A Semantic Concept for The Mapping of Low-level Analysis Data to High-level Scene Descriptions

Ein semantisches Konzept für die Abbildung von low-level Analyseergebnissen auf high-level Szenenbeschreibungen

D I S S E R T A T I O N

zur Erlangung des akademischen Grades
Doktoringenieur
(Dr.-Ing.)

vorgelegt der
Fakultät für Elektrotechnik und Informationstechnik
Technische Universität Ilmenau

von
Dipl.-Ing. Holger Neuhaus
geboren am 31.08.1972 in Rotenburg (Wümme)

vorgelegt am: 18. Juni 2007

Gutachter:

1. Prof. Dr-Ing. Karlheinz Brandenburg
2. Prof. Dr-Ing. Thomas Sikora
3. Dr-Ing. Ulrich-Lorenz Benzler

Verteidigung am: 24. Oktober 2008

urn:nbn:de:gbv:ilm1-2008000197

Abstract

Along with the growing need for security, an increasing amount of surveillance content is created. It is indispensable to index the content in advance in order to enable quick and reliable searches on the output of hundreds or thousands of surveillance sensors installed at a single facility. For this purpose, the concept of Smart Indexing and Retrieval (SIR) enables cost efficient searches by generating high-level meta data. The generation of this meta data has to be done automatically, based on the low-level features extracted by content analysis algorithms. Creating it manually becomes more and more difficult to handle in reasonable time at reasonable costs.

Whereas formerly proposed approaches have been strongly application dependent, in this thesis, a generic concept for mapping the results of the low-level content analysis data to semantic event descriptions and its application is presented. The constituting elements of this approach and their underlying concepts as well as an introduction to their application are shown. The main contribution of the approach are the generality and the early stage at which the step from low-level to high-level representation is taken. This reasoning in the meta data domain is performed on small time frames while the reasoning on complexer scenes is done in the semantic space. Even an unsupervised self-assessment is possible using the semantic approach.

Keywords:
scene description, behavior recognition, event ontology, smart indexing, video retrieval, surveillance

Zusammenfassung

Zusammen mit dem wachsenden Bedarf an Sicherheit wird eine zunehmende Menge an Überwachungsinhalten geschaffen. Um eine schnelle und zuverlässige Suche in den Aufnahmen hunderter oder tausender in einer einzelnen Einrichtung installierten Überwachungssensoren zu ermöglichen, ist die Indizierung dieses Inhalts im Voraus unentbehrlich. Zu diesem Zweck ermöglicht das Konzept des Smart Indexing & Retrieval (SIR) durch die Erzeugung von high-level Metadaten kosteneffiziente Suchen. Da es immer schwieriger wird, diese Daten manuell mit annehmbarem Zeit- und Kostenaufwand zu generieren, muss die Erzeugung dieser Metadaten auf Basis von low-level Analysedaten automatisch erfolgen.

Während bisherige Ansätze stark domänenabhängig sind, wird in dieser Arbeit ein generisches Konzept für die Abbildung der Ergebnisse von low-level Analysedaten auf semantische Szenenbeschreibungen präsentiert. Die konstituierenden Elemente dieses Ansatzes und die ihnen zugrunde liegenden Begriffe werden vorgestellt, und eine Einführung in ihre Anwendung wird gegeben. Der Hauptbeitrag des präsentierten Ansatzes sind dessen Allgemeingültigkeit und die frühe Stufe, auf der der Schritt von der low-level auf die high-level Repräsentation vorgenommen wird. Dieses Schließen in der Metadatendomäne wird in kleinen Zeitfenstern durchgeführt, während das Schließen auf komplexeren Szenen in der semantischen Domäne ausgeführt wird. Durch die Verwendung dieses Ansatzes ist sogar eine unbeaufsichtigte Selbstbewertung der Analyseergebnisse möglich.

Schlagwörter:
Szenenbeschreibung, Verhaltenserkennung, Ereignisontologie, „Kluge“ Indexerstellung, Suche in Videos, Überwachungssysteme

meinen Eltern

Acknowledgments

I am thanking my supervisor Prof. Dr.-ing. Karlheinz Brandenburg for having given me the opportunity to commence in this challenging and most interesting area. Above all, I thank him for his valuable clues and references, and for his guidance and support throughout this work. I am thanking Prof. Dr-Ing. Thomas Sikora for his support.

I thank Dr. Ulrich-Lorenz Benzler for his guidance and mentoring, the unbounded patience and readiness and the administrative support he gave me through these years.

I thank the Robert Bosch GmbH for employing me in the "Doktorandenprogramm" and namely Dr. Lars Placke, Dr. Andreas Engelsberg, Dr. Wolfgang Niem, Dr. Stefan Mueller-Schneiders, and Thomas Jäger at Robert Bosch GmbH Hildesheim for their administrative support, the mentoring, and the willingness for count- and endless discussions. Great gratitude to my lektors Ulrich and Marco. I thank Uwe Daniel, the head of the department CR/AEM, for his support in all formalities.

I thank all the other doctoral candidates at Robert Bosch Hildesheim for the great community and the many soirées to strengthen this community ;-)

My special thanks go to the Kern Team.

I thank Rob Wijnhoven for his cooperation and the productive discussions we had about the topic and for his constructive literary hints.

Vielen Dank an Kerstin Lietz für die Bereitstellung des „Drehortes".

I thank Simon Moser not only for him fortifying me and being a good fellow student and friend but also for his professional advice and support.

Ich danke Andrea & Gert, Ulrike, Soeren und Marco für Unterstüzung, Zuspruch und Verständnis in den „weniger erfolgreichen“ Zeiten.

Weiterhin danke ich meinen damaligen Kommilitonen Marcus Kern, Matthias Kautzner und Helge-Hubertus Hundacker für die äußerst produktive Zusammenarbeit während des gesamten Studiums sowie Christian Weigel für ebendies, als auch ihm und Rossi für die freundliche Unterkunft in Ilmenau.

Cheers to the Australian Outback for inspiration.

Mein größter Dank jedoch gilt meinen Eltern[1], die mich stets mit all ihrer Kraft unterstützt und gefördert haben und mir so den langen Weg, den ich gegangen bin, erst ermöglichten.

[1] My *greatest* gratitude goes to my parents

Contents

List of Figures

List of Tables

Chapter 1

The Smart Indexing and Retrieval Problem

1.1 Introduction

Along with the growing need for security, an increasing amount of surveillance content is created. It is indispensable to index the content in advance in order to enable quick and reliable searches on the output of hundreds or thousands of surveillance sensors installed at a single facility. The aim of Smart Indexing and Retrieval (SIR; also called Semantic-Based Video Retrieval (SBVR), [Hu et al., 2004], Content Based Video Indexing and Retrieval (CBVIR), [Bashir and Khokhar, 2003] or Automatic Forensic Video Retrieval (AFVR), [Hampapur et al., 2004]) is to generate meta data and thus enable efficient search on the content. The generation of this meta data has to be done automatically, based on the low-level features extracted by content analysis algorithms. Creating it manually becomes more and more difficult to handle in reasonable time at reasonable costs. Upon the outputs on- or offline meta data processing is employed to extract higher-level information. The actual challenge is this semantic analysis of the content based on the extracted features. Although several approaches have been taken to address this topic, the search for an all-purpose solution continues:

> If indeed such a useful set of generic actions can be defined, would it be possible to identify corresponding features and matching methods which are, to a large degree, application-independent? ([Gavrila, 1999])

1.2 Problem Statement

Intuitively, an index depends on the kind of retrieval that is *expected* to be performed. Unfortunately, it can not be known in advance, which scenarios will be searched for. When an index is queried for a scenario that wasn't known to be relevant at indexing time, problems will arise inevitably. The index has to be smart or at least generic to enable such a query. This way, together with a smart retrieval, a query for any scenario would be possible. So, what is "Smart Indexing", what is "Smart Retrieval"?

A smart way to index is to index what and how it makes sense. "Smart" in the information theory context refers to reduction of redundancy. "Smart Retrieval" exploits such a Smart Index in the most efficient way, i.e. the index is being interpreted correctly. Specific domain characteristics, e.g. scene layout, as well as user restrictions are taken into account at retrieval time.

A set of other problems arises when it comes to video query by semantic keywords (or even more: free text queries). The greatest difficulty can be found in the mapping of the low-level (pixel) video representation to high-level (human) semantics. I.e. while low-level features are extracted easily, the starting point of a retrieval process is usually the high-level query by the user. The problem is illustrated by the mapping of the low-level features the computer uses on the one hand to the question posed by a human on the other. This is commonly described as "bridging the semantic gap". However, bridging the semantic gap is not only translating high-level queries to low-level features. The essence of a semantic query is understanding of the meaning behind the query. This, of course, is also user dependent, as it involves the definition of terms depending on the domain the user searches in. This has to be considered when processing a query.

Taking into account the preceding, we arrange the data according to ascending semantic level from pixel level (in the visual domain) to semantic meta data. The first level comprises the image processing algorithms. In this thesis this level will be called *level 0* as this is the beginning of the analysis and there is nothing "below" that level. The algorithms of *level 0* are those to segment moving regions, extract features like color or texture, and perform object classification. Output of these algorithms and thus of *level 0* is meta data, e.g. the position of the object in pixel coordinates of the bounding box, a defined label of the object's class etc.

The meta data output of *level 0* is the input for *level 1* on whom basic reasoning is performed. This includes detection of simple events, analysis of direction of movement or the detection of people carrying goods [Abdelkader and Davis, 2002], e.g. a backpack [Haritaoglu et al., 2001]: the reasoning

whether or whether not goods are being carried is based on the shape, which is meta data output of *level 0*.

Level 2 finally is the highest semantic level. This level incorporates the output of *level 1* together with semantic information about the scene to output a semantic description like "theft of a suitcase". *Level 2* represents how a user would describe a scene and is most likely the form in which an operator would perform a query to a retrieval system. See figure 1.1 for an overview on the different levels of semantic.

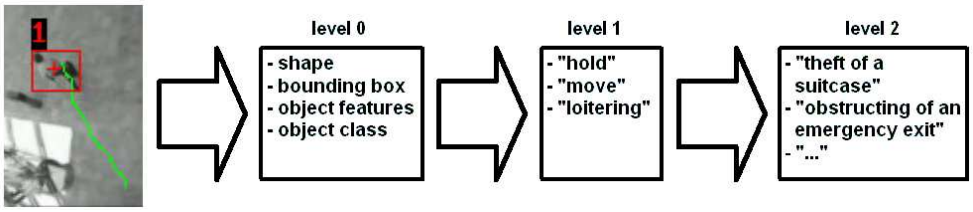


Figure 1.1: The different levels of semantics in scene descriptions.

1.2.1 An overall system

In order to create a system capable of detecting events and combining those recognized events to a semantic scene description, we need

1. a low-level analysis (VCA[1]) algorithm,
2. an ontology to lay down a common description language,
3. an inference mechanism to map the VCA meta data to a semantic label,
4. a tool/mechanism for creating an adequate map of the scene
5. a retrieval interface

These items constitute the system as follows. The VCA algorithm (item 1) analyzes the image sequence and outputs the objects' shapes, tracks, and features like color, texture, etc. Thus, it provides the *level 0* meta data. The inference mechanism (item 3) is the crux of the system: anything that isn't processed soundly here, will effect the consistence of the following stages (as well as inadequacy of the VCA). This part of the system will have to be adapted to the preceding VCA algorithm(s) employed. It outputs *level 1* meta data. The ontology for the common description language (2) is required on the one hand for the reasoning logics and on the other hand for the output

[1]Video Content Analysis

to the user and the mapping of the user's query, respectively. Furthermore, the reasoning logic exploiting the underlying ontology enables (together with item 4) the step to *level 2*. Both the ontology and the reasoning should be independent from the VCA algorithms used as this adaption is already performed when stepping from *level 0* to *level 1*. The retrieval interface (5) finally should be flexible enough to enable the user a variety of queries and map the query to the index.

The *level 0* content analysis algorithms are an important requirement for the following two levels. Still, this area is not in focus of this thesis. This thesis introduces a semantic concept for detection, representation, indexing, and retrieval of events of human and non-human actions based on the output of a VCA algorithm. Thus, the generation of *level 1* and *level 2* is the matter of interest, a generic approach for behavior understanding together with a suitable concept for indexing and retrieval.

1.2.2 Fields of application

A Smart Indexing system should be able to automatically index video content to enable efficient searches. The index has to be of little redundancy but still rich in information. The Smart Retrieval must be able to exploit the Smart Index by combining the indices intelligently and also incorporate additional information such as user restrictions and scene layout.

A Smart Indexing & Retrieval system has to be measured by its retrieval performance. So, when evaluating such a system specific events are being queried and the results are rated.

Together with security systems experts[2], a set of relevant use cases has been identified. At first, the following sub-events are to be detected: pick-up, put-down, and fall-down. The selected sub-events should be capable of composing the defined use cases. In the next step, the following use cases to be retrieved from the index were defined.

1. "Fight": a person falls caused by person passing by/person falls down and doesn't stand up again.
2. "Shopping": did the customer pay or not?
3. "Criminal behavior at an ATM": spy on another person's PIN
4. detect beggar/salesman on street.

[2]to specify, associates of the Bosch Security Systems business unit

1.3 Existing Solutions

Research in the field of automated action recognition and scene description has been going on "for some time now" [Gavrilla, 1997]. See chapter 2 for a complete overview. The following sections show systems and research work related to the presented work in a sense that a model is incorporated in the recognition process, and a (to some extend) semantic description (or label) of a scenario is being generated.

1.3.1 Model-based Event Recognition

[Hongeng et al., 2004] present a concept to represent activities and recognition methods employing this representation. An activity is being composed of action threads, each thread being executed by a single actor. A single thread event represents the characteristics of trajectory and shape. A multi-thread event is composed of several action threads related by temporal constraints and represented by an event graph, similar to interval algebra networks [Allen and Ferguson, 1994]. Mostly, multi-thread events incorporate several actors, i.e. being multi-agent events. Events are organized into several layers of abstraction. The approach is closely related to [Ivanov and Bobick, 2000] as external knowledge is incorporated into the expected structure of the activity model. This model is hierarchical but only models agents, not objects.

[Cupillard et al., 2004] present a system in the framework of the european project ADVISOR [Naylor and Attwood, 2003]. It is an approach for the online recognition of individual, group of people, or crowd behavior in metro surveillance context employing multiple cameras. A hierarchy is used for the representation of scenarios. Input for the scenario recognition are

1. scenario models defined by experts,
2. geometric information of the observed environment, and
3. persons tracked by a vision module, which is supposed to do that correctly.

The formalism is based on three main ideas:

1. define various operators (software modules) for recognition,
2. have all knowledge needed in the corresponding operator, and
3. description of the operator shall be declarative to build an extensible library of operators.

The behavior representation is actor oriented, where an actor is any scene object involved in behavior. The recognition process uses only the knowledge represented by experts through scenario models. Still, scenarios are defined "as one", thus variations have to be handled by different detector modules.

[Bourbakis et al., 2003] define a model for representing, recognizing, and interpreting human activity. The model is based on the hierarchical synergy of three other models: the Local/Global (L-G) graph, the Stochastic Petri Net (SPN) graph and a neural network (NN) model. The authors distinguish between structural knowledge (knowledge about physical state) and functional knowledge (knowledge about change and events), and they model the temporal relationship among a set of different object temporal events in the scene. They develop a Dynamically Multi-Linked Hidden Markov Model (DML-HMM) to interpret group activities involving multiple objects captured in an outdoor scene. The field of application are airport cargo activities with events like "movingCargo", being completely domain dependent.

An approach based on force dynamics is presented by [Siskind, 2001], building a system for recognizing the occurrence of events described by simple spatial-motion verbs in short image sequences. The semantics of these verbs is specified with event-logic expressions that describe changes in the state of force-dynamic relations between the participants of the event.

[Fern, 2004] extends [Siskind, 2001] by developing a supervised learning algorithm for automatically acquiring high-level visual event definitions from low-level force dynamic interpretations of video. A temporal event-description language is introduced: AMA, "*A*nd's of "*M*eet's and *A*nd's" (*MA timelines*)". An *MA timeline* is the succession of one state being true for a time interval *and* a second state being true for a second time interval *meeting* the first. *AMA* is the conjunction of *MA timelines*. Algorithms and complexity bounds are given for the AMA subsumption and generalization problems, a learning method is developed based on these algorithms and applied to learning event definitions from video.

[Ghanem et al., 2004] represent and recognize events using Petri Nets to build an interactive system for querying surveillance video about events. The queries may not be known in advance and have to be composed from primitive events and previously defined queries. Petri Nets are used as both representation and recognition methods. A graphical user interface is used to compose queries which then are mapped into a set of petri nets that represent the components of the query. [Ghanem et al., 2004] also use an event ontology defining states, events, composite events/scenarios, and relations. Objects are assumed to be provided by an intermediate vision layer.

[Xin and Tan, 2005] propose a system that integrates all related information in a hierarchical conceptual model (namely an ontology) as an approach

for event modeling and analysis with semantic representations. This system defines events as significant changes and mappings of conceptual units in the model. Three basic event components form the concept: entities, words, and a set of attributes. The lower level of the framework extracts features (the words) from the content, while the upper level semantical representations of events are received using these words. Events are treated as obvious feature changes; change is the trigger of an event. Different attributes of regions and moving objects are entities. The scene is divided into different regions that are labeled manually. To describe moving objects, the motion states move, halt, stop are used. Trajectories are characterized through 'go straight', 'turn right', 'turn left', 'retrace'. Interactions between moving objects and special regions are described through their spatial relations: occupy, enter, transfer, appear. Interactions of two moving objects are close to, away from, encounter, follow. The hierarchy of ontologies consists of three levels. The first level is the layout of the scene and associated restraints: usable features of the region "Road" include that vehicles and other moving objects can move on it. The next level contains the moving objects' ontology containing the states of motion and the concepts describing interactions mentioned above. On a final level, the semantic ontology represents what occurs in the scene. To measure the similarity of semantic concept, a method using Conceptual Status Vector and Weighted Semantic Distance (WSD) is proposed. The WSD measures the distance between two conceptual status vectors. When the semantic distance is larger than a threshold learned from training data, an event happens and a semantic representation of this event is made.

[Guler et al., 2003] present a video event detection and mining framework for event detection, annotation, and content browsing including a video analysis database. The event detection part regards events in a hierarchical three-level structure consisting of

1. the tracking data,
2. simple behaviors like "wait", "enter", or pick-up", and
3. higher-level activities constituted by these simple behaviors as "meeting", "package drop-off", or "exchange between people".

The event analysis is based on split and merge information. The second and third level of event detection are performed by a two-level Hidden Markov Model, the first (hidden level) representing the second level of the event structure.

1.3.2 Blank Spots

To recognize events, three inputs aside the video stream are necessary:

1. event domain knowledge, i.e. the knowledge about what an event "looks like" in the specific domain,
2. knowledge about the scene layout, and
3. user restrictions such as allowed duration of stay in a sensitive area.

The main challenge in semantic video retrieval is not to know at indexing time, what events will be queried. This way, it wouldn't be wise to directly index online detected complex composite multi-thread multi-agent scenarios. To be able to *retrieve* them nevertheless, their constituting sub-scenarios have to be indexed.

Reasonably, it's only the scenario-independent and thus basic components of events that can be used to enable scenario and user independent indexing. Formerly proposed concepts for behavior understanding and description are application dependent and more or less real-time alert systems. Those approaches employing taxonomies/ontologies or being modular in another way either incorporate scenario-related information and/or user restrictions in their detection algorithms. Scenario representations enabling retrieval incorporating any user restrictions might sprawl when applied as in [Ghanem et al., 2004] - a new representation for each variation. Scenario knowledge shouldn't be held in the event detectors ([Cupillard et al., 2004]) as this makes them scenario dependent and limits the search with varying parameters.

So, what's actually missing is a concept that is generic and can be applied to complex scene descriptions, and thus enables Smart Indexing. Upon this indexing efficient retrieval can be performed, even if the events to be queried were not known at indexing time (Smart Retrieval). Rules and restrictions do not belong in the index but should be set up in the retrieval step. The same goes for contextual information: which scenario represents "stealing" and which does "handover" can only be resolved by an operator retrieving the scenario due to a theft being reported.

1.4 Definition of a Semantic Concept

This work shows the application of the semantic concept that has been introduced in [Neuhaus, 2005]: the mapping of low-level analysis data to semantic labels. The labels are then condensed into generic modules. Finally, a syntax is provided for arranging these modules to a semantic scene description which

can cover simple to complex scenes. By using the modules as an index structure, it is shown how a retrieval system exploiting the possibility of modeling scenarios by arranging the elements can be build. This way, the presented system makes a contribution to the problematic nature of the SIR topic. It is shown how this concept can be applied to a variety of scenarios of scene recognition, description and retrieval. Not only the retrieval application as a "pull-application" of such an event recognition system can be build using the concept evolved here but also the "push"-variant, a real-time-alert system. Most importantly, it is possible to do the advanced reasoning based on the smart index at retrieval time. The presented concept is event-oriented, not actor-oriented: a specific actor becomes of interest no sooner, as (s)he performs a specific action.

The goal of this work is to build a system for indexing *and* retrieval of video surveillance content. Such a system should be adoptable to any VCA algorithm and thus be independent or easily be to be separated from a specific algorithm. To illustrate this, see figure 1.1 for the functional distinction of the generation of enriching semantic information. This work follows the motion detection & tracking algorithm.

1.5 Organization of this work

The remainder of this thesis is organized as follows. Chapter 2 briefly summarizes related work addressing the problem of behavior recognition and scene description in natural language. In chapter 3, the principles and tools utilized in this thesis are listed and explained. Chapter 4 presents the approach to solve the SIR challenge. Section 4.2 introduces the terminology used in that chapter, section 4.3 introduces the constituting elements of the concept, the Event Morphemes. Section 4.6 shows examples for the usage of the presented concept. Section 4.4 shows how Event Morphemes are applied, and section 4.7 shows the realization of Event Morphemes. Chapter 5 presents the results of the concept when processing the use cases and the comparison with related work (sections 5.4.1 and 5.5, respectively). Chapter 6 summarizes the work and points out how the identified weak point of the implementation of the presented concept can be met.

Chapter 2

Recent Advances in the Field of Analysis and Modeling of Object Motion and Behavior

2.1 Introduction

According to [Hu et al., 2004], the process of analyzing, understanding, and describing the content of interest includes these stages: "modeling of environments, detection of motion, classification of moving objects, tracking, understanding and description of behaviors, human identification, and fusion of data from multiple cameras". Figure 2.1 shows an overview on the general framework referred to in [Hu et al., 2004].

The hierarchical arrangement of levels of semantic suggests the order of processing the data to be analyzed using the stages of [Hu et al., 2004]: *level 0* covers 'detection of motion', 'classification of moving objects', 'tracking', 'human identification', and 'fusion of data from multiple cameras', *level 1* deals with the 'understanding of behaviors' and *level 2* incorporates the preceding stages together with the 'modeling of environments' to the 'description of behaviors' and above. See figure 1.1 for an overview on the different levels of semantic.

This chapter is organized as follows. Section 2.2 introduces the techniques applied for event detection so far. Section 2.3 gives an introductory survey on systems deriving simple event or action information directly from motion detection and tracking algorithm, where the detection of events rather is a "side product". Section 2.4 covers systems addressing the task of event and action detection without having an underlying model or semantic representation. Section 2.5 summarizes systems employing models of behavior and/or

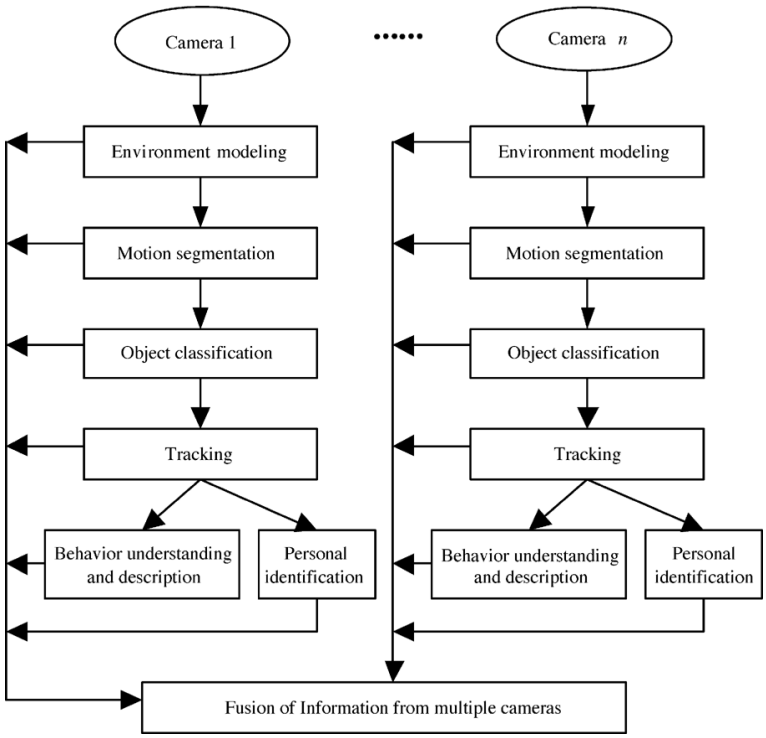


Figure 2.1: General framework of visual surveillance (taken from [Hu et al., 2004]).

scenarios for event recognition and thus being closely related to the topic of this thesis. Section 2.6 presents ontologies and ontology languages for video event descriptions.

2.2 Overview on applied techniques

2.2.1 Probabilistic and Stochastic Techniques

The main characteristic of probabilistic and stochastic techniques is to model explicitly uncertainty using numbers. The section starts by describing Bayesian Classifier techniques, then Neural Networks techniques. Both techniques are adapted to model the uncertainty in the recognition of events depending of visual features at a given time. With Bayesian classifiers, the combination is inferred from the frequency of the observations of events in function of visual features. With Neural Networks, the combination is stochastically adjusted by improving the recognition over a learning set of samples. Finally, Hidden Markov Model techniques applied to human activity recognition are

described. These techniques are usually used to recognize sequences of events.

Bayesian Classifier

In Bayesian analysis, the state of knowledge about the parameters x associated with a model that describes the physical object being studied is summarized by the posterior, which is the probability density function $p(x \mid d)$ of the parameters given the observed data d. Bayes law gives the posterior as

$$p(x_j \mid d) = \frac{p(d \mid x_j)p(x_j)}{\sum_j p(d \mid x)p(x_j)} \tag{2.1}$$

The probability $p(d \mid x)$, called the likelihood, comes from a comparison of the actual data to the data predicted on the basis of the model of the object. The predicted data are generated using a model for how the measurements are related to the object, which we call the measurement model. The prior $p(x)$ expresses what is known about the object, exclusive of the present measurements, and may represent knowledge acquired from previous measurements, specific information regarding the object itself, or simply general knowledge about the parameters, e.g. that they are non-negative. Bayes law says that for a given object model the posterior can be evaluated by combining the likelihood, which requires the data values predicted for that object model, and with the numerical value of the prior. This calculation usually is straightforward. It involves calculating the predicted measurements for the given object model, which we refer to as the forward measurement calculation.

Dynamic scenes are an uncertain environment. Thus Bayesian classifiers are applicable to this problem. If the variables (i.e. object attributes) are conditionally independent, a naive classifier can be used and the Bayesian rule is used to infer the object class. Such a classifier needs to learn the parameters, e.g. $p(ratio \mid car)$ and $p(ratio \mid non{-}car)$. The main advantage of Bayesian classifiers is their capability to model the uncertainty of the recognition by using probabilities. However, there are two drawbacks. First, the a priori probability needs to be learned. Due to the construction of the learning sets this learning stage is time consuming. Secondly, the time when the visual features have to be computed needs to be indicated explicitly. Thus, they are not adapted to model temporal relations.

Hidden Markov Model (HMM)

As Bayesian classifiers, HMMs are also used to model uncertainty of the observed environment and in particular, the uncertainty of temporal relations of events. The principle of this approach is to use the Makovian hypothesis: the probability of being in a given state only depends on the probability of being in the direct previous state. The advantage of HMMs compared to Bayesian classifier and Neural Networks is the ability to recognize sequences of events. However, they are limited in the way they recognize sequences of events where several mobile objects are involved. The probability of being in a state for a mobile object has to be combined with the probability of being in another state for all other mobile objects.

Neural Networks

In essence, neural networks are mathematical constructs that emulate the processes people use to recognize patterns, learn tasks, and solve problems. Neural networks are usually characterized in terms of the number and types of connections between individual processing elements, called neurons, and the learning rules used when data is presented to the network. Every neuron has a transfer function, typically non-linear, that generates a single output value from all of the input values that are applied to the neuron. Every connection has a weight that is applied to the input value associated with the connection. A particular organization of neurons and connections is often referred to as a neural network architecture. The power of neural networks derives from their ability to learn from experience (that is, from historical data collected in some problem domain).

Human behaviors evolve normally in an uncertain environment thus neural networks techniques have been used to cope with this problem. However, it is not efficient to handle complex behaviors involving a large number of physical objects and complex temporal constraints (e.g. synchronized constraint) because it leads to a combinatorial explosion of possible behaviors corresponding to all combinations of physical objects detected in the scene.

2.2.2 Symbolic Techniques

This section presents symbolic techniques for human activity recognitions. These techniques aim at transforming numerical observations into symbolic scenarios.

Automata

Recently, automata have been used to recognize human behaviors in video sequences. Several numerical techniques are used to recognize video events up to the event level (e.g. numerical calculations of basic properties of physical objects, comparison of states at two consecutive instants to recognize events). At the Scenario level, they use an automaton approach for recognizing pre-defined Scenarios. To recognize a Scenario M, the Scenario recognition process creates an automaton representing the Scenario M. The states of this automaton correspond to the states/events/sub scenarios composing M. The transitions of this automaton correspond to the constraints defined between two states. This approach has the advantage of reusing the Scenarios partially recognized at previous instants instead of recalculating them at each instant. Moreover, it also shows the capacity of predicting which Scenarios will happen in the observed scenes. However, it has several drawbacks. For example: (1) if a Scenario M is defined with several physical objects, the Scenario recognition process has to create all the automata corresponding to all combinations of physical objects defined within M for the recognition of M. Moreover, the number of states of a Scenario increases in function of the number of physical objects involved in the Scenario, because these physical objects can evolve in many different Situations.

2.2.3 Petri Net

A Petri Net consists basically of four components, the places, the transitions, the arcs, and the tokens. The tokens are put in the places modeling states. The transitions are enabled when the preceding places are filled sufficiently. The transitions are used to model events that change states. Arcs are the connections between places and transitions.

The advantages of Petri nets are: (1) the capacity of sequencing, parallelism and synchronization, (2) Petri nets allow monitoring and prediction. However, this technique can lead the recognition process to a combinatorial problem when coping with temporal Scenarios defined with several physical objects and with scenes composed of a large number of mobile objects. Moreover, some temporal constraints (e.g. "person B arrives 1 minute after person A left") are difficult to express using this formalism.

2.3 Action-oriented motion analysis and tracking

"The aim is to develop intelligent visual surveillance to replace the traditional passive video surveillance that is proving ineffective as the number of cameras exceeds the capability of human operators to monitor them. In short, the goal of visual surveillance is not only to put cameras in the place of human eyes, but also to accomplish the entire surveillance task as automatically as possible."[Hu et al., 2004]

[Aggarwal and Cai, 1997] state that for human activity recognition the advantage using the template matching technique was its inexpensive computational costs while being relatively sensitive to the variance of the movement duration. Three major areas related to interpreting human motion are defined:

1. motion analysis involving human body parts,
2. tracking of human motion using single or multiple cameras, and
3. recognizing human activities from image sequences.

[Cutler and Davis, 2000] present a system that analyzes periodic motion by segmenting the motion and tracking objects in the foreground. Objects are aligned along the temporal axis. The object's self-similarity as it evolves in time, is being computed. The system also classifies objects using periodicity:

"people",
"dogs" and
"other"

are the distinct classes.

In [Haritaoglu et al., 2000], the real-time visual surveillance system W^4 is introduced. It uses a combination of shape analysis and tracking and constructs models of people's appearances. By this, the system detects and tracks groups of people and watches their behaviors. The System handles occlusion and outdoor environments. A single camera with a gray scale sensor is used. W^4 employs a background model to reduce the influence of changes in dynamic scenes derived from lighting etc.

Residual flow is used in [Lipton, 1999] to analyze rigidity and periodicity of moving objects. It is assumed that rigid objects present little residual flow and non-rigid moving object, e.g. a human being has a higher average residual flow and additionally shows a periodic component. Based on

this, human motion is distinguished from motion of other objects, such as vehicles. To recognize human dynamics in video sequences, [Bregler, 1997] builds motion models of human limbs and joints, which are widely used in tracking ([Hu et al., 2004]). They are effective because the movements of the limbs are strongly constrained. These motion models are employed as a priori knowledge to interpret and recognize human behaviors. Human behavior is decomposed into multiple abstractions, and represents the high-level abstraction by HMM's built from phases of simple movements. This representation is used for both tracking and recognition. [Zhao et al., 2002] use motion models of human limbs and joints, too. A highly structured motion model for ballet dancing under the minimum description length (MDL) paradigm is learned. This motion model resembles a finite-state machine (FSM).

In [Rao and Shah, 2001], a view invariant representation of action consisting of dynamic instants and intervals is presented which is computed using spatio-temporal curvature of a trajectory. This representation is then used to learn human actions without training. Focus of the system are human actions performed by a hand. The trajectory of a hand is represented by a sequence of dynamic instants and intervals. A dynamic instant is an instantaneous entity, which occurs for only one frame, and represents an important change in motion characteristics: speed, direction, acceleration, and curvature. An instant is detected by identifying maxima in spatio-temporal curvature. An interval represents the time-period between two dynamic instants, during which the motion characteristics remain constant. Instants and intervals have physical meanings. Therefore, it is possible to explain an action as a sequence of meaningful instants and intervals. Dynamic instants include

"touching",
"twisting",
"loosening".

Intervals include

"approaching",
"lifting",
"pushing",
"receding".

Example scenarios are

"opening/closing overhead cabinet",

"picking up/putting down a book/phone", and

"erasing a whiteboard".

The subject of tracking in multiple cameras is addressed in [Javed et al., 2000]. The system presented uses spatial relationships between view fields of cameras to establish corresponding relationships of images. [Krumm et al., 2000] use color histograms to match regions. [Brand et al., 1997] use Coupled Hidden Markov models for the recognition of action.

[Bobick and Davis, 2001] use a temporal template for the recognition of human movement – a static vector-image where the vector value at each point is defined as a function of the motion properties at the corresponding spatial location in the image sequence. People carrying objects are detected in [Abdelkader and Davis, 2002] by looking at factors affecting gait perception, e.g., clothing, environments, distance, carried objects such as briefcases. [Haritaoglu et al., 2001] use silhouettes to determine whether people are carrying objects or moving unencumbered. The employed shape analysis algorithm determines whether a person is carrying an object and segments the object from the person so that it can be tracked, e.g. during an exchange of objects between two people. [Cunado et al., 1997] model gait as the movement of an articulated pendulum and use the dynamic Hough transform to extract the lines representing the thigh in each frame. The least squares method is used to smooth the inclination data of the thigh and to fill the missing points caused by self-occlusion of the legs. Phase-weighted magnitude spectra are used as gait features for recognition.

2.4 Scenario analysis

This section covers approaches addressing the scenario analysis problem without underlying models or semantic representations.

In [Ersoy et al., 2004], events are formulated using domain-independent event primitives represented by spatio-temporal relationships between objects. Complex events are expressible as combinations of simple events. The trajectories of objects serve as "atomic entities". The syntax of the event description in the application of a parking lot uses the syntax of [Allen, 1983]:

$$\begin{aligned} drop\_off(x,y) \equiv\ & enterlot(x) \wedge_{<} \\ & ((stop(x) \wedge_{o} exits(y) \wedge_{o} (dist(x,y) < d)) \wedge_{di} leavelot(x) \end{aligned}$$

[Stauffer and Grimson, 2000] employ real-time tracking to learn activity patterns. It is stated that because of the stability and completeness of the representation it was possible to do simple classification based on aspect ratio or size.

In [Piater et al., 2002], the described events are

> "NewTarget"
>
> "ConfirmTarget"
>
> "MoveTarget"

etc., i.e. description of tracker output, and not semantically expressions. From those events, simple scene descriptions are generated.

[Davis, 2004] addresses the problem of the time necessary to identify human actions. The actions in focus are

> "walking",
>
> "running", and
>
> "standing".

A reference framework using the "key feature" from [Jepson and Richards, 1991] is employed and a continual verification process of the selected object likelihood once the action has been detected is proposed.

[Ayers and Shah, 2001] extensively use prior knowledge. This knowledge is employment in tracking, skin detection, and action recognition. As an example,

> "use terminal"

is defined as

> "person sitting near terminal" and "scene change" is detected in mouse but not behind it.

"In mouse" describes the region the (computer) mouse is in.

[Makris and Ellis, 2003] automatically train an activity-based semantic scene model for a surveillance region. The semantic scene models defines regions of activity in the camera view. Regions where particular types of motion-related activity are located have been trained from target trajectories generated from tracking objects through the environment. Those regions are

zones of entry/exit, paths, routes, and junctions. A textual description of the activity based on this system is:

> "At the time t1, pedestrian #440 enters the scene at entry point 1, moving along path A. At junction 2 he/she chooses path B and exits the scene at exit point 3"

[Dee and Hogg, 2005] present a system that analyzes behavior in a surveillance setting. The "goal" of an agent within the scene is evaluated by a subgoal algorithm. If he moves towards an entry/exit he can "see", the behavior is classified as being expected. The more the agent doesn't behave like that, the higher the inexplicability score is. As an application, the system is used to highlight interesting actions to a surveillance operator.

[Toshev et al., 2006] present an algorithm that processes a set of primitive events such as simple spatial relations between objects obtained from a tracking system and outputs frequent event patterns. This work focusses on the problem of detecting frequent complex activities without a model. An event is a spatio-temporal property of an object in a time interval or a change of such a property. Events are formally defined in an event description language ([Brémond et al., 2004]) which enables the definition of complex events in terms of simpler ones and this way build hierarchical structures of events. Such simple events are

> "a vehicle on the road",
>
> "a vehicle on the parking road",
>
> "vehicle on a parking place", and
>
> "person coming out of the vehicle",

building the complex scenario

> "parking manoeuvre".

Thus, primitive events are

> "object in a zone" or
>
> "object near another object".

For the detection of complex events, the data mining Apriori algorithm is adapted which uses the so called Apriori property: the subpatterns of frequent patterns are also frequent. As this property does not hold in case of

similarity because subpatterns of patterns can be less similar than the patterns themselves, a WEAK-APRIORI property is formulated. It decreases the frequency threshold for shorter patterns in order to prevent losing subpatterns of frequent patterns and thus to guarantee their detection in the merge step. The context knowledge is separated from the algorithm to make the approach applicable in different domains. Field of application is the parking lot monitoring domain.

[Porikli and Haga, 2004] introduce a set of time-wise and object-wise statistical features as trajectories, histograms, and HMM's of speed, orientation, location, size, and aspect ratio to build an event detection framework. Not predefined models are mapped to events, but unusual events are detected by analyzing the conformity scores. Thus, usual and unusual are not predefined: usual is laid down as

> "the high recurrence of events that are similar".

(Simulated) scenarios are

> "an object moving in opposite direction to the rest",
>
> "a waiting object where other objects moves", and
>
> "a fast moving object".

[Nascimento et al., 2005] describe an algorithm for segmenting and classifying human activities from video sequences of a shopping center as

> "entering/exiting the shop",
>
> "passing, or browsing in front of a shop window".

These activities are recognized by using a priori knowledge about the layout of the scene.

[Fuentes and Velastin, 2005] present an event detection algorithm based on motion trajectories. Position, speed and people density are used to create low-level representations of predefined events. A semantic descriptions is associated to these events. The system then raises alarms to the surveillance operator. Events detected are

"unattended luggage",

"falls",

"people hiding",

"vandalism",

"fights",

"intrusion in forbidden areas", and

"attacks",

the latter being pre-stages to fight, indicated by a person getting too close to another person by entering her/his social or personal zone.

[Lou et al., 2002] propose a framework for semantic interpretation of vehicle and pedestrian's behaviors. The object trajectories are analyzed using dynamic clustering and classification on which the high level semantic is based. Spatial information is used to combine trajectories into clusters and then dynamic information is employed to arrange the trajectories in each cluster into classes. Those are

"Move Forward",

"Turn Right",

"Turn Left", and

"Stop".

The natural language description follows the rule

> "(The Obj) (Action) in (The place name) [at (high/low/middle) speed]".

The places' names are hand labeled. The application is visual traffic surveillance.

2.5 Event detection and representation

In this section those systems employing an event model to some extent are presented.

[Pinhanez and Bobick, 1998] define a representation for the temporal structure inherent in human actions and a method for using that representation to detect occurrences of actions. The hierarchy developed starts at

the "sensor information" which is followed by "events" constituting "sub-actions" and finally "actions". The temporal structure, the *PNF Propagation* (past - now - fut[1]), is employed to detect and remove inconsistent situations. To employ PNF, interval algebra constraint networks (IA-networks, [Allen, 1984]) are used. Those IA-networks are mapped into PNF-networks. A PNF network is a binary constraint satisfaction network where the domain of all variables is the set of symbols $m =$ `past, now, fut`. The actions presented are

"pick-up bowl",

"wrapping chicken", and

"mixing ingredients".

"pick-up-bowl", i.e, is defined as

"reach-for-bowl" AND "grasp-bowl".

The sensor information is e.g. coded as

"DET:hands-close-sta-bowl" or "DET:bowl-on-table".

To recognize multi-agent action [Hongeng and Nevatia, 2001] create a framework for representing and visually recognizing complex multi-agent action. "Complex" is an action that contains many components occurring in (typically) a partially ordered temporal relation to one another and that are subject to certain logical constraints. "Multi-agent" results in parallel event streams that interact in temporal (typically causal) ways. The task and the domain developed is recognizing American football plays. The approach is driven by the idea of [Grimson and Lozano-Pérez, 1985], that massive low order consistency typically implies correctness. The representational elements are

1. to define a *temporal structure description* of the global behavior ("individual", "local goals", "events") with relations coded as temporal constraints,
2. to define for each basic element a *visual network* that detects the occurrence of goals or events, and
3. to construct a multi-agent belief network reflecting the temporal structure of the action.

[1] for "future"

Temporal relations are expressed with Allen's interval algebra [Allen, 1983].

[Park and Aggarwal, 2004] use event hierarchy as a method to represent two-person interactions at a semantic level with a natural language description. Interactions consist of two single-person actions, which consist of torso and arm/leg movement. For the representation, those triplets are used:

<agent - motion - target>

The system performs its reasoning based on body-part gestures [Park and Aggarwal, 2003] as an elementary event of motion being composed of a sequence of instantaneous poses at each frame. The interaction hierarchy is defined as

interaction - action - gesture (dynamic) - pose (static).

The transformation rules are determined by domain-specific knowledge about human interactions. Examples of human interactions are "hugging" and "punching".

[Kojima et al., 2002] are describing activities by tracking skin regions (facial and hands). By associating visual features of head and hand motion with natural language concepts, syntactic components such as verbs, objects, etc. are determined and translated into natural language. The position of the head implies not only the position where a person is but also a posture whether (s)he is standing or sitting. The direction of the head implies what (s)he is looking at, and the positions of the hands imply gestures and interactions with objects. Other components necessary for a sentence are determined using knowledge about objects and equipment in a scene. A concept hierarchy of body actions is defined from "be" as the root, "move" and "not move", specialized by "move slow/ fast" and "stay high/low", respectively.

[Gritai et al., 2004] propose an approach to matching human actions using semantic correspondence between human bodies. They make implicit use of the laws governing body proportions to derive geometric constraints for matching. Instead of using a single point for representation, the usage of several points on the actor for action recognition is being explored. Instead of two camera views, geometric constraints with respect to two actors performing an action are used. Each point represents the spatial coordinate for an anatomical landmark on the human body. Eight points are required in each frame; at least one must correspond to the body part directly involved in the action. Actions are recognized by measuring the similarity of posture at each corresponding time instant. (Staged) scenarios are

"walking",

"bending down to grasp an object",

"lifting the object",

"walking away", and

"the 'Egyptian' gait".

In [Stern et al., 2003], a system using Fuzzy Expert System models to describe a scene is presented. The terms of description are the number of people and people groups in the scene, their actions as

"walking toward/away from the camera",

"standing still",

"departing from another person",

"walking with another person", or

"joining a group".

The object classification is performed via a Static Expert Model, employing fuzzy rules such as

"if Area $= very - small$ Then $not - a - person$".

For action identification Dynamic Expert System models are used, e.g.

"if X-movement $= slightly - right$ and Y-movement $= almost - no - change$ Then Velocity $= standing$ and Direction $= none$".

[Xiang et al., 2002] present an approach for modeling temporal events on the local intensity temporal history of pixels. Pixel-level events like

"car stopping",

"people browse", or

"object removal"

are detected by Pixel Change History with a background model. On the next hierarchic level, blob-level events are detected using clustering via Expectation-Maximization algorithm. As use case, a shopping scenario is presented:

"take a can and exit without paying".

The scenario is represented by the event classes

"can taken",

"entering and leaving",

"shop keeper",

"browsing", and

"paying".

The event classes are learned automatically and labeled manually.

In [Ivanov and Bobick, 2000] the recognition of activities and interactions between multiple agents is addressed. The recognition problem is hereby divided into two levels:

1. the independent probabilistic event detectors to propose candidate detections of low-level features and
2. taking the output of level 1 as input for a stochastic context-free grammar parsing mechanism.

The advantage of the second level employed is described as to provide longer range temporal constraints, disambiguate uncertain low-level detections, and allow the inclusion of a priori knowledge about the structure of events in a given domain. Level 1 is for the recognition of primitives (statistical), while level 2 recognizes structure (syntactical) of the scene. The system is applied to a parking lot scenario:

"driving into the parking lot and leaving the parking lot on foot",

"people being dropped off/picked up" etc.

The scenarios are constructed from the primitives detected by the tracking event generator:

"car-enter",

"person-enter",

"car-found",

"person-found",

"object-lost", and

"object-stopped".

Thus, these symbols consist of tracking system output and contain little or no semantic labels.

The IBM Smart Surveillance System [Hampapur et al., 2004] (formally PeopleVision) is designed with the intent of making currently developed surveillance systems "smart". The system assumes largely static cameras. Upon the inputs of these cameras, real-time video based alerts are generated:

1. motion detection: movement of any object within a specified zone,
2. directional motion detection: specific direction of movement,
3. abandoned object alarm: objects which are abandoned,
4. object removal: movements of a user-specified object that is not expected to move, and
5. camera move / blind: when the camera has been tampered with.

These alerts are solely based on movement. In addition to real-time alerts, a viewable video index for Automatic Forensic Video Retrieval (AFVR) is generated, containing indices for the number of objects, classification (single person, group of people, vehicles), object properties (color, texture, shape, size), movement properties (position, velocity, trajectory), occlusion parameters (when objects are occluded), background changes due to changes in lighting and stopping of moving objects, and event information: any events that may be flagged by the engine.

[Guler et al., 2003] present a video event detection and mining framework for event detection, annotation, and content browsing including a video analysis database. The event detection part regards events in a hierarchical three-level structure consisting of

1. the tracking data,
2. simple behaviors like

   "wait",
   "enter", or
   pick-up" and

3. higher-level activities constituted by these simple behaviors as

   "meeting",
   "package drop-off", or
   "exchange between people".

The event analysis is based on split and merge information. The second and third level of event detection are performed by a two-level Hidden Markov Model, the first (hidden level) representing the second level of the event structure.

[Foresti et al., 2004] present a system for event classification in parking lots. They distinguish simple and complex events. Simple events are "moving objects", e.g.

"vehicle moving in an allowed area" or

"pedestrians walking with typical trajectories".

Complex events are a "set of temporally consecutive simple events". The object trajectories are approximated by Bezièr Curves. Objects are classified as "vehicles" or "pedestrians". Event recognition is executed by taking into account that an event is characterized by a set of classified objects over a sequence of consecutive frames. Three types of alarms are generated:

normal events,

suspicious events

"pedestrians walking with trajectories not always rectilinear",
"pedestrians moving around a vehicle", and

dangerous events

"pedestrians/vehicles moving/stopping in not allowed areas",
"pedestrians moving with atypical trajectories".

An offline Event Database contains models of those types. The features are the object class and the parameters of the Bezièr fitting. These are used as input for the training of an AHNT (adaptive high order neural tree). An Active Event Database is storing detected simple events. Old events are eliminated via an age counter. An automatic procedure checks if some of these events are spatially or temporally related. If so, a composite event is generated. Composite events are

"vehicle entering",

"vehicle moving",

"person exiting vehicle",

"person moving", or

"person exiting parking area".

Composite events are defined by the operator.

[Hongeng et al., 2004] present a concept to represent activity and recognition methods employing this representation. An activity is being composed of action threads, each thread being executed by a single actor. A single thread event represents characteristics of trajectory and shape, e.g. "approaching a reference person" or "heading toward". A multi-thread event is composed of several action threads related by temporal constraints and is represented by an event graph, similar to interval algebra networks [Allen and Ferguson, 1994]. Mostly, multi-thread events incorporate several actors, i.e. being multi-agent events. Events are being organized into several layers of abstraction (from [Medioni et al., 23]). The approach is closely related to [Ivanov and Bobick, 2000] as external knowledge is incorporated into the expected structure of the activity model. For the scenario recognition, three steps are employed:

1. detect and track moving objects,
2. compute object properties using "User Provided Context" (spatial and task context), and
3. in parallel match scenarios to a "Library of Event Models".

Scenarios are defined from a set of properties or sub-scenarios building a hierarchical structure. The event representation at the scenario level maps to how a human would describe events. Single-thread events are recognized by naive Bayes classifier, complex ones by Bayesian Networks, as well are multi-thread events. Presented are the events

"stand" and

"crouch"

as shaped-based events,

"approach",

"stop at", and

"slow down"

based on trajectory and

"moving along the path"

with regard to the geometrical zone.

[Cupillard et al., 2004] present a system performed in the framework of the european project ADVISOR [Naylor and Attwood, 2003]. It is an approach for the online recognition of individual, group of people, or crowd behavior in metro surveillance context employing multiple cameras. For the representation of scenarios a hierarchy is used, deriving from an "Entity" the "Scenario" with "State" and "Event" on the one branch and "Scene-Object" on the other, with "Static" ("Equipment", "Zone") and "Mobile" ("Person") as its specializations. Input for the scenario recognition are

1. scenario models defined by experts,
2. geometric information of the observed environment, and
3. persons tracked by a vision module, which is supposed to do that correctly.

The system employs a "graph of solutions" for each person with nodes like

"close_to",

"far_from",

"moves_close_to",

"moves_away_from",

"stays_at", and

"vandalism".

The formalism is based on three main ideas:

1. define various operators (software modules) for recognition,
2. have all knowledge needed in the corresponding operator, and
3. description of the operator shall be declarative to build an extensible library of operators.

The behavior representation is actor oriented, where an actor is any scene object involved in behavior: static objects, zones of interest, individuals, group of people, or crowd. For each tracked actor, the behavior recognition module performs three levels of reasoning: "states", "events" and "scenarios".

The recognition process employs four concepts:

basic properties (trajectories, speed, direction),

states (a situation characterizing an actor at a certain time as "an individual is walking" or "the trajectory is straight"),

events (change of state, e.g. "a group enters a zone of interest"), and

scenarios as the combination of states, events, or sub-scenarios.

A framework of operators is used: to recognize a scenario, the operators are arranged hierarchically: the bottom is composed of states while the top (after possible intermediate levels) corresponds to the scenario to be recognized. The output of the operators are boolean: the event is either detected or not detected. The scenarios presented are

"fraud" (jumping over the turnstiles),

"fighting",

"blocking",

"vandalism", and

"overcrowding".

Behaviors are specific scenarios defined by the user. In [Vu et al., 2003], these scenarios are "pre-compiled" for better recognition. Input for the system is contextual a priori knowledge (scenario models, geometric & semantic information about the scene) and the video stream. The recognition process uses only the knowledge represented by experts through scenario models. For each model of a scenario instance, the set of actors with actor variables, the set of sub-scenario instances (elementary scenarios, composed scenarios), and the set of constraints are defined. The system is applied to

"Bank attack" and

"Vandalism against a ticket machine".

[Bourbakis et al., 2003] define a model for representing, recognizing and interpreting human activity. The model is based on the hierarchical synergy of three other models: the Local/Global (L-G) graph, the Stochastic Petri Net (SPN) graph, and a neural network (NN) model. The authors make a distinction between structural knowledge (knowledge about physical state) and functional knowledge (knowledge about change and events) and model the temporal relationship among a set of different object temporal events

in the scene. They develop a Dynamically Multi-Linked Hidden Markov Model (DML-HMM) to interpret group activities involving multiple objects captured in an outdoor scene. The field of application are airport cargo activities with events like

> "movingTruck",
>
> "movingCargo",
>
> "movingCargoLift", or
>
> "movingTruckCargo".

An approach based on force dynamics is presented by [Siskind, 2001], building a system for recognizing the occurrence of events described by simple spatial-motion verbs in short image sequences. The semantics of these verbs are specified with event-logic expressions that describe changes in the states of force-dynamic relations between the participants of the event. The example of "A hand is picking up a block" is described as follows.

> A *pick up* event is characterized as a change from a state where the patient is supported by a substantially constraint with the source to a state where the patient is supported by being attached to the agent.

[Fern, 2004] extends [Siskind, 2001] by developing a supervised learning algorithm for automatically acquiring high-level visual event definitions from low-level force dynamic interpretations of video. A temporal event-description language is introduced: AMA, "*A*nd's of "*M*eet's and *A*nd's" (*MA timelines*)". An *MA timeline* is the succession of one state being true for a time interval *and* a second state being true for a second time interval *meeting* the first. *AMA* is the conjunction of *MA timelines*. Algorithms and complexity bounds are given for the AMA subsumption and generalization problems, a learning method is developed based on these algorithms and applied to learning event definitions from video.

[Ghanem et al., 2004] represent and recognize events using Petri Nets to build an interactive system for querying surveillance video about events. The queries may not be known in advance and have to be composed from primitive events and previously defined queries. Petri Nets are used as both representation and recognition methods.

> "For simple activities whose structure is known in advance and can be easily learned from training data, stochastic inference can

> be used. On the other hand, for higher level events that include temporal combinations of other events, deterministic inference seems preferable."

A graphical user interface is used for formulating queries. These queries are then mapped into a set of petri nets that represent components of the query. [Ghanem et al., 2004] also uses an event ontology defining states, events, composite events/scenarios, and relations. Objects are assumed to be provided by an intermediate vision layer. An event is represented by a transition in the Petri Net; a primitive event by a conditioned transition (with the condition that the primitive event has been detected), a composite event by a hierarchical transition (a predefined Petri Net being used as one block) whose structure is derived from the event structure. The system is used on a parking lot, with events like

"counting cars" and

"car exchange".

[Hakeem et al., 2004] propose a representation of events in videos, based on the CASE representation of natural languages. They point out the importance of causal and temporal relationships between sub-events. In order to capture multi-agent and multi-threaded events, a hierarchical CASE representation of events, CASE^E, is developed. By mapping scenes to an event-tree, events are recognized via sub-tree matching. The presented scenario is railroad crossing.

[Hakeem and Shah, 2004] propose a framework for classification of meeting videos. This framework is utilized to analyze human motion data to perform automatic meeting classification. A rule-based system and a state machine are employed to analyze the videos, utilizing three levels of context hierarchy; movements and their attributes, events (=actions), and behavior. By these, activities are identified and the meeting type is classified, based on the meeting ontology. The rule-based system is the primary framework manager, which recognizes behaviors based on the events detected by the state machine. In the ontology, the relationships between movements form events that have a relationship with each other to form behaviors. Different behaviors form genres and the meeting ontology.

As an approach for event modeling and analysis with semantic representations [Xin and Tan, 2005] propose a system that integrates all related information in a hierarchical conceptual model (namely an ontology) and defines events as significant changes and mappings of conceptual units in the model. Three basic event components form the concept:

entities,

words, and

a set of attributes.

The lower level of the framework extracts features (the words) from the content, while the upper level semantical representations of events are received using these words. Events are treated as obvious feature changes; change is the trigger of an event. Different attributes of regions and moving objects are entities. The scene is divided into different regions that are labeled manually:

grassplot,

road,

sideway,

intersection,

crosswalk.

To describe moving objects, the motion states

“move”,

“halt”,

“stop”

are used. Trajectories are characterized through

“go straight”,

“turn right”,

“turn left”,

“retrace”.

Interactions between moving objects and special regions are described through their spatial relations:

“occupy”,

“enter”,

“transfer”,

“appear”.

Interactions of two moving objects are

> "close to",
> "away from",
> "encounter",
> "follow".

The hierarchy of ontologies consists of three levels. The first level is the layout of the scene and associated restraints: usable features of the region "Road" include that vehicles and other moving objects can move on it. The next level contains the moving objects' ontology containing the states of motion and the concepts describing interactions mentioned above. In a final level, the semantic ontology represents what occurs in the scene. To measure the similarity of semantic concept, a method that uses Conceptual Status Vector and Weighted Semantic Distance (WSD) is proposed. The WSD measures the distance between two conceptual status vectors. When the semantic distance is larger than a threshold learned from training data, an event happens and a semantic representation of this event is created.

2.6 Ontologies for Video Events

[Nevatia et al., 2004] define a formal language for describing an ontology of events, VERL, (Video Event Representation Language) and a language to annotate instances of the events described in VERL: VEML (Video Event Markup Language). Different types of composition of events are defined:

> "if our hypothesis of decomposing complex events into simpler events is valid and if the number of primitive events is limited, we can overcome the complexity of representing the wide variety of events seen in the real world."

Objects have properties, attributes, and relations. States are defined by the object's properties, attributes, and relations at a given time. Events can occur at a time instant or interval. The ontology defines types, derived subtypes, expressions, and operators to describe events. As an example for syntax and inference rules, "carry" is presented/defined as in equation 2.2:

$$PROCESS(carry(x, y, a, b, t), \\ AND(hold(x, y, t), move(x, a, b, t))) \quad (2.2)$$

Primitive events of a mobile object can be

"speed-up",
"slow-down",
"start",
"stop",
"turn-right", or
"turn-left".

VERL is a programming language. VEML encodes instances of VERL in XML. See section 3.2.3 for the syntax of VERL.

[Brémond et al., 2004] introduce another ontology to represent video event knowledge for automatic video interpretation. Two main concepts are embodied:

1. physical objects and
2. video events.

A video event can be

a primitive state,

a composite state,

a primitive event, or

a composite event.

Primitive states are atoms to build other concepts. A primitive event is a change of state. A physical object can be a static object, like a desk or a mobile object, like a person or a car. They have a class, attributes and "liveliness", the ability to either be moved and/or move by initiating their own movement, then being mobile objects. Contextual objects are walls, doors, chairs, suitcases, etc. The relationships between the presented concepts are either vision based, spatial, or spatio-temporal. A syntax to describe these concepts is also proposed.

[Allen, 1983] introduces an interval-based temporal logic and a reasoning algorithm based on constraint propagation. To maintain temporal relations, thirteen relationships to express any relationship which can hold between two intervals are defined:

1. BEFORE
2. MEET
3. OVERLAP
4. DURING
5. START
6. FINISH

and their inverses, and additionally

13. EQUAL

See figure 2.2 for a visual representation of those relations.

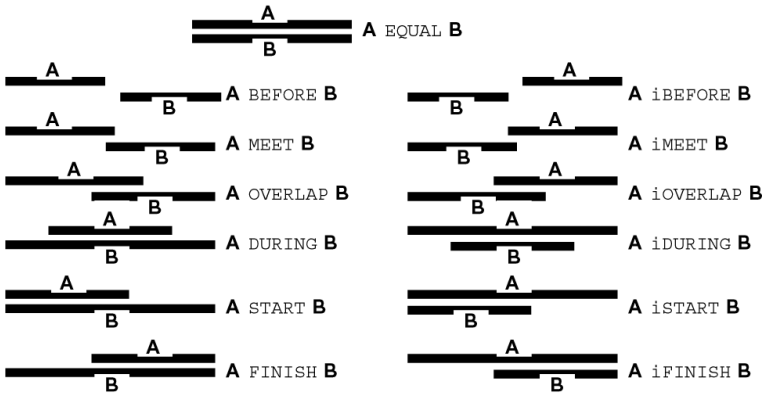


Figure 2.2: The thirteen relationships to express any relationship that can hold between two intervals.

[Allen, 1983] also addresses the problem of 'persistence': a state is true until discovered otherwise. This is modeled by applying constraints to intervals.

Chapter 3

Employed Techniques and Tools

3.1 Introduction

This chapter presents the underlying theoretical fundamentals. It gives an introduction to the principle of ontologies as a tool for knowledge representation and reasoning. The syntax of the ontology language used to express the rules of the semantic concept presented in this work is summarized separately. The employed software is shown as well as the utilized retrieval quality measures are introduced.

3.2 Ontologies

Ontologies were developed in Artificial Intelligence to enable sharing and reuse of knowledge. An ontology provides a shared and common understanding of a domain that can be communicated between people and heterogeneous and widely spread application systems. It is also an explicit conceptualization (i.e. meta information) that describes the semantics of the data.

3.2.1 Ontology design

Reusability is one of the most important features. Research focusses on building technologies enabling the large scaled reuse of ontologies. To achieve the requirements set by reusability an ontology must consist of small modules with a high internal coherence and a limited amount of dependencies between the modules. Gruber has expressed the design principles of the ontology in 1995. There is a need for objective criteria to guide and evaluate the designs. A preliminary set of design criteria for ontologies whose purpose is

knowledge sharing and interoperability among programs based on a shared conceptualization are [Gruber, 1995]:

1. **Clarity**: An ontology should effectively communicate the intended meaning of defined terms. Definitions should be objective. While the motivation for defining a concept might arise from social situations or computational requirements, the definition should be independent of social or computational context. Formalism is a means to this end. When possible a definition should be stated in logical axioms. Where possible, a complete definition (a predicate defined by necessary and sufficient conditions) is preferred over a partial definition (defined by only necessary or sufficient conditions). All definitions should be documented with natural language.

2. **Coherence**: An ontology should be coherent: that is, it should sanction inferences that are consistent with the definitions. At the least, the defining axioms should be logically consistent. Coherence should also apply to the concepts that are defined informally, such as those described in natural language documentation and examples. If a sentence that can be inferred from the axioms contradicts a definition or example given informally, then the ontology is incoherent.

3. **Extendibility**: An ontology should be designed to anticipate the uses of the shared vocabulary. It should offer a conceptual foundation for a range of anticipated tasks, and the representation should be crafted so that one can extend and specialize the ontology monotonically. In other words, one should be able to define new terms for special uses based on the existing vocabulary, in a way that does not require the revision of the existing definitions.

4. **Minimal encoding bias**: The conceptualization should be specified at the knowledge level without depending on a particular symbol-level encoding. An encoding bias results when representation choices are made purely for the convenience of notation or implementation. Encoding bias should be minimized, because knowledge-sharing agents may be implemented in different representation systems and styles of representation.

5. **Minimal ontological commitment**: ontology should require the minimal ontological commitment sufficient to support the intended knowledge sharing activities. Ontology should make as few claims as possible about the world being modeled, allowing the parties committed to the ontology freedom to specialize and instantiate the ontology

as needed. Since ontological commitment is based on consistent use of vocabulary, ontological commitment can be minimized by specifying the weakest theory (allowing most models) and defining only those terms that are essential to the communication of knowledge consistent with that theory.

3.2.2 Ontology Languages

Recently, there has been a growing number of notations to describe the structure and semantics of the exchanged data in the World Wide Web. XML (Extensible Markup Language) is the basis for several of these new standard candidates. It is a simple, flexible text format derived from SGML (ISO 8879). XML is a W3C[1] recommendation designed to meet the challenges of large-scale electronic publishing and is also playing an increasing role in the exchange of a wide variety of data on the Web and elsewhere. It has been adopted as a means of interchanging information between computer programs. In particular it is widely seen as the best solution for the interchange of metadata about stored objects and programs (e.g. the Open Software Description) and for the interchange of commercial information (e.g. Open Financial Exchange) [Hunter, 2001].

The Resource Description Framework (RDF) makes up a standard for describing the semantics of information via metadata descriptions [Lassila and Swick, 1999]. XML schemas give a standard for describing structure and semantics of the data. The transformation language XSL provides a standard for describing mappings between different terminologies [Clark, 1999]. RDF is a foundation for processing metadata. It provides interoperability between applications that exchange machine-readable information on the Web. RDF emphasizes facilities to enable automated processing of Web resources. It is also designed for representing data. RDF provides a basic object-attribute-value data model for meta-data. Other than these intended semantics – described only informally in the standard – RDF makes no data-modeling commitments. In particular, no reserved terms are defined. Just like XML, the RDF data model provides no mechanisms for declaring property names that are to be used [Lassila and Swick, 1999].

The RDF Schema provides a representation formalism and basic ontological modeling primitives. It defines classes, subclasses, subproperties, domain and range restrictions of properties, etc. in a web-based context. Furthermore, it provides information about the interpretation of the statements given in the RDF data model, but it does not constrain the syntactical appearance

[1] World Wide Web Consortium

of an RDF description. The RDF Schema lets developers define a particular vocabulary for RDF data and lets them specify the kinds of objects to which these attributes can be applied. In other words, the RDF Schema mechanism provides a basic type system for RDF models. This type system uses some predefined terms, such as *Class*, *subPropertyOf* and *subClassOf*. RDF Schema expressions are also valid for RDF expressions. RDF objects can be defined as instances of two or more classes using the type property [Davies et al., 2002].

Ontology Inference Layer

The Ontology Inference Layer OIL is a proposal for a web-based representation and inference layer for ontologies. It combines the widely used modeling primitives from frame-based languages with the formal semantics and reasoning services provided by description logic. It is compatible with the RDF Schema and includes a precise semantics for describing term meanings and thus also for describing implied information.

OIL presents a layered approach to a standard ontology language. Each additional layer adds functionality and complexity to the previous layer. This is done to enable that agents (humans or machines) who can only process a lower layer can still partially understand ontologies that are expressed in higher layers [Ont, 2000].

DARPA Agent Markup Language

The DARPA Agent Mark-Up Language (DAML) is designed as an XML-based semantic language that links the information on a page to machine-readable semantics. The goal of the DAML program is to create technologies that enable software agents to dynamically identify and understand information sources, and to provide interoperability between agents in a semantic manner. DAML research plan includes six tasks [DAM, 2000]:

1. To create an Agent Mark-Up Language (DAML) built upon XML that allows users to provide machine-readable semantic annotations for specific communities of interest.

2. To create tools that embed DAML markup on to web pages and other information sources in a manner that is transparent and beneficial to the users.

3. To use these tools to build up, instantiate, operate, and test different sets of agent-based programs that markup and use DAML.

4. To measure, via empirical experimentation, the productivity improvements provided by these tools.

5. To apply these tools to third party agent development, military-specific problems, and support for the intelligence community so as to evolve DAML technologies towards large-scale use.

6. To insert the DAML to the commercial and military markets via partnerships with industrial and defense-related (C2 and intelligence) organizations.

DAML+OIL

DAML+OIL is a semantic mark-up language built on earlier W3C standards such as RDF and RDF Schema. It extends these languages with richer modeling primitives. DAML+OIL provides modeling primitives commonly found in frame-based languages [Genesereth and Fikes, 1987].

The language has well defined semantics. A DAML+OIL knowledge base is a collection of RDF triplets. DAML+OIL prescribes a specific meaning for triples that use the DAML+OIL vocabulary. The model-theoretic semantics specifiy exactly which triples are assigned a specific meaning, and what this meaning is. DAML+OIL only provides a semantic interpretation for those parts of an RDF graph that instantiate the schema. Any additional RDF statements, resulting in additional RDF triplets, are perfectly allowed, but DAML+OIL is silent on the semantic consequences (or lack thereof) of such additional triples [Smith et al., 2002].

Web Ontology Language

The Web Ontology Language OWL is intended to provide a language that can be used to describe the classes and the relations between them that are inherent in Web documents and applications. The Web Ontology Language OWL is a semantic markup language for publishing and sharing ontologies on the World Wide Web. OWL is developed as a vocabulary extension of and is derived from the DAML+OIL Web Ontology Language. In OWL, an ontology is a set of definitions of classes and properties, and constraints on the way those classes and properties can be employed. An OWL ontology may include the following elements:

- taxonomic relations between classes,

- datatype properties, descriptions of attributes of elements of classes,

- object properties, descriptions of relations between elements of classes and, to a lesser degree,
- instances of classes, and
- instances of properties.

Datatype properties and object properties are collectively the properties of a class. OWL is a set of three, increasingly complex languages: OWL Lite, OWL DL and OWL Full.

OWL Lite has been defined with the intention of creating a simple language that will satisfy users primarily needing a classification hierarchy and simple constraint features. For example, while it supports cardinality constraints, it only permits cardinality values of 0 or 1. For these reasons, it should be simpler to provide tool support for OWL Lite than for its more complex relatives.

OWL DL includes the complete OWL vocabulary, interpreted under a number of simple constraints. Primary among these is type separation. Class identifiers cannot simultaneously be properties or individuals. Similarly, properties cannot be individuals. OWL DL is so named due to its correspondence with description logic.

OWL Full includes the complete OWL vocabulary. It interprets this vocabulary more broadly than OWL DL, with the freedom provided by RDF. In OWL Full, a class can be treated simultaneously as a collection of individuals (the class extension) and as an individual in its own right (the class intention). Another significant difference from OWL DL is that a DatatypeProperty can be marked as an InverseFunctionalProperty. These are differences that will be of interest to the advanced user [Smith et al., 2002].

3.2.3 VERL

In the summer and fall of 2003 the Advanced Research and Development Activity (ARDA) of the U.S. Government sponsored the "Challenge Project on Video Event Taxonomy". The result was a formal language for describing an ontology of events, called VERL (Video Event Representation Language). A companion language called VEML (Video Event Markup Language) to annotate instances of the events described in VERL, was also developed. This section shows the syntax and application of VERL and VEML, as presented in [Bolles and Nevatia, 2004].

Introduction

VERL is intended to be a language for representing events for the purpose of designing an ontology of the domain of an application and for annotating data with the categories in that ontology. The first version of the language, VERL 1.0, was defined as part of an ARDA 2003 challenge project and described in a January 2004 report. This document describes modifications of the language in response to feedback from users of the language in defining ontologies. It corrects some errors, provides additional constructs, and clarifies semantics. This language is VERL 2.0.

The Syntax of VERL This sections sketches how to use VERL in its two primary tasks-annotating data and defining composite events. The former is done first because it is easier to understand.

The language is very simple, but this simplicity is deceptive. Section 3.2.3 provides a careful development of the language, giving its semantics.

Annotating Data An annotation is a pair consisting of a thing in a VERL ontology and a designation of a location in the video data.

$$< thing,\ loc >$$

The *thing* describes a state or event, or an entity such as a physical object. Nothing is said here about how the locations are to be specified.

Types There are three basic types in the language. Everything is a *thing*. There are two types of things. The type *ent* encompasses entities and generally may be thought of as physical objects, although in some applications it can be used more broadly. The type ev encompasses states and events. A state is a property of something holding over a period of time. Normally, a person would be of type *ent*, and his running would be of type *ev*. The type hierarchy is

$$\begin{array}{c} thing \\ / \ \backslash \\ ent\ ev \end{array}$$

In specific applications it is possible to expand this hierarchy to more specific types. For example, one might introduce person and vehicle as subtypes of type *ent*.

VERL Expressions Constants may be of any one of the three types. Variables may range over any one of the three types. A VERL expression (*vexpr*) is defined as follows:

A constant or variable is a vexpr.

$$vexpr \rightarrow constant \mid variable$$

For example, "John," "X1," "Fire-1," and "E1" may all be vexprs. The type of the vexpr is the type of the constant or variable. Thus, "John" is an entity constant, "E1" will be an event variable if, for example, it refers to John's running, and so on.

A function symbol applied to the appropriate number of vexprs as arguments is a vexpr.

$$vexpr \rightarrow fct\ ''('' \ [\ vexpr\ \{'',''\ \ vexpr\ \}*\ ]\ '')''$$

Square brackets [...] indicate something is optional; here the function may have no arguments. Curly brackets {...} group elements together. The Kleene star * means zero or more instances of. The arguments must be of the right type.

A predicate symbol applied to the appropriate number of arguments is a vexpr.

$$vexpr \rightarrow pred\ ''('' \ [\ vexpr\ \{'',''\ \ vexpr\ \}*\ ]\ '')''$$

The arguments must be of the right type. The result is always of type *ev*. For example, if *change* is a predicate symbol relating two things of type *ev* and *E1* and *E2* are event variables, then $change(E1, E2)$ is a vexpr of type *ev*.

A logical operator applied to the appropriate number of vexprs of type ev is a vexpr.

$$\begin{aligned} vexpr \rightarrow & \text{''}AND\text{''}\ \text{''}(\text{''}\ vexpr\ \{\text{''},\text{''}\ vexpr\ \}*\text{''})\text{''}\ | \\ & \text{''}OR\text{''}\ \text{''}(\text{''}\ vexpr\ \{\text{''},\text{''}\ vexpr\ \}*\text{''})\text{''}\ | \\ & \text{''}IMPLY\text{''}\ \text{''}(\text{''}\ vexpr\ \text{''},\text{''}\ vexpr\ \text{''})\text{''}\ |\text{''}\ NOT\text{''}\ \text{''}(\text{''}\ vexpr\ \text{''})\text{''}\ | \\ & \text{''}EQUIV\text{''}\ \text{''}(\text{''}\ vexpr\ \text{''},\text{''}\ vexpr\ \text{''})\text{''} \end{aligned}$$

AND and *OR* take one or more arguments. *IMPLY* and *EQUIV* take two arguments. *NOT* takes one argument. The result is always of type *ev*. For example,

$$AND(change(E1, E2),\ change(E3, E4))$$

is a vexpr, and the result is the event consisting of the aggregate of $change(E1, E2)$ and $change(E3, E4)$). $NOT(change(E1, E2))$ is a vexpr and the result is the state of absence of the event $change(E1, E2)$.

A constant or variable can be used as a label on a vexpr.

$$vexpr \rightarrow \{constant \mid variable\}\text{''}:\text{''}\ vexpr$$

The resulting vexpr refers to the same thing as its constituent vexpr and of course is of the same type. The label can then be used elsewhere to refer to that thing.

Variables occurring in annotations are assumed to be existentially quantified with the outermost scope. Variables in annotations are thus equivalent to constants, and no formal distinction between the two need be made.

This completes the basic syntax of VERL, as would be required for annotating data.

Defining Composite Events The basic operator for defining composite events is *PROCESS*. It takes a predication and a vexpr as its two arguments. The predication is a predicate applied to the appropriate number of arguments, where the arguments have an optional type specification.

$defn \rightarrow'' \ PROCESS'' \ ''('' pred''('' [argspec\{'','' argspec\}*'')'' \ [\,'','' vexpr]'')''$
$argspec \rightarrow [type] \ variable$

The second argument of *PROCESS* is optional, and if it is missing, it is assumed the process is primitive, i.e., in this application it is directly implemented in software. For example, if there is the predicate *located-at* relating a thing to an entity, and a predicate *change* relating two things of type *ev*, then the predicate *move* is defined as follows:

$$PROCESS(move(thing\ x,\ ent\ y, ent\ z),$$
$$change(located-at(x,y),\ located-at(x,z)))$$

That is, for a thing x to move from entity y to entity z, there is a change in x's location from y to z.

PROCESS is equivalent to implication. The above statement could be read as "If there is a change from x being located at y to x being located at z, then there is a moving of x from y to z." Variables occurring in the antecedent (the second argument of *PROCESS*) are taken to be universally quantified over the scope of the whole definition. Variables occurring only in the consequent (the first argument of *PROCESS*) are existentially quantified.

Thus,

$$PROCESS(p(x,z),\ q(x,y))$$

will be interpreted as

$$(Ax,z)[p(x,z) \rightarrow\ (Ey)\ q(x,y)]$$

Three other operators can be used optionally in place of *PROCESS-PRIMITIVE*, *SINGLE-THREAD*, and *MULTITHREAD*. *PRIMITIVE* can be used when there is no second argument to the *PROCESS* definition. *SINGLE-THREAD* means that all constituent events in the definition happen sequentially, without overlap. *MULTITHREAD* can be used when there is no such constraint.

For example, assume located-at and change are primitive predicates. Then a move event is a single-thread event.

$$\begin{aligned} PRIMITIVE(located-at(thing\ x,\ ent\ y)) \\ PRIMITIVE(change(ev\ e1,\ ev\ e2)) \\ SINGLE-THREAD(move(thing\ x,\ ent\ y,\ ent\ z), \\ change(located-at(x,y),\ located-at(x,z)) \end{aligned}$$

Defining Subtypes The user may want to define subtypes of the types *ent* and *ev*. This can be done with the *SUBTYPE* operator. It takes two arguments, the name of the subtype and the name of the supertype.

$$SUBTYPE(type, type)$$

For example, to state that *ent* has subtypes *mobile* and *immobile* and *mobile* has subtypes *vehicle* and *human*, one would write

$$\begin{aligned} SUBTYPE(mobile, ent) \\ SUBTYPE(immobile, ent) \\ SUBTYPE(vehicle, mobile) \\ SUBTYPE(human, mobile) \end{aligned}$$

Sibling types in the type hierarchy should be mutually exclusive. Thus, if you were to specify *mobile* and *container* as subtypes of *ent*, you could not have mobile containers. It is often better to treat such concepts as properties rather than as types. In this case, the definitions would be

$$PRIMITIVE(container(ent\ x))$$
$$PRIMITIVE(mobile(ent\ x))$$

Inference Rules In addition to annotations of specific events and definitions of composite properties, relations, and events, one may also want to specify inference rules that allow to draw conclusions from what is recognized in the data. For this the operator *RULE* takes two vexprs of type *ev* as its arguments.

$$RULE(vexpr, vexpr)$$

For example, suppose defining $carry(x, y, a, b, t)$ (x carries y from a to b during time interval t), as x holds y during time interval t and x moves from a to b during time interval t.

$$PROCESS(carry(x, y, a, b, t), AND(hold(x, y, t), move(x, a, b, t)))$$

Then to say that when x carries y from a to b during t, y also moves from a to b during t:

$$RULE(carry(x, y, a, b, t), move(y, a, b, t))$$

Variables in the antecedent of the implication will be interpreted as universally quantified; variables that occur in the consequent but not in the antecedent will be interpreted as existentially quantified. Thus,

$$RULE(p(x,z), q(x,y))$$

will be interpreted as

$$(Ax,z)[p(x,z) \rightarrow (Ey)\ q(x,y)]$$

Comments Lines beginning with // are comments.

Control Structures Just as in programming languages, control structures are essential for building up complex composite events and processes; VERL provides *Sequence*, *Conditional*, *Repeat-Until*, and *While-Do* constructs.

Sequence takes an arbitrary number of events as its arguments. It says that these events occur in sequence, not overlapping. The order in the sequence is the same as the order of the arguments. The resulting vexpr describes the composite event consisting of all the argument events occurring in sequence.

$$Sequence(e1,\ e2,\ ...)$$

Conditional takes two or three events as its arguments. It says that if the first argument holds or obtains, then the second argument happens; otherwise, the third argument happens, if there is a third argument.

$$Conditional(e1,\ e2)$$
$$Conditional(e1,\ e2,\ e3)$$

The resulting vexpr describes a piece of behavior that has been recognized to operate in this fashion.

Repeat-Until takes two things of type *ev* as its arguments. The resulting vexpr describes the composite event in which the first argument is repeated until the second argument holds or obtains.

$$Repeat - Until(e1,\ e2)$$

Normally, *e1* is the sort of event that changes the world in a way that affects whether or not *e2* holds or obtains. Both *e1* and *e2* are types of events or states, rather than specific instances.

While-Do takes two things of type *ev* as its arguments. The resulting vexpr describes the composite event in which the second argument is repeated as long as the first argument holds or obtains.

$$While - Do(e1, e2)$$

While-Do can be defined in terms of other operators:

$$\begin{aligned} &PROCESS(While - Do(ev\ e1,\ ev\ e2),\\ &\qquad Conditional(e1,\\ &\qquad\qquad Repeat - Until(e2,\ NOT(e1)))) \end{aligned}$$

This completes the summary of the syntax of VERL. The following paragraphs present several classes of predicates that are useful in building up and relating complex structured events.

Equality Equality is a useful relation fordefining complex events which specify, for example, cases in which the same participant plays more than one role. Equality is represented with the predicate *Equal* that takes two entities as its arguments and returns a value of "true" if they are identical. More precisely, it returns an eventuality that exists when the two entities are identical.

$$Equal(thing, thing)$$

Equality statements can be given labels, in which case the label refers to the state of the two things being equal.

Temporal Relations Representing the temporal relations among component events is crucial in recognizing composite events. For perhaps most applications, describing the relations among the temporal intervals occupied by the component events, according to *Allen's interval algebra*, is sufficient. This is because agents are responding primarily to the actions of other agents or the behavior of moving objects. Even in a case where one of the threads is precisely timed, such as a conveyor belt in a factory, the worker is responding primarily to the appearance of the part rather than to the passage of a certain amount of time.

Times come in two varieties – *instants* and *intervals*. Thus,

$$SUBTYPE(temporalEntity, thing)$$
$$SUBTYPE(instant, temporalEntity)$$
$$SUBTYPE(interval, temporalEntity)$$

Of two distinct instants, one is *before* the other. The predicate *after* is the inverse of *before*.

$$before(t1, t2), after(t1, t2), Equal(t1, t2)$$

An instant t and an interval T can be in several possible relations:

$$begins(t, T), inside(t, T), ends(t, T)$$

It may be that none of these is true.
There are six possible basic relations that can obtain between two intervals:

$$before(T1, T2), meets(T1, T2), overlaps(T1, T2), begins(T1, T2),$$
$$contains(T1, T2), ends(T1, T2)$$

These form the basis of Allen's interval algebra. They can be defined in terms of begins, inside, and ends relations between instants and intervals. There are two possible relations between events and times: Some events happen instantaneously. In this case,

$$atTime(e, t)$$

where t is an instant. Some events happen across intervals, with a duration. In this case

$$during(e, T)$$

where T is an interval. The predication

$$timeSpan(T, e)$$

says that T is the entire temporal entity or sequence of temporal entities during which e occurs.

The OWL-Time ontology [Hobbs and Pan, 2004] provides a rich elaboration of these concepts and includes treatments of measures of duration, clock and calendar terms, and temporal aggregates. In some cases, this richer theory of time is required.

Many actions are rhythmic, and recognizing this is an important part of recognizing the higher-level event. Tapping one's foot to music is like this, as is marching in unison. A rhythmic event can be characterized as an iterative event in which the iterations occupy time intervals of equal duration. Rhythm is often used to coordinate multithread iterative action.

Three Other Useful Predicates The predicate *change* says that there is a change from one thing of type *ev* to another thing of type *ev*.

$$change(e1, e2)$$

For example, as seen above, a change of location can be represented in this way.

$$change(located - at(x, y, t1), located - at(x, z, t2))$$

The predicates cause and enable can also be used to link predicates together.

$$cause(e1, e2)$$
$$enable(e1, e2)$$

The second argument of *cause* and both arguments of *enable* are of type *ev*. The first argument of cause is of type *thing*. Events can cause other events. But there are also some entities that can function as a cause of an event. For example, it is most convenient to view a dog as the cause of its own movements.

Of course, *cause* and *enable* are not directly observable, but they are often involved in what is to be inferred from observables.
Examples:
To say that B is located at C:

$$located - at(B, C)$$

To say that B moves (changes location) from C to D:

$$change(located - at(B, C), located - at(B, D))$$

To say that A causes B to move (change locations) from C to D:

$$cause(A, change(located-at(B, C), located-at(B, D)))$$

Assume defining the predicates *rise* and *fall*. To say that A causes B to rise and C to fall, where the rising starts before and overlaps with the falling:

$$cause(A, AND(e1 : rise(B), e2 : fall(C), overlaps(e1, e2)))$$

This describes the situation where A causes not only B's rising and C's falling but also the overlapping of the two events. If the overlapping of the two events is accidental and not specifically caused by A,

$$AND(cause(A, AND(e1 : rise(B), e2 : fall(C))), overlaps(e1, e2))$$

The Semantics of VERL

Types, Terms, and Expressions Rather than dealing immediately with VERL expressions (vexprs), we start with a more conventional treatment of expressions and terms.

The type system is as before. Everything is a thing. There are two kinds of things – entities (ent) and events, states, and conditions (ev). As axioms:

$$(Ax)[thing(x) \leftrightarrow [ent(x)\ v\ ev(x)]]$$
$$(Ax)[ent(x) \leftrightarrow\sim ev(x)]$$

That is, x is a thing if and only if it is an ent or an ev, and ents and evs are mutually exclusive. Users can define subtypes of ent and ev, but

they should be mutually exclusive. Often it is safer to define something as a property rather than as a type. A term can be a constant, a variable, or a function symbol applied to the appropriate number of terms as arguments.

$$term \rightarrow\ constant \mid variable \mid fct''(''[term\{'',''\, term\}*]'')''$$

The constants can be of any one of the three types, and variables can range over any one of the three types. The function determines the type of the term that results from function application. Terms denote things in the world being described in VERL. An expression can be a predicate applied to the appropriate number of terms as arguments.

$$expr \rightarrow\ pred\ ''(''\ [term\{'',''\ \ term\ \}*]\ '')''$$

Such an expression is true if the predicate is true for the things denoted by the terms. A predicate applied to some set of arguments will be a "predication." In addition, an expression can be a logical operator applied to the appropriate number of expressions as operands.

$$\begin{aligned} expr \rightarrow & ''AND''\ ''(''[expr\{,expr\}*]'')''\ \mid \\ & ''OR''\ ''(''[expr\{,expr\}*]'')''\ \mid''\ NOT''\ ''(''expr'')''\ \mid \\ & ''IMPLY''\ ''(''expr'',''\, expr'')''\ \mid \\ & ''EQUIV''\ ''(''expr'',''\, expr'')'' \end{aligned}$$

AND can be applied to an arbitrary number of expressions and is true if all the expressions are true. *OR* can be applied to any number of expressions and is true if any one of the expressions is true. *NOT* is applied to one expression and is true if the expression is false. *IMPLY* is applied to two expressions and is true if the first expression is false or the second expression is true. *EQUIV* is applied to two expressions and is true if they are both true or both false.

When annotating video data and saying that an entity or event is at a particular location in the data – $at - video - loc(e, l)$ – thus asserting a

predication about that entity or event and a location in the data. A set of annotations for a video source is a conjunction of such predications. [Bolles and Nevatia, 2004]

3.3 The VCA Algorithm

It is not in focus of this thesis to develop an image processing algorithm of it's own. The VCA input is rather provided by the system presented by [Müller-Schneiders et al., 2005]. The detection and description of moving objects is based on an object-oriented, statistical multi-feature analysis of video sequences. This analysis is self-adapting to an observed scene ([Meyer et al., 1996]). The system is able to identify split and merge behaviors, where a single object splits into two or more objects and two or more objects merge into one object. Additionally, objects that are or become idle are detected: when the motion vector of the object becomes too small, the object is predicated to be *idle*. The system can also "discriminate between removed versus abandoned objects. It does this by analyzing the change in the amount of edge energy associated with the boundaries of the foreground region [...]. Barring extremely cluttered environments, if there are significantly more edges then an object has been added. Conversely, less associated edge energy suggests that an object has been removed" ([Connell et al., 2004]).

Thus, the system provides the following outputs: 1) object regions, 2) the objects' tracks, 3) an "Idle" flag for non-moving objects, and 4) distinction between abandoned/removed objects. Those are -through an MPEG-7 document- the inputs for the system presented in this thesis.

3.4 Retrieval quality measures

Precision and recall are the basic measures used in evaluating search strategies. These measures assume:

1. There is a set of records in the database which is relevant to the search topic
2. Records are assumed to be either relevant or irrelevant (these measures do not allow for degrees of relevancy).
3. The actual retrieval set may not perfectly match the set of relevant records.

3.4.1 Recall

RECALL is the ratio of the number of relevant records retrieved to the total number of relevant records in the database. It is usually expressed as a percentage.

Recall = (Number Retrieved and Relevant) / (Number Possible Relevant).

3.4.2 Precision

PRECISION is the ratio of the number of relevant records retrieved to the total number of irrelevant and relevant records retrieved. It is usually expressed as a percentage.

Precision = (Number Retrieved and Relevant) / (Number Total Retrieved).

3.5 The "Ground Truth"

There is no deterministic methodology for understanding what is relevant to a user's search. Thus, for the evaluation of a retrieval system, the relevant documents have to be selected by hand. In the case of an event detection system, the Ground Truth consists of

- the type of event,
- the time window the event occurred in, and
- the ID the VCA system assigned the object(s) performing the events to.

Chapter 4

A Semantic Concept for the Mapping of low-level Analysis Data to high-level Scene Descriptions

> **Morpheme**
> From Wikipedia, the free encyclopedia.
>
> In Linguistics, a morpheme is the smallest meaningful unit in a given language. This is the definition established in 1933 by the American linguist Leonard Bloomfield[1].

4.1 Introduction

This chapter presents a concept for event modeling and detection. The concept has been introduced by [Neuhaus, 2005]. The development and application of this approach as well as a proof of concept are now presented in this and the following chapters.

The remainder of this chapter is organized as follows. In section 4.2 terms used (and introduced) are listed and explained. In section 4.3 and it's subsections the constituting elements of the concept, Event Morphemes, along with the underlying concepts and ontology are presented. In section 4.6

[1]English Example: The word "unbelievable" has three morphemes "un-", (negatory) a bound morpheme, "-believe-" a free morpheme, and "-able". "un-" is also a prefix, "-able" is a suffix. Both are affixes.

an example of an Event Morpheme - based description of a scene is given. Section 4.7 shows the implementation of the Event Morphemes that is used to proof the concept in the subsequent chapters.

4.2 Terminology

At first there are some terms that need clear definition before usage. This is mainly due to the blending of content analysis matters with high-level concepts. Most of the content analysis' terms have their corresponding counterpart on the semantics' side but it is not a *1-to-1* match.

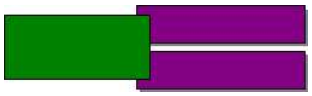
Low-level and high-level *Low-level* describes features that can be extracted automatically by means of signal processing. The extraction and segmentation of moving objects from image sequence' pixels would be low-level. *High-level* describes semantic meanings and concepts a user employs when (s)he describes a scene. In terms of automated content analysis and indexing, "high-level" describes the process of drawing conclusions from further analysis of the low-level features. Thus (post-hoc) analysis of low-level with semantic output features will here be called "high-level".

Features and attributes The terms *features* and *attributes*, as used here, are to describe the characteristics of the low-level data and the high-level semantic concepts, respectively. Feature is commonly used in the field of content analysis, anyways. For the semantic concept presented here, attribute describes the elements of the semantic representation. This can be the number of objects, direction of movement, or idle time of an object. Which these are and how they are used will be shown in section 4.4.2.

Moving region and object The *moving region* is the output of the level 0 (see section 1.2), the content analysis system. It is an arbitrary shaped cluster of pixels that -in theory- represents a moving (or recently moving; i.e. idle) object. Reality, however, shows that current content analysis is not always there (yet). Thus, an *object* in the "real-world" has to be distinguished from a moving region. This is not only for reasons of semantics, but also because *one* moving region may represent more than *one* moving object. A simple example for this case would be a person carrying a suitcase, as de-

picted in figure 4.1.

(a) real-world example



(b) schematic representation

Figure 4.1: A moving region representing two objects.

On the other hand, an object may be represented (or be part of) more than one moving region. When a person e.g. passes behind a truck and re-appears on the other side, some content analysis systems will assign a new id although it is the same object. The schematic representation of this case is depicted in figure 4.2.

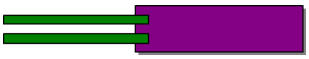
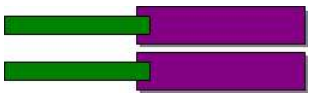


Figure 4.2: An object consisting of two moving regions (schematic representation).

Another conceivable scenario would be the handover of a suitcase: At first, the object 'suitcase' is represented by the same moving region as the first person carrying it (see figure 4.1). Next, the suitcase is standing on the floor, being represented by it's own moving region, as depicted in figure 4.3.

When the suitcase is carried away by a second person, the corresponding moving region will contain the suitcase and it's porter. This is analog to figure 4.1. Figure 4.4 shows the resulting schematic representation of the whole suitcase handover scenario. Temporal (or spatial) relations are not represented, only the associations of the objects to moving regions representing them is depicted.

(a) real-world example



(b) schematic representation

Figure 4.3: Two objects with one corresponding moving region each.

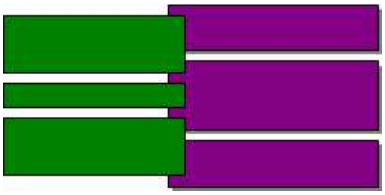


Figure 4.4: Suitcase handover representation.

Event Generally, *event* represents what is actually happening at a specific point in time. In this thesis, "event" represents only an elementary part of a sequence of actions and is comparable to "primitive events"[Ghanem et al., 2004] in literature.

Event Morpheme *Event Morphemes* appear to have no correspondence in literature. Basically, they represent an object's "point of view" of an event and thus link the objects to the corresponding events. Their relation is always a *1-to-1* relation, as depicted in figure 4.5.

(a) One object - one Event Morpheme - one event

(b) Two objects - two Event Morphemes - one event

Figure 4.5: The Event Morpheme's *1-to-1* relation.

The detailed concept and application of Event Morphemes follows in section 4.3.

Meta Knowledge The term *Meta Knowledge* as used in this thesis incorporates what is commonly known as "a priori knowledge" together with general *event morphology* and context-specific *user restrictions*. The a priori knowledge could be a map of the scene (see section 4.3.1) etc. while the general event morphology describes context independent knowledge about how events piece together (see section 4.3.1) and the user restrictions (see section 4.3.1) describe e.g. location dependent allowed durations of stay.

Semantic Module The *Semantic Module* is the output of the ontology-based inference mechanism. It is comparable to related works' "composite event"[Ghanem et al., 2004] or "process"[Nevatia et al., 2004]. In most cases, it exploits the Meta Knowledge to reason about the events by incorporating specific inference rules as described in section 4.3.2. In simple cases a Semantic Module solely uses succession/combinations of events. Actually, the Semantic Module is what the database may actually be queried for. A Semantic Module may constitute itself by one event, as depicted in figure 4.6 a), or it may incorporate two or more events (figure 4.6 b)), depending on the complexity of the process.

(a) A Semantic Module consisting of one event.

(b) A Semantic Module consisting of two events.

Figure 4.6: From event to Semantic Module

The "amount" of Meta Knowledge used in the inference process cannot be measured in number of pieces. Therefore, in figure 4.7 the Meta Knowledge is represented by a bar across the Semantic Modules. A Semantic Module may represent *level 1* or *level 2*.

Scene A *Scene* as used in this work consists of Semantic Modules representing events in their context specific instance. One may think of it as a "chain of events" (not to be confused with the Event Morpheme Path as introduced in section 4.3.6) or of what is commonly known as "scenario". Figure 4.7 depicts the connection between moving regions, objects, Event

Morphemes, events, Meta Knowledge and Semantic Module and the scene. A description of a scene is *level 2*.

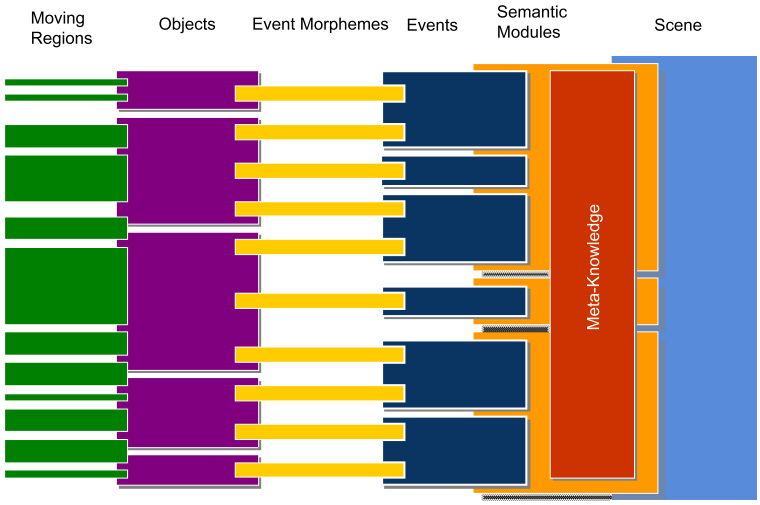


Figure 4.7: Schematic representation of the context between VCA's moving regions to higher-level scene description.

4.3 Event Morphemes

The idea behind Event Morphemes is the decomposition of a Scene into semantically meaningful parts (small, but not atomic), each with a "main object's" perspective. Such a part would not consist of more than the main object, its attributes, and the object(s) the main objects interacts with (if any). The attributes are the representations of the features extracted by the employed content analysis algorithm. An Event Morpheme represents a time window in which these attributes are analyzed and interpreted (see section 4.4.2). This inference process creates a semantic label describing the represented time window.

The semantic concept *"Event Morphemes"* thus represents a semantic intermediate layer. It maps (on an early stage) the low-level meta data to a semantic label .

An Event Morpheme may have a predecessor, a successor, and one or more companions. An example for a companion is "hold *while* walk" which

represents “carry” (see section 4.3.2). The roles of the successor and predecessor are explained in section 4.4.

4.3.1 Meta Knowledge

The Meta Knowledge represents auxiliary knowledge necessary for a more distinctive event recognition. It includes not only knowledge about what an event is composed of (event morphology) and the layout of the scene (map of the scene etc.), but also user restrictions for events which depend on the user's interpretation of e.g. “loitering”, i.e. after *what time* does “stay” turn into “loiter”. The role of event morphology, the map of the scene and user restrictions are pointed out in the following sections.

Event morphology

Event morphology describes the common knowledge about what an event is composed of in a given domain. Actually, these are the underlying rules and definitions for the inference mechanism described in section 4.3.2. Such a rule would e.g. be

> “walk” AND “hold” EQUALS carry.

Another example would be

> “stay” IN FRONT OF “emergency exit” EQUALS “obstructing”.

“Loitering” is an example for which more input is needed. The definition of this event could be

> “stay” FOR “specific time” IN “ specific area” EQUALS “loitering”.

How long “a specific time” is and where “a specific area” is, is not part of the event morphology. These informations come into play as user restriction (section 4.3.1), when the operator refines the query.

A map of the scene

A map of the scene is indispensable for location-dependent events. If e.g. a car is being parked in its designated parking space, it is an allowed event. A car being parked in the middle of a fire lane and thus obstructing it would be unwanted. Another example of incorporating the map of the scene into the

event reasoning would be knowing that where object "X" is moving, there is a river. From this, the system can conclude that the moving object is a swimming (or floating) object.

Similar to a map of the scene is knowledge about the camera's perspective and parameters, the average object size and actually everything commonly described as "a priori knowledge". In this thesis, it is the user-independent knowledge concerning the location and setup of the scene. The scene map and the semantic description of the locations therein are the requirement for the step from *level 1* to *level 2*.

User restrictions

User restrictions include all those parameters for the reasoning process that depend on the specific scene. "Loitering", as introduced in section 4.3.1, is an example. The time a person is allowed to stay at a certain place depends on the place and on the situation. The "specific time" may be shorter in front of an ATM than it would be on the lawn outside the facility's fence. The user restrictions will have to be defined at query time (in a retrieval application) or have to be parametrized when setting up a real time alert system.

4.3.2 Semantic Module

The *Semantic Module* is the reasoning mechanism, where higher level scene descriptions are generated. Picture a scene that – when described just using events – would read like this:

> A person is moving.
> The person is holding a suitcase.

In the video event representation language, VERL (introduced in [Nevatia et al., 2004]), complex events are composed from simple events. In this thesis complex events correspond to Semantic Module. When using the rule in equation 4.1

$$PROCESS(carry(x, y, a, b, t), \\ AND(hold(x, y, t), move(x, a, b, t))) \quad (4.1)$$

the above description of a scene yields as

> A person is carrying a suitcase,

which is a much more compact and intuitive description. So, the Semantic Module condenses events to statements the user would actually look for. The basis of the Semantic Modules are inference rules as in equation 4.1.

The richer the set of Event Morpheme Classes (see section 4.3.3) and the more extensive the inference rules are, the more and the finer scenes can be indexed. These inference rules' inputs, of course, have to match the semantic label of the Event Morpheme. The output should fit common ontologies and thus be applicable for other conceivable semantic driven applications. This way it would exploit the concept of incorporation of different ontologies.

4.3.3 Event Morpheme Taxonomy

The semantic labels of a meta data pattern represented by an Event Morpheme are organized in hierarchical classes. "State" and "transition" are the two types of classes. States are represented by Event Morphemes with semantic labels such as "move" or "hold", while transitions would be "put-down" or "pick-up". "Transition" is sub-dividable into "start" and "stop" for those transitions that begin or end states, respectively. These properties concern the principle of Semantic Interpolation (see section 4.3.4). An exemplary implementation of such an approach as invented in [Neuhaus, 2005], in VERL notation, would be

$$SUBTYPE(state, ev)$$
$$SUBTYPE(transition, ev)$$

$$SUBTYPE(start, transition)$$
$$SUBTYPE(stop, transition)$$

$$SUBTYPE(pick-up, start)$$
$$SUBTYPE(put-down, stop)$$

$$SUBTYPE(move, state)$$
$$SUBTYPE(hold, state)$$

To benefit in full of the concept of the Event Morpheme Taxonomy, the taxonomy has to be adapted to the VCA algorithm employed for the event detection. The finer the Event Morpheme's predicate shall be, the less confident

the classification will be. The advantage of arranging the Event Morphemes' semantic labels hierarchically is enabling a "maybe" statement. This way, it is possible to fall back on a more general label when the underlying inference mechanism produces a lower confidence for the more specific one.

Another motivation for a taxonomial arrangement of Event Morphemes are first experimental results. The "pick-up"-detector uses the "removed" classification of the VCA algorithm (see section 3.3) as input. This is combined with split and merge information: the moving regions the 'removed' object "split-off" from in tracking within a specified time interval from the detection of the removal are those removing. If all these rules are met, the Event Morpheme "pick-up^{-1}" is generated for the removed object.

When looking at scenarios resulting in the detection of "pick-up," is to be seen that not only actual events of e.g. a person picking up a suitcase (fig, 4.8 a) appear, but also a car pulling out of a parking space or a person getting up after having sat for a while (figures 4.8 b and 4.8 c, respectively).

(a) *pick-up* (b) $get - up$ (c) $pull - out$

Figure 4.8: Three phenotypes of *take-out*.

Thus, similar patterns in the VCA output have different semantics. It's just something being removed from the scene, either by itself (car, person) or another (the suitcase, by the person). This results in this initial approach for the Event Morpheme Taxonomy:

$$SUBTYPE(move, state)$$
$$SUBTYPE(hold, state)$$
$$SUBTYPE(bring - in, transition)$$
$$SUBTYPE(take - out, transition)$$
$$SUBTYPE(put - down, bring - in)$$
$$SUBTYPE(pick - up, take - out)$$

"Pick-up" has been replaced by it's superclass "take-out", and "put-down" by "bring-in". Note that the classes "start" and "stop" as sub-classes of "transition" are deleted as "pick-up" *starts* "hold", but "pull-out" *stops* "parking". The transition's property of the implication of the beginning or end of a state therefore cannot be modeled by means of taxonomy; it has to be done via the definition of rules as introduced in section 4.3.4 and this way the development of an ontology.

4.3.4 Semantic Interpolation

The principle of Semantic Interpolation describes the process of asserting that objects are present in the scene that are not explicitly detected by the content analysis at all times. An example would be a brought-in suitcase: the content analysis detects the put-down of the suitcase, yet it doesn't detect the suitcase while the person who brings it in is carrying it. Thus, it has to be inserted by the inference module. The rules for this are as follows.

$$
\begin{aligned}
&SUBTYPE(appear(ent\ e, atTime(t)), transition)\\
&SUBTYPE(hold^{-1}(ent\ eperson\ p), state)\\
&PRIMITIVE(starts(transition\ e, state\ s))\\
&PRIMITIVE(stops(transition\ e, state\ s))\\
&RULE(begins(pick-up^{-1}, hold^{-1}))\\
&RULE(ends(put-down^{-1}, hold^{-1}))\\
&IMPLY(AND(hold^{-1}(ent\ e1, person\ p1), appear(p1, atTime(t1))),\\
&\qquad\qquad appear(e1, atTime(t1)))
\end{aligned}
$$

The suitcase appears together with the person carrying it. Practically, this means that the moving region of the person before the put-down is being "copied" and preceded to the suitcase's history. Another example for the application of Semantic Interpolation is the detection of a $pull-out$: the corresponding predecessor is $parking$. However, when the $slot-in$ has been detected also, this would be a case of persistence: "a state is true until discovered otherwise" [Allen, 1983].

4.3.5 Reasoning-out false positives

The principle of the 'early' step from the low-level representation to a semantic label entails another advantage. It becomes possible to *know* which results are false positives for the determinated class of Event Morpheme. If e.g. a presumed brought-in (bring-in^{-1}) object existed before the inferred put-down, it is quite obvious, that this assertion is a false positive. It can be either caused by segmentation failure or a similar pattern in the VCA output as the respective class of Event Morpheme. Another basis for reasoning-out false positives is the object class: some actions are restrained to certain classes.

4.3.6 Event Morpheme Path

All conceivable events and scenes can be modeled by Event Morphemes. To achieve this, not only a rich vocabulary of Event Morphemes and a sound reasoning concept (the Semantic Module, see section 4.3.2) are of need. A representation of the succession of the Event Morphemes is necessary, too. This representation is the *Event Morpheme Path*.

The Event Morpheme Path is not to be confused with "chain of events" as this would be described more appropriately as "Semantic Module Path". It is the succession of Event Morphemes. As well as Event Morphemes themselves, there can be parallel Event Morpheme Paths, representing multi-threaded Semantic Modules. When using an indexing system based on Event Morphemes and combining Event Morphemes to Event Morpheme Paths, it becomes possible to query such an index and search for events ($\hat{=}$ Semantic Modules) that haven't been thought of at indexing time.

4.4 Application of Event Morhpemes

This section shows how Event Morphemes can be applied to model a scene and thus create a description of it. Special notion is given to implementational aspects, such as the separate central storage of a list of all objects or how low-level features are mapped to a semantic label.

4.4.1 Separate list of objects

An Event Morpheme holds as one of its attributes the main object and the object(s) the main object interacts with (if any). To avoid redundancy, all object related data is stored in one separate *Object List*. This Object List may (as in section 4.7) match with the VCA results. Links to the objects are

then stored in the Event Morphemes. The object data consists of the low-level features such as the corresponding moving region, color and textural features. When these features are chosen in a way that object recognition is possible upon them (at least to some extend), the list will enable multi-camera searches. This, of course, demands additional matching of the object features when analyzing track data and by this the merging of corresponding object tracks. The objects classes are available in the Object List, too. They are, on the one hand, another important piece for a *level 2* description but also necessary for the process of reasoning out false positives, as described in section 4.3.5.

4.4.2 Event Morpheme Template

The Event Morpheme Template is, practically speaking, a matrix where the entries are the output features of *level 0*. With these inputs as attributes, the Event Morpheme is created. During that process, an inference algorithm produces the semantic label. Table 4.1 shows the principle of the Event Morpheme Template.

Attributes								
object split	object merge	distance covered	change in orientation	deformation	duration of time frame	sensitive area		
		≈ 0					stay	*Semantic labels*
		> 0					move	
			•				tip-over	
	•						meet	
•				•			pick-up	
•				•			put-down	

Table 4.1: The attributes of the instantiated Event Morpheme and the corresponding semantic labels

Event Morphemes are instantiated when visual changes in a combination corresponding to table 4.1 take place. The duration of the Event Morpheme

is determined by the next change "undoing" the instantiating one. So, at first a temporal segmentation is carried out. For the resulting time window, the attributes of the Event Morpheme as shown in table 4.1 are analyzed. Based on these, a first classification and -if necessary- a refinement of the temporal segmentation is performed. "Classification" stands for attempting to determine what class of action (see section 4.3.3) is taking place and which objects are involved. This way, an Event Morpheme for each object participating in an event is created.

In the second step, the succession of the Event Morphemes (the *Event Morpheme Path*) is analyzed. This implies looking at the predecessor(s) and successor(s) of each Event Morpheme to refine it's classification or even instantiate additional Event Morphemes. If, e.g. an Event Morpheme "pick-up" is detected in the succession, a new Event Morpheme "hold" will be created (Semantic Interpolation).

4.4.3 Event Morphemes' sphere of responsibility for reasoning

Several points have to be borne in mind when considering at which stage which reasoning should take place. One has to take into consideration what features and attributes affect the decision, e.g., about the specialization of the class of the Event Morphemes. Rules to reason out false positives in the VCA algorithm results can be handled in the Event Morpheme detection stage. Those assertions derived from "reliable" VCA output such as object class are inferred in the Semantic Module. The advantage of the latter would be that growing domain knowledge can be incorporated by simply adapting the underlying ontology and not having to touch the program code of the Event Morpheme detectors (and re-compile them). This, again, underlines the generic nature of the Event Morphemes as an indexing structure.

4.5 Where is the index?

The entire concept presented so far is a concept for indexing and retrieval. So, as a last step, the separation between the indexing and retrieval part has to be done. In other words: where is the index? As the Retrieval will need all the information held in the Event Morphemes, the index should map exactly this information. Thus, the dividing line goes "through" the Event Morphemes (see figure 4.9). When building an index like this, the generic nature of the Event Morpheme approach can be exploited in total by the Smart Retrieval without losing access to low-level features.

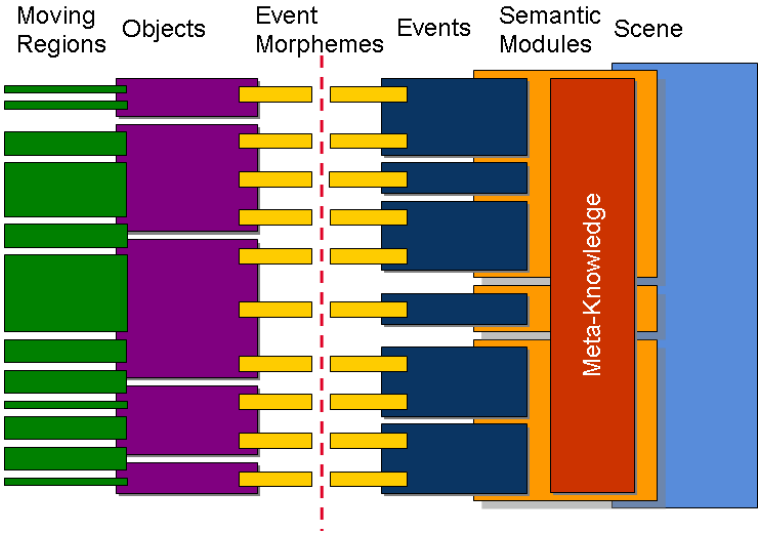


Figure 4.9: New Indexing Approach: The Event Morphemes *are* the index.

4.6 An example

Figure 4.10 shows the key frames of a scene that could be described as

> A suitcase is passed from one person to another.

A person enters with a suitcase. He puts down the suitcase and walks over to the chair. After the person sat a for while, a second person enters. He picks up the suitcase and leaves.

The three corresponding Event Morpheme Paths would read as follows.

> For the **first person**: "walk" WHILE "hold" - "put-down" - "walk" - "sit-down" - "sit"
>
> For the **second person**: "walk" - "pick-up" - "walk" WHILE "hold"
>
> Finally, the **suitcase**: "being held" - "being put-down" - "stand" - "being picked-up" - "being held"

Another interpretation of the scene could certainly be

> A left-behind suitcase is stolen

Figure 4.10: The suitcase handover scene.

In cases like this, the advantage of breaking the events into small parts emerges. Either of the two interpretations of the scene can be modeled, and it is simple to query a database. Whether this sequence is showing a handover or a theft may just turn out when the first person reports a burglary.

Other and more complex scenes can be modeled as well. It could even be conceivable to model scenes spreading over multiple cameras. This, of course, demands a sound list of object and reliable object features for recognition. If this was the case, a scenario like

Person shopping at a supermarket

would be modeled as (only person's point of view):

"exiting car"
"walk" (to shopping baskets)
"pick-up" (shopping basket)
"walk" WHILE "hold"
"grab" (into shelf)
"hold" (goods)
"put into" (goods into basket)

"walk" WHILE "hold" (into cashier area)

"walk" WHILE "hold" (into parking lot)

"put into" (goods in car)

"put down" (shopping basket)

"enter car"

4.7 Implementing Event Morphemes

4.7.1 An overall system

In order to create a system capable of detecting events and combining those recognized events to compile a semantic scene description, we need

1. a low-level analysis (VCA[2]) algorithm,
2. an ontology to lay down a common description language,
3. an inference mechanism to map the VCA- meta data to a semantic label,
4. a tool/mechanism for creating the map of the scene
5. a retrieval interface

These items constitute the system as follows. The VCA algorithm (item 1) analyzes the image sequence and outputs the objects' shapes, tracks, and features like color, texture, etc. Thus, it provides the *level 0* meta data. The utilized VCA algorithm is described in section 3.3. The inference mechanism (item 3) is the crux of the system: anything that isn't processed soundly here, will effect the consistence of the following stages (as well as inadequacy of the VCA). This part of the system will have to be adapted to the preceding VCA algorithm(s) employed. It outputs *level 1* meta data. The ontology for the common description language (2) is required on the one hand for the reasoning logics and on the other hand for the output to the user and the mapping of the user's query, respectively. Furthermore, the reasoning logic exploiting the underlying ontology enable (together with item 4) the step to *level 2*. Both the ontology and the reasoning should be independent from the VCA algorithms used as this adaption is already performed when stepping from *level 0* to *level 1*. The retrieval interface (5) finally should be flexible enough to enable the user a variety of queries and map the query to the index.

[2]Video Content Analysis

4.7.2 The Event Morpheme Ontology

Event Morphemes hold assessable patterns of the pixel-level or transformed attributes of the semantic (event) concepts they represent. These attributes play an important role in the reasoning process as they enable the specialization of semantic concepts. Therefore, the underlying ontology has to provide concepts and mechanisms to model, define, and interpret those values correctly. The following ontology in VERL notation shows a basic vocabulary of Event Morphemes.

$SUBTYPE(person, ent)$
$SUBTYPE(object, ent)$
$SUBTYPE(state, ev)$
$SUBTYPE(move, state)$
$SUBTYPE(hold, state)$
$SUBTYPE(event, ev)$
$SUBTYPE(bring-in, event)$
$SUBTYPE(take-out, event)$
$SUBTYPE(put-down, bring-in)$
$SUBTYPE(pick-up, take-out)$
$PRIMITIVE(starts(event\ e, state\ s))$
$PRIMITIVE(stops(event\ e, state\ s))$

4.7.3 The Event Morpheme detector modules

As stated in section 4.7.1, the Event Morpheme detection modules have to be adapted to the employed VCA algorithm. So, to present a proof of concept, the mapping of the VCA algorithm demanded in section 3.3 is described here. Note that the following algorithms only fit this particular software. Detector modules for detection of

1. take-out
2. bring-in
3. fall
4. non-straight (movement)

were implemented. Figure 4.11 shows schematically how the mapping is done. A detailed desciption follows in the subsequent sections.

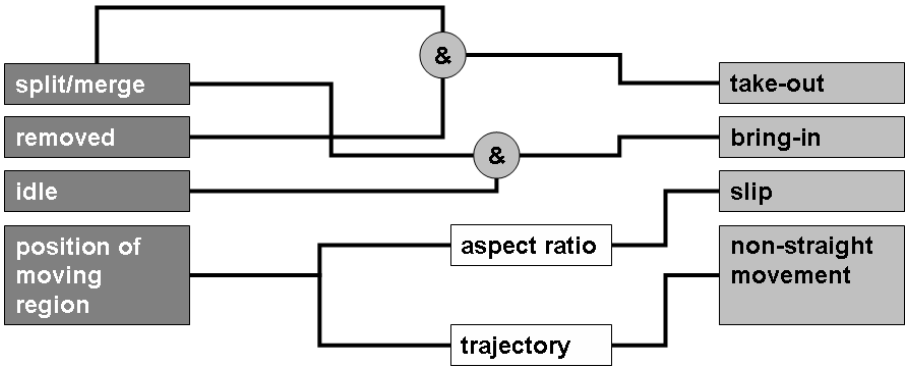


Figure 4.11: Mapping VCA output to Event Morphemes.

The modules have been implemented using MATLAB^{TM}. The Event Morphemes were detected in a post-hoc analysis step, *after* the VCA processed the sequences.

take-out

For the detection of the Event Morpheme *take-out* two informations are of need: the removed-classification and the split lists of all objects. When an object is classified by the VCA algorithm as "removed" *and* appears in the split list of another object a specific time before the detection of the "removal", it is predicated to be taken out. Thus, the Event Morphemes *take-out* and *take-out*$^{-1}$ are generated for the object the taken-out object appears in its split list and the taken-out object, respectively. Algorithm 4.1 shows in principle the algorithm.

bring-in

The detection of the Event Morpheme *bring-in* is quite similar to the detection of *take-out*. Instead of the "removed"- classification, the "idle"- classification is used. When an object is classified by the VCA algorithm as "idle" *and* appears in the split list of another object a specific time before, it is predicated to be brought. Thus, the Event Morphemes *bring-in* and *bring-in*$^{-1}$ are generated for the object the brought-in object appears in its split list and the brought-in object itself, respectively. Algorithm 4.2 shows

```
removedObjects ← getObjectsWithEvent('Removed');
for i ← 1 to number_of_removed_objects do
    occurrenceOfRemoval(i) ← getEventTime( removedObjects(i),
    'Removed');
    objectsContainingRemovedObjectInSplitList(i) ←
    findObjectInSplitLists( removedObjects(i) );
end
for i ← 1 to numberOfRemovedObjects do
    for j ← 1 to
    numberOfObjectsContainingRemovedObjectInSplitList do
        occurrenceOfSplit ← getSplitTime(j);
        if ( occurrenceOfRemoval(i) >= occurrenceOfSplit ) && (
        occurrenceOfRemoval(i) <= occurrenceOfSplit + ϵ ) then
            Object i has been taken out by object j
        end
    end
end
```

Algorithm 4.1: Detection module for *take-out*

in principle the algorithm.

```
idleObjects ← getObjectsWithEvent('Idle');
for i ← 1 to number_of_removed_objects do
    occurrenceOfIdle(i) ← getEventTime( idleObjects(i), 'Idle');
    objectsContainingIdleObjectInSplitList(i) ← findObjectInSplitLists(
    idleObjects(i) );
end
for i ← 1 to numberOfIdleObjects do
    for j ← 1 to numberOfObjectsContainingIdleObjectInSplitList
    do
        occurrenceOfSplit ← getSplitTime(j);
        if occurrenceOfIdle(i) >= occurrenceOfSplit) then
            Object i has been brought in by object j
        end
    end
end
```

Algorithm 4.2: Detection module for *bring-in*

As the "idle"-classification also applies to objects that recently moved and then stay still for a specific time, the results of algorithm 4.2 have to be

post-processed semantically: false positives are to be reasoned-out. This is done by eliminating those objects presumed to be brought-in that existed in the track lists of the VCA output or the object trajectory gives evidence of movement. Thus, algorithm 4.2 has to be extended as shown in algorithm 4.3.

```
idleObjects ← getObjectsWithEvent('Idle');
for i ← 1 to number_of_removed_objects do
    occurrenceOfIdle(i) ← getEventTime( idleObjects(i), 'Idle');
    objectsContainingIdleObjectInSplitList(i) ← findObjectInSplitLists(
    idleObjects(i) );
end
for i ← 1 to numberOfIdleObjects do
    for j ← 1 to numberOfObjectsContainingIdleObjectInSplitList
    do
        occurrenceOfSplit ← getSplitTime(j);
        if occurrenceOfIdle(i) >= occurrenceOfSplit ) then
            if ( getFirstOccurrence( idleObjects(i) ) ==
            occurrenceOfIdle(i) ) && ( getTrackLength( idleObjects( i )
            < 1 ) then
                Object i has been brought in by object j
            end
        end
    end
end
```

Algorithm 4.3: Extended detection module for *bring-in*

The resulting improvement is shown in section 5.3. The same restraints are put on the results of the detection of *take-out*. Thus, algorithm 4.1 is adapted accordingly.

fall

The detection of fall analyzes the change of the aspect ratio of the object's bounding box. If this change is larger than a specified threshold and the object is a person, the object (person) is predicated to have fallen (see algorithm 4.4).

```
setTimeWindow(Δ_t);
foreach timeWindow do
    bboxAspectRatio ← calculateAspectRatio( BoundingBox )
end
if ( bboxAspectRatio >= threshold ) && (objectClass == 'Person')
then
    Person fell
end
```

Algorithm 4.4: Detection module for *fall*

non-straight

The Event Morpheme *non-straight* is detected by analyzing the object's trajectory. First, the trajectory is normed to a specific number of vertices within the analysis (time) window. Then, a pseudo-differentiation of the trajectory is calculated. Exploiting the fact that the second differentiation of a straight line vanishes, a threshold is applied to the outcome. If it is reached, the object is predicated to move in a non-straight way.

```
1  setTimeWindow(Δ_t);
2  foreach timeWindow Δ_{t_i} do
3      foreach Trajectory of Δ_{t_i} do
4          normedTrajectory ← normTrajectory( TrajectoryOf_Δ_{t_i},
           numberOfVertices );
5          pseudoStraightness ← diff( normedTrajectory, 2 ) /
           numberOfVertices;
6          if pseudoStraightness > threshold then
7              Object is not moving straight
8          end
9      end
10 end
```

Algorithm 4.5: Detection module for $non-straight$

$normTrajectory()$ in line 4 is taken from [ISO/MPEG, 2002] (see eq.

4.2), without correspondence determination.

1. Set $V_i = c(i)$ for $0 \leq i \leq N_C - 1$ and $N_P = N_C$
2. If $N_P = N$, terminate.
3. Find $i(= i_{min})$ which minimizes T_i.
4. Set $V_{(i_{min}-1) mod N_P} = V_{(i_{min}-1) mod N_P} + \Delta(i)$ and $V_{(i_{min}+1) mod N_P} + \Delta(i)$.
5. Set $V_{i_{min}} = V_{i_{min}+1}, V_{i_{min}+1} = V_{i_{min}+2}, \ldots, V_{N_C-2} = V_{N_C-1}$ and $N_P = N_P - 1$. Go to Step 2.

(4.2)

$diff(X, n)$ in line 5 of algorithm 4.5 is a MATLAB™ function that calculates differences between adjacent elements of X, applying recursively n times, resulting in the n^{th} difference.

Chapter 5

Results

5.1 Introduction

This chapter presents the results of an exemplary implementation of an Event Morpheme-based event detection system. It is shown how the detectors introduced in section 4.7.3 perform in different scenarios and their application in the recognition of complex scenarios. It is also demonstrated how detection results are improved by reasoning-out false positives automatically. The results that will be presented here as a proof of concept have been obtained in several steps. At first, a training set consisting of 304 sequences with various scenarios was processed. Next, the Event Morphemes detection has been developed on the output of the VCA algorithm processing these sequences. Finally, the detection has been performed on the test set containing the use case sequences. The remainder of this chapter is organized as follows. In section 5.2 the realization of the use cases identified in section 1.2.2 is presented. Section 5.3 shows how the technique of reasoning-out improves the precision of the results. The results of event detection in the sequences of the test set are presented in section 5.4.1. Finally, section 5.5 compares the Event Morpheme approach with competitive approaches.

5.2 Realization of Use Cases

As mentioned in section 1.2.2, together with security systems experts, a set of relevant use cases has been identified. At first, the following sub-events are to be detected: pick-up, put-down, and falling down. The selected sub-events should be capable of composing the defined use cases. So, the following use cases to be retrieved from the index, were defined.

1. “Fight”: a person falls caused by a person passing by/person falls down and doesn’t stand up again.
2. “Shopping”: did the costumer pay or not?
3. “Criminal behavior at an ATM”: spy on another person’s PIN.
4. detect beggar/salesman on street.

These scenarios were staged in different locations: 1) a newsstand, 2) an outdoor pathway, and 3) indoors. The following sections explain how the use cases where put into action.

5.2.1 Fight

The fight scenario was recorded indoors. Two persons approached each other or passed by one another. One of the person fell. See picture 5.1 for a shot of the scene.

Figure 5.1: A person falls caused by another.

5.2.2 Shopping

The main issue in the shopping scenario was “Did the costumer pay or not?”. To test this scenario, a set of sequences were shot at a convenient store. A person standing at a shelf with newspapers and magazines picks up a magazine, and then either puts it back, pays for it and leaves, or leaves without paying for it. Figure 5.2 shows a frame of the scene.

Figure 5.2: The shopping scenario.

5.2.3 Criminal Behavior at an ATM

The ATM scenario was recorded indoors. The issue of "spying on another person's PIN" was staged by a person pretending to operate an ATM (actually a whiteboard). Another person approaches closely and (pretends to) spies on what person one is doing. See figure 5.3 for the setup of the scene.

Figure 5.3: A person spying on another's PIN.

5.2.4 Detect Beggar/Salesman on Street

The scenario "beggar/salesman" was recorded outdoors. The setup of the scene was a line of people walking down a footpath. Another person comes toward those people and approaches each one of them. This results in a non-straight trajectory where the others' trajectories are straight. Figure 5.4 shows the scenery.

Figure 5.4: A person walking from one person to another.

5.3 Reasoning out false positives

This section shows the results for the application of the reason-out strategy as introduced in section 4.7.3. Whether the object detected as taken-out/brought-in existed before the detection of the "removal"/"idle" or if the object's trajectory gave proof of movement, the result of the detector output was classified as 'false positive'.

The following improvements were achieved on the training set. The false positive rate decreased and, thus, the precision increased. This was for the detection of *take-out* a decrease of false positives by 20,2% resulting in an increase of 17,7% in the precision. For the detection of *bring-in*, the results were even more significant. The false positive rate dropped by 35,8% and this way improved the precision by 29,0%. See table B.1 in appendix B for the results on the training sequences. The reasoning-out strategy solely effects the precision as the recall is affected by the false negatives. Unfortunately, the latter are not to be reduced by post-hoc analysis.

5.4 Event Morpheme detection

For the detection of the scenarios described above, a demonstrator software called "postHocMPEG7Analysis" was developed. See figure 5.5 for a screenshot.

The software was developed using MATLAB^{TM}. The desired class of the Event Morpheme can be selected via the user interface. The implemented classes are

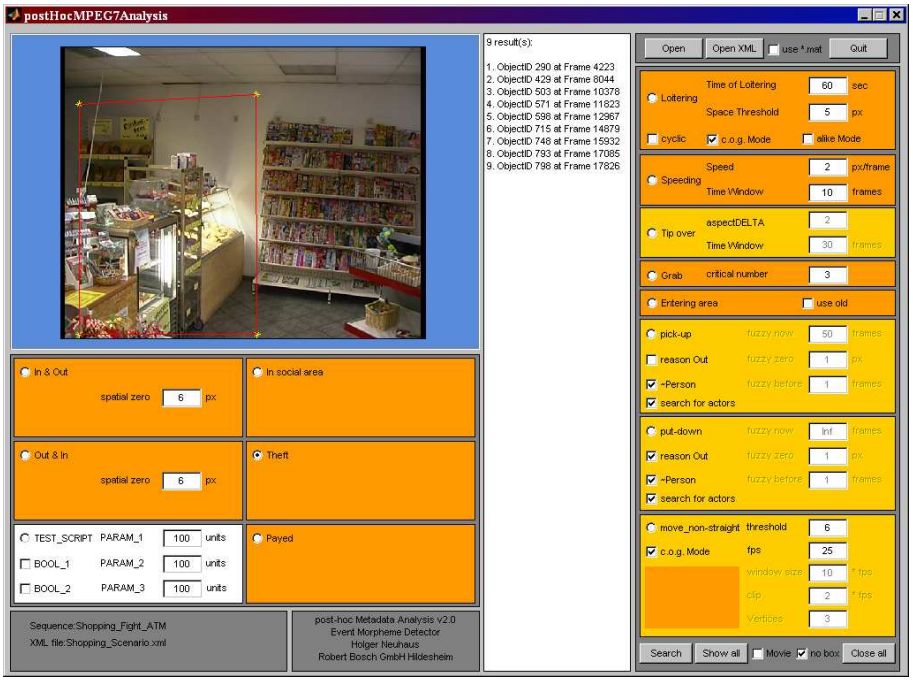


Figure 5.5: The postHoc analysis tool.

fall ("Tip over")

take-out ("pick-up")

bring-in ("put-down")

non-straight ("move_non-straight")

Additionally, the following Semantic Modules are available.

1. In social area[1]
2. Theft
3. Pay
4. Loitering
5. Speeding

[1]term adopted from [Fuentes and Velastin, 2005]

6. Grab
7. Entering area
8. In & Out
9. Out & In

Items 1 to 3 are those relevant to the test scenarios. "In social area" applies to the ATM scene: the social area is drawn in the video display. The second person entering this area triggers the event. "Theft" (item 2) is the succession of *pick-up* and *disappear*[2] without a *put-down* or an "entering the cashier zone" preceding; the cashier zone is drawn in the video display. "Pay" (item 3) is the succession of *pick-up*, "entering the cashier zone" and *disappear*, without the detection of *put-down*. Items 4 to 9 are not relevant for the detection of the use cases.

5.4.1 Results for the test sequences

This section presents the results for the detection of the identified use cases. These results were validated against a hand-annotated ground truth. The rate of true and false positives and of false negatives and the resulting values for precision and recall are shown in table 5.1.

Scenario		Ground Truth	true positives	false positives	false negatives	Recall	Precision
Shopping	pick-up	29	27	1	2	93,1	96,4
	put-down	9	7	0	2	77,8	100
	pay	8	7	0	1	87,5	100
	theft	12	11	1	1	91,7	91,7
Beggar[3]		2	2	0	0	100	100
Fight		11	10	0	1	90,9	100
ATM		10	9	0	1	90	100

Table 5.1: Results of the detection of events in the test sequences

[2] *disappear* is the end of the object track.
[3] tracks labeled manually

The detection of the Event Morpheme *non-straight* for the use case "detect Beggar/Salesman on street" failed due to the early overlap of the persons staging the scenario. So, the detector was tested with manually labeled tracks. With this input, the beggar was detected correctly. The one false positive for "theft" was caused by a missed "put-down".

5.4.2 Error bounds

This section approximates the Confidence Radii for the total recall and precision values calculated from the results above. The Confidence Radius determines a given interval, in which - with a certain confidence - the results *actually* may lie. As total recall, the presented system achieves 90 %, and for total precision 97 %. First, the Confidence Radius for recall is calculated [Papula, 1997].

Using a confidence level of 95% ($\gamma = 0,95$), from the constraint

$$P(-c \leq\ U \leq\ c) = \gamma = 0,95, \tag{5.1}$$

U being the actual value for recall, c calculates as

$$\begin{aligned} P(-c \leq\ U \leq\ c) = \Phi(c) - \Phi(-c) = \Phi(c) - [1 - \Phi(c)] = \\ 2 \cdot \Phi(c) - 1 = 0,95 \\ \Phi(c) = 0,975 \longrightarrow c = u_{0,975} = 1,960. \end{aligned} \tag{5.2}$$

Thus, the constant is the quantile $u_{0,975} = 1,960$ of the normal distribution.

With $n = 81$ and $k = 73$ the unknown value of p estimates

$$\hat{p} = \frac{k}{n} = \frac{73}{81} = 0,901. \tag{5.3}$$

The constraint for an extensive sample ($\Delta > 9$) is not totally met, yet close enough:

$$\Delta = n \cdot \hat{p} \cdot (1 - \hat{p}) = 81 \cdot 0,901 \cdot (1 - 0,901) = 7,21. \tag{5.4}$$

Now, the Confidence Radius Θ calculates as

$$\Theta = \frac{c}{n} \cdot \sqrt{\Delta} = \frac{1,96}{81} \cdot \sqrt{7,21} = 0,06 \approx 6\%. \tag{5.5}$$

Proceeding accordingly for the precision values ($n = 75, k = 73$), the confidence radius yields to $\approx 3\%$.

5.5 Comparison

In this section, the results presented in section 5.4.1 are compared to results published in literature. Where not available, the competitors' results were transposed to precision/recall values using the results given. In the following, the results are shown as published.

[Brand et al., 1997] tested their system using 52 gestures of 3 types:

"single whip"
"cobra"
"brush knee"

The classification accuracy for the proposed system was 94,2%.

[Cupillard et al., 2004] used the scenarios

"fraud"
"vandalism"
"fighting"
"blocking"
"overcrowding"

They achieved the following detection rates. 6 of 6 for fraud, 4 of 4 for vandalism, 20 of 24 for fighting, 13 of 13 for blocking, and 2 of 2 for overcrowding. False alarms were 2 for fraud, 4 for fighting, and 1 for blocking.

[Guler et al., 2003] tested with "ten significant events such as package exchange over multiple views, people parking but not leaving their cars, people tailgating through access controlled check points" The correct detection rate for the "Real Training Set", real life video data, was 92,31%.

[Hakeem and Shah, 2004] classified staged meeting scenarios consisting of events as

"hand raised"
"hand lowered"
"pick up object"
"put down object"
"hand extended"
"hand retraced"
"hand waving"
"moderator present"
"hand pointing"
"head shakes/nods"

Those ten events were used to detect three classes of meetings:

"argument"
"object passing"
"voting"

[Rao and Shah, 2001] tested the proposed system on the events

"pick up"
"put down"

and on the detection of similar actions. The true positive rate for the detection of “pick up” was 9 with 6 false negatives and 1 false positive. “Put down” resulted in 7 true positives, 1 false negative and 1 false positive.

[Xiang et al., 2002] trained their system with the events

“can taken”

“enter & leave”

“shopkeeper”

“browsing”

“paying”

The results on the test set for “can taken” consisted of 10 true positives with no false positive or negative (100% of 10, no false detection). Enter & leave resulted in 11 true positives, 3 false positives and 7 false nagatives (61,1% of 18, 3 false detections). Shopkeeper achieved 4 true positives, 1 false positive and 8 false negatives (33,3% of 12, 1 false detection). The results for the detecction of browsing are 5 true positives, 9 false positives and 3 false negatives (62,5% of 8, 9 false detections). Paying resulted in 6 true positives, 6 false positives and no false negative (100% of 6, 6 false detections). The comparison is summarized in table 5.2.

System	Ground Truth	different scenarios	true positives	false positives	false negatives	Recall	Precision	Technology
[Brand et al., 1997]	52	3					94,2	Coupled Hidden Markov Models
[Cupillard et al., 2004]	45	5	42	7	3	93,3	85,7	three-stage formalism (operators-knowledge-description)
[Guler et al., 2003]		10					92,3	HMM
[Hakeem and Shah, 2004]	122	3	115	6	7	94,3	95,0	rule-based system and state machine
[Neuhaus, 2006][4]	81	7	73	2	8	90,1	97,3	(see chapter 4)
[Rao and Shah, 2001]	23	2	16	2	7	69,6	88,9	HMM
[Xiang et al., 2002]	54	5	36	19	18	66,7	65,5	EM for clustering, selection using Minimum Description Length (MDL)

Table 5.2: Comparison of the systems

[4] *This work*, to be precise!

5.6 Mapping other approaches' events to Event Morphemes

The concept of Event Morphemes is proposed to be a generic semantic framework for the mapping of low-level analysis data to high-level scene descriptions. This way, it must be possible to model *any* conceivable event. Table 5.3 exemplarily shows that this is the case for some scenarios used in literature.

[Hongeng et al., 2004]:	
"approaching a reference person"	*approach* is an Event Morpheme
"heading toward"	*heading toward* is synonymous for approach
[Bourbakis et al., 2003]:	
"movingTruck"	an Event Morpheme with knowledge of the objects' class
"movingCargo"	same as above
"movingCargoLift"	
"movingTruckCargo"	
[Siskind, 2001]:	
"A hand is picking up a block"	two Event Morphemes with knowledge of the objects' class
[Ghanem et al., 2004]:	
"car exchange"	Semantic Module: the car is being driven away by a driver NOT being the one that brought it in
[Guler et al., 2003]:	
"wait"	Event Morpheme *stay*
"enter"	Semantic Module: *enter* implies the knowledge (Meta Knowledge) of an entry
"pick-up"	Event Morpheme
"package drop-off"	see the example in section 4.6
"exchange between people"	

Table 5.3: Mapping other approaches' events to Event Morphemes

Chapter 6

Conclusion and outlook

Wenn einer, der mit Mühe kaum
Geklettert ist auf einen Baum
Schon meint, dass er ein Vogel wär,
So irrt sich der.[1]
(Wilhelm Busch)

6.1 Introduction

In this thesis the application of the generic concept for mapping the results of low-level content analysis data to semantic event descriptions is presented. The constituting elements of this approach and their underlying concepts as well as an introduction to their application are shown. A scenario or events are not regarded as whole, but decomposed into small semantically meaningful parts (small but not atomic). By arranging these parts, the *Event Morphemes*, any scenario can be build.

The main contribution of the approach are the generality and the early stage at which the step from low-level to high-level representation is taken. This reasoning in the meta data domain is performed on small time frames. The reasoning on complexer scenes is done in the semantic space. Even an unsupervised self-assessment is possible using the semantic approach. The results will be assessed in the following section, section 6.2. Section 6.3 points out future research directions.

[1] If someone climbs laboriously into the branches of a tree and thinks himself a bird to be: wrong is he.

6.2 Assessing the results

The proposed concept was validated by implementing an Event Morpheme-based event detection system. The overall detection result of the system comes to 97,3% and 90,1% for precision and recall, respectively. This shows that the concept of Event Morphemes as an index for large surveillance content databases is exeedingly suitable for the modelling of relevant surveillance scenarios.

The results show that the presented concept's performance is at least equal to the state of the art. The difference to the other approaches in comparison is that the system presented here is a post-hoc meta data analysis system. It is to be expected that the incorporation of the Event Morpheme detectors within the VCA algorithm will increase the performance of the system. The information available directly in the VCA module allows to incorporate more input to the detector modules and even confidence measures for the recognition. This would also exploit the concept of arranging Event Morpheme classes in a taxonomy with increasing semantic predicate (see section 4.3.3). There are VCA systems that perform long-term analysis such as background models, 2,5 D maps or entry/exit detection and thus an (semi-) automated map of the scene creation. Such systems would even reduce the incorporation of the user in the event recognition process and this way enable the development of a more integrated system.

The system clearly relies on the content analysis algorithm. This not only means that improvement of the event detection includes improving the VCA. It also entails that the assessment of the retrieval performance is a possible approach to evaluate the performance of VCA algorithms.

The drawback of other approaches is the lack of generality. This is resolved by splitting up the information necessary to recognize an event into three parts. General domain knowledge about how an event "looks like", scenario specific information such as the layout of the scene etc., and search specific user restrictions are regarded separately. They are incorporated not until an actual query is posted.

An index for the search in large databases is set up best using Event Morphemes as semantic intermediate layer rather than searching directly on the low-level meta data. Faster searches are possible on the pre-processed index, with no decrease in generality.

The concept of Event Morphemes is capable of modeling the other approaches' events. Even the output of those systems based on tracking of limbs such as [Brand et al., 1997] and [Hakeem and Shah, 2004] can be mapped to Event Morphemes.

6.3 Outlook

An important question is how accurate semantic scene descriptions based on currently available content analysis algorithms can be. Another thing to be evaluated is the amount of event classes that can be recognized in real time. This would provide a flexibly configurable system to alert the surveillance operator when actions of interest take place, a kind of semantic filter, so to speak.

A conceivable extension would be the incorporation of audio as a second modality. Multimodal analysis leads to better performance in multimedia analysis [Snoek and Worring, 2003]. For the field of surveillance, this might be a different story, as surveillance scenes are neither composed nor do they have a musical soundtrack. So, the modality "audio" might be adding new events rather than influence the confidence of the visual clues. But this, of course, has to be investigated further.

On the semantic side, the development of a complete ontology is a matter of future research. The potential output of content analysis algorithms has to evolve in diversity to make the ontology complete and not limit it to events based on trajectory analysis and change detection in the background (e.g. removal). Of course, the concept of the Event Morphemes already is applicable, but still, yet, it has more potential.

Bibliography

The DARPA Agent Markup Language Homepage, 2000. URL `http://www.daml.org`.

Description of oil, 2000. URL `http://oil.semanticweb.org/`.

Chiraz Ben Abdelkader and Larry Davis. Detection of people carrying objects: a motion-based recognition approach. In *5th International conference on Automatic face and gesture recognition*, May 2002.

J. K. Aggarwal and Q. Cai. Human motion analysis: A review. *Computer Vision and Image Understanding*, 73(3):428 – 440, March 1997.

J. F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832 – 843, 1983.

J. F. Allen and G. Ferguson. Actions and events in temporal intervals. *Journal of Logic Computing*, 4(5):531 – 579, 1994.

James F. Allen. Towards a general theory of action and time. *Artificial Intelligence*, 23:123 – 154, 1984.

D. Ayers and M. Shah. Monitoring human behavior from video taken in an office environment. *Image and Vision Computing*, 19(12):833 – 846, October 2001.

Faisal I. Bashir and Ashfaq A. Khokhar. Video content modeling techniques: An overview. Technical report, Dept. of CS/ECE, UIC, 2003.

A. F. Bobick and J. W. Davis. The recognition of human movement using temporal templates. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 23(3):257 – 267, March 2001.

Bob Bolles and Ram Nevatia. A hierarchical video event ontology in owl. Arda challenge project final report, 2004. Reporting Period: June 1, 2004 - September 30, 2004.

Nikolaos G. Bourbakis, James R. Gattiker, and George Bebis. A synergistic model for representing and interpreting human activities and events from video. *International Journal on Artificial Intelligence Tools*, 12(1):101 – 116, 2003.

M. Brand, N. Oliver, and A. Pentland. Coupled hidden markov models for complex action recognition. In *Computer Vision and Pattern Recognition*, pages 994 – 999, June 1997.

C. Bregler. Learning and recognizing human dynamics in video sequences. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 568 – 574, June 1997.

François Brémond, Nicolas Maillot, Monique Thonnat, and Van-Thinh Vu. Ontologies for video events. Rr-5189, INRIA Sophia-Antipolis Research Unit, May 2004.

James Clark. XSL Transformations (XSLT), November 1999. URL `http://www.w3.org/TR/xslt`.

J. Connell, A. W. Senior, A. Hampapur, Y.-L. Tian, L. Brown, and S. Pankanti. Detection and tracking in the IBM PeopleVision System. In *IEEE International Conference on Multimedia and Expo (ICME '04)*, pages 1403 – 1406 Vol.2, June 2004.

D. Cunado, M. S. Nixon, and J. N. Carter. Using gait as a biometric, via phase-weighted magnitude spectra. In *1st Int. Conf. on Audio- and Video-Based Biometric Person Authentication*, pages 95 – 102, 1997.

F. Cupillard, A. Avanzi, F. Brémond, and M. Thonnat. Video understanding for metro surveillance. In *IEEE International Conference on Networking, Sensing and Control, Volume: 1*, pages 186 – 191, March 2004.

R. Cutler and L. S. Davis. Robust real-time periodic motion detection, analysis, and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):781 – 796, August 2000.

John Davies, Dieter Fensel, and Frank van Harmelen, editors. *Towards the Semantic Web*. Wiley, 2002.

J. W. Davis. Sequential reliable-inference for rapid detection of human actions. In *Computer Vision and Pattern Recognition Workshop*, pages 111 – 118, June 2004.

H. M. Dee and D. C. Hogg. On the feasibility of using a cognitive model to filter surveillance data. In *IEEE International Conference on Advanced Video and Signal-Based Surveillance, Como, Italy*, 2005.

I. Ersoy, F. Bunyak, and S. R. Subramanya. A framework for trajectory based visual event retrieval. In *International Conference on Information Technology: Coding and Computing (ITCC 2004), Volume: 2*, pages 23 – 27, 2004.

Alan Fern. *Learning Models and Formulas of a Temporal Event Logic*. PhD thesis, Purdue University, August 2004.

G.L. Foresti, C. Micheloni, and L. Snidaro. Event classification for automatic visual-based surveillance of parking lots. In *17th International Conference on Pattern Recognition, Volume: 3*, pages 314 – 317, August 2004.

Luis M. Fuentes and Sergio A. Velastin. Tracking-based event detection for CCTV systems. *Pattern Analysis and Applications*, 7(4), 2005.

D. M. Gavrila. The visual analysis of human movement: A survey. *Computer Vision and Image Understanding*, 73(1):82–98, 1999.

D. M. Gavrilla. The visual analysis of human movement: A survey. *Computer Vision and Image Understanding: CVIU*, 73(1):82 – 98, 1997.

M. R. Genesereth and R. E. Fikes. *Logical Foundations of Artificial Intelligence*. Morgan Kaufmann Publishers, 1987.

Nagia Ghanem, Daniel DeMenthon, David Doermann, and Larry Davis. Representation and recognition of events in surveillance video using petri nets. In *IEEE Workshop on Event Detection and Recognition*, page 112, June 2004.

W. Eric L. Grimson and Tomás Lozano-Pérez. Recognition and localization of overlapping parts from sparse data. A.i. memo 841, Massachusetts Institute of Technology Artificial Intelligence Laboratory, June 1985.

Alexei Gritai, Yaser Sheikh, and Mubarak Shah. On the use of anthropometry in the invariant analysis of human actions. In *ICPR (2)*, pages 923–926, 2004.

T: R: Gruber. Toward principles for the design of ontologies used for knowledge sharing. In N. Guarino and R. Poli, editors, *Formal Ontology in Conceptual Analysis and Knowledge Representation*. Kluwer Academic Publishers, 1995.

Sadiye Guler, Winnie H. Liang, and Ian A. Pushee. A video event detection and mining framework. In *Conference on Computer Vision and Pattern Recognition Workshop*, volume 4, pages 42–49, 2003.

Asaad Hakeem and Mubarak Shah. Ontology and taxonomy collaborated framework for meeting classification. In *17th conference of the International Conference on Pattern Recognition (ICPR)*, 2004.

Asnnd Hakeem, Yaser Sheik, and Mubarak Shah. CASE^E: A hierarchical event representation for the analysis of videos. In *The Nineteenth National Conference on Artificial Intelligence (AAAI)*, pages 263–268, 2004.

A. Hampapur, L. Brown, J. Connell, M. Lu, H. Merkl, S. Pankanti, A.W. Senior, C. Shu, and Y-L Tian. The IBM Smart Surveillance System. In *IEEE CVPR, Washington D.C.*, June 2004.

Ismail Haritaoglu, David Harwood, and Larry S. Davis. w^4: Real-time surveillance of people and their activities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):809 – 830, August 2000.

Ismail Haritaoglu, Ross Cutler, David Harwood, and Larry Davis. Backpack: Detection of people carrying objects using silhouettes. *Computer Vision and Image Understanding*, 81(3):385 – 397, March 2001.

Jerry R. Hobbs and Feng Pan. An ontology of time for the semantic web. *ACM Transactions on Asian Language Processing (TALIP): Special issue on Temporal Information Processing*, 3(1):66–85, March 2004.

Somboon Hongeng and Ramakant Nevatia. Multi-agent event recognition. In *International Conference on Computer Vision (ICCV'01)*, volume 2, July 2001.

Somboon Hongeng, Ram Nevatia, and François Brémond. Video-based event recognition: activity representation and probabilistic recognition methods. *Computer Vision and Image Understanding*, 96(2):129 – 162, November 2004.

Weiming Hu, Tieniu Tan, Liang Wang, and Steve Maybank. A survey on visual surveillance of object motion and behaviors. In *IEEE Transactions on Systems, Man and Cybernetics, Part C*, pages 334–352, Aug. 2004.

Jane Hunter. Adding multimedia to the semantic web - building an mpeg-7 ontology. International Semantic Web Working Symposium (SWWS), Jul. 2001.

ISO/MPEG. N4579, Text of ISO/IEC Draft Technical Report 15398-8 Information Technology - Multimedia Content Description Interface - Part 8 Extraction and Use of MPEG-7 Descriptions, March 2002.

Yuri A. Ivanov and Aaron F. Bobick. Recognition of visual activities and interactions by stochastic parsing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):852 – 872, August 2000.

O. Javed, S. Khan, Z. Rasheed, and M Shah. Camera handoff: Tracking in multiple uncalibrated stationary cameras. In *Workshop on Human Motion*, pages 113 – 118, December 2000.

A. Jepson and W. Richards. What makes a good feature? *Spatial Vision in Humans and Robots*, pages 89 – 125, 1991.

A. Kojima, T. Tamura, and K. Fukunaga. Textual description of human activities by tracking head and hand motions. In *16th International Conference of Pattern Recognition, Vol.2*, pages 1073 – 1077, August 2002.

J. Krumm, S. Harris, B. Meyers, B. Brumitt, M. Hale, and S. Shafer. Multi-camera multi-person tracking for easyliving. In *Third IEEE International Workshop on Visual Surveillance*, pages 3 – 10, July 2000.

O. Lassila and R. Swick. Resource description framework (rdf), January 1999. URL `http://www.w3.org/TR/PR-rdf-syntax/`.

Alan J. Lipton. Local application of optic flow to analyse rigid versus non-rigid motion. In *ICCV99 Workshop on Frame-Rate Applications*, September 1999.

Jiangung Lou, Qifeng Liu, Tieniu Tan, and Weiming Hu. Semantic interpretation of object activities in a surveillance system. In *IEEE 16th Int. Conf. Pattern Recognition*, volume 3, pages 777–780, 2002.

Dimitrios Makris and Tim Ellis. Automatic learning of an activity-based semantic scene model. In *IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS'03)*, pages 183 – 1488, July 2003.

Gérard Medioni, Isaac Cohen, Brémond, Somboon Hongeng, and Ramakant Nevatia. Event detection and analysis from video streams. In 8, editor, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, August, 2001 23. 873 – 889.

M. Meyer, M. Hötter, and T. Ohmacht. A new system for video-based detection of moving objects and its integration into digital networks. In *Security Technology, 30th Annual International Carnahan Conference*, Oct. 1996.

Stefan Müller-Schneiders, Thomas Jäger, Hartmut S. Loos, and Wolfgang Niem. Performance evaluation of a real time video surveillance system. In *2nd Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, October 2005.

Jacinto C. Nascimento, Mário A. T. Figueiredo, and Jorge S. Marques. Motion segmentation for activity surveillance. In *1st ISR Workshop on Systems, Decision and Control Robotic Monitoring and Surveillance, Lisbon*, June 2005.

M. Naylor and C. I. Attwood. ADVISOR Annotated Digital Video for Intelligent Surveillance and Optimised Retrieval. Ist1999-11287 : Advisor, ADVISOR, May 2003.

Holger Neuhaus. A semantic concept for the mapping of low-level analysis data to high-level scene description. In *Fourth International Workshop on Content-Based Multimedia Indexing (CBMI 2005)*, ISBN 952-15-1364-0, 21-23 June 2005. SuviSoft Oy Ltd. CDRom.

Nevatia, Hobbs, and Bolles. An ontology for video event representation. In *Conference on Computer Vision and Pattern Recognition Workshop*, pages 119 – 128, June 2004.

Lothar Papula. *Mathematik für Ingenieure und Naturwissenschaftler*, volume 3. Vieweg, 1997.

Sangho Park and J. K. Aggarwal. Recognition of two-person interactions using a hierarchical bayesian network. In *First ACM SIGMM international workshop on Video surveillance*, pages 65 – 76, 2003.

Sangho Park and Jake K. Aggarwal. Semantic-level understanding of human actions and interactions using event hierarchy. In *Conference on Computer Vision and Pattern Recognition Workshop*, pages 12 – 20, June 2004.

J.H. Piater, S. Richetto, and J. L. Crowley. Event-based activity analysis in live video using a generic object tracker. In *3rd IEEE Int. Workshop on PETS*, pages 1 – 8, June 2002.

C. Pinhanez and A. Bobick. Human action detection using pnf propagation of temporal constraints. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 898 – 904, June 1998.

Fatih Porikli and Tetsuji Haga. Event detection by eigenvector decomposition using object and frame features. In *Conference on Computer Vision and Pattern Recognition (CVPRW)*, volume 7, page 114, June 2004.

Cen Rao and Mubarak Shah. View-invariant representation and learning of human action. In *IEEE Workshop on Detection and Recognition of Events in Video*, July 2001.

J. M. Siskind. Grounding the lexical semantics of verbs in visual perception using force dynamics and event logic. *Journal of Artificial Intelligence Research (JAIR)*, 15:31 – 90, August 2001.

Michael K. Smith, Deborah McGuinness, Raphael Volz, and Chris Welty. Web ontology language (OWL) guide, November 2002. URL `http://www.w3.org/TR/2002/WD-owl-guide-20021104/`.

C.G.M. Snoek and M. Worring. Multimodal video indexing: A review of the state-of-the-art. *Multimedia Tools and Applications*, 2003. URL `http://citeseer.nj.nec.com/article/snoek03multimodal.html`.

Chris Stauffer and W. Eric L. Grimson. Learning patterns of activity using real-time tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):747 – 757, August 2000.

H. Stern, U. Kartoun, and A. Shmilovici. An expert system for surveillance picture understanding. In *NATO ASI, Data Fusion for Situation Monitoring, Incident Detection, Alert and Response Management*, 2003.

Alexander Toshev, François Brémond, and Monique Thonnat. An Apriori-based method for frequent composite event discovery in videos. In *ICVS*, page 10, 2006.

Van-Thinh Vu, François Brémond, and Monique Thonnat. Automatic video interpretation: A recognition algorithm for temporal scenarios based on pre-compiled scenario models. In *3rd International Conference on Vision System (ICVS'03)*, pages 523 – 533, April 2003.

T. Xiang, S. Gong, and D. Parkinson. Autonomous visual events detection and classification without explicit object-centered segmentation and tracking. In *British Machine Vision Conference (BMVC), Vol. 1*, pages 233 – 242, September 2002.

Lun Xin and Tieniu Tan. Ontology-based hierarchical conceptual model for semantic representation of events in dynamic scenes. In *2nd Joint*

IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2005., 15-16 Oct. 2005. Paper submission deadline: June 30^{th}, 2005.

Tao Zhao, Tianshu Wang, and Heung-Yeung Shum. Learning a highly structured motion model for 3d human tracking. In *The 5th Asian Conference on Computer Vision*, January 2002.

Anhang A

Zusammenfassung in deutscher Sprache

A.1 Einleitung

Zusammen mit dem wachsenden Bedürfnis nach Sicherheit wird eine zunehmende Menge Überwachungsinhalts geschaffen. Die Indizierung des Inhalts im Voraus ist unentbehrlich, um eine schnelle und zuverlässige Suche in den Aufnahmen von Hunderten oder Tausenden der in einer einzelnen Einrichtung installierten Überwachungssensoren (Kameras etc.) zu ermöglichen. Das Ziel des Smart Indexing & Retrieval (SIR; auch Semantic-Based Video Retrieval (SBVR), [Hu et al., 2004], Content Based Video Indexing and Retrieval (CBVIR), [Bashir and Khokhar, 2003] oder Automatic Forensic Video Retrieval (AFVR), [Hampapur et al., 2004]) ist die Erzeugung von Metadaten und somit die Ermöglichung effizienter Suche. Die Erzeugung dieser Metadaten muss automatisch auf Basis der Content-Analyse Algorithmen erfolgen, da die manuelle Erstellung in vernünftiger Zeit und zu vernünftigen Kosten immer schwieriger wird. Die eigentliche Herausforderung hierbei ist die semantische Analyse des Inhalts. Obwohl dieses Thema in mehreren Ansätzen adressiert wurde, geht die Suche nach einer Allzwecklösung weiter:

> Wenn in der Tat solch ein nützlicher Satz von generischen Aktionen definiert werden kann, würde es möglich sein, entsprechende Eigenschaften und Korrespondenzmethoden zu identifizieren, welche zu einem hohen Maße anwendungsunabhäng sind? (Gavrila [1999])

A.1.1 Problemstellung

Intuitiv hängt ein Index von der Art der Retrieval-Anfrage ab, die *erwartet* wird, durchgeführt zu werden. Leider ist nicht im Voraus bekannt, wonach gesucht werden wird. Wenn ein Index nach einem Szenario durchsucht wird, von dem zum Zeitpunkt der Indexerstellung nicht bekannt war, dass es relevant sein würde, entstehen unvermeidlich Probleme. Der Index muss „smart“ oder zumindest generisch sein, um solch eine Anfrage zu ermöglichen. Somit wäre, zusammen mit „Smart Retrival“, eine Suche nach jedem Szenario möglich. Also, was ist „Smart Indexing“, was ist „Smart Retrieval“?

Ein kluger Weise, einen Index zu erstellen, ist zu indizieren, was und wie es Sinn macht. „Klug“ im Sinne der Informationstheorie würde die Reduzierung der Redundanz bedeuten. „Smart Retrieval“ nutzt solch einen Smart Index auf die effizienteste Weise aus, d.h. der Index wird richtig interpretiert. Spezifische Domänen-Eigenschaften, z.B. Szenenanordnung, ebenso wie Benutzerbeschränkungen werden zur Retrieval-Zeit in Betracht gezogen.

Eine Reihe anderer Probleme entsteht, wenn eine Anfrage über semantische Schlüsselwörter (oder sogar freie Text-Anfragen) kommt. Die größte Schwierigkeit liegt in der Abbildung der low-level Bildrepresentation (Pixel) zu high-level (menschlicher) Semantik. Das bedeutet, dass während low-level Merkmale leicht extrahiert werden können, der Ausgangspunkt eines Retrievalprozesses normalerweise die semantische Frage des Benutzers ist. Die Abbildung von low-level Merkmalen, die der Computer verwendet einerseits auf die Anfrage durch einen Menschen andererseits illustriert das Problem, das allgemein als „Überbrücken der semantischen Lücke[1]“ beschrieben wird. Allerdings bedeutet die semantische Lücke zu überbrücken nicht allein, high-level Anfragen auf low-level Merkmale abzubilden. Die Essenz einer semantischen Frage ist das Verstehen der Bedeutung hinter der Frage. Das ist selbstverständlich auch benutzerabhängig, da es die Definition von Ausdrücken einschließt, die von der Domäne, in der der Nutzer sucht, abhängen. Das muss betrachtet werden, wenn eine Frage bearbeitet wird.

Das Vorangehende in Betracht ziehend, werden die Informationen ansteigend gemäß ihres semantischen Niveaus vom Pixel-Level (in der visuellen Domäne) zu semantischen Metadaten angeordnet. Die erste Ebene schließt die Bildverarbeitungsalgorithmen ein. In dieser These wird dieses Niveau *Niveau 0* genannt, da es der Anfang der Analyse ist und es nichts „unter“ diesem Niveau gibt. Die Algorithmen des *Niveau 0* sind jene, die bewegte Objekte segmentieren, Merkmale wie Farbe oder Textur extrahieren und Objektklassifikation durchführen. Ergebnis dieser Algorithmen und somit des

[1] bridging the semantic gap

Niveau 0 sind Metadaten, z.B die Position des Gegenstandes in Bildpunkt-Koordinaten, ein definierter Label der Klasse des Objekts etc.

Die Ergebnisse von *Niveau 0* sind der Input für *Niveau 1*, auf dem grundlegendes Schließen durchgeführt wird. Dies schließt das Erkennen von einfachen Ereignissen, Analyse der Richtung der Bewegung oder die Erkennung solcher Personen ein, die Waren [Abdelkader and Davis, 2002], z.B ein Rucksack [Haritaoglu et al., 2001] tragen: das Schließen ob Waren getragen werden oder nicht, beruht auf der Objektkontour, die die Metadatenausgabe von *Niveau 0* ist.

Niveau 2 ist schließlich das höchste semantische Niveau. Dieses Niveau vereinigt die Ergebnisse von *Niveau 1* zusammen mit semantischer Information über die Szene zu einer semantischen Beschreibung wie „Diebstahl eines Koffers“. Bild A.1 zeigt einen Überblick über die verschiedenen semantischen Niveaus.

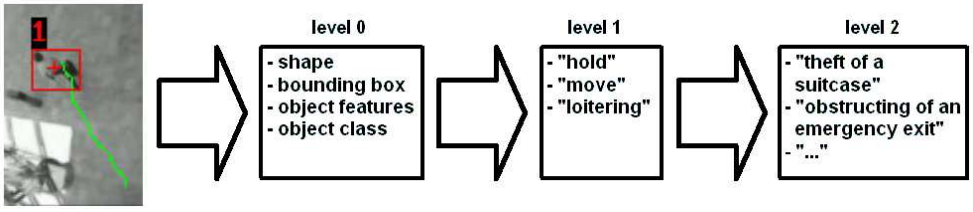


Abbildung A.1: Die verschiedenen Niveaus der Semantik in Szenenbeschreibungen.

Niveau 2 repräsentiert, wie ein Benutzer eine Szene beschreiben würde und die Form, die am wahrscheinlichsten ist, in der ein Nutzer eine Anfrage an ein Retrievalsystem durchführen würde.

A.1.2 Ein Gesamtsystem

Um ein System zu schaffen, das fähig ist, Ereignisse zu entdecken und diese Ereignisse zu einer semantischen Szenenbeschreibung zu verbinden, wird benötigt

1. einen low-level Analyse (VCA[2]) Algorithmus,
2. eine Ontologie für die Definition einer allgemeinen Beschreibungssprache,
3. ein Interferenzmechanismus, der die VCA-Metadaten auf ein semantisches Label abbildet,

[2]Video Content Analysis; Videoinhaltsanalyse

4. ein Werkzeug/Mechanismus, um eine entsprechende Karte der Szene zu erzeugen sowie
5. ein Retrievalinterface.

Diese Teile bilden das System wie folgt. Der VCA Algorithmus (1) analysiert die Bildfolge und gibt die Formen von Objekten, deren Trajektorien und Merkmale wie Farbe, Textur etc aus. So werden *Niveau 0* Metadaten erzeugt. Der Interferenzmechanismus (3) ist der springende Punkt des Systems: etwas, das hier nicht richtig bearbeitet wird, beeinflusst die Konsistenz der folgenden Stufen (ebenso wie Unzulänglichkeiten des VCA). Dieser Teil des Systems muss an den eingesetzten VCA angepasst werden. Ausgabe sind *Niveau 1* Metadaten. Die Ontologie für die allgemeine Beschreibungssprache (2) wird einerseits für das logische Schließen und andererseits für die Ausgabe an den Benutzer sowie die Abbildung der Frage des Benutzers auf systeminterne Termini benötigt. Ausserdem ermöglicht die Interferenzlogik, die die zugrunde liegende Ontologie verwendet, (zusammen mit 4) den Schritt auf *Niveau 2*. Sowohl die Ontologie als auch das Schließen sollten von den verwendeten VCA Algorithmen unabhängig sein, da diese Anpassung bereits am Übergang von *Niveau 0* zu *Niveau 1* durchgeführt wurde. Das Retrievalinterface (5) soll schließlich flexibel genug sein, um den Benutzer zu einer Vielfalt von Fragen zu befähigen und die Frage auf den Index abzubilden.

Die VCA Algorithmen auf *Niveau 0* sind eine wichtige Voraussetzung für die folgenden Stufen. Dieses Gebiet ist jedoch nicht im Fokus dieser Arbeit. Diese Arbeit führt ein semantisches Konzept für die Detektion, Repräsentation, Indexierung und Retrieval von Ereignissen auf Basis der Ausgaben eines VCA Algorithmus ein. So ist die Generierung von *Niveau 1* und *Niveau 2* von Interesse, ein generischer Ansatz für das Erkennen von Verhalten, zusammen mit einem passenden Konzept für Indexing & Retrieval.

A.1.3 Anwendungsgebiete

Ein *Smart Indexing* System soll zur automatischen Indizierung von Videoinhalt imstande sein, um effiziente Suchen zu ermöglichen. Der Index muss von geringer Redundanz, gleichwohl reich an Information sein. *Smart Retrieval* muss imstande sein, den Smart Index durch Kombinieren der Indices intelligent auszunutzen und zusätzliche Information wie Benutzerbeschränkungen und Szenenlayout mit einzubeziehen.

Ein Smart Indexing & Retrievalsystem muss an seiner Retrievalleistung gemessen werden. Wenn solch ein System also beurteilt wird, werden spezifische Ereignisse gesucht, und die Ergebnisse werden bewertet.

Zusammen mit Sicherheitssystemexperten[3] ist eine Reihe relevanter Use-Cases identifiziert worden. Zuerst sollen die folgenden Subereignisse detektiert werden: Aufnehmen, Abstellen und Hinfallen. Die ausgewählten Subereignisse sollen ermöglichen, die definierten Use-Cases zusammenzusetzen. Im Folgenden wurden die wiederzufindenden Fälle wurden definiert.

1. „Kampf“: eine Person fällt, verursacht durch eine vorbeigehende Person/ Person fällt hin und steht nicht wieder auf.
2. „Einkauf“: zahlt der Kunde oder nicht?
3. „Kriminelles Verhalten an einem Geldautomaten“: Ausspionieren der PIN einer anderen Person.
4. detektiere Bettler/Verkäufer auf der Straße.

A.2 Existierende Lösungen

Forschung im Feld der automatisierten Aktionenerkennung und Szenenbeschreibung läuft „bereits seit einiger Zeit“ [Gavrilla, 1997]. Für einen vollständigen Überblick sei auf Kapitel 2 verwiesen. Die folgenden Abschnitte zeigen Systeme und Forschungsarbeiten, die mit der präsentierten Arbeit in dem Sinne verwandt sind, als dass ein Modell dem Erkennungsprozess zu Grunde gelegt wird und (zu einem gewissen Maße) eine semantische Beschreibung der Szene erzeugt wird.

A.2.1 Modell-basierte Ereigniserkennung

[Hongeng et al., 2004] präsentieren ein Konzept, das Aktionen und Erkennungsmethoden repräsentiert, die diese Repräsentation einsetzen. Eine Aktion wird aus Aktionssträngen[4] zusammengesetzt, jeder Strang wird von einem einzelnen Akteur ausgeführt. Ein „Ein-Strang-Ereignis“ repräsentiert die Eigenschaften der Trajektorie und Form. Ein „Multi-Strang-Ereignis“ wird aus mehreren Aktionssträngen unter zeitlichen Einschränkungen zusammengesetzt und repräsentiert durch einen Ereignisgraphen, ähnlich Interval Algebra Netzwerken [Allen and Ferguson, 1994]. Ereignisse werden in mehreren Abstraktionsschichten angeordnet. Dieser Ansatz ist eng verwandt mit [Ivanov and Bobick, 2000], da externes Wissen in die erwartete Struktur des Aktionssmodells einbezogen wird. Dieses Modell ist hierarchisch aber modelliert nur Agenten, nicht Gegenstände.

[3] um genauer zu sein, Mitarbeiter des Bosch Sicherheitssystem-Produktbereichs
[4] action threads

[Cupillard et al., 2004] präsentieren ein System im Rahmen des europäischen Projekts ADVISOR [Naylor and Attwood, 2003]. Es ist ein Ansatz für die Online Erkennung eines Individuums, einer Gruppe von Leuten oder dem Verhalten von Menschenmengen im U-Bahn-Überwachungskontext, wobei mehrere Kameras einsetzt werden. Für die Repräsentation der Szene wird eine Hierarchie verwendet. Ausgang für die Szenarioerkennung sind

1. durch Experten definierte Szenariomodelle,
2. geometrische Information der beobachteten Umgebung, und
3. durch ein Vision Modul (VCA) verfolgte Personen, von dem angenommen wird, dass es das richtig tut.

Der Formalismus beruht auf drei Hauptideen:

1. definiere verschiedenartige Operatoren (Software-Module) für die Erkennung,
2. habe das ganze erforderliche Wissen im entsprechenden Operator, und
3. Beschreibung des Operators soll deklarativ sein, um eine erweiterbare Bibliothek von Operatoren zu erstellen.

Szenarien werden „im Ganzen“ definiert, also müssen Variationen durch verschiedene Detektormodule behandelt werden.

[Bourbakis et al., 2003] definieren ein Modell für die Darstellung, Erkennung und Interpretation menschlicher Aktionen. Das Modell beruht auf der hierarchischen Synergie von drei anderen Modellen: der Local/Global (L-G) Graph, dem Stochastische Petri-Netz (SPN) Graph und einem Neuronales Netzwerk (NN) Modell. Die Autoren machen eine Unterscheidung zwischen struktureller Kenntnis (Kenntnis über physischen Zustand) und funktionaler Kenntnis (Kenntnis über Änderungen und Ereignisse). Sie entwickeln Dynamische Multi-linked Hidden Markov Modelle (DML-HMM), um Gruppenaktionen zu interpretieren. Das Feld der Anwendung sind Ladungstätigkeiten am Flughafen mit Ereignissen wie „movingTruckCargo“, womit dieses System vollkommen domänenabhängig ist.

Ein auf Kräftedynamik gegründeter Ansatz wird durch [Siskind, 2001] präsentiert. Sie erstellen ein System zur Erkennung von Ereignissen, die durch einfache Verben der räumlichen Bewegung in kurzen Bildfolgen beschrieben sind. Die Semantik dieser Verben wird mit Event-Logikausdrücken angegeben, die Änderungen in kräftedynamischen Beziehungen zwischen den Teilnehmern des Ereignisses beschreiben.

[Fern, 2004] erweitert [Siskind, 2001] um die Entwicklung eines überwachten Lernalgorithmus, um high-level visuelle Ereignisse von low-level kräftedynamischer Repräsentation des Videos automatisch zu erhalten. Eine zeitliche Ereignisbeschreibungssprache wird eingeführt: AMA, „*A* nd's von „*M* eet's und *A* nd's“ (*MA-Timelines*)“. Eine *MA-Timeline* ist die Abfolge eines Zustandes, der einige Zeit hält *und* ein zweiter Zustand, der einige Zeit hält und den ersten *trifft*. *AMA* ist die Verbindung von *MA-timelines*. Eine Lernmethode wird basierend auf diesen Algorithmen entwickelt und auf Lernen von Ereignisdefinitionen aus Videos angewandt.

[Ghanem et al., 2004] repräsentieren und erkennen Ereignisse unter Verwendung von Petri- Netzen. Hieraus erstellen sie ein interaktives System, mit dem in Überwachungsvideos nach Ereignissen gesucht wird. Die Fragen sind nicht im Voraus bekannt und aus primitiven Ereignissen zusammengesetzt. Petri Netze werden sowohl für Repräsentation als auch Erkennungsmethodik verwandt. Eine graphische Benutzerschnittstelle wird verwendet, um Fragen zu formulieren. Diese Fragen werden dann auf eine Reihe von Petri-Netzen abgebildet. [Ghanem et al., 2004] nutzt auch eine Ereignisontologie, die Zustände, Ereignisse, zusammengesetzte Ereignisse/Szenen und Beziehungen definiert. Das System wird auf einem Parkplatz mit Ereignissen wie „Autos zählen“ und „Autoaustausch“ eingesetzt.

Xin and Tan [2005] schlagen als einen Ansatz für die Modellierung von Ereignissen und Analyse mit semantischen Repräsentationen ein System vor, das die zusammenhängende Information in ein hierarchisches Begriffsmodell (namentlich eine Ontologie) integriert und Ereignisse als bedeutende Änderungen und Abbildungen von Begriffseinheiten im Modell definiert. Drei grundlegende Ereignisbestandteile formen das Konzept: Entitäten, Wörter und eine Reihe von Attributen. Ereignisse sind repräsentiert als offensichtliche Eigenschaftsänderungen. Verschiedene Attribute von Regionen und bewegten Objekten sind Entitäten. Die Szene wird in verschiedene Gebiete geteilt, die manuell etikettiert werden. Wechselwirkungen zwischen bewegten Objekten und speziellen Gebieten werden durch ihre räumlichen Beziehungen beschrieben. Die Hierarchie der Ontologie besteht aus drei Niveaus. Die erste Ebene ist die Anordnung der Szene und zugehörige Einschränkungen. Das folgende Niveau enthält die Ontologie der bewegten Objekte, die die Zustände der Bewegung und die Konzepte, die obig erwähnte Wechselwirkungen beschreibt. Im dritten Niveau repräsentiert die semantische Ontologie das, was in der Szene vorkommt.

[Guler et al., 2003] präsentieren ein Framework für Ereigniserkennung in Videos und deren Speicherung für Detektion, Annotation und Browsing. Der Detektorteil betrachtet Ereignisse in einer hierarchischen Struktur bestehend aus

1. den Trackingdaten,
2. einfachem Verhalten wie „warten“, „gehen“ oder „aufnehmen“ und
3. Tätigkeiten des höheren Niveaus zusammengesetzt aus diesen einfachen Verhalten, wie „Meeting“, „Paket absetzen“ oder „Austausch zwischen Personen“.

A.2.2 Offene Punkte

Um Ereignisse erkennen zu können, sind neben dem Video drei Inputs erforderlich:

1. Domänenwissen, d. h. die Kenntnis darüber, wie ein Ereignis in der spezifischen Domäne „aussieht“,
2. Kenntnis über die Anordnung der Szenen und
3. Benutzerbeschränkungen wie z.B. erlaubte Dauer des Aufenthalts in einem sensitiven Gebiet.

Die Hauptherausforderung des semantischen Videoretrievals ist die Tatsache, dass zum Zeitpunkt der Indexerstellung nicht bekannt sein kann, welche Ereignisse gefragt werden. Somit wäre es nicht klug, direkt online detektierte Multi-Strang-Multi-Agenten Szenarien zu indizieren. Um dennoch nach ihnen suchen zu können, müssen sinnvolle Sub-Ereignisse indiziert werden.

Angemessen ist hierbei, nur szenariounabhängige und somit grundlegende Bestandteile von Ereignissen zu verwenden. Bisher vorgeschlagene Ansätze für Ereigniserkennung sind mehr oder weniger anwendungsspezifische Echtzeitsysteme. Die Ansätze, die Taxonomie/Ontologie einsetzen, tragen Information über Szenen bzw. Benutzerbeschränkungen direkt in ihren Detektionsalgorithmen.

Was also wirklich fehlt, ist eine Repräsentation, die generisch ist und auf komplexe Szenenbeschreibungen angewandt werden kann und somit Smart Indexing ermöglicht. Auf in dieser Weise erstellten Indices kann effizient Retrieval durchgeführt werden, selbst wenn die gesuchten Ereignisse zum Indizierungszeitpunkt nicht bekannt waren. Regeln und Beschränkungen gehören nicht in den Index, sie sollen im Retrievalschritt aufgestellt werden. Dasselbe gilt für Kontextinformation: welches Szenario „Diebstahl“ repräsentiert, und welches „Austausch“ kann nur vom Anwender aufgelöst werden, dem ein gestohlener Koffer gemeldet wird.

A.3 Mittel und Wege der Lösung

A.3.1 Einführung

Dieser Abschnitt präsentiert die theoretischen Grundlagen der Arbeit. Es werden Ontologien als ein Werkzeug für Wissensrepräsentation und Inferenz vorgestellt. Die Syntax der Ontologie-Sprache, die die Regeln des semantischen Konzepts in dieser Arbeit formuliert, wird gesondert zusammengefasst. Die eingesetzte Software wird gezeigt, ebenso wie die verwendeten Retrievalqualitätsmaße.

A.3.2 Ontologie

Ontologien wurden in der Künstlichen Intelligenz entwickelt, um das Teilen und die Wiederverwendung von Wissen zu ermöglichen. Eine Ontologie stellt ein geteiltes und allgemeines Verstehen einer Domäne bereit, die zwischen Menschen und heterogen verteilten Anwendungssystemen kommuniziert werden kann. Dies ist auch eine explizite Konzeptualisierung (d. h. Metainformation) welche die Semantik der Daten beschreibt.

Ontologie Design

Reusability (Wiederverwendbarkeit) ist eine der wichtigsten Eigenschaften. Die Forschung ist darauf ausgerichtet, Technologien zu entwickeln, die eine Wiederverwendung von Ontologien ermöglichen. Um diese Anforderungen zu erreichen, muss eine Ontologie aus kleinen Modulen bestehen, die eine hohe innere Kohäsion und eine beschränkte Menge an Abhängigkeiten zwischen den Modulen haben. Gruber hat die Designprinzipien der Ontologie 1995 ausgedrückt. Es gibt eine Notwendigkeit objektiver Kriterien, um Designs durchzuführen und auszuwerten. Ein einleitender Satz von Designkriterien für Ontologien, deren Zweck das Teilen von Wissen und Interoperabilität unter auf einer geteilten Konzeptualisierung gegründeten Programmen ist, sind [Gruber, 1995]:

1. **Klarheit**: Die in einer Ontologie beschriebenen Definitionen sollten objektiv und unabhängig vom sozialen oder technischen Kontext sein. Eine Definition sollte (wenn möglich) in logischen Axiomen gemacht und in einer natürlichen Sprache dokumentiert werden.

2. **Kohärenz**: Eine Ontologie sollte zusammenhängend sein. Das bedeutet, dass die Inferenzen in keinem Widerspruch zu den gemachten Definitionen stehen. Die für die Definitionen eingesetzten Axiome sollten logisch konsistent sein.

3. **Erweiterbarkeit**: Es sollte möglich sein, über das existierende Vokabular neue Terme zu definieren, ohne dass die existierenden Definitionen wiederholt geprüft und korrigiert werden müssen.

4. **Minimale Verwendung von Implementationsdetails**: Die Konzipierung einer Ontologie sollte nicht auf einer teilweisen Kodierung abhngig von der späteren Implementierung erfolgen.

5. **Minimale ontologische Festlegungen**: Eine Ontologie sollte eine minimale, aber ausreichende Menge an Behauptungen über die Domäne haben, die sie modelliert.

A.3.3 VERL

Im Sommer und Herbst 2003 sponserte die Advanced Research and Development Activity (ARDA) der US-amerikanischen Regierung das „Callenge Projekt Videoereignistaxonomie“. Das Ergebnis war eine formale Sprache zur Beschreibung von Ontologie von Ereignissen, genannt VERL (Video Event Representation Language). Eine dazugehörige Sprache, genannt VEML (Video Event Markup Language), für die Annotation der Instanzen der in VERL beschriebenen Ereignisse, wurde ebenfalls entwickelt. Syntax und Anwendung von VERL und VEML, wie präsentiert, in [Bolles and Nevatia, 2004] findet sich in Abschnitt 3.2.3.

A.3.4 Der VCA Algorithmus

Es ist nicht im Fokus dieser Arbeit, einen Bildverarbeitungsalgorithmus zu entwickeln. Der VCA-Input wird durch das von [Müller-Schneiders et al., 2005] präsentierte System bereitgestellt. Die Detektion und Beschreibung von bewegten Objekten beruhen auf einer objektorientierten, statistischen Multifeatureanalyse von Videosequenzen. Diese Analyse ist selbstadaptiv zu einer beobachteten Szene (Meyer et al. [1996]). Das System ist fähig, „Abspaltungen“ und „Verschmelzungen“ zu identifizieren, wobei sich ein einzelner Gegenstand in zwei oder mehr Gegenstände aufspaltet, bzw. zwei oder mehr Gegenstände zu einem Gegenstand verschmelzen. Zusätzlich werden Gegenstände detektiert, die ruhend sind oder wurden. Das System kann auch zwischen entfernten und liegen gelassenen Gegenständen unterscheiden.

So stellt das System die folgenden Outputs bereit

1. Objekt Regionen,

2. die Spuren (Tracks) der Objekte,

3. ein „ruhend“ Flag und

4. Unterscheidung zwischen abgestellten/entfernten Objekten.

Diese sind - durch ein MPEG-7 Dokument - der Input für das in dieser Arbeit präsentierte System.

A.3.5 Maße für Retrievalqualität

Precision und Recall sind die grundlegenden im Auswerten von Suchstrategien benutzten Maße. Diese nehmen an:

1. Es gibt eine Reihe von Einträgen einer Datenbank, die für eine Suche relevant sind

2. Einträge werden angenommen, entweder relevant oder irrelevant zu sein.

3. Der tatsächliche Retrievalsatz kann nicht dem Satz von relevanten Einträgen vollkommen gleichen.

Recall

RECALL ist das Verhältnis der Anzahl von erhaltenen relevanten Einträgen zur Gesamtzahl von relevanten Einträgen in der Datenbank. Es wird normalerweise als Prozentsatz ausgedrückt.

Recall = (Anzahl Erhaltener und relevanter) / (Gesamtzahl möglicher Relevanter).

Precision

PRECISION ist das Verhältnis der Anzahl von erhaltenen relevanten Einträgen zur Gesamtanzahl von irrelevanten und relevanten erhaltenen Einträgen. Es wird normalerweise als Prozentsatz ausgedrückt.

Precision = (Anzahl Erhaltener und relevanter) / (Gesamtzahl Erhaltener).

Die „Ground Truth“

Es gibt keine deterministische Methodik, um zu verstehen, was für eine Suche des Benutzers relevant ist. Für die Bewertung eines Retrievalsystems müssen

also die relevanten Dokumente per Hand ausgewählt werden. Im Fall eines Ereignisdetektionssystems besteht die Ground Truth[5] aus:

- dem Typ des Ereignisses,
- dem Zeitfenster, in dem das Ereignis auftrat und
- der dem Objekt, das das Ereignis ausführt, durch das VCA System zugewiesenen ID.

A.4 Ein semantisches Konzept für die Abbildung von low-level Analyseergebnissen auf high-level Szenenbeschreibungen

Morphem
aus Wikipedia, der freien Enzyklopädie.

Ein Morphem ist die kleinste bedeutungstragende Einheit einer Sprache auf der Inhalts- und Formebene in der langue, dem Sprachsystem. Es lässt sich auch als kleinste semantisch interpretierbare Konstituente eines Wortes bezeichnen.

A.4.1 Einführung

Diese Arbeit präsentiert ein Konzept für die Repräsentation und Detektion von Ereignissen. Das Konzept ist durch [Neuhaus, 2005] eingeführt worden. Die Entwicklung und Anwendung dieses Ansatzes ebenso wie ein „proof of concept“ werden präsentiert.

A.4.2 Terminologie

Zuerst gibt es einige Ausdrücke, die vor Gebrauch klar definiert werden müssen. Die meisten Ausdrücke von Inhaltsanalysen haben ihr entsprechendes Gegenstück auf semantischer Seite, aber es ist keine *1-zu-1* Entsprechung.

[5] der „Gold Standard“, die „alles umfassende Wahrheit“

low-level und high-level *Low-level* beschreibt die Eigenschaften, die automatisch mittels Signalverarbeitung (z.B. Bildverarbeitung) extrahiert werden können. Die Extraktion und Segmentierung von Objekten in Bildfolgen ist low-level. *High-level* beschreibt semantische Bedeutungen und Begriffe die ein Benutzer einsetzt, wenn er eine Szene beschreibt. In Bezug auf automatisierte Inhaltsanalyse und Indizieren beschreibt „high-level“ den Prozess, Schlüsse aus weiterer Analyse von low-level Eigenschaften zu ziehen.

Eigenschaften und Attribute Die Ausdrücke *Eigenschaften* und *Attribute* sollen, wie hier verwendet, die Eigenschaften von low-level Daten bzw. den high-level semantischen Begriffen beschreiben. Eigenschaft wird im Feld der Inhaltsanalyse ohnehin verwendet. Für das semantische Konzept, das hier präsentiert wird, beschreibt „Attribut“ die Elemente der semantischen Repräsentation. Das kann die Anzahl von Objekten, die Richtung der Bewegung, oder die „ruhend“ Zeit eines Objektes sein.

Bewegte Region und Objekt Die *bewegte Region* ist der Output vom *Niveau 0*, des VCA-Systems. Es ist eine beliebig geformte Menge von Bildpunkten, die - in Theorie - ein Objekt repräsentieren. Die Realität zeigt jedoch, dass gegenwärtige Inhaltsanalyse dies noch nicht leisten kann. So muss ein *Objekt* in der „realen Welt“ von einer *bewegten Region* unterschieden werden. Dies nicht nur aus Gründen der Semantik, sondern auch, weil *eine* bewegte Region mehr als *ein* bewegtes Objekt repräsentieren kann. Ein einfaches Beispiel für diesen Fall wäre eine Person, die einen Koffer trägt.

Andererseits kann ein *Objekt* durch mehr als eine bewegte Region repräsentiert sein sein. Wenn eine Person z.B. hinter einen Lastwagen geht und auf der anderen Seite wieder erscheint, werden einige Inhaltsanalyse-Systeme einen neuen ID zuteilen, obwohl es dieselbe Person ist.

Ein anderes denkbares Szenario wäre die Ablieferung eines Koffers: zuerst wird der Gegenstand ‘Koffer’ durch dieselbe bewegte Region wie die Person vertreten, die ihn trägt. Dann steht der Koffer auf dem Fußboden, repräsentiert durch seine eigene Region. Wenn der Koffer schließlich von einer zweiten Person weggetragen wird, wird deren bewegte Region auch den Koffer repräsentieren.

Ereignis Im Allgemeinen beschreibt *Ereignis* das, was zu einem spezifischen Zeitpunkt geschieht. In dieser Arbeit repräsentiert „Ereignis“ nur einen elementaren Teil einer Reihenfolge von Aktionen und ist mit „primitiven Er-

eignissen“[Ghanem et al., 2004] der Literatur vergleichbar.

Event Morphem[6] *Event Morpheme* scheinen keine Übereinstimmung in der Literatur zu haben. Im Grunde repräsentieren sie den „Standpunkt“ eines Objekts in einem Ereignis und verbinden so die Objekte mit den entsprechenden Ereignissen. Ihre Beziehung ist immer eine *1-zu-1* Beziehung: ein Objekt - ein Event Morphem - ein Ereignis.

Meta Knowledge[7] Der Ausdruck *Meta Knowledge*, so wie er in dieser Arbeit verwendet wird, vereinigt das, was allgemein als „a priori Wissen“ bezeichnet wird zusammen mit allgemeiner *Ereignismorphologie* und kontextspezifischen *Benutzerbeschränkungen*. Das a priori Wissen könnte z.B. eine Karte der Szene sein, während die allgemeine Ereignismorphologie kontextunabhängiges Wissen darüber ist, wie Ereignisse zusammen gesetzt sind. Benutzerbeschränkungen beschreiben z.B. die ortsbhängige erlaubte Dauer eines Aufenthaltes.

Semantisches Modul Das *Semantische Modul* ist der Output des ontologiebasierten Interferenzmechanismus. Es ist vergleichbar zum „zusammengesetzten Ereignis“[Ghanem et al., 2004] oder „Prozess“[Nevatia et al., 2004]. In den meisten Fällen nutzt es das Meta Knowledge, um über die Ereignisse durch Verbinden spezifischer Interferenzregeln zu schließen. Tatsächlich ist das Semantische Modul das, wonach die Datenbank abgefragt werden wird.

Szene Eine *Szene*, wie in dieser Arbeit verwandt, besteht aus Semantischen Modulen, die Ereignisse in ihrer kontextspezifischen Instanz repräsentieren. Man kann dies als eine „Kette von Ereignissen“ verstehen. Eine Beschreibung einer Szene ist *Niveau 2*.

A.4.3 Event Morpheme

Die Idee hinter Event Morphemen ist die Zerlegung einer Szene in semantisch aussagekräftige Teile (klein, aber nicht atomar), jedes aus der Perspek-

[6] „Ereignismorphem“
[7] „Metawissen“

tive eines „Hauptobjekts“. Ein Event Morphem besteht aus dem Hauptobjekt, dessen Attributen, und dem/den Objekt(en), mit dem/denen das Hauptobjekt interagiert (ggf.). Die Attribute sind die Repräsentationen der durch den eingesetzten Contentanalyse-Algorithmus extrahierten Eigenschaften. Ein Event Morphem repräsentiert ein Zeitfenster, in dem diese Attribute analysiert und interpretiert werden. Dieser Interferenzprozess erzeugt einen semantischen Label, der ein solches Zeitfenster repräsentiert.

Das semantische Konzept *„Event Morpheme“* repräsentiert also eine semantische Zwischenschicht. Es bildet (auf einer frühen Stufe) die low-level Metadaten auf ein semantisches Etikett ab.

Eine Karte der Szene Eine Karte der Szene ist für die Erkennung ortsabhängiger Ereignisse unentbehrlich. Wenn z.B. ein Auto in einem ausgewiesenen Parkplatz abgestellt wird, ist dies ein erlaubtes Ereignis. Ein Auto, das in der Mitte einer Feuer-Gasse abgestellt wird und sie somit versperrt, würde unerwünscht sein. Ein anderes Beispiel, die Karte der Szene zu verwenden, wäre zu wissen, dass dort, wo sich Objekt „X“ bewegt, einen Fluss ist. Daraus kann das System schließen, dass das bewegte Objekt schwimmt.

Ähnlich einer Karte der Szene ist Kenntnis über die Perspektive der Kamera und deren Parameter, die durchschnittliche Objektgröße und alles das, was allgemein als „a priori Wissen“ beschrieben wird. In dieser Arbeit ist es die benutzerunabhängige Kenntnis bezüglich des Ortes und der Aufteilung der Szene. Die Karte der Szene und die semantische Beschreibung der Orte sind darin die Voraussetzung für den Schritt von *Niveau 1* auf *Niveau 2*.

Event Morphem Taxonomie

Die semantischen Label eines durch Event Morpheme repräsentierten Datenmusters werden in hierarchischen Klassen angeordnet. „Zustand“ und „Übergang“ sind die zwei Klassentypen. Zustände werden durch Event Morpheme mit semantischen Labeln wie „Bewegung“ oder „Halt“ repräsentiert, während Übergänge „abgestellt“ oder „aufgenommen“ wären. „Übergang“ ist unterteilbar in „Start“ und „Stop“ für jene Übergänge, die Zustände beginnen beziehungsweise beenden. Dieses Eigenschaften betreffen das Prinzip der Semantischen Interpolation (siehe Abschnitt A.4.3). Eine beispielhafte Implementierung eines solchen Ansatzes, wie eingeführt in [Neuhaus, 2005], in VERL Notierung wäre:

$SUBTYPE(Zustand, ev)$
$SUBTYPE(Uebergang, ev)$

$SUBTYPE(Start, Uebergang)$
$SUBTYPE(Stop, Uebergang)$

$SUBTYPE(Aufnehmen, Start)$
$SUBTYPE(Abstellen, Stop)$

$SUBTYPE(Bewegung, Zustand)$
$SUBTYPE(Halt, Zustand)$

Um voll vom Konzept der Event Morphem Taxonomie zu profitieren, muss die Taxonomie dem eingesetzten VCA Algorithmus angepasst werden. Je feiner das Event Morphem Prädikat sein soll, desto unsicherer wird die Klassifizierung sein. Der Vorteil, die semantischen Label der Event Morpheme hierarchisch zu ordnen, ist das Ermöglichen einer „vielleicht Aussage“. Auf diese Weise ist es möglich, auf das allgemeinere Label zu fallen, wenn der zugrunde liegende Interferenzmechanismus eine geringere Wahrscheinlichkeit für den Spezifischeren aufweist.

Eine andere Motivation für eine taxonomische Anordnung von Event Morphemen sind die ersten experimentellen Ergebnisse. Wenn Szenarien betrachtet werden, in denen „aufnehmen“ detektiert wird, zeigt sich, dass dabei nicht nur tatsächlich Ereignisse mit Aufheben vorkommen, sondern auch solche von Autos, die aus einer Parkbucht ausparken oder Personen, die aufstehen, nachdem sie längere Zeit gesessen haben. So haben ähnliche Muster im VCA Output unterschiedliche Semantik. Es wird etwas aus der Szene „entfernt“. Es ergibt sich dieser initiale Ansatz für die Event Morphem Taxonomie:

$SUBTYPE(Bewegung, Zustand)$
$SUBTYPE(Halt, Zustand)$
$SUBTYPE(einbringen, Uebergang)$
$SUBTYPE(entfernen, Uebergang)$
$SUBTYPE(abstellen, einbringen)$
$SUBTYPE(aufnehmen, entfernen)$

Semantische Interpolation

Das Prinzip der Semantischen Interpolation beschreibt den Prozess, die Anwesenheit eines Objektes zu erkennen, das durch die Inhaltsanalyse nicht zu jeder Zeit explizit entdeckt wird. Ein Beispiel ist ein hereingetragener Koffer: der VCA Algorithmus detektiert zwar das Abstellen, jedoch nicht den getragenen Koffer. So muss er durch das Interferenzmodul eingefügt werden.

Logisches Ausschließen von Fehlerkennungen (Reasoning Out)

Das Prinzip des 'frühen' Schrittes von der low-level Repräsentation auf einen semantischen Label hat einen weiteren Vorteil. Es wird möglich, zu *wissen*, welche Ergebnisse Fehlerkennungen für eine bestimmte Klasse des Event Morphems sind. Wenn z.B. ein angenommen hereingebrachtes Objekt existierte, bevor das Abstellen detektiert wurde, ist offensichtlich, dass diese Behauptung falsch ist.

A.4.4 Wo ist der Index?

Das komplette Konzept, das hier präsentiert wird, ist ein Konzept für Indizieren und Retrieval. Also, als ein letzter Schritt muss die Trennung zwischen dem Index und dem Retrievalteil getan werden. Mit anderen Worten: wo ist der Index? Da für das Retrieval, die Information, so wie sie in den Event Morphemen enthalten ist, benötigt wird, sollte genau das im Index stehen. So geht die Linie also „durch“ die Event Morpheme (siehe Bild A.2). Wenn ein Index auf dem generischen Prinzip der Event Morpheme erstellt wird, so wird Smart Retrieval ermöglicht, ohne den Zugriff auf low-level Eigenschaften zu verlieren.

A.5 Ergebnisse

A.5.1 Einführung

Die Ergebnisse der exemplarischen Implementierung eines Event Morphembasierten Ereignisdetektionssystems zeigen, dass in verschiedenen Szenarien Ereignisse unterschiedlicher Komplexität erkannt werden. Es wird auch demonstriert, wie die Ergebnisse verbessert werden, indem falsche Treffer durch

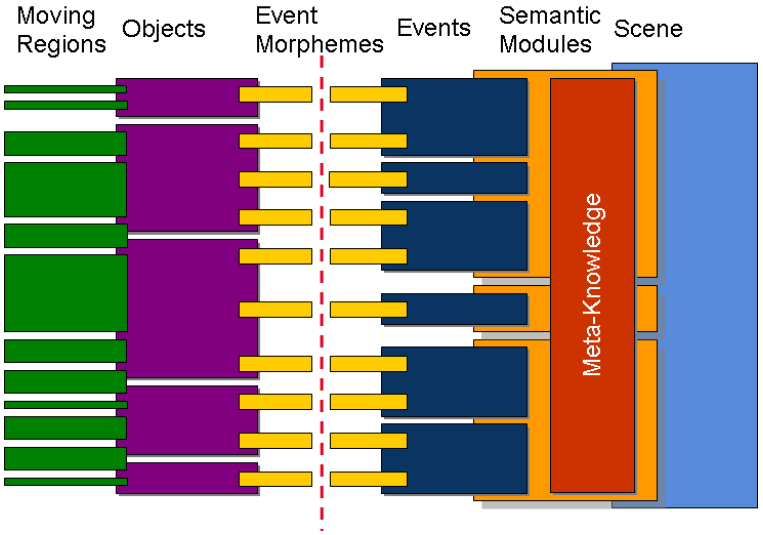


Abbildung A.2: Die Event Morpheme *sind* der Index

logisches Schließen eliminiert werden. Die Ergebnisse, die hier als ein „proof-of-concept“ präsentiert werden, sind in mehreren Schritten erlangt worden. Zuerst wurde ein Lehrsatz bestehend aus 304 Sequenzen mit verschiedenartige Szenarien bearbeitet. Dann ist die Event Morphem Detektion auf Basis des Outputs des VCA Algorithmus entwickelt worden, der diese Sequenzen durchrechnete. Schließlich ist die Detektion auf dem Testsatz durchgeführt worden, der die Anwendungsfälle enthält.

A.5.2 Realisierung der Anwendungsfälle

Wie Eingangs erwähnt wurden zusammen mit Sicherheitssystemexperten eine Reihe relevanter Anwendungsfälle identifiziert. Es sollen die folgenden Subereignisse entdeckt werden: Aufnehmen, Abstellen uns Hinfallen. Die ausgewählten Subereignisse sollen dazu geeignet sein, die definierten Anwendungsfälle zusammenzusetzen. Diese waren:

1. „Kampf“: eine Person fällt, verursacht durch eine vorbeigehende Person/ Person fällt hin und steht nicht wieder auf.
2. „Einkauf“: zahlt der Kunde oder nicht?
3. „Kriminelles Verhalten an einem Geldautomaten“: Ausspionieren der PIN einer anderen Person.
4. detektiere Bettler/Verkäufer auf der Straße.

A.5.3 Ausschließen von Fehlerkennungen („reasoning-out“)

Dieser Abschnitt zeigt die Ergebnisse der Anwendung Reasoning-out Strategie. Die folgenden Verbesserungen wurden auf dem Lehrsatz erreicht. Die Fehlerkennungsrate nahm ab, und somit nahm die Genauigkeit zu. Das war für die Aufdeckung von *entfernen* eine Abnahme von Fehldetektionen um 20,2 %, resultierend in einer Zunahme von 17,7 % für Precision. Für die Aufdeckung von *einbringen* waren die Ergebnisse noch signifikanter. Die Fehlererkennungsrate fiel um 35,8 % und somit verbesserte sich der Wert für Precision um 29,0 %. Das Ausschließen von Fehlerkennungen beeinflusst allein die Precision; der Recall wird durch die nicht Erkannten beeinflusst. Leider können die Letzteren nicht mittels post-hoc Analyse reduziert werden.

A.5.4 Ergebnisse für die Testsequenzen

Dies sind die Ergebnisse für die Detektion der identifizierten Anwendungsfälle. Diese Ergebnisse wurden gegen eine von Hand kommentierte Ground Truth verifiziert. Die Erkennungsraten und die resultierenden Werte für Precision und Recall sind in Tabelle A.1 dargestellt.

Szenario		Ground Truth	richtig Erkannte	falsch Erkannte	nicht Erkannte	Recall	Precision
Einkaufen	aufnehmen	29	27	1	2	93,1	96,4
	abstellen	9	7	0	2	77,8	100
	bezahlen	8	7	0	1	87,5	100
	Diebstahl	12	11	1	1	91,7	91,7
Bettler[8]		2	2	0	0	100	100
Kampf		11	10	0	1	90,9	100
Bankautomat		10	9	0	1	90	100

Tabelle A.1: Ergebnisse für die Detektion der definierten Anwendungsfälle

Vergleich mit anderen Ansätzen

Tabelle A.2 zeigt das präsentierte System im Vergleich mit anderen Ansätzen aus der Literatur:

[8]Objekte von Hand gekennzeichnet

System	Ground Truth	Szenarien	richtig Erkannte	falsch Erkannte	nicht Erkannte	Recall	Precision	Verfahren
[Brand et al., 1997]	52	3					94,2	Coupled Hidden Markov Models
[Cupillard et al., 2004]	45	5	42	7	3	93,3	85,7	three-stage formalism (operators-knowledge-description)
[Guler et al., 2003]		10					92,3	HMM
[Hakeem and Shah, 2004]	122	3	115	6	7	94,3	95,0	Regelbasiertes System und Zustandsautomat
Neuhaus [2006][9]	81	7	73	2	8	90,1	97,3	(siehe Kapitel 4)
[Rao and Shah, 2001]	23	2	16	2	7	69,6	88,9	HMM
[Xiang et al., 2002]	54	5	36	19	18	66,7	65,5	EM zur Clusterung, Selektion durch Minimum Description Length (MDL)

Tabelle A.2: Vergleich der Systeme

[9] *Diese Arbeit*, um genau zu sein!

A.6 Auswertung und Ausblick

A.6.1 Einführung

In dieser Arbeit wird die Anwendung des generischen Konzepts für die Abbildung von low-level Analysedaten auf semantische Szenenbeschreibungen, wie eingeführt in [Neuhaus, 2005], präsentiert. Die konstituierenden Elemente dieses Ansatzes und ihre zugrunde liegenden Begriffe, ebenso wie eine Einführung in ihre Anwendung werden gezeigt. Ein Szenario oder Ereignisse werden nicht als Ganzes betrachtet, sondern in kleine semantisch bedeutungsvolle Teile (klein aber nicht atomar) zerlegt. Durch Ordnen dieser Teile, der *Event Morpheme*, kann jedes Szenario unter Verwendung dieser Teile repräsentiert werden.

Der Hauptbeitrag des Ansatzes ist die Allgemeingültigkeit und die frühe Stufe, auf der der Schritt von low-level auf high-level Repräsetationen vorgenommen wird. Das Schließen in der Metadatendomäne wird in kleinen Zeitfenstern durchgeführt. Das Schließen komplexerer Szenen wird in der semantischen Domäne vollzogen.

A.6.2 Bewertung der Ergebnisse

Das gesamte Detektionsergebnis des Systems kommt auf 97,3 % für Precision und 90,1 % für Recall. Das zeigt, dass das Konzept der Event Morpheme als ein Index für große Überwachungsinhalt-Datenbanken passend ist für das Modellieren der relevanten Überwachungsszenarien.

Die Ergebnisse zeigen, dass die Leistung des präsentierten Konzepts mindestens dem Stand der Technik gleich ist. Der Unterschied zu den anderen Ansätzen im Vergleich besteht darin, dass das System hier ein post-hoc Metadatenanalysesystem ist.

Das System ist klar abhängig vom verwendeten VCA Algorithmus. Das bedeutet nicht nur, dass Verbesserung der Ereignisaufdeckung ein Verbessern der VCA einschließt. Es hat auch zur Folge, dass die Einschätzung der Retrievalleistung ein möglicher Ansatz ist, um die Leistung von VCA Algorithmen zu bewerten.

Der Nachteil anderer Annäherungen ist der Mangel an der Allgemeingültigkeit. Das wird durch das Aufteilen der Information gelöst: allgemeines Domänenwissen darüber, wie ein Ereignis „aussieht“, spezifische Informationen wie die Anordnung der Szene und spezifische Benutzerbeschränkungen werden gesondert betrachtet. Sie werden vereinigt, wenn eine tatsächliche Anfrage gestellt wird. Das Konzept der Event Morpheme ist dazu fähig, die Ereignisse der anderen Ansätze zu modellieren.

A.6.3 Ausblick

Eine wichtige Frage ist, wie genau semantische Szene-Beschreibungen auf Basis zur Zeit vorhandener VCA Algorithmen sein können. Eine andere Sache, die zu evaluieren ist, ist die Menge von Ereignisklassen, die in Echtzeit erkannt werden können. Das würde ein flexibel konfigurierbares System bereitstellen, das den Surveillanceoperator alarmiert, wenn Ereignisse von Interesse stattfinden - eine Art semantischer Filter sozusagen.

Eine denkbare Erweiterung wäre die Miteinbeziehung von Audio als eine zweite Modalität. Multimodale Analyse führt zu besserer Leistung bei der Multimediaanalyse [Snoek and Worring, 2003]. Für das Feld der Überwachung könnte das anders sein, da Überwachungsszenarien nicht „nach Drehbuch“ zusammengesetzt werden. Dennoch könnte Audio neue Ereignisse hinzufügen. Aber das muss noch untersucht werden.

Auf der semantischen Seite ist die Entwicklung einer vollständigen Ontologie eine Angelegenheit der zukünftigen Forschung. Der potentielle Output von Inhaltsanalysealgorithmen muss sich breit entwickeln, um die Ontologie vollständig zu machen. Selbstverständlich ist das Konzept der Event Morpheme bereits jetzt anwendbar, es hat jedoch noch mehr Potential.

Appendix B

Results for training sequences

These are the results obtained during the development of the detector modules for *bring-in* and *take-out*. "fuzzy now" names the number of frames that have passed between the detection of "idle/removed" and the occurrence of the split (see section 4.3.3).

	true positives					false positives					false negatives				
fuzzy now	1	8	15	25	∞	1	8	15	25	∞	1	8	15	25	∞
TOTAL	0	1	15	28	44	33	54	112	164	274	85	84	70	60	47
Precision	0	1,82	11,81	14,58	13,84										
Recall	0	1,18	17,65	31,82	48,35										
take-out	0	0	10	19	23	3	8	44	74	109	49	49	39	32	28
Precision	0	0	18,52	20,43	17,42										
Recall	0	0	20,41	37,25	45,10										
bring-in	0	1	2	5	17	30	46	68	90	165	31	30	29	27	18
						2	14	26	46	199					
Precision	0	2,13	2,86	5,26	9,34										
Recall	0	3,23	6,45	15,63	48,57										
reasoned-out	0	0	0	0	2	5	19	33	51	93					
new bring-in	0	1	2	5	15	25	27	35	39	72	31	30	29	27	20
Precision	0	3,57	5,41	11,36	17,24										
Recall	0	3,23	6,45	15,63	42,86										
reasoned-out	0	0	0	0	1	2	9	41	72	92					
new take-out	0	0	10	19	22	1	-1	3	2	17	49	49	39	32	29
Precision	0	0	76,92	90,48	56,41										
Recall	0	0	20,41	37,25	43,14										
reasoned-out	0	0	0	0	3	7	28	74	123	185					
new TOTAL	0	1	15	28	41	26	26	38	41	89	85	84	70	60	50
Precision	0	3,70	28,30	40,58	31,54										
Recall	0	1,18	17,65	31,812	45,05										

Table B.1: Results of the detection of events in the training sequences

Anhang C

Thesen

1. Ereignisse lassen sich detektieren, in dem nicht nach dem ganzen Ereignis gesucht wird, sondern das zu suchende Ereignis in semantisch sinnvolle Teile aufgelöst wird und nach der Abfolge dieser Teile gesucht wird.

2. Fehlerkennungsraten in Content Analyse Systemen können durch logisches Schließen auf semantischer Ebene gesenkt werden.

3. Es existiert ein Vokabular für die Synthese beliebiger Ereignisse

4. Ein solches Vokabular lässt sich auf die Ausgabe eines jeden Content Analyse Systems anwenden.

Appendix D

Propositions

1. It is possible to detect scenarios by resolving the sought-after scenario in semantically meaningful parts and search for the succession of these parts.
2. It is possible to reduce false positive rates in content analysis algorithms by reasoning on the semantic level.
3. A vocabulary for the synthesis of arbitrary scenarios does exist.
4. Such a vocabulary is adaptable to the output of any content analysis system.

Appendix E

Used abbreviations and symbols

Term	Meaning
SIR	Smart Indexing & Retrieval
VERL	Video Event Representation Language
GMM	Gaussian Mixture Model
HMM	Hidden Markov Model
DBN	Dynamic Bayesian Network
VCA	Video Content Analysis
www	World Wide Web
W3C	World Wide Web Consortium
MPEG	Moving Picture Experts Group
MPEG-7	Multimedia Content Description Interface
XML	Extensible Markup Language
HTML	Hypertext Markup Language
RDF	Resource Description Framework
OIL	Ontology Interference Layer
DARPA	Defense Advanced Research Projects Agency
DAML	DARPA Agent Markup Language
OWL	Web Ontology Language

Erklärung

Ich versichere, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus anderen Quellen direkt oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet.

Weitere Personen waren an der inhaltlich-materiellen Erstellung der vorliegenden Arbeit nicht beteiligt. Insbesondere habe ich hierfür nicht die entgeltliche Hilfe von Vermittlungs- bzw. Beratungsdiensten (Promotionsberater oder anderer Personen) in Anspruch genommen. Niemand hat von mir unmittelbar oder mittelbar geldwerte Leistungen für Arbeiten erhalten, die im Zusammenhang mit dem Inhalte der vorgelegten Dissertation stehen.

Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder ähnlicher Form einer Prüfungsbehörde vorgelegt. Ich bin darauf hingewiesen worden, dass die Unrichtigkeit der vorstehenden Erklärung als Täuschungsversuch angesehen wird und den erfolglosen Abbruch des Promotionsverfahrens zu Folge hat.

5. November 2008 Holger Neuhaus