SEMI-AUTOMATIC SEMANTIC VIDEO ANNOTATION TOOL

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

BY

MERVE AYDINLILAR

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

DECEMBER 2011

Approval of the thesis:

SEMI-AUTOMATIC SEMANTIC VIDEO ANNOTATION TOOL

submitted by **MERVE AYDINLILAR** 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. Adnan Yazıcı _____
Head of Department, **Computer Engineering**

Prof. Dr. Adnan Yazıcı _____
Supervisor, **Department of Computer Engineering, METU**

Examining Committee Members:

Assoc. Prof. Dr. Ahmet Coşar
Department of Computer Engineering, METU _____

Prof. Dr. Adnan Yazıcı
Department of Computer Engineering, METU _____

Assist. Prof. Dr. Murat Koyuncu
Department of Information Systems Engineering, Atılım University _____

Assist. Prof. Dr. Mustafa Sert
Department of Computer Engineering, Başkent University _____

Dr. Dilek Küçük
TÜBİTAK UZAY _____

Date: _____

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: MERVE AYDINLILAR

Signature :

ABSTRACT

SEMI-AUTOMATIC SEMANTIC VIDEO ANNOTATION TOOL

Aydınlılar, Merve
M.Sc., Department of Computer Engineering
Supervisor : Prof. Dr. Adnan Yazıcı

December 2011, 58 pages

Semantic annotation of video content is necessary for indexing and retrieval tasks of video management systems. Currently, it is not possible to extract all high-level semantic information from video data automatically. Video annotation tools assist users to generate annotations to represent video data. Generated annotations can also be used for testing and evaluation of content based retrieval systems. In this study, a semi-automatic semantic video annotation tool is presented. Generated annotations are in MPEG-7 metadata format to ensure interoperability. With the help of image processing and pattern recognition solutions, annotation process is partly automated and annotation time is reduced. Annotations can be done for spatio-temporal decompositions of video data. Extraction of low-level visual descriptions are included to obtain complete descriptions.

Keywords: video annotation, MPEG-7, semi-automatic annotation

ÖZ

YARI OTOMATİK ANLAMSAL VİDEO BETİMLEME ARACI

Aydınlılar, Merve
Yüksek Lisans, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi : Prof. Dr. Adnan Yazıcı

Aralık 2011, 58 sayfa

Videoların anlamsal olarak betimlenmesi otomatik video yönetim sistemlerinin dizin ve erişim görevlerinin yerine getirilmesi açından önem taşımaktadır. Gelinen noktada, video verisindeki tüm anlamsal verinin otomatik olarak çıkarılması mümkün değildir. Video betimleme araçları kullanıcıların video verisini betimleyen açıklamalar oluşturmalarına yardımcı olur. Oluşturulan betimlemeler, içerik tabanlı video erişimi sağlayan sistemlerin test edilmesinde ve değerlendirilmesinde de kullanılır. Bu çalışmada, yarıotomatik anlamsal video betimleme aracı sunulmuştur. Bu aracın ürettiği betimlemelerin, farklı sistemlerde de kullanılabirliğinin sağlanması için uluslararası bir standart olan MPEG-7 formatı kullanılmıştır. Görüntü işleme ve örüntü tanıma tekniklerinin yardımıyla, betimleme işlemi kısmen otomatikleştirilerek, betimleme için gereken süre kısaltılmıştır. Betimlemeler video verisinin uzamsal ve zamansal bölümlerini tanımlayacak şekilde yapılabilmektedir. Alt düzey görsel öznitelikler betimlemelerin bütünlüğünün sağlanması amacıyla eklenmiştir.

Anahtar Kelimeler: video betimlenmesi, MPEG-7, yarı otomatik betimleme

To my friends and family

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my supervisor Prof. Dr. Adnan Yazıcı for his support and encouragement during this study. I would like to thank Dr. Dilek Küçük, Assoc. Prof. Dr. Ahmet Coşar, Assist. Prof. Dr. Murat Koyuncu, Assist. Prof. Dr. Mustafa Sert and Turgay Yılmaz for attending examining committee and reviewing my thesis.

I am truly indebted and thankful to my friends Aykut Kızıcı, Okan Akalın, Utku Erdoğu, Kerem Hadımlı, Elvan Gülen, Can Hoşgör, Kazım Işık, Tuba Demirtaş, Görkem Demirtaş, Nazif İlker Erçin, Uygar Yüzsüren, Can Eroğul, Selma Süloğlu and Aslıhan Örüm. This thesis would not have been possible without them.

I am grateful to my colleagues and friends, Özgür Kaya, Gökdeniz Karadağ, Hüsnü Yıldız and İlkcan Keleş for their support and encouragement.

I would like to thank my parents, my sister Melike, and my grandparents for their interest and best wishes.

Finally I would like to thank TÜBİTAK BİDEB for financially supporting this study.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

Advances in information technology pave the way for creation and consumption of large amounts of video data. While reduced costs of hardware and publicly available video editing software turn every individual into a potential video shooter, broadband Internet connections and social networking services make created video widely accessible. In addition to these, professionally created content like news reports, TV series, music videos, etc. are digitally available. This huge amount of information is useless if it can not be presented to users effectively. However, querying and retrieval of video content is much more complex than textual data. Currently video content analysis systems can successfully detect certain concepts and objects, but for general domains success rates are usually very low. At this point, manual annotation of video is necessary for intelligent content management services. Also, automatic content analysis systems need manual annotations for training on content descriptions and comparing results with ground truth. Video annotation tools provide necessary environment for annotators to create video content descriptions.

Video annotation tools take video data as input and with aid of the annotator, create video content descriptions. These content descriptions can be both low-level and high-level. Low-level descriptions can be extracted from video data directly without user assistance. Commonly used low-level descriptions are color and texture descriptions. Low-level descriptions are important for multimedia content management systems to support query-by-example. High-level descriptions are related to semantic information about the video content and usually can not be generated automatically. Objects and events in video content are examples of high-level descriptions. Annotator can use free text, keywords, predefined vocabulary of application domain or ontology for semantic annotations. Systems that support query-by-keyword need this kind of annotation for semantic querying of video content.

Video content descriptions should be interoperable in order to exchange descriptions between systems and reuse annotated data. MPEG-7 [1] is a widely used, comprehensive multimedia metadata description standard that can be used to solve interoperability issues of video annotations.

In order to describe video content effectively, video should be decomposed into structural units. Video annotation tools may support temporal and spatial decompositions to annotate video content in fine detail.

Manual annotation is time consuming and annotation can be automated in several ways to reduce annotation time. But the results of these automated tools should be editable by the annotator to generate true descriptions.

In this study, a comprehensive semi-automatic semantic video annotation tool is implemented. Main features of the annotation tool can be listed as follows:

- Annotation tool supports both low-level descriptions and high-level semantic descriptions.
- Low-level visual descriptions can be extracted both for whole frame and for a selected region in a frame.
- Video is temporally decomposed into shots; shots are temporally decomposed to key frames and spatio-temporally to moving and still regions. Semantic annotations are allowed at each level.
- Temporal decomposition of video into shots is done automatically and once it is done it can be loaded for further annotations.
- Generated descriptions by annotation tool is in MPEG-7 format and specified by Detailed Audio Visual Profile (DAVP) to ensure interoperability.
- To reduce annotation time, object recognition and object tracking are included. Once an object is selected and annotated in a shot, it will be tracked till the end of the shot, and for other shots object will be re-detected (recognized) and annotated automatically.
- Since human face is always important for video content, faces are detected with a face detection algorithm, and annotated with spatio-temporal information and visual descriptors.

- In order to make semantic annotations faster and use relations between concepts in annotation, user defined Web Ontology Language (OWL) ontologies that describe application domain can be loaded and used.

Organization of the rest of this thesis is as follows. Chapter 2 reviews existing video annotation tools, Chapter 3 briefly describes MPEG-7 standard, Chapter 4 explains how video is represented for description, Chapter 5 presents each module in the annotation tool, Chapter 6 explains how annotation tool differs from other annotation tools and Chapter 7 concludes this study with a discussion of possible future extensions.

CHAPTER 2

RELATED WORK

2.1 Semantic Video Annotation Tools

Currently most of the high-level concepts and related semantic features in videos can not be extracted automatically, and extraction of these kinds of information is not in the scope of standards like MPEG-7 [1]. Manual or semi-automatic semantic annotation tools are developed to annotate multimedia content. Basically, semantic video annotation tools take video as input, provide necessary tools to user for annotation, and output metadata about annotated video. In order to be used by content retrieval systems, created metadata should be structured, reusable, descriptive, fine grained and able to describe video content at any level of detail.

According to a survey performed on semantic annotation tools [2], the criteria to evaluate and compare semantic annotation tools can be divided into three categories.

- Input & Output
  - Annotation Vocabulary: Annotations can be done either with predefined lexicon or ontology, or keywords and free text assigned by user at annotation time.
  - Metadata Format: representation format of annotation.
  - Content Type: supported video formats.
- Annotation Level
  - Metadata Type: divided into four types, content descriptive metadata (semantic information), structural metadata (spatio-temporal decomposition), media metadata (low-level features), administrative (annotation creation time, annotator)

  - Granularity: division of annotation content, it may be entire video, shots, frames, regions within frames, or moving regions.
  - Localization: manual drawing and/or segmentation
  - Annotation Expressivity: concepts and/or relation between concepts.

- Miscellaneous
  - Application Type: web based or stand alone.
  - License: kind of license.
  - Collaboration: supports collaborative annotation by multiple users or not.

2.2 Existing Semantic Video Annotation Tools

This section gives a brief overview of existing video annotation tools.

VIA (Video Image Annotation Tool)[1], is developed by MK-lab[2]. VIA is a stand-alone application. Recently it has become an open source project. Annotation level of VIA is entire video, shot, frame and region. It supports live video annotation and frame by frame annotation. Free text annotation is enabled. It supports MPEG-1 and MPEG-2 video formats. Created metadata is saved in XML or text.

VideoAnnEx [3] is an MPEG-7 video annotation tool developed by IBM. Annotation level of VideoAnnEx is entire video, shot, key frames and regions in key frames. Annotation vocabulary is predefined and includes user loaded lexicon and free text. It supports automatic shot detection and key frame extraction. Once shot detection and key frame extraction are done, it can be loaded whenever it is needed. Another property of the tool is that, it can find frames similar to the annotated frame, so the overall annotation time is decreased. VideoAnnEx is also a stand alone application with MPEG-1 and MPEG-2 video format support.

Ontolog [4] is developed by Jon Heggland from Norwegian University of Science and Technology. Annotation vocabulary of Ontolog is user defined Resource Description Framework Schema (RDFS) ontologies. Created metadata format is Resource Description Framework

[1] http://mklab.iti.gr/via/
[2] http://mklab.iti.gr

(RDF). Annotation level of Ontolog is entire video and temporal video segments. It is a stand alone Java application.

Advene [5] is developed by LIRIS laboratory[3]. Advene allows users to define structure of the annotation to be created, in order to meet user specific needs. Advene supports a package system that contains metadata and media content itself. Annotation level is entire video and temporal video segments. Created annotations are saved in XML format. Advene is an open source stand alone project.

Elan [6] is developed by Max Plank Institude for Psycholinguistics, Nijmegen, Netherlands. Elan is a linguistic oriented tool so its annotation vocabulary is wide, it separates annotations to sentence, word or gloss, comment, translation or description of features of the content. Elan supports multi-layer annotation, where each layer is called a tier. Tiers can be in any language so, the tool supports multi-language in the same description. Created annotations are saved in XML format.

SVAS [7] is developed by Joanneum Research Institute of Information Systems and Information Management[4]. SVAS creates annotations in MPEG-7 format. It is composed of two separated tools: Media Analyzer and SVAT(Semantic Video Annotation Tool). Media Analyzer part extracts information related to shots and key frames automatically, without user interaction and the created metadata is stored in MPEG-7 format. Annotation level is entire video, shots, frames and regions. For region annotation, automatic segmentation or manual drawing can be used. Additionally, once an object is selected in a frame, SVAT searches the object in the entire video, and labels its location in discovered frames. This approach decreases manual annotation time. Semantic annotations are described by semantic description scheme of MPEG-7.

[3] http://liris.cnrs.fr
[4] http://www.joanneum.at/en/jr.html

CHAPTER 3

MPEG-7 STANDARD

3.1 Introduction

MPEG-7 [8], formally called Multimedia Content Description Interface, is an ISO/IEC standard that describes multimedia information representation. MPEG-7 is developed by Moving Pictures Experts Group, which has published widely used standards such as MPEG-1, MPEG-2 and MPEG-4. While former standards describe multimedia content itself, MPEG-7 describes metadata of multimedia content.

MPEG-7 has very wide application areas, including digital libraries, multimedia editing and cultural services. MPEG-7 descriptions can be used stand-alone without media content itself, or as links to the content. Many data types can be associated with MPEG-7, such as audio, image, graphics, video, etc. MPEG-7 descriptions are independent of the medium that stores multimedia content, which may be either paper or hard disk. MPEG-7 uses object-based modeling for multimedia description, and all described objects is accessible from content. MPEG-7 can describe any content regardless of its format. Analogue content can be described as well as digital one. Digital data may be compressed or uncompressed. Any compression approach and format can be used. MPEG-7 supports different abstraction descriptions for different levels of information. While abstraction of low-level features includes statistical features, abstraction of high-level features is related to semantic descriptions.

The scope of MPEG-7 is standardizing format of description and its decoding. Feature extraction techniques, encoding, usage of the description, content analysis tools are not part of the standard. Beside these, MPEG-7 is extensible, description tools can be extended in a standard way, for application specific purposes.

MPEG-7 has four different tools for representing media content:

- Descriptors: Descriptors are used to represent features. A feature is represented by one or more descriptors. Color layout is an example for color descriptor.
- Description Schemes: Description schemes consist of descriptors and other description schemes.
- Description Definition Language: Language that allows users to define application domain specific description schemes and descriptors.
- System Tools: Tools that are used for binarization, storage, transportation of descriptors.

Currently, MPEG-7 standard consists of twelve parts:

- Systems: framework of the standard, textual and binary formats of descriptions.
- Description Definition Language: specifies MPEG-7 schema language.
- Visual: descriptors and description schemes that deal with visual data only.
- Audio: descriptors and description schemes that deal with audio data only.
- Multimedia Description Schemes: descriptors and description schemes for general data.
- Reference Software: reference implementation of standard tools.
- Conformance Testing: standardize test procedures for preceding parts of the standard.
- Extraction and Use of MPEG-7 Descriptions: a technical report about extraction and usage of description tools.
- Profiles and Levels: collection of standard profiles and levels.
- Schema Definition: standard schema definition of description tools.
- MPEG-7 Profile Schemas: schema definitions of the standard profiles.
- Query Format: syntax and semantics of query format tools.

3.2 Description Definition Language

MPEG-7 Description Definition Language (DDL) [9], is the core part of the MPEG-7 standard. DDL is a schema definition language which is used for defining MPEG-7 documents and validation of MPEG-7 descriptors, description schemes and descriptions. Besides defining MPEG-7 description schemes and descriptors, DDL allows users to create their own description definitions and schemas by using existing descriptors and description schemes.

DDL has been developed for expressing MPEG-7 schemas, which are used for defining class of MPEG-7 documents. MPEG-7 documents are basically XML documents that can be validated by MPEG-7 schemas. DDL is composed of

- XML Schema Structural Components
- XML Schema Data Types
- MPEG-7 Specific Extensions

The most important XML schema structural components for MPEG-7 are namespaces, element declarations, attribute declarations, type definitions, and group definitions. Namespaces are used to uniquely identify descriptors and description schemes in a MPEG-7 schema. It is mandatory for MPEG-7 schemas to contain namespace information. Element and attribute declarations provide elements and attributes specific names and types in documents. Type definitions are used to create new types based on existing simple and complex types. Creating and naming of attribute and element groups are specified by group definitions.

XML schema data types are

- Built-in primitive data types: examples are string, double, duration etc.
- Built-in derived data types: data types derived from primitive data types with some constraints, such as negative integer, long int etc.
- Facets: constraints that can be applied for type derivation, examples are numeric facets and length facets.
- List data type: sequences of atomic types

- Union data type: a data type selected from a set of types

MPEG-7 specific extensions are required for XML Schema to met MPEG-7 DDL requirements. These extensions are

- Array and matrix data types
- *basicTimePoint* and *basicDuration* built-in derived types

Array and matrix data types are added to XML Schema to restrict sizes of multidimensional arrays in schema definition and at time of instantiation. Extension to XML Schema, *dimension* facet, restricts size of multidimensional arrays by definining fixed size arrays. *dim* attribute on the other hand, parametrizes sizes of multidimensional arrays, so sizes of multidimensional arrays can be determined at instantiation time. *basicTimePoint* data type specifies a time point and *basicDuration* data type specifies duration of a time interval.

3.3 Multimedia Description Schemes

MPEG-7 description schemes enable constitution of compound descriptions by combining descriptors and description schemes. MPEG-7 Multimedia Description Schemes [10], which are part of the MPEG-7 standard, deal with audio-visual and textual data. The difference between multimedia descriptors and description schemes is, while descriptors describe low-level features of multimedia such as color and texture; description schemes describe high-level features like objects and events. Multimedia Description Schemes can be divided into six parts: Basic Elements, Content Description, Content Management, Content Organization, Navigation and Access, and User Interaction. Content Organization, Navigation and Access, and User Interaction are not directly related to the annotation tool, other parts are explained in the following sections.

3.3.1 Basic Elements

Basic elements of multimedia description schemes are schema tools, linking and media localization, basic data types, and basic description tools. These basic elements are fundamental constructs of MPEG-7 description schemes.

3.3.1.1 Schema Tools

Schema Tools differs from other basic elements in a way that while other basic elements are related to describing multimedia content, schema tools deal with creation and management of multimedia descriptions. MPEG-7 root and top-level elements are the most important schema tools. Every valid MPEG-7 document must contain MPEG-7 root element as document root. Top-level elements are directly attached to root element, and each of them describes different description tasks. Content Entity Description, Content Abstraction Description and Content Management are important tasks described by top-level elements.

3.3.1.2 Basic Data Types

In addition to types defined by MPEG-7 DDL, Basic Data Types are defined to meet multimedia content description requirements. These data types are *integers* and *reals* with constrained values, *vectors* and *matrices* with arbitrary dimensions, *probabiltyvectors* and *matrices* that represent probability distributions, and *strings* that are codes of commonly used elements like currency.

3.3.1.3 Linking and Localization Tools

Linking and Localization Tools provide a linking mechanism between MPEG-7 descriptors and multimedia content and a time representation within multimedia content.

- *References* provide reference mechanisms between descriptions with *Reference* data type. This data type accomplishes this task in three ways: *idref*, within the same document by using ID attribute of description, *xpath*, by descriptions path in the document tree, *href*, by using URI, this type can refer to a description among documents.
- *UniqueIdentifiers* identify multimedia content or descriptions with UniqueID data type.
- *MediaLocators* link multimedia descriptor to multimedia content with media locator element of descriptor. Types of media locators are *MediaLocator*, *TemporalSegment Locator*, and *ImageLocator*. *MediaLocator* links multimedia content to its physical lo-

cation by an URI. *TemporalSegmentLocator* specifies a segment within audio or video, either using start and stop time points or segment's offset and length. *ImageLocator* marks a frame in a video using time points.

- *Time* specifies temporal information. Both media time and world time can be represented. *mediaTimePoint* data type defines a time instance while *mediaDuration* data type defines a time duration. MPEG-7 time representations are
  - *SimpleTime* which represents a time point,
  - *RelativeTime* which represents a time point relative to a time base,
  - *IncrementalTime* which specifies a time period of equally divided time intervals.

3.3.1.4 Basic Tools

Basic tools are primitive MPEG-7 tools that can be used to form composite tools. Some of these basic tools described below:

Relations and Graphs Relation description scheme defines relations between descriptions. Relations are directed and each description in the relation is classified as source or target. Fuzzy relations are supported by assigning values to relations. Graph description scheme is used for representing graphs. Node data type represents nodes of the graph and edges are represented with relation elements.

Text Annotation Text annotation describes media content with natural language. MPEG-7 specifies four kinds of annotation within Text annotation data type, which are

- Free Text: Free text annotations are unstructured, human readable annotations. Language of these annotations can be specified by *xml* : *lang* attribute
- Keyword Annotation: Since it is hard to process natural languages by computer, important words of text are selected as keywords, and Keyword annotation data type represents these keywords.
- Structured Annotation: Structured Annotation aims to capture semantic structure of text annotation by using who, whatobject, whataction, where, when, why and how elements.

- Dependency Structure: Dependency Structure represents linguistic structure of text annotation with a dependency tree.

Classification Scheme Since it is impossible to define every domain within MPEG-7, Classification scheme is used to define domain vocabularies as set of terms and other classification schemes. Classification scheme consists of Term elements which holds TermID as attribute and Name elements.

People and Places Description tools for people and places are

- Agents: All entities that can be subject of an action are defined as agents.
  - Person: Real or fictional person described with name, affiliation and address information.
  - PersonGroup: Group of people which group members can be identified and have a common title.
  - Organization: Group of people which group members can not be easily identified and have a common title.
- Places: Existing, historical or fictional places described with name, role, geographic position, country, region and postal address.

Affective Response Affective response describes viewers response to media. These responses can be emotional like excitement, anger etc. and described by relative numeric values. Affective responses are evaluated within video segments, so can be used for constructing story shape of a video.

Ordering Descriptions Ordering descriptions define ordering of media data on some criteria called OrderingKey. Ordering descriptions has Selector element that defines data element to be ordered and Field element describes the criteria.

3.3.2 Content Management

Content Management tools describe media content in terms of its media information, creation information and usage information. The same content can be captured more than once using different tools, or with different encodings. Each of these records corresponds to different content entities. Also each content entity can have different copies in different formats and encodings, which forms a media profile of the content entity. Original media is called master profile and each copy with different formats and encoding is a new profile.

3.3.2.1 Media Information

Media Information describes multimedia features with MediaInformation description scheme which consists of MediaIdentification Descriptor (unique id and information about content entity) and MediaProfile description schemes. MediaProfile description scheme consists of these description units:

- *MediaFormatDescriptor* represents coding parameters and format.
- *MediaInstanceDescriptionscheme* describes different instances of the media profile by locating and identifying with an id.
- *MediaTranscodingHintsDescriptor* contains hints for transcoding.
- *MediaQualityDescriptor* rates the media profile by its quality.

3.3.2.2 Content Creation

Content Creation describes the information related to generation of the content entity by CreationInformation description scheme. This description scheme includes Creation description scheme (creation place, date etc.), Classification description scheme (subject, genre etc.) and RelatedMaterial description scheme.

3.3.2.3 Content Usage

Usage information of the content entity is described by UsageInformation description scheme. This description scheme contains Rights data type (right holders and access rights), Financial data type (costs and income), Availability description scheme (access availability), and UsageRecord description scheme (past use of the content).

3.3.3 Content Structure

MPEG-7 Content Structure tools describe segment entities and relations between them. A segment entity is spatial, temporal, or spatio-temporal units of multimedia content. Segment entities can be connected spatially and temporally. Table 3.1 gives an overview of some important Segment entities.

Table 3.1: MPEG-7 segment entities

StillRegion	Spatial region of 2D image or frame
VideoSegment	Group of frames in a video
MovingRegion	A spatiotemporal region in a video segment
AudioSegment	An interval of audio sequence
AudioVisualSegment	Both audio and video in same interval
AudioVisualRegion	Spatiotemporal segment of audiovisual data
Mosaic	Mosaic/Panaromic view
StillRegion3D	3D region of a 3D image
ImageText	Still text region
VideoText	Moving text region
InkSegment	Ink content segment
Multimedia Segment	Compound segments
AnalyticClip	Shot from video
AnalyticTransitions	Transitions between shots

Properties of segment entities, such as creation information, color and audio features, can also be described by MPEG-7, like other media contents. Beside these, Mask Descriptor describes spatio-temporal connectivity between separated segments and MatchingHintDecriptor represents the importance of description for a segment entity.

For disconnected segments, SegmentDecomposition description scheme describes decomposition of segment entity into subsegments. SpatialSegmentDecomposition, TemporalSeg-

mentDecomposition, SpatioTemporalSegmentDecomposition and MediaSourceDecomposition description schemes are decomposition tools for segment entities. Subsegments are not required to span the whole segment, with SegmentDecomposition it can be stated whether there are gaps between subsegments or not.

Relations between segment entities are described by SpatialRelation Classification Scheme and TemporalRelation Classification Scheme. SpatialRelation represents spatial relations like left, right, above etc. TemporalRelation represents temporal relations like follows, overlaps, precedes etc.

3.3.4 Content Semantics

MPEG-7 describes media content in terms of semantic entities, attributes of semantic entities and relationships between them. Media contents consist of one or more semantic units which are called narrative worlds. These non-overlapping narrative worlds cover everything related to describe semantics of the content, background, context etc. Narrative worlds are described by Semantic description scheme.

Abstraction modeling is necessary for semantic descriptions. In this context, abstraction is generalizing instances to higher level concepts or classes. In MPEG-7, there are two kinds of abstraction, namely media abstraction and formal abstraction. Media abstraction is taking a media instance's semantic description and using this description to describe similar media. News bulletins on different channels are example to media abstraction. Formal abstraction is substituting instances in the description to more general classes. These substitutions can be more than one level. For example Anna can be abstracted as any woman and any woman can be abstracted as any person. AbstractionLevel data type represents the type of abstraction. AbstractionLevel data type has dimension attribute. For media abstraction, dimension of AbstractionLevel is zero. For formal abstraction dimension of AbstractionLevel is one or more. Higher abstraction dimensions are used for abstractions of abstractions.

Semantic Entities are all kinds of semantic descriptions such as events, objects, places etc. These semantic entities and relationships between them are elements of semantic description scheme. Description schemes for semantic entities are the following ones:

- Object: describes objects in narrative worlds.

- AgentObject: describes an object in action as a person.
- Event: describes event in a narrative world.
- SemanticPlace: describes a place in narrative world, can be fictional.
- SemanticTime: describes time in narrative world with duration.
- SemanticState: represents information about narrative worlds state.
- Concept: collections of properties.

Semantic Attributes describes features of semantic entities of multimedia data. SemanticBase description scheme's elements are Label, Definition, Property and Media Occurrence Label element assigns label to semantic entity, Definition element contains textual information about semantic entity, Property element involves adjectives that describes semantic entity, and Media Occurrence element locates semantic entity spatio-temporally in media content.

Semantic Relation tools represents relations between semantic entities. Some of the semantic relations described by MPEG-7 are similar, opposite, resultOf, representedBy, influences, etc. Semantic relations may describe localization of semantic entities, how they relate to the story, how they relate to each other. Also with SpatialRelation CS and TemporalRelation CS, semantic entities spatio-temporal relations (left, precedes) with other semantic entities can be described.

3.4 Visual Descriptors

Visual information of audiovisual content is described by MPEG-7 visual descriptors [8]. There are two kinds of visual descriptors: general and domain specific. While general visual descriptors include color, shape, texture descriptors; face recognition descriptor is a domain specific descriptor. Visual descriptors are widely used for image and video retrieval applications.

3.4.1 Color Descriptors

Color descriptors [11] are important visual descriptors and widely used for image and video retrieval applications. Such applications select these color descriptors according to their ability to reflect characteristics of the described region, complexity of the extraction process to obtain descriptor, size of the descriptor, scalability and interoperability of the descriptor, etc. Table 3.2 lists MPEG-7 color descriptors.

Table 3.2: MPEG-7 color descriptors

Color Space Descriptor	Describes color spaces, which are • HSV (Hue Saturation Value) • HMMD (Hue-Min-Max-Diff) • RGB • YCbCr • Monochrome
Color Quantization Descriptor	Partitioning of given color space into discrete bins
Dominant Color Descriptor	Represents small number of dominant colors with their statistical information
Scalable Color Descriptor	HSV Color histogram encoded with Haar Transform
Group of Frames or Group of Pictures Descriptor	Aggregation of scalable color descriptor of a group of pictures or video frames
Color Structure Descriptor	Localized HMMD color histogram
Color Layout Descriptor	Grid-based spatial layout of representative colors

3.4.1.1 Dominant Color Descriptor

For multimedia database indexing and query-by-example applications, Dominant Color Descriptor is a widely used descriptor, which summarizes color information of a video frame, still image or regions of these. This descriptor uses color spaces defined by Color Space Descriptor that defines RGB, YCbCr, HVS and HMMD color spaces and RGB is the default one. For quantization of color space Dominant Color Descriptor uses Color Quantization Descriptor. Dominant Color Descriptor is composed of a number of significant colors, color

components of each of these colors, percentage of these colors, spatial coherency, and color variance. For extraction of Dominant Color Descriptor, Generalized Lloyd Algorithm is used in order to divide pixel color values into clusters by minimizing distortion. Initially the algorithm starts with a single cluster that contains all image pixels. This cluster's center of mass, which is a dominant color, is calculated and this clustering and center of mass calculations are repeated until distortions of clusters are below a threshold.

3.4.1.2 Scalable Color Descriptor

Scalable Color Descriptor is color histogram of an image, which is calculated in HSV color space. This histogram is encoded for decreasing storage requirements. This descriptor's extraction process starts with calculating color histogram in 256 bins. Hue component of HSV color space is represented with 16 bins and other components Saturation and Value are represented with 4 bins each. Then 1-D Haar transforms are applied repeatedly to original histogram and resulting coefficients are used to represent this descriptor.

3.4.1.3 Color Structure Descriptor

Color Structure Descriptor gives both color layout information and local color structure information of an image. Extraction process of this descriptor is as follows: a structuring element, which is a small rectangular pixel block, is used as a moving window over the image. For each location of the structuring element, colors of pixels of underneath image region are added to corresponding histogram bins. Then this histogram is normalized and quantized to form Color Structure Descriptor.

3.4.1.4 Color Layout Descriptor

Color Layout Descriptor is a compact representation of spatial distribution of the color of an image that uses YCbCr color space. Extraction process of this descriptor starts with partition of image into 8x8 blocks. After that, a representative color for each block is selected. Then Discrete Cosine Transform (DCT) is applied to the resulting array. Transformed array is ordered with zig-zag re-ordering and high frequency DCT coefficients are eliminated.

Low frequency DCT coefficients of Y, Cb and Cr values are used for representation of this descriptor.

3.4.2 Texture Descriptors

Texture descriptors [12] are one of the most expressive visual descriptors with color descriptors. MPEG-7 texture descriptors are used for similarity matching (query by example) and browsing applications. Table 3.3 lists MPEG-7 texture descriptors.

Table 3.3: MPEG-7 texture descriptors

Homogeneous Texture Descriptor	Represents texture with mean energy and energy deviation from a set of frequency channels
Texture Browsing Descriptor	Perceptual characterization of texture in terms of regularity, coarseness and directionality
Edge Histogram Descriptor	Spatial distribution of edge types

3.4.2.1 Homogeneous Texture Descriptor

Homogeneous Texture Descriptor represents statistical distribution of image texture by using mean energy and energy deviation of frequency layout channels. Descriptor is a vector with size 62. Extraction starts with calculating mean and standard deviation of the image. Then Fourier transform is applied to the image. 2-D frequency domain is partitioned into 30 channels and modeled with Gabor functions. Then mean energy and energy deviation for each channel are calculated and these values correspond to the remaining numbers in Homogeneous Texture Descriptor vector.

3.4.2.2 Texture Browsing Descriptor

Texture Browsing Descriptor describes textured regions regularity, direction and coarseness. This descriptor is a five element vector. First element is a ranking of regularity, second and third elements are used for the description of directionality, and the last two elements describe coarseness of the texture. For extraction of this descriptor, image is filtered with Gabor filter. Then, for dominant direction estimation, directed histogram is calculated on filtered image

and peak points of the histogram are taken as dominant directions. Filtered image is rotated at these obtained dominant directions and then with Radon transform projections on dominant directions are calculated. Projection analysis results in regularity and coarseness estimation of the texture.

3.4.2.3 Edge Histogram Descriptor

Edge Histogram Descriptor represents spatial distributions of edges of an image. For extraction, image is divided into 16 sub-images. For each sub-image edges are extracted and classified into five categories with respect to their angle between edge and x-axis. Then a histogram is generated with these classes as histogram bins and edge counts are calculated as histogram bin values. At last step, resulting histogram bin values are nonlinearly quantized.

3.4.3 Shape Descriptors

MPEG-7 shape descriptors [13] are used in similarity search and retrieval. Shape descriptors are important for object detection because shape reveals characteristics of the object, unlike other visual descriptors like color and texture. Table 3.4 lists shape descriptors of the MPEG-7 standard.

Table 3.4: MPEG-7 shape descriptors

Region-Based Descriptor	Expresses pixel distribution within a region
Contour-Based Descriptor	Curvature Scale-Space representation of contour
3-D Shape Descriptor	Histogram of geometrical properties of 3-D surface

3.4.3.1 Region-Based Descriptor

Region-Based Descriptor represents an object's shape by using both edge pixels and filling pixels. This descriptor is extracted with Angular Radial Transform (ART). ART divides region into complex valued basis functions. These transforms' normalized and quantized coefficients are used to represent the shape. This descriptor is suitable for describing shapes with holes.

3.4.3.2 Contour-Based Descriptor

Contour-Based Descriptor uses Curvature Scale-Space (CSS) based representation of shape. CSS representation is extracted by repeatedly calculating the curvature of the contour while smoothing the contour. This descriptor consists of the following fields: number of peaks, global curvature, prototype curvature, highest peak (absolute), peakX (X position of a peak), height of the peak (relative). Global curvature is circularity and eccentricity of the contour, while prototype is circularity and eccentricity of the smoothed contour. This scale and rotation invariant descriptor is useful for similarity based retrieval applications.

3.4.4 Motion Descriptors

Motion descriptors [14] of the standard are used for representing motion features of a video sequence. These descriptors enable motion-based queries for content based retrieval applications. Motion descriptors are listed in Table 3.5.

Table 3.5: MPEG-7 motion descriptors

Motion Activity Descriptor	Intensity or pace of action
Camera Motion Descriptor	Movement of the camera
Motion Trajectory Descriptor	Movement of objects in time
Parametric Motion Descriptor	Change of regions over time

3.4.4.1 Motion Activity Descriptor

Motion Activity Descriptor represents the degree of activity in a video. Fields of this descriptor are intensity of activity, direction of activity, spatial distribution of the activity and temporal distribution of activity. Only first field is mandatory, other fields are optional. Intensity of activity field denotes the degree of activity and takes values between 1 and 5. Extraction of this field can be done with statistical properties of motion vector magnitudes.

3.4.4.2 Parametric Motion Descriptor

Parametric Motion Descriptor describes an object's motion by representing it in one of the following motion models: translational, rotation/scaling, affine, perspective, quadratic. Extraction of this descriptor aims to minimize the difference between the actual motion and selected model.

CHAPTER 4

VIDEO REPRESENTATION

4.1 Video Modeling

Video data is a collection of still pictures that represents scenes. Video differs from audio and image data types because it contains both spatial and temporal information. In order to annotate video effectively, video should be decomposed into smaller spatio-temporal units. Video can be divided into temporal segments called shots. A shot is an uninterrupted sequence of frames of a video, recorded with a single camera. It is defined by start time and duration. Transitions between shots can be either abrupt or gradual. Video is composed of shot frames and transitional frames between shots. Transitional frames are highly variable and do not contain semantic information.

Besides shots, video can be decomposed into semantic temporal units, namely events. Durations of events are arbitrary and collection of events might not cover the entire video. Events may have sub-events. For example, event 'Leaving Parking Lot' in a video that has taken by a parking-lot's surveillance camera, can be divided into sub-events 'Getting by the Car', 'Open Front Door' and 'Move the Car'. Temporal relations between events and sub-events can be defined.

Each shot is temporally decomposed into key frames. Key frames are also video segments which are defined by time points. One or more key frames can be used to summarize a shot. Key frames can be described with low-level visual features and high-level semantic descriptions.

A moving region is a spatio-temporal decomposition of a shot. It can be located with temporal

descriptors in a shot. If the annotator is interested in the movement of the object in the moving region, motion descriptors are used to describe its movement. Moving regions are spatially decomposed into still regions. Each of these still regions corresponds to a semantic object and is represented as bounding rectangles and described with visual and semantic descriptors. A still region can denote either a face or an object.

4.2 Representation of Video Content with MPEG-7 Descriptions

MPEG-7 is a comprehensive and flexible standard that aims to describe almost any type of multimedia data in detail. Unfortunately this expressiveness brings complexity and ambiguity to media descriptions and restricts usage of MPEG-7 descriptions in real world applications [15]. In order to overcome this problem, profiling is proposed.

4.2.1 MPEG-7 Profiles

Profiling is basically selecting descriptors and description schemes for a specific application domain. Reducing the number of description elements reduces complexity. Another benefit of using profiling is that it prevents the semantically same descriptions to have different MPEG-7 documents. This ambiguity causes interoperability problems between automatic systems.

A profile can be described in three steps [16]:

- Description tool selection
- Description tool constraints
- Semantic constraints

Description tool selection refers to the determination of the descriptors and the description schemes to be used for the selected domain. This step reduces the number of description tools. Second step, description tool constraints, specify the number of elements for description tools and usage of attributes. Third step, semantic constraints, which are expressed in natural language, specifies how to use selected and constrained tools to describe multimedia content in the selected application domain. There are three standardized MPEG-7 profiles defined in Part 9 of the standard [17]:

- Simple Metadata Profile (SMP): Describes simple metadata tagging, can be used for audio image and video clips.

- User Description Profile (UDP): Describes personal preferences and usage patterns of multimedia content users in order to support personalized multimedia usage.

- Core Description Profile (CDP): Describes general multimedia content for creation, consumption, distribution, and archiving.

None of these standard profiles is suitable for descriptions created by the presented video annotation tool, since they do not contain visual descriptors.

There are two commonly used but not standardized profiles for multimedia descriptions: TRECVID Profile and Detailed Audio Visual Profile (DAVP) [18]. TRECVID Profile is used by TRECVID Video Retrieval Evaluation[1] participants, in order to generate interoperable MPEG-7 documents of video data selected for the contest. TRECVID Profile specifies video shot decomposition and key frames represent these shots. In the following section DAVP is presented.

4.2.2 Detailed Audio Visual Profile

Detailed Audio Visual Profile (DAVP) [19] is proposed for describing single audiovisual content entities. The profile allows decomposition of audiovisual data by supporting spatio-temporal structuring tools. Also, at many levels of decomposition, low-level feature descriptors and textual annotations are allowed. In Figure 4.1, DAVP top level elements and content entity types are shown.

Since DAVP describes audiovisual content, ContentEntityType is the main description type. Other allowed description types are Classification Description, Semantic Description, and Summary Description. ContentEntityType supports two types of content type, ImageType and AudioVisualType. Video and audio content is considered as decomposition of AudioVisualType. Most of the textual annotation types such as free and structural textual annotation tools are included for semantic descriptions. Media, creation and usage description tools are

[1] http://trecvid.nist.gov

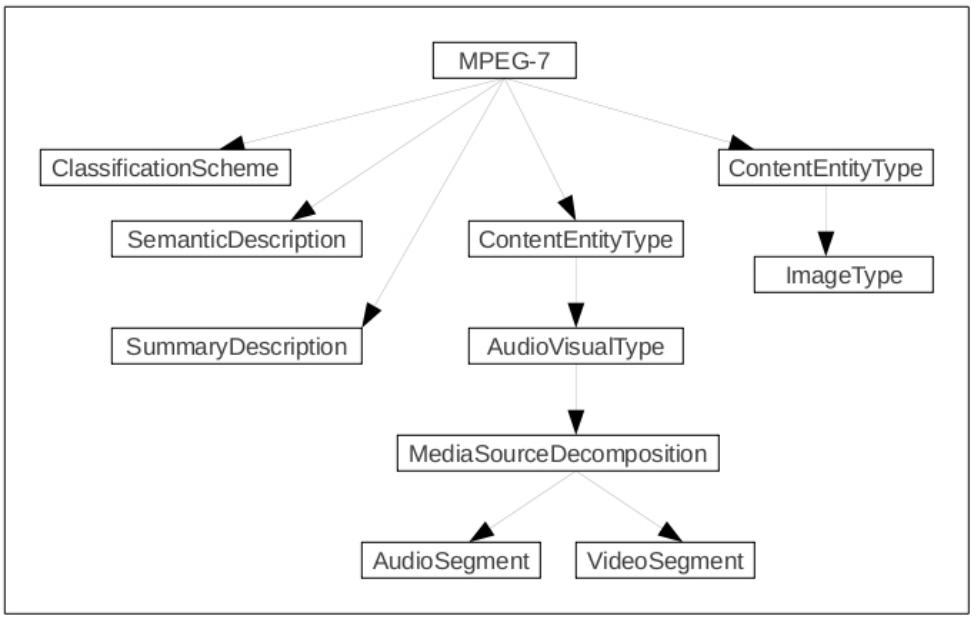

Figure 4.1: Top level elements of DAVP

fully supported. Structure description tools are important for content decomposition and included to the profile with the exception of ink and handwriting tools. From navigation and access tools of MPEG-7, summarization tools are included. Semantic constraints of DAVP specify structuring descriptions usage and ensure that description is modular and organized.

In Figure 4.2, the parts of DAVP profile used in the annotation tool is shown. Since the annotation tool's input is video data, AudioSegment (MediaSourceDecomposition of AudioVisualType) and ImageType are not included.

In Figure 4.3, descriptions of key frames and moving regions using DAVP are shown.

In DAVP, there is no specific key frame type; instead key frames are considered as video segments with a time point. For each shot there may be one or more key frames. With MediaLocator description, key frames can be linked to external files. Low-level features of key frames are described by VisualDescriptor. TextualAnnotation and Semantic description are included for describing high-level semantic features of key frames.

MovingRegion type in the profile, corresponds to all kinds of objects in the video. All of the objects in the video have spatial locations and temporal time points (every object should be in the video for at least one frame). If an object appears in the video more than one frame

and the annotator needs to represent its motion, MovingRegionDS is used, otherwise object is described by StillRegionDS. Like key frames, objects low-level features are described by VisualDescriptor and high-level semantic features are described by TextualAnnotation and semantic description.

Profiling of MPEG-7 descriptors provides interoperability for descriptions created with the annotation tool and produced MPEG-7 metadata for video data can be used by other applications using the same profile, directly. The annotation tool that we propose employs DAVP rather than coming up with a new profile for the same application domain. In various application domains, DAVP has been successfully used by many projects [19].

The reason of choosing DAVP as MPEG-7 profile for the proposed annotation tool is its ability to represent video in a fine level of granularity. All levels of annotation structures, video, shot, frame, still and moving regions can be represented and semantically annotated. Beside these, for frames, still and moving regions' low-level feature descriptors are included in the profile. The annotation tool does not use all of the descriptions in the profile, but for video description all mandatory parts are included in order to obtain descriptions that conform to DAVP.

MovingRegion type in the profile corresponds to all kinds of objects in the video. All of the of objects in the video have spatial locations and temporal time points (every object should be in the video for at least one frame). If an object appears in the video more than one frame and the annotator needs to represent its motion, MovingRegionDS is used, otherwise object is described by StillRegionDS. Like key frames, low-level features of the objects are described by VisualDescriptor and high-level semantic features are described by TextualAnnotation and semantic description.

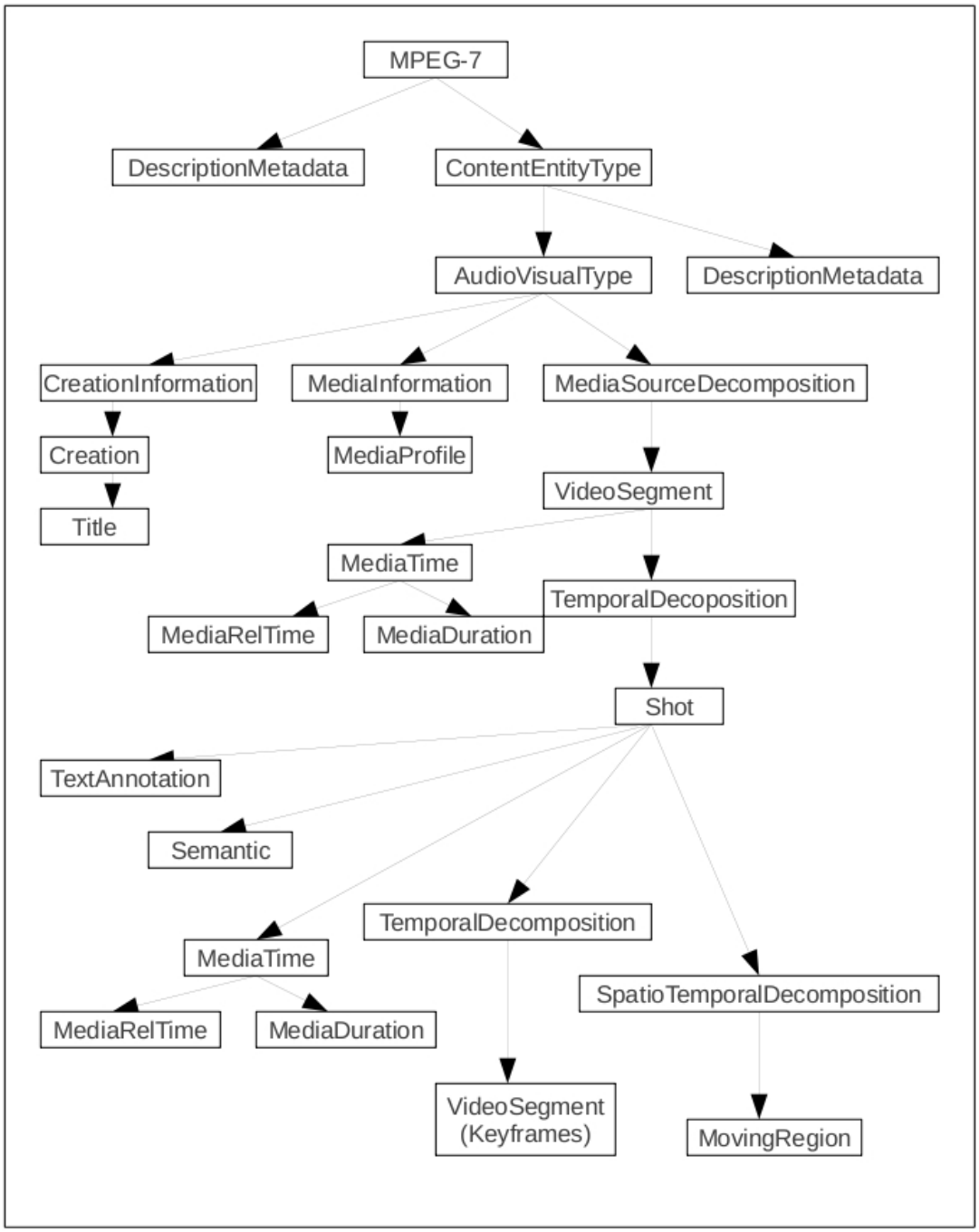

Figure 4.2: DAVP parts used by the annotation tool

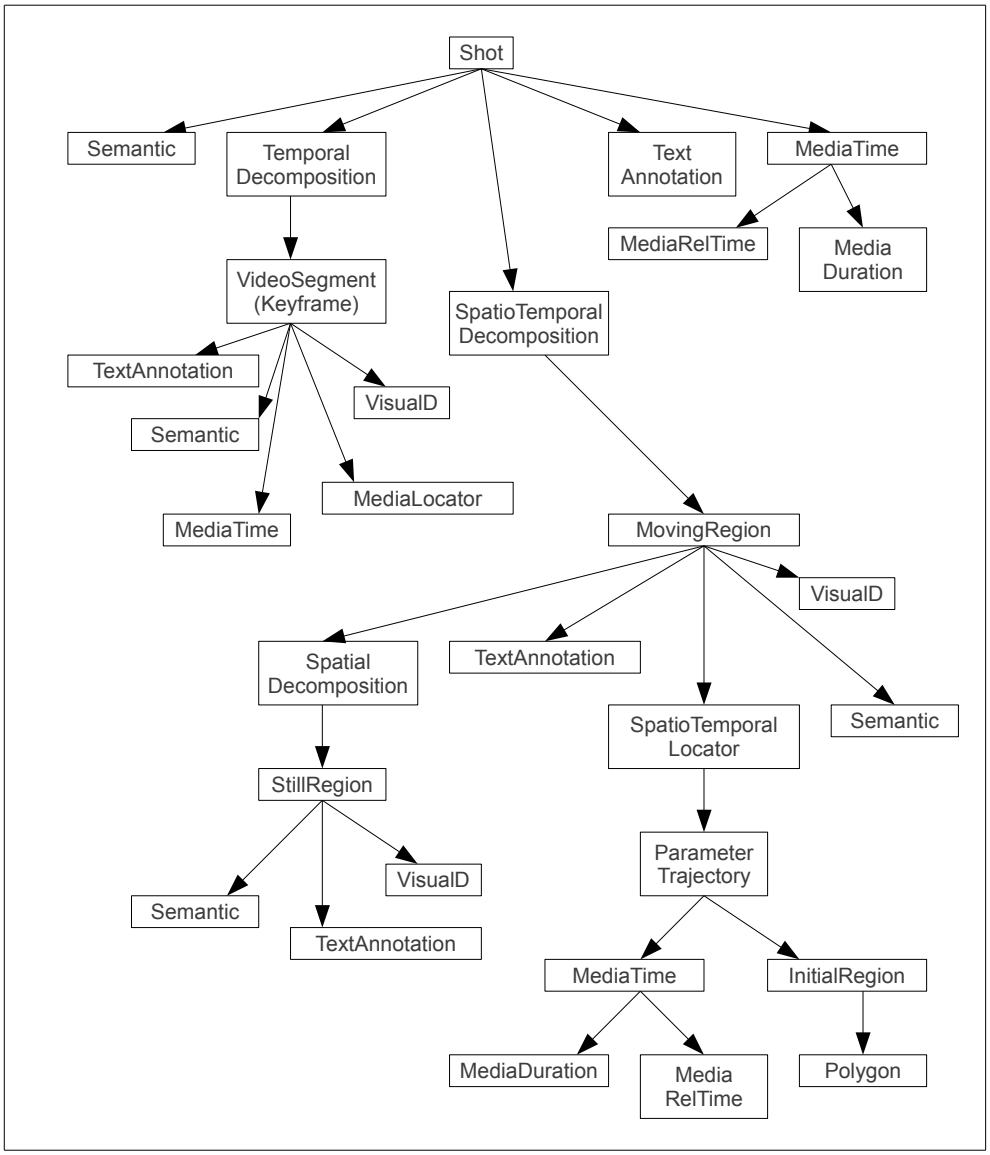

Figure 4.3: Key frame and moving region descriptors

CHAPTER 5

SYSTEM OVERVIEW

5.1 Overall System Capabilities

In this chapter, the capabilities of the implemented semi-automatic semantic video annotation system are explained in detail. Implementation is done in C++ and the application is stand-alone. The annotation tool has a video player in order to display the annotated video. With the tool the annotator can convert and save all frames in the video. After a video is loaded, the automatic shot detection module decomposes the video into shots. Each shot's first frame, last frame and I-frames between first and last frames are considered as key frames. Low-level feature extraction of frames and still regions with XM Reference Software [20] is added. For manual annotation, rectangular region selection is added. To make annotations practical, annotator can load an ontology. In order to reduce manual annotation, face detection, object tracking and object detection features are added. For implementation parts related to image processing, OpenCV[1] library is used. The results of all these annotations are used to represent video as an MPEG-7 document. The annotation tool takes video file and optionally ontology document as input, and outputs an MPEG-7 document that is generated semi-automatically as illustrated in Figure 5.1.

Annotation tool consists of ten modules: core management module, shot boundary detection module, ontology module, event annotation module, region annotation module, low-level extraction module, face detection module, object tracking module, object recognition module, and finally MPEG-7 exporter module. These modules and their interactions are illustrated in Figure 5.2. The core management module takes video file as input, loads the video, sends

[1] http://opencv.willowgarage.com

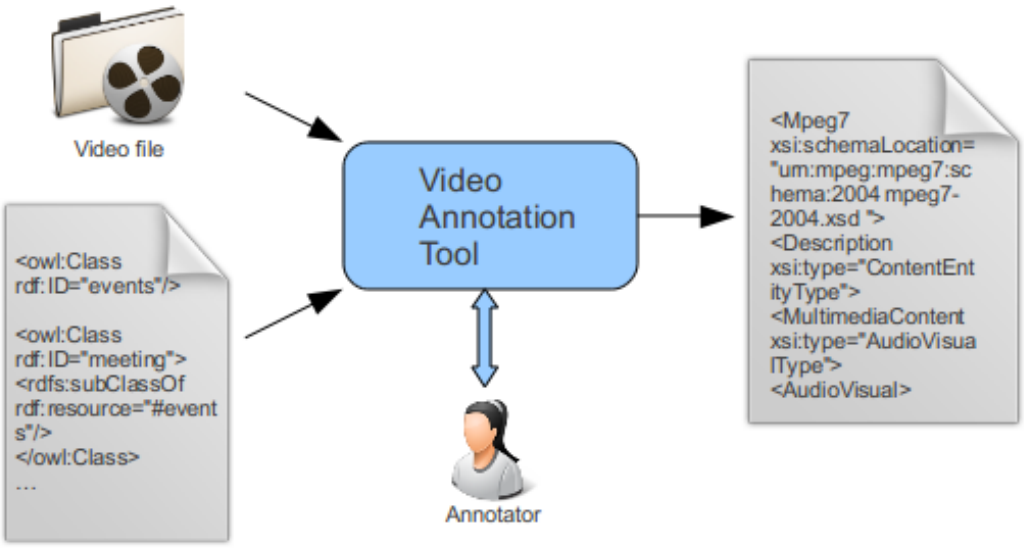

Figure 5.1: Input output diagram of the annotation tool

video information to shot boundary detection module, shot boundary detection divides video into shots, and sends shots' boundary frame numbers to core management module. Ontology module takes OWL ontology file as input, parses this file and display it to the annotator. When the annotator uses the ontology structure for annotation, ontology module sends this semantic annotation information to the core module and either core module uses it directly e.g. for shot annotation, or transfers it to the related module, for example region annotation module. When the annotator selects a frame for region annotation, core module sends this request to region annotation module, and after annotation of selected frame is finished, information related to these regions are sent to core module. Similarly, for low-level feature detection module, face detection module, event annotation module and object tracking module; core module gets annotator's request, transmits the request to related module, after module finishes its operation, it informs core module and core module updates annotation information. But for object recognition process, annotator sends recognition request to region annotation module, this request is taken by the core module and the core module delivers this request to object recognition module. Results of object recognition are displayed to the annotator for verification and verified results are transferred to the core module. When annotator needs to export annotated data to MPEG-7 document, the core module transfers annotation data to MPEG-7 exporter module.

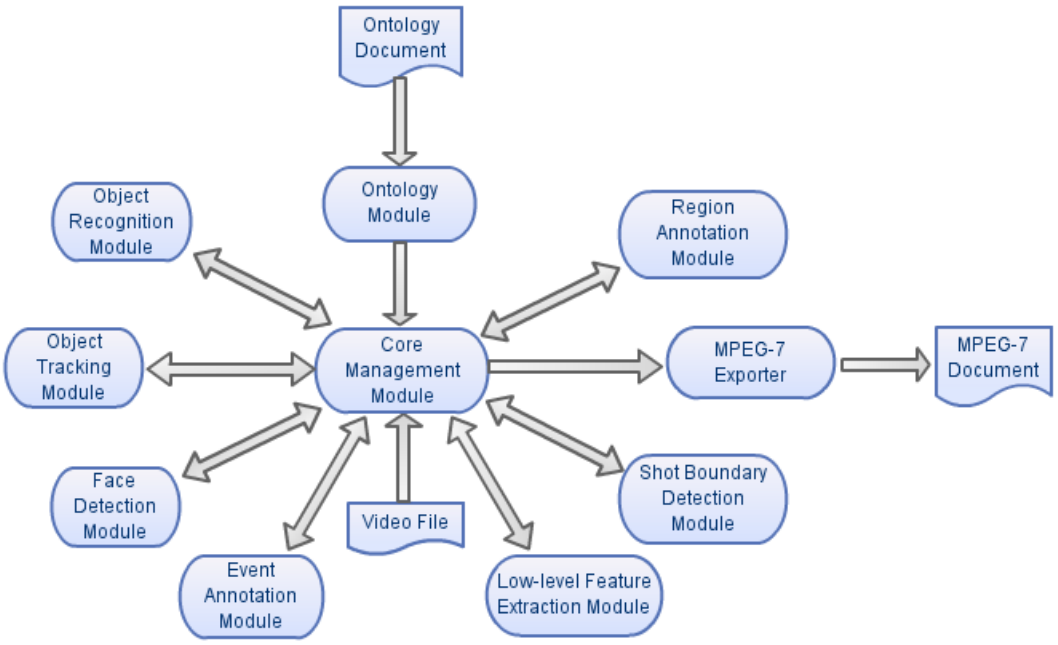

Figure 5.2: Module interaction diagram

Figure 5.3 shows the flow diagram of the annotation process. The annotation process starts with selecting the video file to be annotated. Selected video file is loaded to the video player and the tool checks if there exists shot boundary information file that has been generated from previous annotations. If this file exists, shot boundary and key frame information is loaded from file, else shot boundary boundary detection process is initialized. When the number of shots, start and end frames of these shots are determined; I-frames for each shot is identified and saved as key frames. After shot boundary and key frame information is obtained, each shots' first frames and key frames are displayed to the annotator as a list. After key frames are identified and saved, Speeded Up Robust Features (SURF) [21] are extracted from key frames. After that, OWL ontology document is loaded and parsed. Parsed ontology is displayed to the annotator. The annotator can modify loaded ontology by adding new classes or instances or removing existing ones. Ontology classes and instances are hierarchically represented, and this representation is used for the annotation of video components. After this step, the annotator can choose next annotation step. If event annotation is chosen event annotation window is displayed, and from this window new events and sub events can be added. These events can be chosen from the ontology classes which are subclass of class *events*. After events are

added to event list, annotator can mark their start and stop time points, duration of the events are displayed. Temporal relations between events can be inserted from the same window. After event annotation, facial annotation can be accomplished with face detection. With annotator's request, potentially facial regions on the key frames are marked. The annotator can edit the results; delete regions which do not contain faces or add new regions which can not be detected automatically and mark them as faces. Then chosen low-level features of selected facial regions can be extracted. If the annotator needs to annotate moving regions from video, shot that contain the moving region is selected and with object tracking, this region is tracked through the shot and information related to region's motion is saved. If the annotator needs to annotate still regions from video, frame that contains the region of interest is selected by the annotator and boundary of the region is chosen by the annotator. This region can be annotated semantically using the loaded ontology or free text. This annotated object can be searched in other key frames using object recognition. Results of object recognition are displayed and annotator can edit the results of recognition by deleting or correcting boundaries of these regions. After that, verified regions are saved as still regions. These still regions' low-level features can be extracted. Then, all annotations that are carried out so far, are exported as an MPEG-7 document.

Besides MPEG-7 document, the core module displays accomplished annotations in human readable form as plain text. This text, which can be called as video summary, includes general information about the video, video decompositions and high-level semantics related to these decompositions. Frame rate, number of total frames, and format (e.g. MPEG-1, MPEG-2 etc.) of the video stated as general information about the video. Following to these, events, sub-events their start time, duration and relations between events are indicated. Information related to the shots are also stated in the video summary. Total number of shots, start and end frames of each shot, key frames of the shots, number of facial regions and objects, and information related to moving regions in a shot are indicated. Furthermore, semantic information related to events, shots, key frames, still and moving regions are displayed. Video summary does not contain low-level information about video components. Compared to MPEG-7 document, video summary is not structured and compatible with other systems. It is generated in order to give quick insight to the annotator about the video.

The following sections explain each module that constitutes the annotation tool in detail.

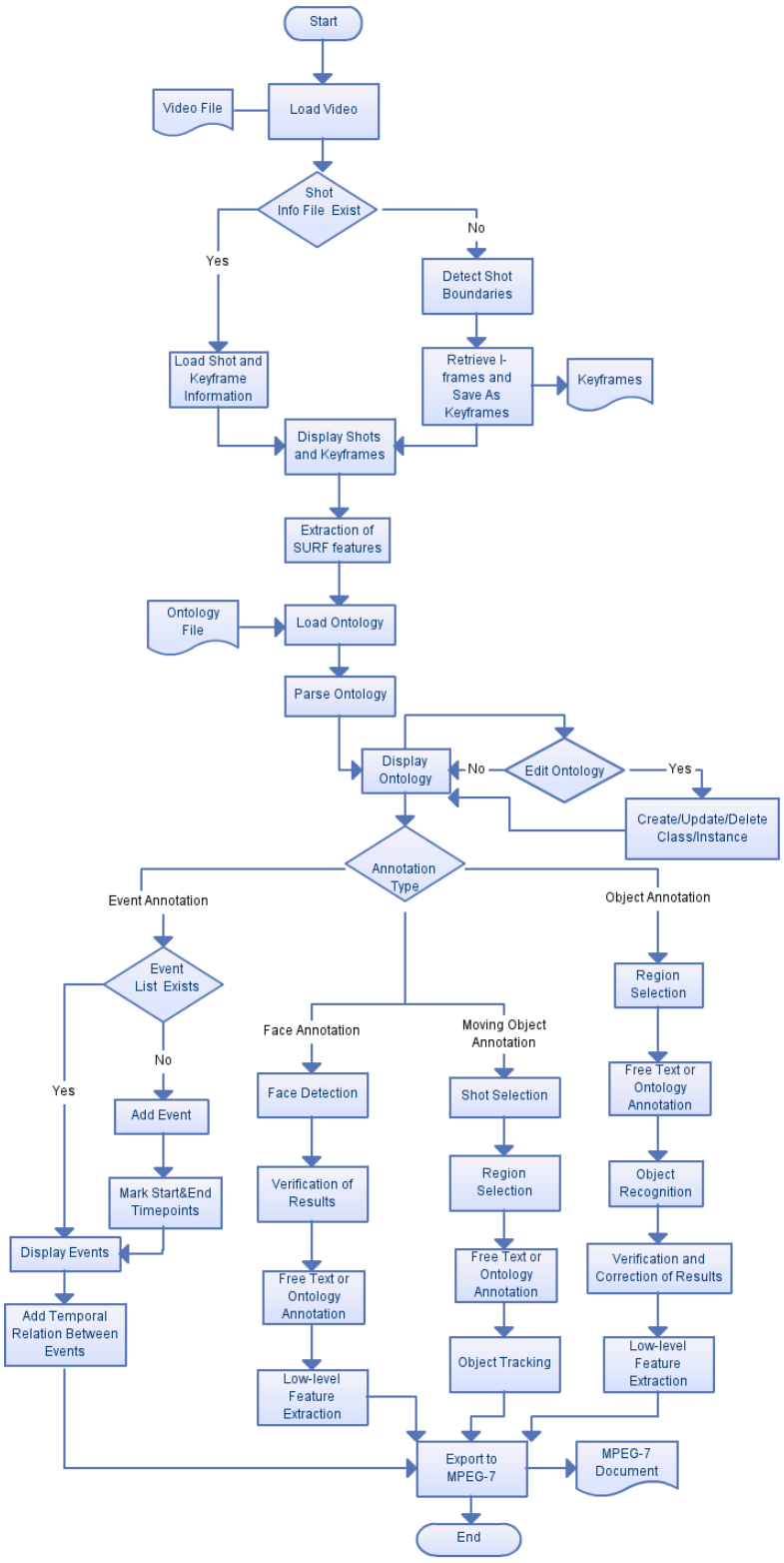

Figure 5.3: Flow diagram of annotation process using the annotation tool

5.2 The User Interface Module

The user interface module provides communication with the annotator, so it must be self explanatory and simple to use. QT [2] is used for UI implementation. QT is a cross-platform UI framework written in C++. It has high runtime performance, easy to use and well documented.

In the main window of the annotation tool, as shown in Figure 5.4, there is a menu at the top, video player is placed under the top menu. On the right of video player, shots of the video are displayed after shot detection. Each shot is represented with its first frame. When one of the shots is clicked by the annotator a new horizontal list of frames appears to the right of the list of shots. This new list shows key frames of the shots. When the annotator clicks one of the frames, a menu is displayed.

With this menu the annotator can select the next process; region annotation or low level feature extraction. The region annotation window shows the selected frame in its original size to make the annotation easier. The annotator can select the important regions with bounding rectangles on this window. After drawing the bounding rectangle, the annotator can right-click to the selected region to open a menu. With this menu the annotator can delete the rectangle, save the frame region to a file as image, annotate the selected region with free text, mark that region as face, search that object in other key frames or extract low level features of the region. In Figure 5.5, region annotation window is shown with two selected rectangular regions and the right-click menu.

In the main window, annotated video components are listed to the annotator in order to give a summary of current status of the annotation. Listed components are events, shots, key frames of these shots, still regions and moving regions.

[2] http://qt.nokia.com/products

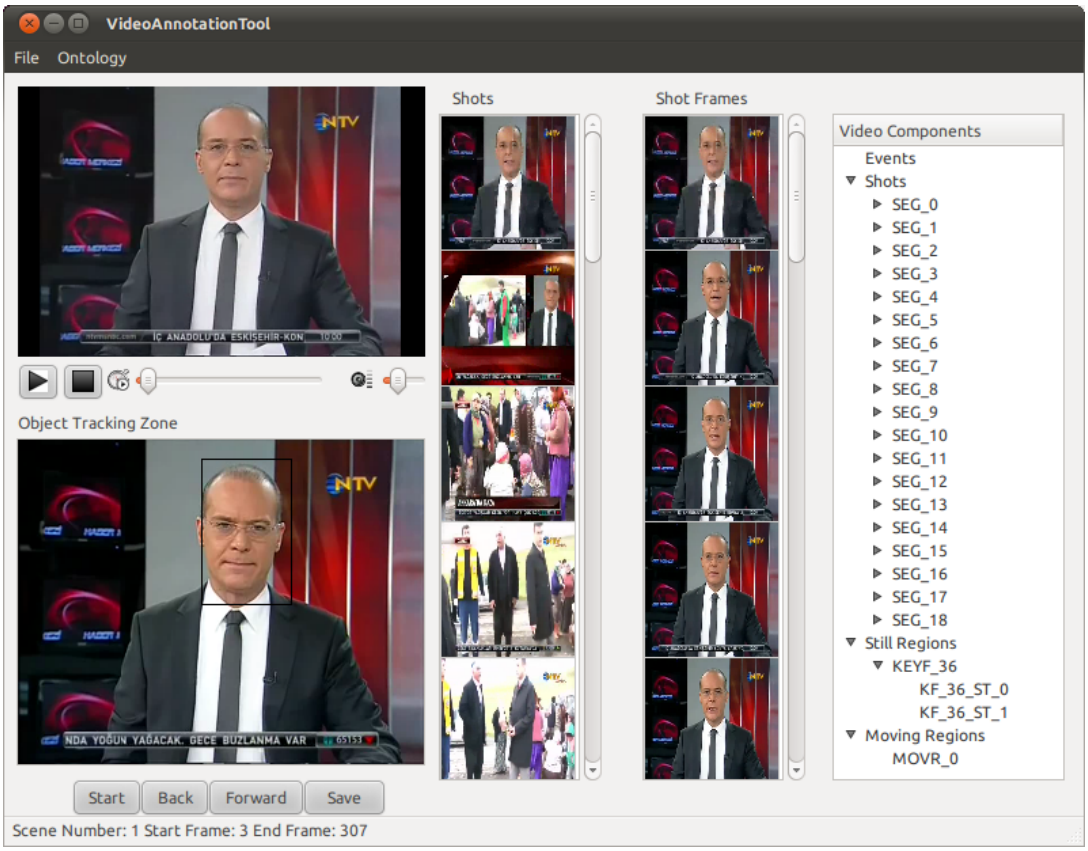

Figure 5.4: Main window of the annotation tool

Figure 5.5: Region annotation window

5.3 The Video Player

The video player is included to display the video to be annotated to the annotator. It is implemented with QT Phonon Video Player[3]. Phonon video player is currently using by well known video players MPlayer[4] and Dragon Player[5]. The video player used in the annotation tool is simple. There are play/pause and stop buttons and two sliders, one for position of the video and one for volume.

5.4 Saving Video as Frames

The annotation tool can save all frames of the video as images. Saved frames may be used for further image processing purposes. FFmpeg[6] is used for video processing. FFmpeg is a very powerful tool and also cross-platform. Libavcodec library, which is a part of FFmpeg, is used as codec library, so all video codecs that are supported by FFmpeg are also supported by the

[3] http://doc.qt.nokia.com/latest/phonon-overview.html
[4] http://www.mplayerhq.hu
[5] http://www.kde.org/applications/multimedia/dragonplayer
[6] http://ffmpeg.org

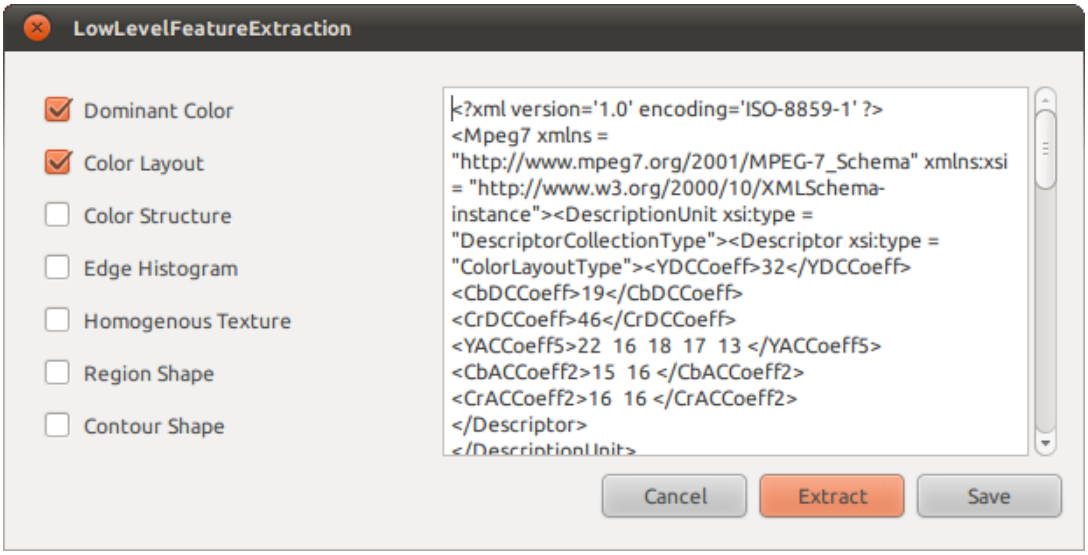

Figure 5.6: Low level feature extraction window

annotation tool.

5.5 The Low-Level Feature Extraction

The low-level feature extraction feature is included in the annotation tool for achieving complete descriptions. Supported low-level features are Dominant Color, Color Layout, Color Structure, Edge Histogram, Homogeneous Texture, Region Shape, and Contour Shape. Each of these descriptors are explained in Chapter 3. XM Reference Software [20] is used for extracting these features. If the annotator selects low level feature extraction from region annotation window menu, a low level extraction window is displayed. Figure 5.6 shows the low level extraction window.

In this window, there is a list of MPEG-7 features (Color Layout, Dominant Color, Edge Histogram, etc.). The annotator can select one or more features and by using the XM Reference Software, the annotation tool extracts selected features and shows them to annotator as text. The annotator can save these features to be exported to annotation result document.

5.6 Automatic Shot Detection

As stated before in Chapter 4, a shot is a sequence of frames captured by a single camera in a single continuous action, and shot boundaries are transitions between shots. Shot boundary detection is the first step of annotation process. The algorithm in [22], known as Edge Change Ratio is a widely used shot boundary detection algorithm that is robust to different parameter values, which is important for variable video data. To accomplish shot boundary detection with Edge Change Ratio, frames are captured from video file and frame edges are extracted with Canny Edge Detection algorithm using OpenCV. The Edge Detection Ratio, ρ, between two frames are calculated with following equations:

$$\rho = max(\rho_{in}, \rho_{out}) \tag{5.1}$$

$$\rho_{out} = 1 - \frac{\sum E[x,y].\overline{E}'[x,y]}{\sum E[x,y]} \tag{5.2}$$

$$\rho_{in} = 1 - \frac{\sum \overline{E}[x,y].E'[x,y]}{\sum E[x,y]} \tag{5.3}$$

where E and E' are binary edge images of consecutive frames, $\overline{E}$ and $\overline{E'}$ are diamond shape dilated versions of these binary edge images. ρ_{out} denotes exiting edge pixels while ρ_{in} denotes entering edge pixels.

Figure 5.7 shows edge change ratios of consecutive frames of a news story video. Selected video consists of 1944 frames, and peek points of the plot correspond video shot boundaries. The areas where high values are grouped indicate gradual changes, and abrupt changes are single high values. Automatic Shot Detection Module, which runs in a separate thread, takes video file from the Core Module, opens the video file, calculates Edge Change Ratio for all frames and determines edge boundaries with these values. Then, Automatic Shot Detection Module sends the frame numbers of start and end frames of each shot. There may be gaps between the shots because of gradual changes; so the last frame of shot S_N and the first frame of S_{N+1} may not be consecutive. Experiments on test videos show that the used method needs to extract edge pixels for each frame which is computationally expensive. However, it gives accurate shot detection results as proposed in [22].

The Core module gets these frame numbers and sends them to the Video Frames Module. The

Video Frames Module saves shot start and end frames and I-frames between these frames. These frames are considered as key frames of a shot. Each shot's start frame number, end frame number and key frame numbers are saved to a text file. For each video file, the Core module checks this text file and loads shot boundary information if exists.

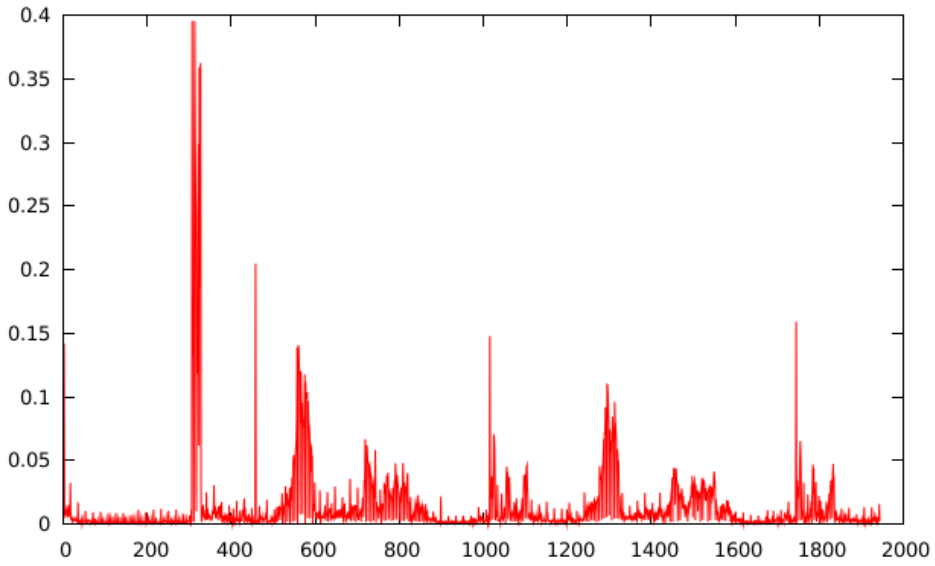

Figure 5.7: Edge change ratio plot of a news story video, x-axis denotes frame numbers while y-axis shows edge change ratios

5.7 Object Tracking

Object tracking is the problem of labeling a moving object's position in consecutive frames of a video [23]. Object tracking is an important task of computer vision. Mainly used areas of object tracking are automatic object detection, automated surveillance, video indexing, human-computer interaction, traffic monitoring, and vehicle navigation. Tracking is a hard task due to arbitrary object shapes with complex trajectory. Also noise and illumination changes in videos and real-time processing necessities require solutions to be robust and efficient. Existing tracking methods differ in object representation, used image features and modeling of motion, appearance and shape of the tracked object.

The annotation tool adopts OpenCV's Continuously Adaptive Mean Shift (CAMSHIFT) implementation for tracking. CAMSHIFT [24] is a computationally efficient color object track-

ing algorithm. Mean shift algorithm [25] is the basis of CAMSHIFT algorithm. Mean shift is developed for finding nearest peak (dominant mode) of a probability distribution. This algorithm requires an initial search window size and the position of the search window. After that, the dominant mode is calculated and search windows' center is adjusted to the mode point. Until convergence, search window's center is moved with calculated dominant mode point. It is proved that mean shift converges [26]. This robust non-parametric technique is designed for static distributions. However object tracking in video requires dynamically changing distributions since the position of the tracked object changes and object's size gets bigger when it gets closer to the camera, and it gets smaller when the object moves away from the camera. Therefore CAMSHIFT algorithm does not use fixed size window, it adopts its window size through video frames. CAMSHIFT operates on color probability image and it uses HSV as the color space. It takes initial search window size and position, constructs a calculation window which has the same center with the search window but greater in size; finds peak point within calculation window, updates the search windows' center to peak point and repeats this process until search window's center position does not change. The algorithm is able to track object in real time (30 frames per second) [24].

In the main window of the annotation tool, under the video player, there is object tracking zone. The object tracking zone has a display part and functional buttons. The display part is initially empty, when the annotator selects a shot, the first frame of the shot is displayed in the object tracking zone. Like region annotation part, the annotator can select the object as a bounding box for object tracking. After selecting the object and pressing the start button, used object tracking algorithm tracks the object through the shot. In consecutive frames, tracked object is shown by the bounding box drawn. If the annotator accepts this tracking result, object tracking module sends the moving region to the core module.

Object tracking requires annotator to select the object to be tracked. In order to automatically detect the object to be tracked, object tracking module employs the following approach: for still backgrounds (for example surveillance camera videos) a moving object which newly entered into the scene can be detected with motion vectors [27]. MPEG encoding suggests two types of compression for video: DCT and motion compensation. For motion compensation, MPEG compression calculates motion vectors for each macroblock of P-frames and B-frames. A motion vector for a macroblock shows in which direction and magnitude the position of the macroblock will change in the next frame. Motion vectors are calculated while encoding

the video, so they are already available in the video and detection with these vectors are extremely fast. However, there are misleading motion vectors that are caused by noise. These motion vectors are eliminated by looking their magnitude and their neighbor motion vectors. A possible moving object is effectively detected and segmented by using the motion vectors. Unfortunately, for videos that contain dynamic backgrounds, motion vectors do not give any useful information about the moving objects in the scene.

5.8 Face Detection

Face detection is the problem of detecting and labeling any number of faces in a given image [28]. Dynamics of human face and variety among human faces make face detection a hard computer vision problem. Solutions must be robust to illumination changes, scale (distance of face to camera), and orientation of face. According to [28] face detection techniques can be divided into two main approaches: feature based methods and image-based methods.

Feature-based methods use either low-level analysis (like color features), feature analysis or active shape models. These types of methods are suitable for real-time applications where color and motion information is present.

Image-based methods, on the other side, are suitable for static, gray-scale images. They use linear subspace methods, neural networks or statistical approaches which are all computationally expensive and not suitable for real-time applications.

The annotation tool uses the method in [29] with the improvement in [30] as the face detector. The implementation of the OpenCV library is used. In [29], the authors suggest a feature-based, real-time face detection framework. The method extracts features from image rather than using pixel intensities, in order to encode domain knowledge and to make method faster. Haar-like features are chosen. These features are basically two or more same sized rectangular regions, and values of the features are calculated by the difference of the sum of the pixels under these rectangular regions. To accomplish face detection task in real-time, rather than using image intensities directly, intermediate representation of the image called integral image is used. The integral image corresponds to double integral of the original image, row-wise and column-wise. It can be computed with a few operations per pixel in a single pass over the image. The reason for using integral image is that it enables Haar-like features to be

computed in constant time at any scale and location. Haar-like features are simple to compute but there are too many and majority of them should be eliminated. AdaBoost algorithm [31] is modified to use as both for feature selection and classifier training. Each feature is given to AdaBoost as classifier and the algorithm returns a set of these features as weak classifiers. In this way, feature selection is accomplished. With the selected features, weak classifiers are obtained, and with weak classifiers a cascade of classifiers is built. Classifier cascade aims to eliminate non facial sub-windows with minimum calculation. A sub-window can be marked as face if and only if every weak classifier in cascade approves it. If one of the classifiers determines that a sub-window does not contain facial region, that sub-window is rejected before other classifiers' decisions calculated. In the classifier cascade, low level classifiers are simple to compute and have low false negative values. Majority of sub-windows which do not contain faces are rejected by these classifiers and almost all sub-windows with facial images are accepted. High level classifiers are more complex and have higher false positive rates. These strong and complex classifiers deal with hard examples since simple ones are rejected by simpler classifiers of the cascade. Therefore the cascade of classifiers highly reduces computation time while preserving high detection rates.

This work is further improved in [30] with a new set of Haar-like features and a post optimization procedure. Introduced rotated features, which also can be computed effectively like original ones, reduces false positive rates. Also, post-optimization on each cascade classifier reduces false positive rates further. It is stated that this optimization procedure affects computation time comparably.

The annotation tool uses classifiers that are trained by [32] and they are available in the OpenCV. According to the study, for the training data set, 1000 facial images are used, then by manipulating these facial images with rotation, scaling, mirroring and shifting, 5000 facial images are generated. In addition to these facial images, 3000 non-facial (background) images are used.

Faces are always important for video annotation, in order to simplify and automate the face annotations, the annotation tool offers the annotator to automatically detect faces in key frames. After the request of the annotator, the face detection algorithm is applied to all key frames, and detected regions are added as region annotation of the frames. These facial regions can also be further processed like manually drawn rectangles. False positives can be deleted by

the annotator and if there are other facial regions that could not be detected automatically, the annotator can select these regions and mark them as faces. In the generated output document, these facial regions are indicated as faces.

Face detection is applied to all key frames, so the running time of the used method is important. Experiments show that available implementation of Viola's method [32] with training data, is fast, robust and effective.

5.9 Object Recognition

In computer vision, object recognition is the problem of detecting and labeling a given object in an image or video sequence. Selected object may be occur in any part of the image, in any orientation and scale, even partly occluded by other objects in the scene. These requirements make object recognition a challenging problem. The annotation tool uses local invariant features rather than global features to accomplish this task. According to [33], global features like color histograms are not suitable for object recognition problem because they process whole image, mix foreground and background information. For studio-isolated objects where background is easily detected, this approach can be useful. Like global features, image segments and sliding window approach are insufficient and inefficient for object recognition.

A local feature is a point or small image patch that has one or more different properties from its local neighborhood [33]. These properties are commonly intensity, color and texture. Local features are very useful for object recognition because they can be used as robust image representation and do not require segmentation, which is a challenging problem itself. Local features do not correspond to semantically meaningful object parts because they are low-level, but analyzing their statistics helps recognizing the interest object. Local features should be repeatable (can be obtained both from object image and scene image), distinct (to represent the object), local (to eliminate occlusion effect), sufficiently large in number, efficiently computable and accurate.

Speeded Up Robust Features (SURF) [21] is a scale and rotation invariant local feature detector and descriptor. The object recognition process with SURF has three main steps: detecting interest points in object and scene images, representing these points with small neighborhoods as a feature vector, and finally matching these feature vectors in order to find correspondence

between object and scene.

In an image, regions like corners, blobs and T-junctions are distinctive regions and interest points are selected from these regions. Feature detector aims to detect the same interest points in different images with different conditions like illumination, noise etc. The detector is based on an approximation of Hessian matrix. Determinant of Hessian is used for both selecting the location and the scale. In order to constitute the Hessian matrix, convolution of Gaussian second order derivatives should be discretized and cropped. Box filters are used to discretize and approximate the second order Gaussian. Integral image [29] is used to make these computations faster. Rather than using image pyramids for multi-scale analysis, filters with different scales are applied to the original image.

After locating the interest points, the SURF descriptor assigns reproducible orientations to circular region around the interest point and aligns a square region on these circular regions, then extracts the descriptor from these square regions. Orientation assignment on interest points provides the rotation invariant feature descriptors. Orientation vector is calculated by two Haar wavelet responses (vertical and horizontal) on the integral image and then dominant orientation is found by calculating the sum of all responses within a sliding orientation window. After square region centered at interest point is oriented to calculated dominant orientation, feature descriptor divides this region to sub-regions. Then for each of these sub-regions, wavelet responses are calculated according to orientation. The feature vector consists of summation of these responses and has dimension of 64.

Last step is matching feature vectors of different images (for object recognition, object and scene images). Matching is done by comparing the distance between feature vectors. The dimension of the feature vector directly affects the matching time and the SURF feature vectors has small dimensions compared to other methods like SIFT [34].

OpenCV library's implementation of SURF is integrated to the annotation tool. With annotator's request, SURF descriptors detected and extracted for all key frames of the video. When the annotator selects a region and initiates the process of re-detection of selected object, the core module sends this information to the object recognition module. This module extracts the SURF descriptors of the object and tries to match object descriptors with descriptors extracted from key frames beforehand. If there are significant number of matching descriptors between a key frame and the object, a rectangle that encloses these descriptors in the key frame is

drawn. After matching and the localization of detected objects is done, object recognition module presents recognition results as a list to the annotator. Annotator can remove irrelevant key frames from this list and correct localization results if any. After these corrections, the object recognition module sends recognition results to the core module. The core module converts these results to still regions and give the reference of the initially selected object's id to these new regions to indicate their relation.

Experiments on the test videos show that extraction of SURF features of all frames in the video requires a considerable amount of time. However, once SURF extraction results are obtained, they can be used for all object recognition processes. Also, the annotation tool extracts SURF features in a separate thread, so the annotator can continue annotation while SURF features are extracted. Experiments also show that matching of SURF features are fast and accurate as stated in [21].

5.10 Events and Temporal Relations

As stated in Chapter 4, events are semantic temporal decomposition of videos. The annotation tool enables annotation of events and temporal relations between them.

The event annotation window of the annotation tool is shown in Figure 5.8. Event annotation window has a video player at the top to watch and identify the start and stop times of the semantic events. Events and sub-events are listed below the player. Each event's name, duration in seconds and visual proportional length are presented. Temporal relations between events can also be defined through this window. Relation types are Temporal Normative Structural Relations of MPEG-7. Precedes, follows, meets, during etc. are examples of these temporal relations.

The event annotation process consists of following steps:

- Insert a new event or sub-event
- Start the video
- When the video is at an event start time, mark the start time of the event
- If the event is finished, mark the finish time of the event

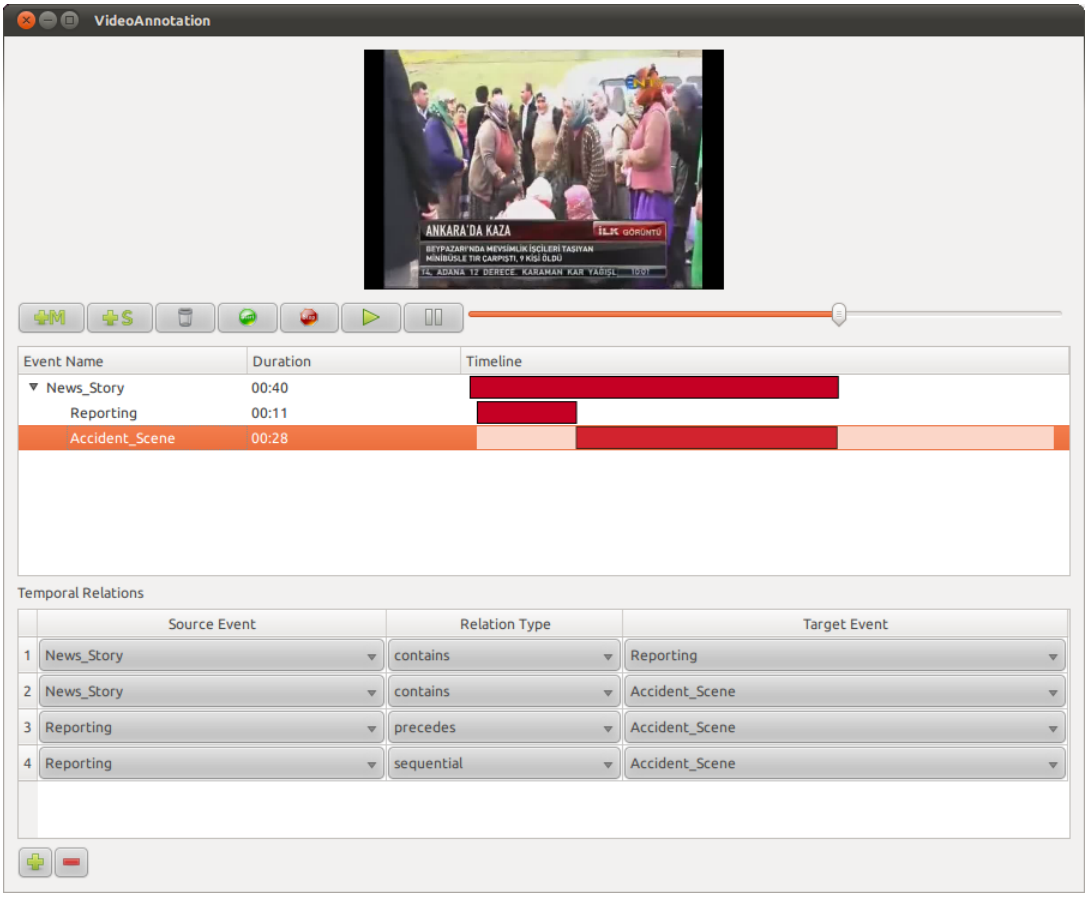

Figure 5.8: Event annotation window

- After annotation of the events are done, add temporal relationphysical
- Select source event, relation type, target event
- Save the event annotations and relations

When the annotator finishes event annotation, annotation information is sent to the core module, the core module updates video components' summary, and when the annotator wants to export annotations, the core module sends the event annotations to the export module.

5.11 Ontology

The annotator can load an Web Ontology Language (OWL) ontology to make the annotations easier. An OWL parser is implemented to parse and list the ontology classes to the annotator.

Ontologies to be used for annotation are restricted. Each defined class should be a subclass of either *events*, *concepts*, *objects*, *semantic place*, or *semantic time* class. Figure 5.9 shows the representation of an example ontology to the annotator.

Ontology classes are represented with *C* and instances are represented with *I*. The annotator can modify the loaded ontology by adding new classes and instances or removing existing ones. This ontology representation can be used for the annotation of events, shots, frames and regions of frames with simply dragging ontology items onto the video components. This highly reduces the annotation time compared to typing each annotation, also hierarchy between classes, which can be considered as *abstraction level* information is preserved. Each video component can be annotated with one or more ontology item.

5.12 Exporting as MPEG-7 Document

When the annotator decides to finish the annotation of the video, the annotation tool saves annotations to a single MPEG-7 document which is based on video representation and MPEG-7 Profile described in Chapter 4. This document aims to describe the video data in a standardized way. It includes temporal decomposition of video, as events, shots and key frames. Start times and durations of events and shots, and time points of key frames are indicated in the final document. Moving regions are spatio-temporal decomposition of shots; faces and objects are still regions that are considered as spatial decomposition of moving regions that are single frame long. High level semantic information for events, shots, key frames, moving regions, objects and faces are included. Moving regions are defined in this document with initial region, duration, and motion type. Still regions are defined with location of the region on the frame, low-level features of the region, semantic annotations and with location of the region if region is saved as a picture. Unique ids are given to all video components and used for referencing these components.

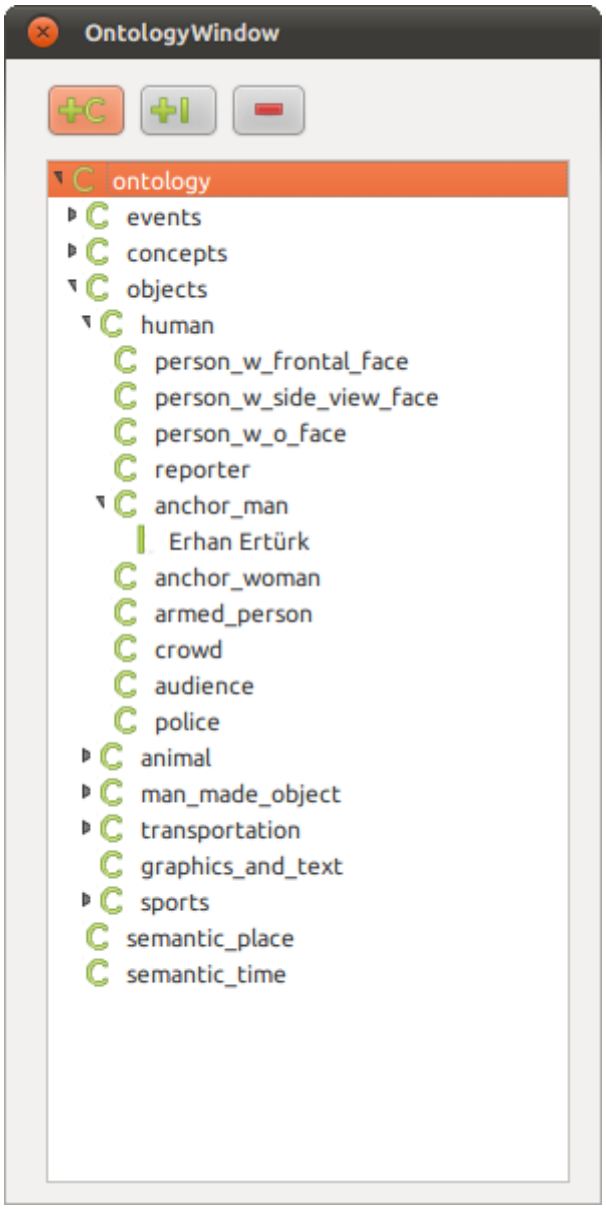

Figure 5.9: Ontology window

CHAPTER 6

DISCUSSIONS

Table 6.1 compares the presented annotation tool with other existing video annotation tools. The table is prepared based on the information from survey in [2]. The table compares annotation tools with input-output format and Annotation Level. Input output format is divided into $MetadataFormat$ and $AnnotationVocabulary$. $MetadataFormat$ denotes generated video descriptions' format. Presented annotation tool uses MPEG-7 with DAVP (Detailed Audio Visual Profile) in order to accomplish compatibility. $AnnotationVocabulary$ shows user entered (U) and tool embedded (T) vocabularies for semantic annotation. Free text annotation, keyword annotation and user given restricted OWL ontology are used as annotation vocabulary in the presented tool. *Granularity* denotes spatio-temporal decompositions of video data that can be annotated. Presented annotation tool enables annotation to be done for all possible levels, whole video, events, shots, key frames, as well as still and moving regions. *Localisation* shows how these spatio-temporal decompositions are indicated. While shots, events, key frames and moving regions are localized with a time interval, still regions are localized both temporally with time points and spatially by rectangular regions in the tool. $Expressivity$ is the scope of the semantic annotations supported, concepts or relations between concepts. Video components are related with semantic concepts and relations between these components, like spatio-temporal decompositions, temporal relations are included in the annotation tool. $Low-levelFeatures$ denotes extraction of low-level visual features, it can be either frame based or region based. Presented tool supports both region based and frame based low-level feature extraction. *Automation* is the annotation tool's effort to make annotation time shorter, by using annotators input for deduction of similar annotations. The annotation tool utilizes automatic shot detection to find shot boundaries, face detection to mark facial regions, object recognition to find similar objects in different key frames, and object tracking to follow

interested object through a shot.

Among listed video annotation tools, VideoAnnEx, SVAT and presented annotation tool generate annotation documents in MPEG-7 format. Presented annotation tool uses the same MPEG-7 profile, DAVP, with SVAT. VideoAnnEx does not use a MPEG-7 profile. VIA, Ontolog, VideoAnnEx and the presented tool utilize structured semantic hierarchies for semantic annotations. VideoAnnEx uses lexicon, while presented tool and the others use ontology. With lexicon, only semantic classes can be defined, however with ontology both semantic classes and instances can be defined. Besides other video decompositions like shots, frames, still and moving regions, presented annotation tool supports semantic event annotation with temporal relations. Among considered video annotation tools, VIA and presented annotation tool allow annotators to extract MPEG-7 low-level features from subregion of features. This property is useful for query by example applications when the selected subregion corresponds to the interested semantic object.

VideoAnnEx, SVAT and the presented annotation tool offers annotators to automatize and shorten the annotation process. First level of the automation process is automatic shot detection. VideoAnnEx uses color histogram based shot boundary detection [3], while the presented annotation tool uses edge change ratio. Edge change ratio is a slower method compared to color histogram based methods, since it requires to extract edge pixels from all frames. However, edge change ratio gives more accurate results compared to color histogram based methods [22]. SVAT and the presented annotation tool aim to shorten annotation time by re-detecting the selected object. While SVAT uses SIFT features to re-detect objects, the presented annotation tool uses SURF features. According to [21], SURF has higher recognition rate than SIFT.

Table 6.1: Comparison of selected video annotation tools

Tool	Input Output		Annotation Level				
	Metadata Format	Annotation Vocabulary	Granularity	Localization	Expressivity	Low-level Features	Automation
VIA	XML	U:OWL, free text T:XML	video, shot, frame, moving and still region	Time interval, free hand, polygon, rectangle	concepts	frame, region	-
Ontolog	RDF	U: RDFS T:Dublincore, OWL	Video, shot, frame, still region	Time interval	Concepts, relations	-	-
VideoAnnEx	MPEG-7/ XML	U: lexicon XML, free text T: MPEG-7	Video, shot, frame, still region	Time interval, rectangle	Concepts, relations	-	shot detection, detection of similar frames
SVAT	MPEG-7 /XML	U: free text, keyword T: MPEG-7	video, shot, still region	time interval	concepts	frame	shot detection, object, shot recognition
Presented Tool	MPEG-7	U: OWL, free text T: MPEG-7	video, event, shot, frame, moving and still region	Time interval, rectangle	Concepts, relations	frame, region	shot detection, object recognition, object tracking, face detection

CHAPTER 7

CONCLUSION

Generating MPEG-7 documents manually from scratch is very time consuming and may lead to erroneous documents. In this study, we presented a semiautomatic semantic video annotation tool, which enables annotators to generate MPEG-7 representations of videos effectively. Presented annotation tool differs from other annotation tools in terms of the automation of the annotation process with the help of image processing and pattern recognition tools. These components are namely automatic shot detection, object recognition, face detection and object tracking. While these components shorten the annotation process, the annotation tool gives the opportunity to the annotator to verify and correct obtained results. Effectiveness of this automation depends on the performance of the used methods directly.

While usage of MPEG-7 as metadata format enables interoperability with other multimedia systems, employing commonly used MPEG-7 profile increases this interoperability further. The way video data modeled which is compatible with the profile, provides annotation to be done in every possible level, from the whole video to still and moving regions.

Extraction of low-level MPEG-7 visual descriptions from frames and selected regions are useful for content based retrieval applications and important for the completeness of the annotations. These descriptors are also widely used in image processing, computer vision, and pattern recognition.

Employing user defined ontologies as annotation vocabularies help the annotation tool to express the concepts and relations between concepts in the generated descriptions for specific application domains. This also reduces the annotation time since it avoids retyping of the concepts.

The annotation tool deals only with visual data, but for multi-modal multimedia applications, annotations for audio data is also required. The annotation tool can be extended to handle audio annotations.

The annotation tool does not offer any solution related to storage of generated MPEG-7 documents. For the storage of generated MPEG-7 documents, a native XML database can be used and annotations can be queried within this database.

REFERENCES

[1] "MPEG-7: ISO/IEC 15938. multimedia content descritpion interface," 2001.

[2] S. Dasiopoulou, E. Giannakidou, G. Litos, P. Malasioti, and Y. Kompatsiaris, "A survey of semantic image and video annotation tools," *Knowledge-driven multimedia information extraction and ontology evolution*, pp. 196–239, 2011.

[3] C. Lin, B. Tseng, and J. Smith, "VideoAnnEx: IBM MPEG-7 annotation tool for multimedia indexing and concept learning," in *IEEE International Conference on Multimedia and Expo*, 2003.

[4] J. Heggland, "Ontolog: Temporal annotation using ad hoc ontologies and application profiles," *Research and Advanced Technology for Digital Libraries*, pp. 5–17, 2002.

[5] O. Aubert and Y. Prié, "Advene: an open-source framework for integrating and visualising audiovisual metadata," in *Proceedings of the 15th international conference on Multimedia*, pp. 1005–1008, ACM, 2007.

[6] H. Sloetjes and P. Wittenburg, "Annotation by category: Elan and iso dcr," *Proceedings of the Sixth International Language Resources and Evaluation (LREC 08), Marrakech, Morocco, may*, 2008.

[7] P. Schallauer, S. Ober, and H. Neuschmied, "Efficient semantic video annotation by object and shot re-detection," in *Posters and Demos Session, 2nd International Conference on Semantic and Digital Media Technologies (SAMT), Koblenz, Germany*, 2008.

[8] B. Manjunath, P. Salembier, and T. Sikora, *Introduction to MPEG-7: multimedia content description interface*, vol. 1. John Wiley & Sons Inc, 2002.

[9] J. Hunter, "An overview of the MPEG-7 description definition language (ddl)," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 6, pp. 765–772, 2001.

[10] P. Salembier and J. Smith, "MPEG-7 multimedia description schemes," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 6, pp. 748–759, 2001.

[11] L. Cieplinski, "MPEG-7 color descriptors and their applications," in *Computer Analysis of Images and Patterns*, pp. 11–20, Springer, 2001.

[12] P. Wu, Y. Ro, C. Won, and Y. Choi, "Texture descriptors in MPEG-7," in *Computer Analysis of Images and Patterns*, pp. 21–28, Springer, 2001.

[13] M. Bober, "MPEG-7 visual shape descriptors," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 11, no. 6, pp. 716–719, 2001.

[14] S. Jeannin and A. Divakaran, "MPEG-7 visual motion descriptors," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 11, no. 6, pp. 720–724, 2001.

[15] R. Troncy, W. Bailer, M. Hausenblas, P. Hofmair, and R. Schlatte, "Enabling multimedia metadata interoperability by defining formal semantics of MPEG-7 profiles," *Semantic Multimedia*, pp. 41–55, 2006.

[16] M. R. Group, "MPEG-7 interoperability, conformance testing and profiling, v.2. ISO/IEC JTC1/SC29/WG11 N4039. Singapore," March 2001.

[17] M. R. Group, "MPEG-7 profiles and levels, ISO/IEC JTC1/SC29 15938-9," 2005.

[18] R. Troncy, W. Bailer, M. Hausenblas, and M. Höffernig, "VAMP: semantic validation for MPEG-7 profile descriptions," *CWI Information Systems [INS]*, no. E0705, 2007.

[19] W. Bailer and P. Schallauer, "The detailed audiovisual profile: Enabling interoperability between MPEG-7 based systems," in *In 12 th International MultiMedia Modelling Conference*, pp. 217–224, 2006.

[20] M. R. Group, "Reference software, ISO/IEC JTC1/SC29 15938-6," 2003.

[21] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded up robust features," *Computer Vision–ECCV*, pp. 404–417, 2006.

[22] R. Zabih, J. Miller, and K. Mai, "A feature-based algorithm for detecting and classifying production effects," *Multimedia systems*, vol. 7, no. 2, pp. 119–128, 1999.

[23] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Computing Surveys (CSUR)*, vol. 38, no. 4, p. 13, 2006.

[24] G. Bradski, "Computer vision face tracking for use in a perceptual user interface," *Intel Technology Journal*, vol. Q2, 1998.

[25] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 8, pp. 790–799, 1995.

[26] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, no. 5, pp. 603–619, 2002.

[27] H. Zen, T. Hasegawa, and S. Ozawa, "Moving object detection from mpeg coded picture," in *Image Processing, 1999. ICIP 99. Proceedings. 1999 International Conference on*, vol. 4, pp. 25–29, IEEE, 1999.

[28] E. Hjelmås and B. Low, "Face detection: A survey," *Computer vision and image understanding*, vol. 83, no. 3, pp. 236–274, 2001.

[29] P. Viola and M. Jones, "Robust real-time face detection," *International journal of computer vision*, vol. 57, no. 2, pp. 137–154, 2004.

[30] R. Lienhart and J. Maydt, "An extended set of haar-like features for rapid object detection," in *Proceedings of International Conference on Image Processing*, vol. 1, pp. I–900, IEEE, 2002.

[31] R. Schapire, Y. Freund, P. Bartlett, and W. Lee, "Boosting the margin: A new explanation for the effectiveness of voting methods," *The annals of statistics*, vol. 26, no. 5, pp. 1651–1686, 1998.

[32] R. Lienhart, A. Kuranov, and V. Pisarevsky, "Empirical analysis of detection cascades of boosted classifiers for rapid object detection," *Pattern Recognition*, pp. 297–304, 2003.

[33] T. Tuytelaars and K. Mikolajczyk, "Local invariant feature detectors: a survey," *Foundations and Trends® in Computer Graphics and Vision*, vol. 3, no. 3, pp. 177–280, 2008.

[34] D. Lowe, "Object recognition from local scale-invariant features," in *The Proceedings of the Seventh International Conference on Computer Vision*, vol. 2, pp. 1150–1157, IEEE, 1999.