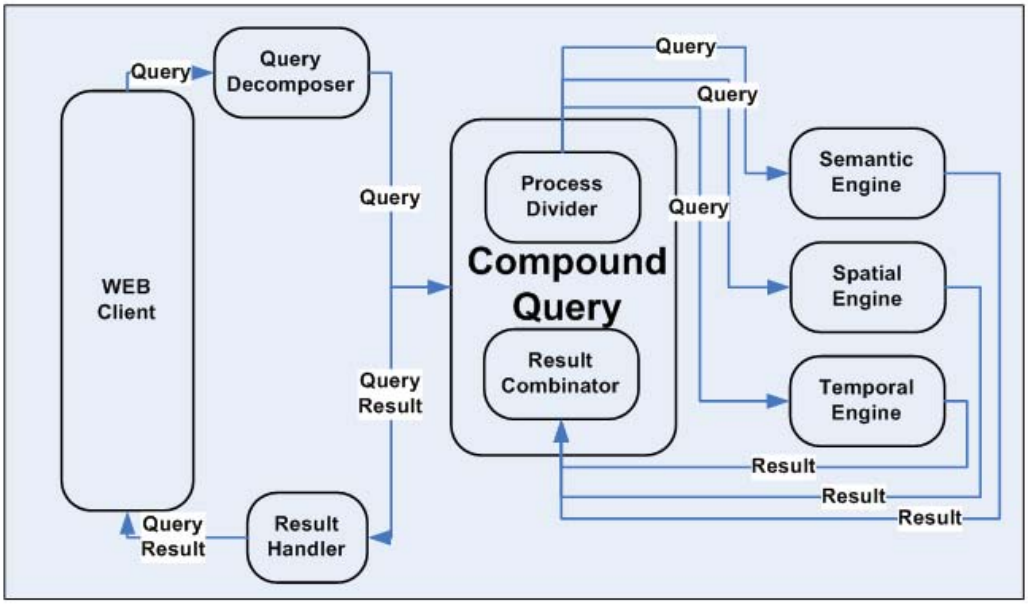


Figure 5.12: Compound Query Engine

There are three operators defined in OntoVMS, for combining the result sets of different query types; AND (as intersection operator), OR (as union operator) and FILTER (for filtering result set). FILTER operator differs from the other two operators, because filter operator is defined for two query combinations; concept - spatial and concept - appearance query combinations. Other operators are defined for all combination types. Unfortunately, all query engines use different format in their result sets. For instance spatial relations are stored as *west(10.2, ahmet, mehmet)*. On the other hand temporal relations do not use keyframes therefore they are stored as *precedes(ahmet, mehmet)*. For that reason, union and intersection operators are defined separately for each combination pair. Combination types and their results can be summarizes as:

1. ***Boolean vs Boolean*** $\Longrightarrow$ ***Boolean***

   Boolean query results can be produced by spatial or temporal engines. For instance if the user makes a query like "*If Ahmet is to the left of the waste-basket at frame K*", spatial engine returns boolean type result; true or false. In the combination process of two different boolean type, truth table is used. Intersection operator behaves like AND operator, and Union operator behaves like OR operator as shown in Figure5.13. Therefore combination result returns a boolean type answer, true or false.

First Operand	Second Operand	Result
false	false	false
false	true	false
true	false	false
true	true	true

(a) AND (Intersection Operator)

First Operand	Second Operand	Result
false	false	false
false	true	true
true	false	true
true	true	true

(b) OR (Union Operator)

Figure 5.13: Boolean vs Boolean Compound Query

2. ***Boolean vs Spatial $\Longrightarrow$ Processed Object Frame List***

   The combination of boolean type and spatial type queries constructs an object-frame list result; due to the AND operator definition, boolean operands are used for filtering the result sets. On the other hand, due to the OntoVMS union operator definition; boolean operands are ineffective, because object-frame list contains more meaningful data than pure true/false results. Combinations can be summarized as:

   For AND operator:

   $false \quad vs \quad spatial\ List \quad \Rightarrow \quad empty\ list$

   $true \quad vs \quad spatial\ List \quad \Rightarrow \quad object - frame\ list$

   For OR operator:

   $false \quad vs \quad spatial\ List \quad \Rightarrow \quad object - frame\ list$

   $true \quad vs \quad spatial\ List \quad \Rightarrow \quad object - frame\ list$

3. ***Boolean vs Appearance $\Longrightarrow$ Processed Object Frame List***

   This compound query characteristic behaves like “Boolean vs Spatial” query combination type. The same operator definitions can be used in this combination.

4. ***Boolean vs Temporal $\Longrightarrow$ Processed Object Frame List***

   This compound query characteristic behaves like “Boolean vs Spatial” query combination type. The same operator definitions can be used in this combination.

5. ***Concept vs Spatial $\Longrightarrow$ Processed Object Frame List***

In OntoVMS framework, concept querying corresponds to retrieving the frame interval where objects or events of interest are seen. For example in a football match, when the first goal event is queried, the specific time interval in which the first goal occurs will be returned. On the other hand, spatial querying corresponds to searching objects' locations and their mutual spatial relationships in the video frames. By definition both of these query types use key frames in their representations. Thus, combination of concept and spatial query types are commonly used in compound querying. Concept querying is used with spatial query type as a filter.

Filter operator is defined in this combination type, because the union or intersection of time intervals and object lists is meaningless for users. The filter operator can be defined as:

If concept time can be formalized as: $[t_{conceptStart}, t_{conceptEnd}]$

If spatial time can be formalized as: $[t_{spatialStart}, t_{spatialEnd}]$

Combination time interval can be calculated as: $[t_{combinationStart}, t_{combinationEnd}]$ where

$$t_{combinationStart} = max(t_{conceptStart}, t_{spatialStart})$$

$$t_{combinationEnd} = min(t_{conceptEnd}, t_{spatialEnd})$$

According to the definition above, the filter operation is applied for selecting the minimum range between the concept and spatial time intervals.

6. ***Concept vs Appearance $\Longrightarrow$ Processed Object Frame List***

   This combination type exhibits the same characteristics with "Concept vs Spatial" combination type; therefore same operator definitions are used. Concept querying is used for filtering the Appearance query results.

7. ***Spatial vs Spatial $\Longrightarrow$ Processed Object Frame List***

   In OntoVMS different spatial queries can be used in order to construct a compound query. For instance the relation *between* is not defined in the basic spatial logic, but this relation can be constructed by using two different spatial relations; combination of *left* and *right* relations.

   A spatial result can be formalized as: *relation(frame,first_object,second_object)*

A processed object-frame result can be formalized as: *(frame,object)*

therefore it can be summarized as:

$spatial\ \ OPERATOR\ \ spatial \Rightarrow (frame, object)$

Intersection and Union operators are defined for combining different spatial queries. In order to demonstrate operator definitions; assume that the user makes a compound query; "*return all objects which are to the left of the house and they are near to the car between time intervals 15.0 - 20.0.*" This query can be decomposed into two different spatial queries "*return all objects which are to the left of the house between time intervals 15.0 - 20.0*" and "*return all objects which are near to car between time intervals 15.0 - 20.0*". Assume the following results are returned for these two queries:

First query results	Second query results
Left(16.0,ahmet,house)	*Near(16.0,ahmet,car)*
Left(17.0,ayse,house)	*Near(19.0,fatma,car)*

The union and intersection of the two queries are:

First Query OR Second Query = *(16.0, ahmet), (17.0,ayse), (19.0, fatma)*

First Query AND Second Query = *(16.0, ahmet)*

8. ***Appearance vs Appearance $\Longrightarrow$ Processed Object Frame List***

It exhibits the same characteristics with "Spatial vs Spatial" combination type. Thus, the same operator definitions can be used in this combination. This type can be used for selecting group of people in a video and their frames, for instance *"return all frames where ali and ahmet appears at the same time"*. Moreover above query capability can be enhanced with adding "concept vs appearance" query combination, like *"return all frames where ali and ahmet appears together at lunch time"*.

9. ***Temporal vs Temporal $\Longrightarrow$ Processed Object List***

"temporal vs temporal" compound queries exhibit same characteristics with "spatial vs spatial" compound querying. Therefore same formalization can be used.

A temporal relation: *relation(frame, first_object,second_object)*

Processed object list: *(frame, object)*

therefore it can be summarized as:

$temporal \;\; OPERATOR \;\; temporal \Rightarrow (frame, object)$

10. ***Spatial vs Temporal $\Longrightarrow$ Processed Object Frame List***

    In OntoVMS, spatial relations can be combined with temporal relations.

    A spatial relation : *relation(frame,first_object,second_object)*

    A temporal relation: *relation(frame, first_object,second_object)*

    Processed object-frame list : *(frame,object)*

    therefore it can be summarized as:

    $spatial \;\; OPERATOR \;\; temporal \Rightarrow (frame, object)$

11. ***Spatial vs Appearance $\Longrightarrow$ Processed Object Frame List***

    It exhibits the same characteristics with "Spatial vs Spatial" combination type. Thus, the same operator definitions can be used in this combination. This type can be used for selecting group of people in a video and their positions related with other objects. It is important to point that spatial relation is dominating the appearance relation, because spatial information covers the appearance information.

12. ***Appearance vs Temporal $\Longrightarrow$ Processed Object Frame List***

    It exhibits the same characteristics with "Spatial vs Temporal" combination type. Thus, the same operator definitions can be used in this combination type.

13. ***Processed vs AnyType $\Longrightarrow$ Processed Object Frame List***

    It is important to point that; the user can use any number of simple queries in order to construct a compound query in OntoVMS. It means

    Query1 AND Query2 AND Query3 AND (Query4 OR Query5)

    is a legal query in our framework. In order to support nested queries "Processed" result type is created. Processed results are constructed from the above compound query

combination types, such as “spatial vs temporal”. With the help of this combination type unlimited number of simple queries can be combined. For example the query above can be calculated as

$$\underbrace{\underbrace{\underbrace{Query1 \;\; AND \;\; Query2}_{Processed1} \;\; AND \;\; Query3}_{Processed2} \;\; AND \;\; (\underbrace{Query4 \;\; OR \;\; Query5}_{Processed3})}_{Processed4}$$

where

$Query1 \;\; AND \;\; Query2 \Rightarrow Processed1$

$Processed1 \;\; AND \;\; Query3 \Rightarrow Processed2$

$Query4 \;\; OR \;\; Query5 \Rightarrow Processed3$

$Processed2 \;\; AND \;\; Processed3 \Rightarrow Processed4$

A processed relation can be formulated as: *(frame,object)*

therefore it can be summarized as:

$processed \;\; OPERATOR \;\; any\ type \Rightarrow (frame, object)$

5.6 OntoVMS Query User Interface

The query interface is handled with user friendly web components.

Combo box values are filled at the initialization phase of the querying process according to domain ontology. Therefore users can select objects or ontological classes easily. Figure 5.14 illustrate the OntoVMS Spatio-Temporal query interface. In this interface if the user is interested in a specific object, he can select the object’s name, or alternatively the user can select class names. Moreover if the user wants to see all objects he can select the question mark symbol (?). According to these input types, engine works differently.

Figure 5.15 exhibits the trajectory query interface of the OntoVMS. In this interface the user can query by using the graphical component of the system.

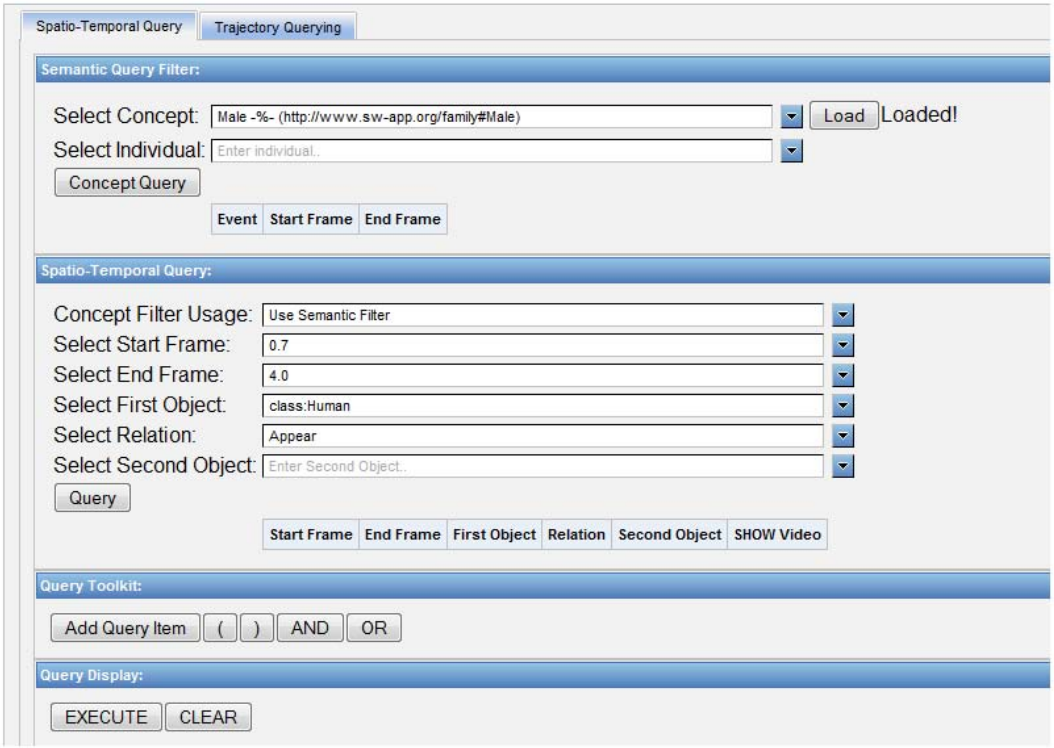


Figure 5.14: Spatio-Temporal Query Interface

5.7 Query Examples

In this section OntoVMS supported query examples are explained. It is important to point that these queries can be combined as compound queries in order to enhance query capability.

Query: *Return all frames where Yılmaz Erdoğan is left of the Demet Akbağ*

OntoVMS Equivalent:

Frame Start ComboBox: ?

Frame End ComboBox: ?

First Object ComboBox: Yılmaz Erdoğan

Relation Combobox: left

Second Object ComboBox: Demet Akbağ

Query: *Return all team_1 players where team_1 players are near team_2_goalkeeper, when*

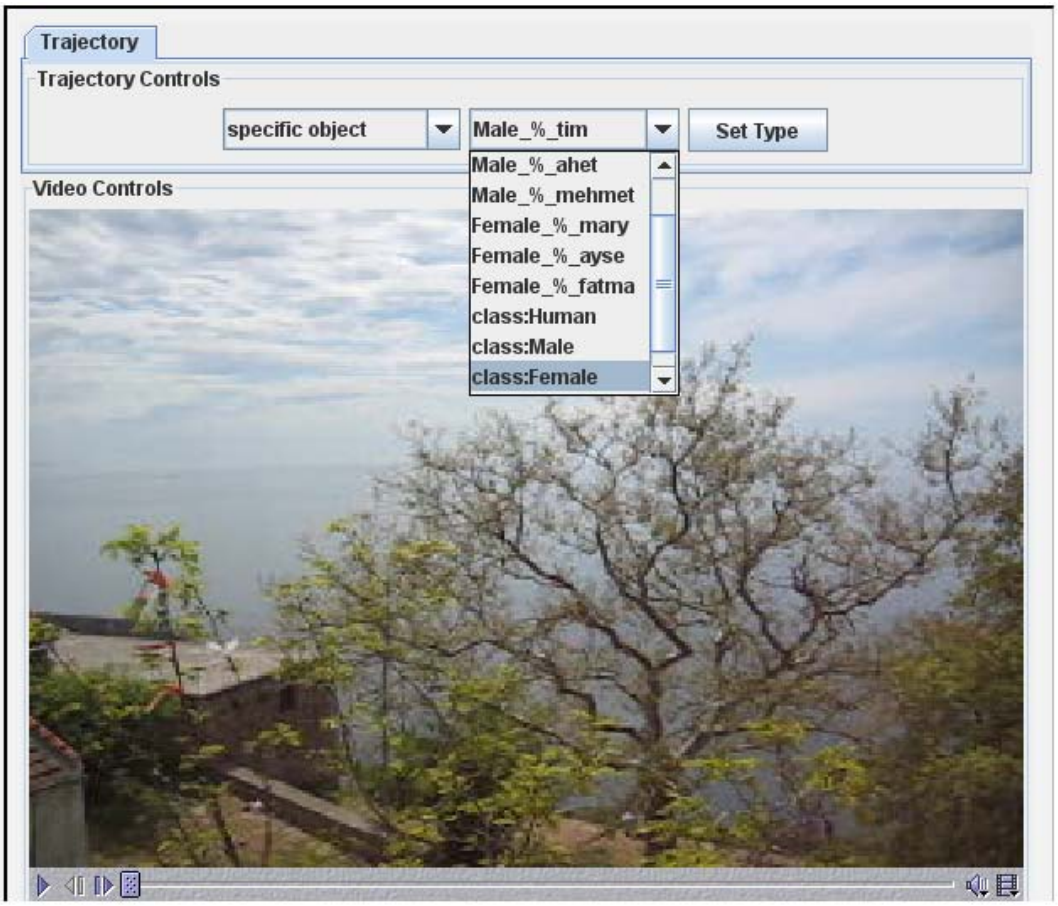


Figure 5.15: Trajectory Query Interface

the first goal occurs.

OntoVMS Equivalent:

First Object ComboBox: class:Team_1

Relation Combobox: near

Second Object ComboBox: Team_2_goalkeeper

Select Concept: goal

Select Event: first_goal

Query: *Return all humans who appear in the same frames with Uğur Dündar.*

OntoVMS Equivalent:

First Object ComboBox: class: Human

Relation Combobox: equal

Second Object ComboBox: Uğur Dündar

Query: *Return all objects which make the selected trajectory.*

OntoVMS Equivalent:

Start Graphic Box : selected start region (Figure 5.4)

Trajectory Type Box: source-destination query

End Graphic Box: selected end region

CHAPTER 6

CASE STUDY

OntoVMS is a domain independent ontology based video management system. In the evaluation period of this framework, the system is used with different domain ontologies in order to test annotation and querying capabilities. Actually the system characteristics are not affected from the selected ontology. For demo purposes the following screenshots are taken from human ontology.

Moreover, in this chapter the use of a semi-automatic annotation tool, face tracker, is explained. This tool is developed by Computer Vision and Intelligent Systems Research Laboratory of the Middle East Technical University. Our framework supplies an infrastructure to this system in order to support semi-automatic data annotation in video files.

In the following sections, the features of the system and user interfaces are explained by screenshots in order to make the system familiar to the end users. The main page of the OntoVMS can be seen in Figure 6.1

OntoVMS user interface consist of three panels; in Section 6.1 admin menu and its features are explained. Section 6.2 explains the usage of OntoVMS Annotation panel and lastly Section 6.3 represents query panel of the system.

6.1 Admin Panel

This module is implemented for managing MPEG-7 and domain ontology operations. System save and load operations can be made by using this panel. Moreover, it supports automatic combination of MPEG-7 and domain ontologies. Admin menu includes the following sub

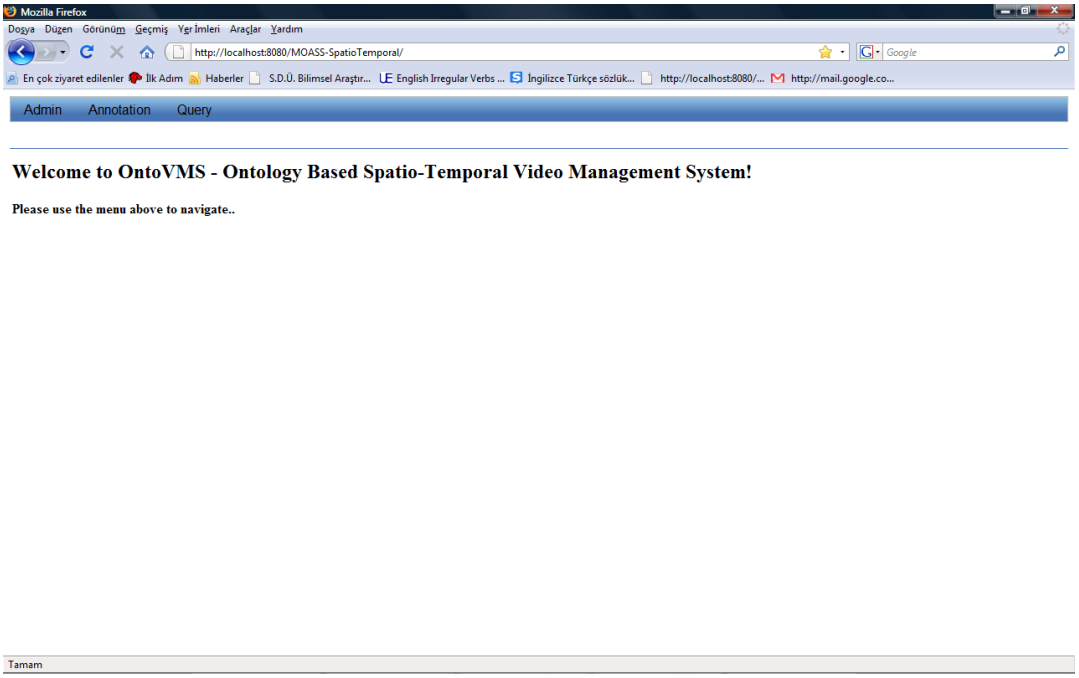


Figure 6.1: Main Page of the OntoVMS

menus.

- Initialize: This module is used for initializing the whole system. All ontological data and annotations are deleted from databases in order to start a new session. After cleaning procedure, MPEG-7 ontology is loaded automatically to the database.

- Import Ontology: This module is used for selecting the domain ontology as it can be seen in Figure 6.2

- Configure Ontology: The user can categorize the concepts of the imported ontology items by using this module [28] . According to this categorization, automatic ontology integration is processed in this module.

- Delete Ontology: It provides an opportunity to delete an imported ontology.

- Save System: This module is used for saving all system data into an OWL file. In this way, the user can store important annotations in a file in order to use in the future.

- Load System: Previously saved systems can be loaded by using this module.

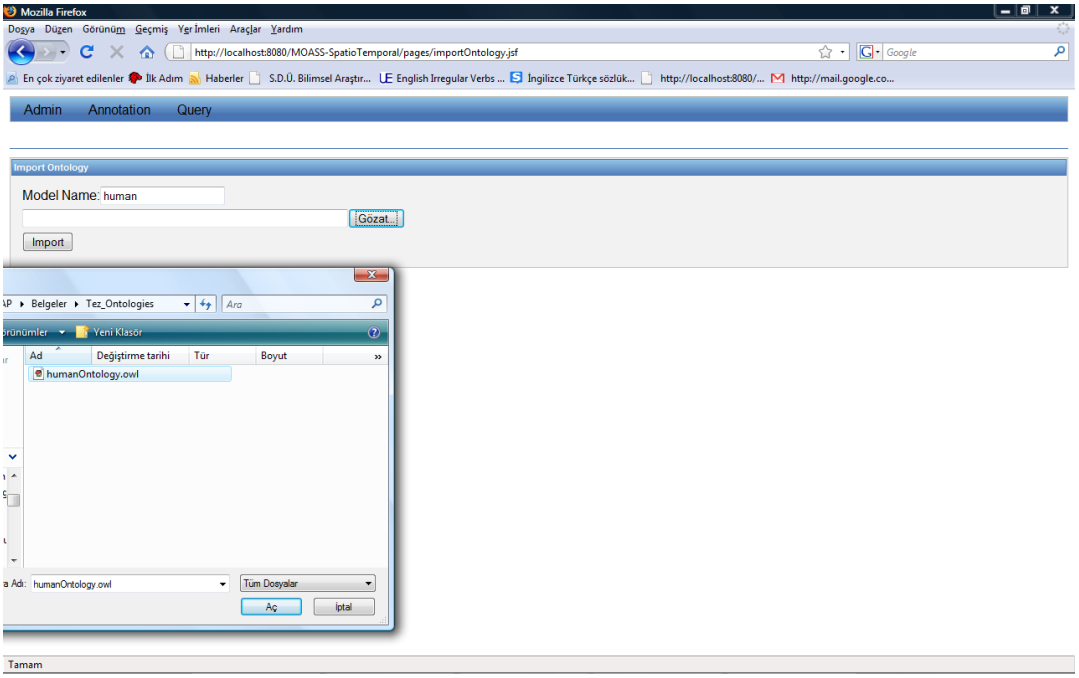


Figure 6.2: Import Ontology Module of the OntoVMS

6.2 Annotation Panel

This panel is used for managing ontological annotations (Figure 6.3). In this panel the user has four options:

1. Define Individuals:

   OntoVMS uses ontological annotation in order to provide ontological querying capability to the users. Therefore, before the annotation process starts, the user should define the objects' ontological class information. For instance in Figure 6.3, "atakan" object is defined as an individual of "Male" class in the related ontology.

2. Semantic Annotation:

   In OntoVMS, semantic annotations are used for defining the time intervals of events. These events are used for filtering time intervals in query operations. Figure 6.4 shows a sample semantic annotation data. In this sample, "atakanSpeaking" event is defined between the time interval 12.0 and 17.0.

3. Manual Annotation:

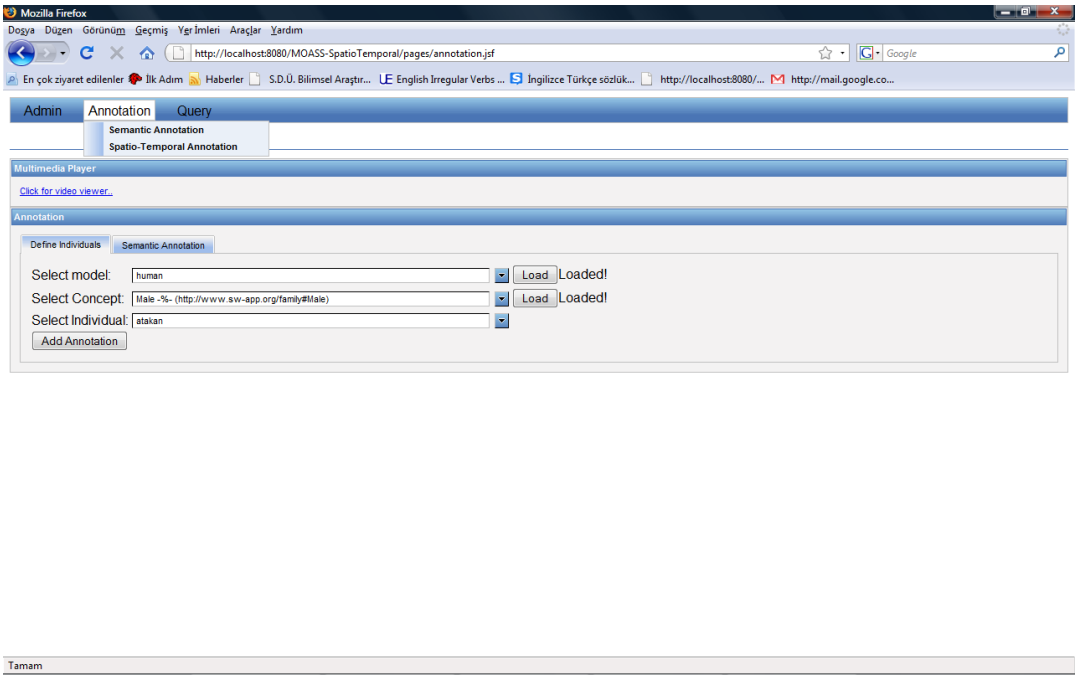


Figure 6.3: Annotation Panel of the OntoVMS

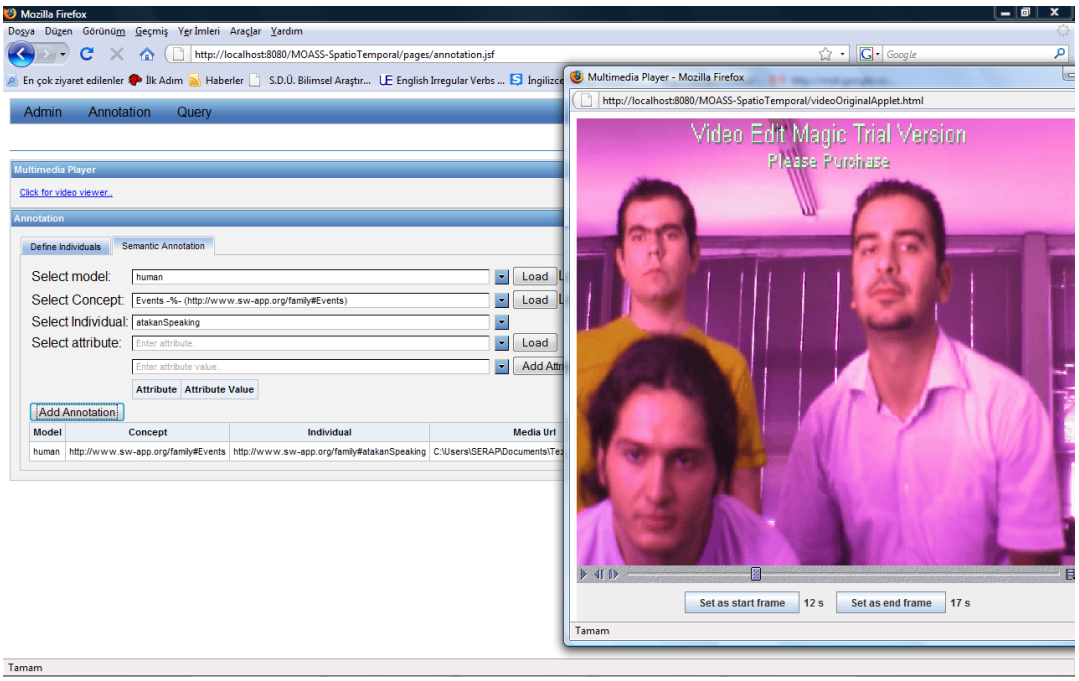


Figure 6.4: Semantic Annotation of OntoVMS

MBR representation is used for manual annotation in OntoVMS. Object names are selected from a combo box according to their ontological classes. In this way ontological querying can be performed.

4. Semi-Automatic Face Tracker Annotation:

   Face tracker is a semi-automatic annotation module, which can track the human faces in a video. It is developed in the Computer Vision and Intelligent Systems Research Laboratory. OntoVMS integrates this application into its annotation module. As shown in in the Figure 6.5 that, this application tracks the human faces in the video file. Red arrows display the orientation of the tracked human faces. The tracked annotations are used in the querying part of the OntoVMS.

Figure 6.5: Annotations of Face Tracker

6.3 Query Panel

The most complicated user interface of the OntoVMS is the query module, because it supports a wide range of query types. The query interface can be categorized into the following sections:

1. Simple Spatio-Temporal Query:

   The user can pose a simple spatial or temporal query by using this interface. Figure 6.6 shows a sample spatial query. In this query sample, the user asks “return all male class objects and their intervals which are to the left of atakan”. The result of the query consists of two data rows; the first result is “mehmet between the time interval 0.1 and 8.3 seconds” and the other result is “fatih between the time interval 0.1 and 18.0 seconds”. If the user presses the “Play” button which is provided at the end of each query result row, the system automatically plays these video intervals in the Video Applet.

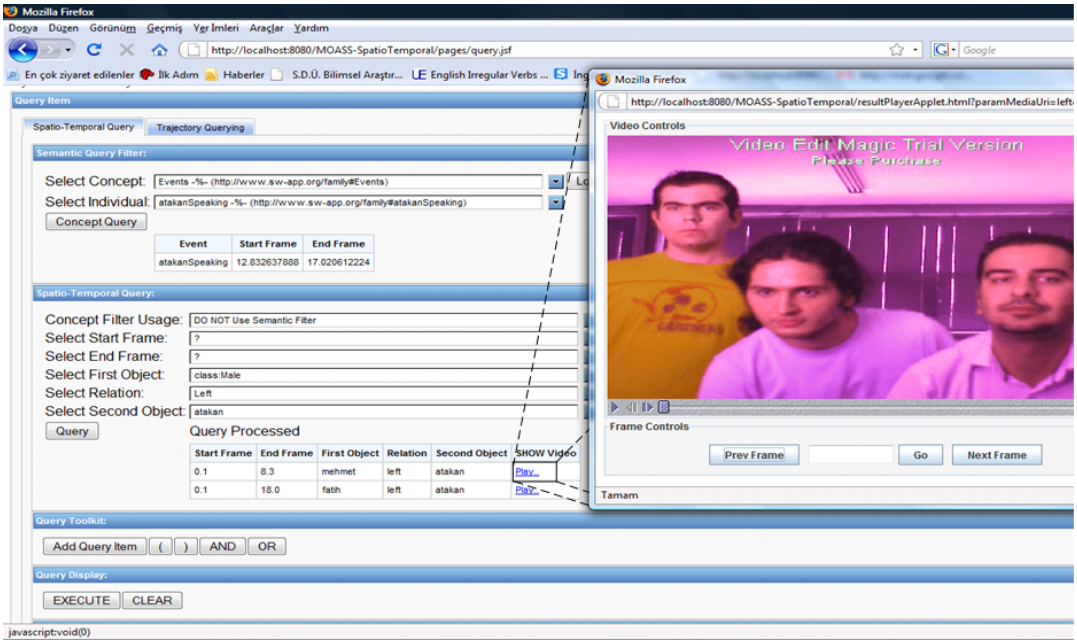


Figure 6.6: Simple Spatial Query Screen of the OntoVMS

Semantic filter usage is another opportunity for the users. For instance the same spatial query is processed with the semantic filter in Figure 6.7. This time the user chooses the “Use Semantic Filter” option, therefore results are changed according to the time interval of the event “atakanSpeaking”. New results are created by intersecting the intervals of the results of the spatial query and the results of semantic event query. Thus, the system returns only one solution for this query type; “fatih between the time interval 12.0 and 17.0 seconds”

The user can select “Appear” relation in order to see the appearance interval of the interested objects. For instance, in Figure 6.8 the user wants to see “all objects which

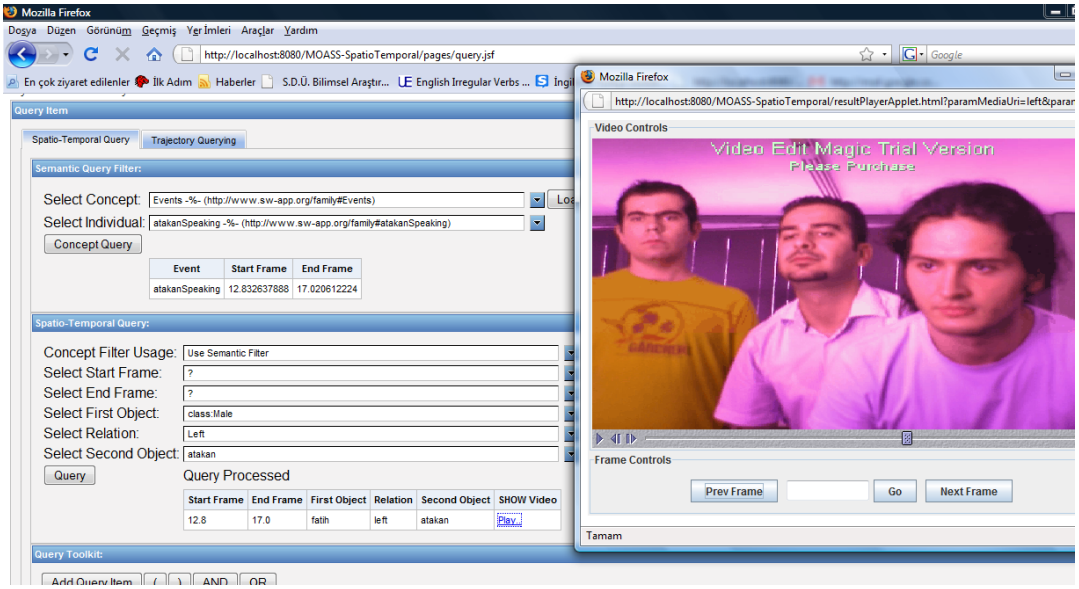


Figure 6.7: Simple Spatial Query with Semantic Filter

appeared in the screen between time interval 11.0 and 15.0". According to the user defined time interval, the system filters the appearance list and returns the correct answers.

2. Compound Spatio-Temporal Query:

   Compound query capability is another option for the users. Users can combine simple queries by using "AND", "OR" operators. For instance in Figure 6.9, the user combines two spatial relations; "left" and "right" relations, in order to make "between" query. In this example, two queries, namely "all male class objects which are to the right of fatih" and "all male class objects which are to the left of mehmet" queries are combined with "AND" operator. The system intersects the results of these queries and returns "atakan" object as a query result with time intervals 10.5 - 18.0.

3. Trajectory Query:

   OntoVMS supports three different trajectory query types (Figure 6.10). Source-Destination query type is used for finding trajectory of objects, if their start and end regions are selected. For instance in Figure 6.11, selected regions are well-matched for "mehmet" object trajectory, thus system returns "mehmet" as the query result. Another query type is direction query. In this query type objects can be searched according to their trajectory directions. Lastly, the user can ask the trajectory of an object of interest.

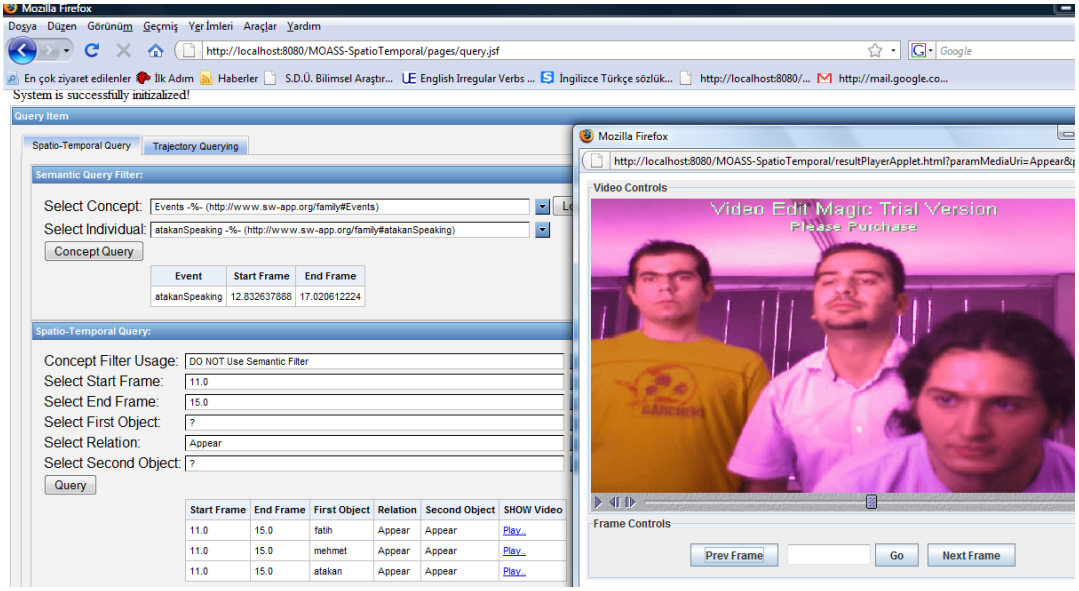


Figure 6.8: Appearance Screen of OntoVMS

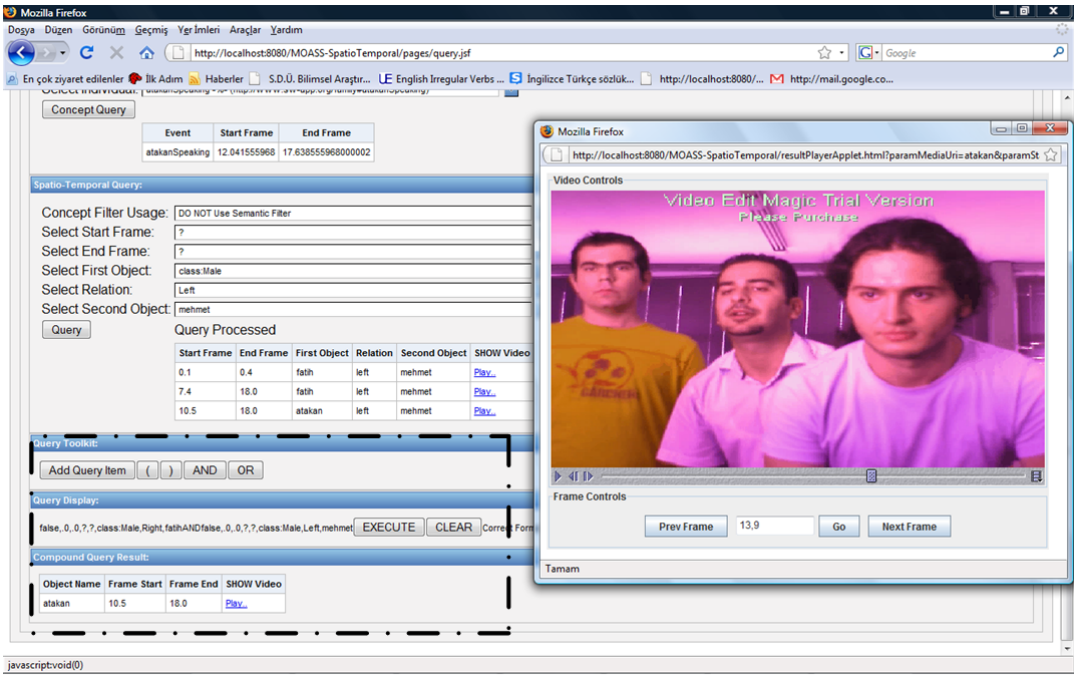


Figure 6.9: Compound Query String of the OntoVMS

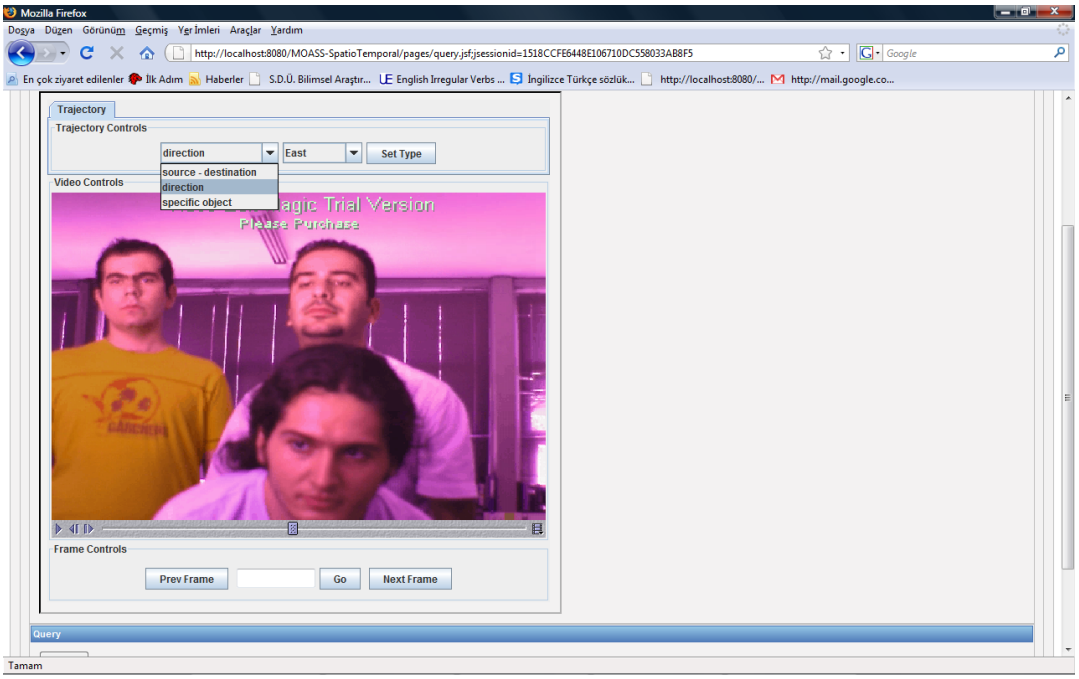


Figure 6.10: Trajectory Query Types of OntoVMS

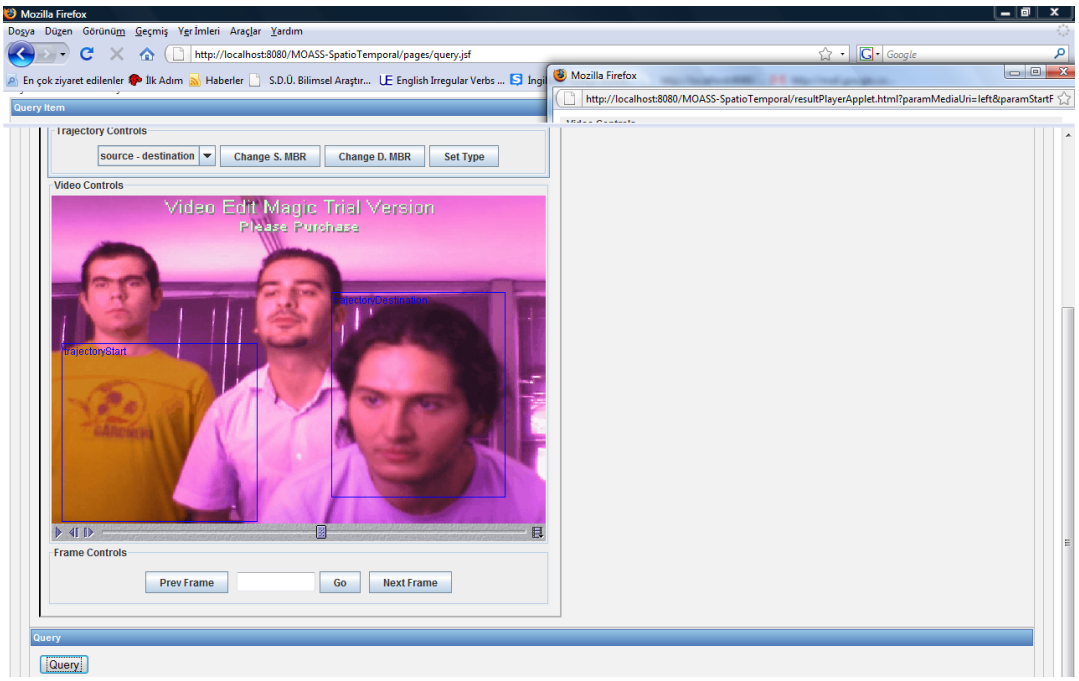


Figure 6.11: Source-Destination Trajectory Query of OntoVMS

CHAPTER 7

CONCLUSION

In this thesis we propose a framework for an ontology-based spatio-temporal video management system. This framework is developed as a solution to semantic video content retrieval problem. OntoVMS differs from the existing spatio-temporal querying systems; because it offers an ontological querying capability. SPARQL is used as the ontology query language in order to reach ontological classes and their individual values. The user can use any domain ontology with our system because this is a generic framework and for that reason it works domain independently. In this thesis, knowledge-base structure is used to store the extracted spatio-temporal relations. User queries are transformed to Prolog codes in order to process in JPL Prolog API. Moreover OntoVMS is a web application therefore web applets are used in its implementation.

This framework currently serves conceptual, spatial, temporal and trajectory querying capability. Current version of the OntoVMS does not support 3D relations; this capability can be added as a future work.

The framework supports videos only, however the architecture of this system is designed to support new attachments in the future. Image, Audio data and text data usage can be added to the system in order to enhance capabilities of fact extraction, video summarization and concept annotation.

OntoVMS uses MPEG-7 ontology and domain ontology to enhance spatio-temporal querying capability. Domain ontologies are uploaded and attached into MPEG-7 ontology automatically. It would be better if there is an ontology creation or modification user interface. Ontology management engine can be added as a future work in order enhances system capabilities.

Current version can be used for ontological concept annotation or spatio-temporal relation annotations. User friendly interfaces simplify user process but full-automatic or semi-automatic object extractor module would be a better solution. In the current version we use semi-automatic face extraction module to automatically recognize and annotate faces. This module can be extended to extract objects in addition to faces. If such a module exists then the system can support semi-automatic annotation.

In this thesis, user friendly query interfaces are used to pose queries to the system. However, speaking natural language query interface would be a better solution. A natural language processing engine can be added to enhance system capabilities.

To summarize, we have developed a functional framework which can be used for modeling and querying spatio-temporal and semantic content in videos. This system enhances its query capabilities by ontology support. New features can be added to the system, because this framework provides an infrastructure which allows extensions.

REFERENCES

[1] Sibel Adali, Kasim S. Candan, Kasim S. C, Su shing Chen, Kutluhan Erol, and V. S. Subrahmanian. Advanced video information system: Data structures and query processing. *Multimedia Systems*, 4:172–186, 1996.

[2] James F. Allen. Maintaining knowledge about temporal intervals. *Commun. ACM*, 26(11):832–843, 1983.

[3] Allen's Interval Algebra. http://en.wikipedia.org/wiki/Allens_Interval_Algebra, last visited on 22 Aug. 2009.

[4] Hayati Çam. Query processing for an mpeg-7 compliant video database. Master's thesis, Bilkent University, 2008.

[5] Ediz Şaykol. Web-based user interface for query specification in a video database system. Master's thesis, Bilkent University, 2001.

[6] Mehmet Emin Dönderler. *Data Modeling and Querying for Video Databases*. PhD thesis, Bilkent University, 2002.

[7] Mehmet Emin Dönderler, Ediz Saykol, Umut Arslan, Özgür Ulusoy, and Ugur Güdükbay. Bilvideo: Design and implementation of a video database management system. *Multimedia Tools and Applications*, 27:79–104, 2005.

[8] Mehmet Emin Dönderler, Özgür Ulusoy, and Ugur Güdükbay. A rule-based video database system architecture. *INFORMATION SCIENCES*, 143:13–45, 2002.

[9] Mehmet Emin Dönderler, Özgür Ulusoy, and Ugur Güdükbay. Rule-based spatiotemporal query processing for video databases. *The VLDB Journal*, 13:86–103, 2004.

[10] Mehmet E. Donderler, Ozgur Ulusoy, and Ugur Güdükbay. A rule-based approach to represent spatio-temporal relations in video data. In *In Video Data. ADVIS'2000 Proceedings, LNCS Vol. 1909, T. Yakhno (Ed*, pages 409–418. Springer-Verlag, 2000.

[11] Mehmet Emin Donderler, Ediz Saykol, Ozgur Ulusoy, and Ugur Gudukbay. Bilvideo: A video database management system. *IEEE Multimedia*, 10:66–70, 2002.

[12] Ersin Esen, Yağız Yaşaroğlu, Özgür Önür, Medeni Soysal, Serhat Tekinalp, and A.Aydın Alatan. Mpeg-7 uyumlu video yönetim sistemi. In *Sinyal İşleme ve İletişim Uygulamaları Kurultayı*, İstanbul, 2003.

[13] R. Garcia and O. Celma. Semantic integration and retrieval of multimedia metadata. Galway, Ireland, 2005.

[14] Ralf Hartmut Güting. An introduction to spatial database systems. *The VLDB Journal*, 3(4):357–399, 1994.

[15] Jane Hunter. Mpeg-7: Behind the scenes. Technical report, 1999.

[16] Jane Hunter. Adding multimedia to the semantic web - building an mpeg-7 ontology. In *In International Semantic Web Working Symposium (SWWS*, pages 261–281, 2001.

[17] Jane Hunter. Enhancing the semantic interoperability of multimedia through a core ontology . *IIEEE Transactions on Circuits and Systems for Video Technology, Special Issue on Conceptual and Dynamical Aspects of Multimedia Content Description*, 13:49–58, 2003.

[18] M. Koprulu, N.K. Cicekli, and A. Yazici. Spatio-temporal querying in video databases. In *Information Sciences*, pages 131–152, 2004.

[19] John Z. Li, M. Tamer Özsu, and Duane Szafron. Spatial reasoning rules in multimedia management systems. In *In Proc. International Conference on Multimedia Modeling*, pages 119–133, 1996.

[20] MPEG-7 Overview. http://www.chiariglione.org/mpeg/standards/mpeg-7/mpeg-7.htm, last visited on 25 Aug. 2009.

[21] Use Case: MPEG-7 metadata interoperability. http://www.w3.org/2005 /Incubator/mmsem/wiki/MPEG-7_metadata_interoperability_Use_case, last visited on 15 Aug. 2009.

[22] MPEG-7. http://en.wikipedia.org/wiki/MPEG-7, last visited on 26 Aug. 2009.

[23] Hakan Öztarak and Adnan Yazici. Structural and event based multimodal video data modeling. In *ADVIS*, pages 264–273, 2006.

[24] Dimitris Papadias, Nikos Mamoulis, and Vasilis Delis. Approximate spatio-temporal retrieval. *ACM Trans. Inf. Syst.*, 19(1):53–96, 2001.

[25] M. Perry, A.P. Sheth, F. Hakimpour, and P. Jain. Supporting complex thematic, spatial and temporal queries over semantic web data. pages 228–246, 2007.

[26] W. Ren, S. Singh, M. Singh, and Y. S. Zhu. State-of-the-art on spatio-temporal information-based video retrieval. *Pattern Recogn.*, 42(2):267–282, 2009.

[27] Shunting-yard algorithm. http://en.wikipedia.org/wiki/Shunting-yard_algorithm, last visited on 23 Aug. 2009.

[28] Hilal Tarakçı. An ontology-based multimedia information management system. Master's thesis, METU, 2008.

[29] R. Troncy, O. Celma, S. Little, R. Garcia, and C. Tsinaraki. Mpeg-7 based multimedia ontologies: Interoperability support or interoperability issue? Genova, Italy, 2007.

[30] Chrisa Tsinaraki, Panagiotis Polydoros, and Stavros Christodoulakis. Interoperability support for ontology-based video retrieval applications. In *In 3 rd International Conference on Image and Video Retrieval*, pages 582–591, 2004.

[31] Gulay Unel, Mehmet Emin Donderler, Ozgur Ulusoy, and Ugur Gudukbay. An efficient query optimization strategy for spatio-temporal. *Journal of Systems and Software*, 73:113–131, 2002.

[32] Sujal Subhash Wattamwar and Hiranmay Ghosh. Spatio-temporal query for multimedia databases. In *MS '08: Proceeding of the 2nd ACM workshop on Multimedia semantics*, pages 48–55, New York, NY, USA, 2008. ACM.

[33] Yakup Yildirim. *Automatic semantic content extraction in videos using a spatio-temporal ontology model*. PhD thesis, METU, 2009.

[34] Yakup Yildirim, Turgay Yilmaz, and Adnan Yazici. Ontology-supported object and event extraction with a genetic algorithms approach for object classification. In *CIVR '07: Proceedings of the 6th ACM international conference on Image and video retrieval*, pages 202–209, New York, NY, USA, 2007. ACM.

APPENDIX A

SPATIAL INFERENCE RULES

A.1 List of Spatial Inference Rules

These rule sets are used in reasoning mechanism of OntoVMS Knowledge-Base structure. Following rules are adapted from [8].

Rule Set 1.1 *(Inverse property) The relations west, north, north-west, north-east, right and above are inverses of east, south, south-east, south-west, left and below, respectively.*

(a) $west(A, B) \Leftrightarrow east(B, A)$

(b) $north(A, B) \Leftrightarrow south(B, A)$

(c) $north - west(A, B) \Leftrightarrow south - east(B, A)$

(d) $north - east(A, B) \Leftrightarrow south - west(B, A)$

(e) $right(A, B) \Leftrightarrow left(B, A)$

(f) $above(A, B) \Leftrightarrow below(B, A)$

Rule Set 1.2 *(Transivity) If* $ß \in S$*, where S is the set of directional relations, then*

$ß(A, B) \wedge ß(B, C) \Rightarrow ß(A, C)$

Rule Set 1.3 *The relations right, left, above and below can be expressed by other directional relations.*

(a) $east(A, B) \vee north - east(A, B) \vee south - east(A, B) \Leftrightarrow right(A, B)$

(b) $west(A, B) \vee north - west(A, B) \vee south - west(A, B) \Leftrightarrow left(A, B)$

(*c*) $north(A,B) \vee north-east(A,B) \vee north-west(A,B) \Leftrightarrow above(A,B)$

(*d*) $south(A,B) \vee south-east(A,B) \vee south-west(A,B) \Leftrightarrow below(A,B)$

Rule Set 2.1 *(Inverse Property). The relations inside and cover are inverses of contains and covered-by, respectively.*

(*a*) $inside(A,B) \Leftrightarrow contains(B,A)$

(*b*) $cover(A,B) \Leftrightarrow covered-by(B,A)$

Rule Set 2.2 *(Reflexivity). The relations equal and overlap are reflexive.*

(*a*) $equal(A,A)$

(*b*) $overlap(A,A)$

Rule Set 2.3 *(Symmetry). The relations equal, overlap, disjoint and touch are symmetric.*

(*a*) $equal(A,B) \Leftrightarrow equal(B,A)$

(*a*) $overlap(A,B) \Leftrightarrow overlap(B,A)$

(*a*) $disjoint(A,B) \Leftrightarrow disjoint(B,A)$

(*a*) $touch(A,B) \Leftrightarrow touch(B,A)$

Rule Set 2.4 *(Transitivity). The relations inside and equal are transitive.*

(*a*) $inside(A,B) \vee inside(B,C) \Leftrightarrow inside(A,C)$

(*b*) $equal(A,B) \vee equal(B,C) \Leftrightarrow equal(A,C)$

Rule Set 2.5. *The relations inside, equal and cover imply the relation overlap.*

(*a*) $inside(A,B) \Rightarrow overlap(A,B)$

(*b*) $equal(A,B) \Rightarrow overlap(A,B)$

(*c*) $cover(A,B) \Rightarrow overlap(A,B)$

APPENDIX B

TEMPORAL INFERENCE RULES

B.1 List of Temporal Inference Rules

These rule sets are used in reasoning mechanism of OntoVMS Knowledge-Base structure. According to Allen algebra following temporal inference rules can be defined.

Rule 1 : *(Inverse property) The relations precedes, meets, overlaps, finished-by contains and starts are inverses of preceded-by, met-by, overlapped-by, finishes, during and started-by respectively.*

(a) $precedes(A, B) \Leftrightarrow preceded - by(B, A)$

(b) $meets(A, B) \Leftrightarrow met - by(B, A)$

(c) $overlaps(A, B) \Leftrightarrow overlapped - by(B, A)$

(d) $finished - by(A, B) \Leftrightarrow finishes(B, A)$

(e) $contains(A, B) \Leftrightarrow during(B, A)$

(f) $starts(A, B) \Leftrightarrow started - by(B, A)$

Rule 2 *(Transivity) If* $ß \in S$*, where S is the set of precedes, preceded-by, finishes, finished-by, during, contains, starts, started-by and equal, then*

$ß(A, B) \wedge ß(B, C) \Rightarrow ß(A, C)$