

# **Unsupervised Learning of Event and Object Classes from Video**

**by**

*Muralikrishna Sridhar*

**Submitted in accordance with the requirements  
for the degree of Doctor of Philosophy.**



**UNIVERSITY OF LEEDS**

**The University of Leeds  
School of Computing**

**December 2010**

**The candidate confirms that the work submitted is his own and that the appropriate credit has been given where reference has been made to the work of others. This copy has been supplied on the understanding that it is copyright material and that no quotation from the thesis may be published without proper acknowledgement.**

# Acknowledgements

First and foremost, I would like to thank my supervisors Anthony Cohn and David Hogg for providing excellent guidance through the course of my PhD. I have enjoyed every supervisory meeting with them, as they encouraged me to propose, discuss and develop novel ideas. At the same time, they have helped me develop a sense of criticality, that is crucial for the practice of good science. I thank them for all their contributions of time, ideas, and funding to make my Ph.D. experience productive and stimulating.

My sincere thanks to University of Leeds for providing the necessary facilities and support to carry on my work. Thanks to my lab mates who have helped me during the finishing stages of the thesis. My special thanks to Patrick Ott whose support is invaluable. I gratefully acknowledge the funding sources that made my Ph.D. work possible. I was funded by the EPSRC (EP/D061334/1) and the EU (FP7 Project 214975, Co-Friend). I also thank colleagues in the Co-friend project for their support.

Lastly, I would like to thank my family for all their love and encouragement. I owe my grandfather, who has been a source of inspiration and for making me enthusiastic about science from a very early age. He always had the confidence that I would make novel contributions in my PhD. I owe my grandmother for her persistent efforts to help me with my studies at school. I owe my mother for her untiring encouragement and support, that I have always relied on during the most difficult times. It is due to her untiring efforts that I have been able to pursue my dreams. Most of all I owe my wife Bhavani for being a supportive, patient and a wonderful companion through this entire journey. Her positive attitude and clarity of thought have always been an inspiration, and has given me the much needed strength and confidence, especially during the last stages of my PhD.

# Abstract

We present a method for unsupervised learning of event classes from videos in which multiple activities may occur simultaneously. Unsupervised discovery of event classes avoids the need to hand-crafted event classes and thereby makes it possible in principle to scale-up to the huge number of event classes that occur in the real world. Research into an unsupervised approach has important consequences for tasks such as video understanding and summarization, modelling usual and unusual behaviour and video indexing for retrieval. These tasks are becoming increasingly important for scenarios such as surveillance, video search, robotic vision and sports highlights extraction as a consequence of the increasing proliferation of videos.

The proposed approach is underpinned by a generative probabilistic model for events and a graphical representation for the qualitative spatial relationships between objects and their temporal evolution. Given a set of tracks for the objects within a scene, a set of event classes is derived from the most likely decomposition of the ‘activity graph’ of spatio-temporal relationships between all pairs of objects into a set of labelled events involving subsets of these objects.

The posterior probability of candidate solutions favours decompositions in which events of the same class have a similar relational structure, together with three other measures of well-formedness. A Markov Chain Monte Carlo (MCMC) procedure is used to efficiently search for the MAP solution. This search moves between possible decompositions of the activity graph into sets of unlabelled events and at each move adds a close to optimal labellings (for this decomposition) using spectral clustering.

Experiments on simulated and real data show that the discovered event classes are often semantically meaningful and correspond well with ground-truth event classes assigned by hand.

Event Learning is followed by learning of functional object categories. Equivalence classes of objects are discovered on the basis of their similar functional role in multiple event instantiations. Objects are represented in a multidimensional space that captures their functional role in all the events. Unsupervised learning in this space results in functional object-categories.

Experiments in the domain of aircraft handling suggests that our spatio-temporal representation together with the learning techniques are a promising framework for learning functional object-categories from video.

# Declarations

Some parts of the work presented in this thesis have been published in the following articles:

**Sridhar, M. and Cohn, A. G. and Hogg, D. C.**, Learning Functional Object-Categories from a Relational Spatio-Temporal Representation, In *Proceedings of the European Conference on Artificial Intelligence*, 2008.

**Sridhar, M. and Cohn, A. G. and Hogg, D. C.**, Unsupervised Learning of Event Classes from Video, In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2010.

**Sridhar, M. and Cohn, A. G. and Hogg, D. C.**, Discovering an Event Taxonomy from Video using Qualitative Spatio-temporal Graphs, In *Proceedings of the European Conference on Artificial Intelligence*, 2010.

**Sridhar, M. and Cohn, A. G. and Hogg, D. C.**, Relational Graph Mining for Learning Events from Video, In *Proceedings of the Starting AI Researcher Symposium*, 2010.



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Characterizing Activities . . . . .	2
1.2	Goals and Challenges . . . . .	4
1.3	Approach . . . . .	6
1.4	Novelty and Significance . . . . .	7
1.5	Thesis Overview . . . . .	8
<b>2</b>	<b>Related Work</b>	<b>10</b>
2.1	Three Aspects of Event Analysis . . . . .	11
2.1.1	Domain and Task . . . . .	12
2.1.2	Learning . . . . .	12
2.1.3	Characterizing Complexity . . . . .	14
2.2	Pattern Recognition Techniques . . . . .	16
2.2.1	Supervised Learning Approaches . . . . .	16
2.2.2	Unsupervised Learning Approaches . . . . .	17
2.3	State Space Methods . . . . .	20
2.3.1	Supervised Approaches . . . . .	20
2.3.2	Unsupervised Approaches . . . . .	22
2.4	Grammars . . . . .	25
2.4.1	Supervised Approaches . . . . .	26
2.4.2	Unsupervised Approaches . . . . .	27
2.5	Logical and Relational Models . . . . .	29
2.5.1	Manually Defined Setting . . . . .	30
2.5.2	Supervised Approaches using ILP . . . . .	30
2.5.3	Unsupervised Approaches . . . . .	32
2.6	Learning with Relational Graphs . . . . .	33
2.6.1	Graph Mining . . . . .	34
2.6.2	Graph Classification and Clustering . . . . .	37

2.6.3	Graph Sampling . . . . .	38
2.7	Learning of Object Classes . . . . .	39
2.7.1	Visual Appearance based Object Classes . . . . .	40
2.7.2	Affordance Based Object Categories . . . . .	42
2.8	Qualitative Spatio-Temporal Relations . . . . .	43
2.9	Characterizing the Proposed Approach. . . . .	45
2.10	Applicability of Related Approaches . . . . .	46
2.11	Conclusions . . . . .	48
<b>3</b>	<b>Representation of Interactions</b>	<b>49</b>
3.1	Introduction . . . . .	49
3.2	Qualitative Spatio-Temporal Relationships . . . . .	51
3.3	Interactions and Interaction Graphs . . . . .	54
3.3.1	Interaction Graph . . . . .	58
3.4	Comparing Interactions Spatio-Temporally. . . . .	61
3.5	Interactions Embedded in Space and Time . . . . .	66
3.5.1	Most Likely Interaction Graph for a Set of Tracks . . . . .	66
3.5.2	Likely Embeddings of an Interaction Graph . . . . .	71
3.6	Conclusion . . . . .	72
<b>4</b>	<b>Learning Events from Activities</b>	<b>73</b>
4.1	Introduction . . . . .	73
4.1.1	Overview of the Unsupervised Event Learning Framework . . . . .	74
4.2	Event-like Properties in a Simple Setting . . . . .	77
4.3	A Generative Process for a Complex Setting . . . . .	79
4.4	Search for the Most Likely Interpretation . . . . .	87
4.4.1	Searching for the Optimal Interpretation . . . . .	88
4.5	Interactivity . . . . .	92
4.6	Conclusion . . . . .	96
<b>5</b>	<b>Learning Functional Object Classes</b>	<b>97</b>
5.1	Introduction . . . . .	97
5.2	Unsupervised Learning of Functional Object Classes . . . . .	98
5.2.1	A Procedure for Mining Functional Roles . . . . .	104
5.3	Functional Relationships . . . . .	105
5.4	Conclusions . . . . .	106

<b>6</b>	<b>Evaluation of Activity Understanding</b>	<b>107</b>
6.1	Introduction . . . . .	107
6.2	Data Sets, Ground Truth and Evaluation Procedures . . . . .	108
6.2.1	Synthetic Datasets . . . . .	108
6.2.2	Real Datasets . . . . .	108
6.2.3	Processing the Video Data Sets to Obtain Tracks . . . . .	109
6.3	Evaluation of the HMM based Procedure . . . . .	111
6.4	Experimental Settings, Parameter Estimation and Evaluation Scores . . . . .	113
6.5	Evaluation of Event Learning on Real Data . . . . .	121
6.6	Evaluating Functional Object Classes and Functional Relationships . . . . .	126
6.7	Overall Conclusion . . . . .	128
<b>7</b>	<b>Summary and Future Work</b>	<b>129</b>
7.1	Contributions . . . . .	129
7.1.1	Representation of Activities . . . . .	129
7.1.2	A Generative Process for Modelling Activities . . . . .	131
7.1.3	Unsupervised Event Learning . . . . .	131
7.1.4	Functional Object Classes and Functional Relationships . . . . .	132
7.2	Future Work . . . . .	132
7.3	Concluding Remarks . . . . .	134
<b>A</b>	<b>The Metropolis Hastings Algorithm</b>	<b>135</b>
<b>B</b>	<b>Generation of Synthetic Datasets</b>	<b>137</b>
<b>C</b>	<b>Evaluation Scores for Event Covers and Event Classes</b>	<b>139</b>
<b>D</b>	<b>Temporal Relationships between two Intervals</b>	<b>141</b>
	<b>Bibliography</b>	<b>142</b>

# List of Figures

1.1	Three different domains with associated event and object classes. . . . .	2
2.1	Clustering of events, where events are trajectories. . . . .	18
2.2	Clustering of events which are a sequence of qualitatively interesting states. . . . .	19
2.3	A Bayesian network for detecting complex events. . . . .	21
2.4	Clustering of events into event classes. . . . .	23
2.5	Dynamic Bayesian networks for modelling multiple parallel processes. . . . .	24
2.6	Topic models for discovering events in a scene. . . . .	26
2.7	Background knowledge of positive examples (+), negative examples (-) and facts. . . . .	31
2.8	Edge expansion lattice, in which one edge is added at a time from the top in order to obtain the target graph at the bottom. . . . .	35
2.9	Allen’s temporal relationships between pairs of intervals. . . . .	44
2.10	Topological (RCC-5) and directional relationships are shown. . . . .	44
3.1	An example of an interaction and its embedding. . . . .	50
3.2	The CNG of the qualitative spatial relations used in this work. . . . .	51
3.3	Co-temporal spatial relationships and temporally overlapping episodes. . . . .	54
3.4	An illustration of two ways of representing our notion of an interaction. . . . .	56
3.5	The interaction graph for representing the interaction in Fig. 3.4. . . . .	59
3.6	Two sub-interactions and their respective interaction graphs are illustrated. . . . .	62
3.7	An interaction graph is represented as a bag of sub-graphs . . . . .	64
3.8	The non-parametric representation of cluster densities are illustrated. . . . .	65
3.9	Two circles for formulating the notion of region based distance between two objects. . . . .	68
3.10	Observation models of a HMM for qualitative spatial relationships is shown. . . . .	69
3.11	A state transition diagram is shown for the HMM. . . . .	70
3.12	Two embeddings for an interaction graph. . . . .	72

4.1	An illustration of event classes and events. . . . .	75
4.2	Illustration of a simple setting of event classes and events. . . . .	78
4.3	An activity graph as an intermediate representation between event classes and a set of tracks. . . . .	82
4.4	Four types of moves for the MCMC search. . . . .	91
4.5	The notion of interactivity is illustrated with three scenarios. . . . .	93
4.6	Illustrating the computation of interactivity with windows on layer 3 nodes. . . . .	94
5.1	The relationship between event activities and their object classes is illustrated. . . . .	98
5.2	A scenario where several occurrences of the <i>taking away</i> event classes can be found. . . . .	99
5.3	Interaction, interaction graphs and predicates for the <i>taking-away</i> event class. . . . .	100
5.4	A lattice that represents event classes with different degrees of instantiations. . . . .	102
6.1	An image from the airport apron dataset. . . . .	109
6.2	A segment with which the HMM based procedure for obtaining spatial relationships is trained and evaluated. . . . .	112
6.3	Trace of various properties with respect to the Markov chain. . . . .	116
6.4	A plot of accuracy with respect to variation of the meta-parameters for expected size, interactivity and frequency. . . . .	118
6.5	Evaluation of the robustness of the generative process to an increases in the presence of each complicating factor. . . . .	120
6.6	Representative occurrences of prototypes, coincidence and recovered noisy event are shown for event class 1. . . . .	122
6.7	Representative occurrences of prototypes, coincidence and recovered noisy event are shown for event class 2. . . . .	124
6.8	Example of two coincidental interactions, which are not classified as belonging to any of the learned event classes. . . . .	125
6.9	The five functional object classes that were learned are shown. . . . .	127

# List of Tables

5.1	An object by event role matrix is illustrated. . . . .	104
6.1	Table showing the accuracies for the three event-like properties and their combination. . . . .	114
6.2	Quantitive evaluation of the baseline and generalized model with respect to two pre-defined event classes. . . . .	125
6.3	Evaluation of functional object classes with respect to a predefined set of object classes. . . . .	127

# Chapter 1

## Introduction

---

Humans have a distinctive ability to observe and understand the apparently complex world of *activities* in simple terms. Even though activities may present themselves visually as a continuously changing stream of complex phenomena, the human mind tends to organize them compactly in terms of meaningful abstractions such as *event classes* and *object classes*. For example, kitchen activities are simply comprehended in terms of event classes that represent procedures involved in making hot drinks and associated object classes such as spoons, sugar bowls and cups. Activities for three domains with their associated event and object classes are illustrated in Fig. 1.1.

Event and object classes are closely related to the idea of *interactions* that is central to this work. It is intuitive to think of an event class as describing a similar type of interactions between objects, for the purpose of reaching a goal. These objects are naturally organized into *functional object classes* such that those of the same class (e.g. spoons) play a similar functional role (e.g. transferring sugar) whenever an event belonging to the associated event class (coffee making) occurs. Consequently, objects of the same class (e.g. spoons) interact in a very similar way with objects of other classes (e.g. cups, sugar bowls), across event occurrences of the coffee making event class.

These close connections between event classes, object classes and interactions have inspired the following question that is addressed in this thesis: if we were to point a camera at activities for a certain domain and assume no prior knowledge of event and object classes for any particular domain, is it possible for a machine to learn activities, more precisely to analyse interactions and learn the event classes, object classes and their relationships?

This thesis proposes a framework in which a machine can be programmed to understand about activities for a domain in an *unsupervised way* i.e. without having to encode

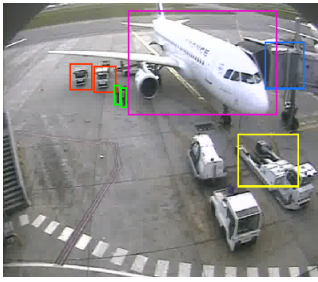
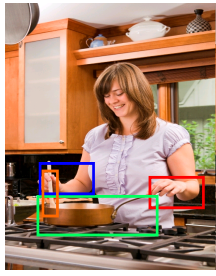
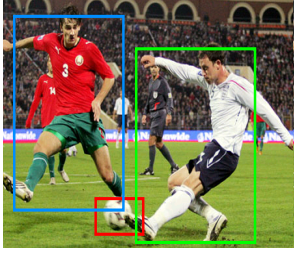
Activities			
Event Classes	Unloading Bridge On	Baking a cake Making Coffee	Tackle Goal kick
Object Classes	Plane Trolley	Spoon Cup	Player Ball

Figure 1.1: Three different domains - aircraft apron, kitchen, football match - where objects are interacting with each other. Examples of associated event and object classes are given below each of the respective domains.

knowledge about the event and object classes of any particular domain. The ability to understand activities can be potentially useful for machines that learn from observation and help humans carry out activities efficiently. Learning from observation can enable a machine to adapt when placed in new circumstances, predict what is going to happen and react accordingly, transfer learned knowledge to related domains, and invent new ways of performing certain tasks. Moreover, since events, objects and their relationships are so fundamentally related to natural language categories such as verbs, nouns and sentences that relate them, this research task represents a small but potentially significant step toward bridging the gap between perception, understanding and natural language.

Section 1.1 describes the key insights and the concepts that are central to the thesis. Section 1.2 details the goals and the challenges that need to be overcome to meet the goals. Section 1.3 describes an overview of the approach adopted in the thesis. Section 1.4 describes the novelty of the proposed approach and the significance of the work. Section 1.5 concludes this chapter by discussing an overview of the rest of the thesis.

## 1.1 Characterizing Activities

The following are the key concepts used to characterize activities: (i) domain; (ii) qualitative spatio-temporal relationships; (iii) interactions; (iv) event classes; (v) events; (vi)



complicating factors such as overlap, coincidences and noise;(vii) functional object classes; (viii) functional relationships. Each of these concepts are described below.

**Activities and Domains.** First we clarify our understanding of the word activities. The phrase *human activities* or more simply *activities* is generally used to refer to the entire diversity of things that humans do. A domain such as a kitchen, airport, workshop involve a subset of human activities directed towards a finite set of semantically meaningful goals. For example, a kitchen domain may be characterized by activities that are directed towards goals such as preparing meals, or washing vessels and extended hours of kitchen activities may be observed and recorded in the form of a video. It is assumed here that activities are generally planned and executed by people who have an understanding of a domain, e.g. a cook for a kitchen, airport authorities for aircraft handling activities.

**Qualitative Spatio-temporal Relationships.** An important premise in this work is that occurrences of interest in activities are often accompanied by changes in qualitative spatial states (e.g. near, far) between sequences of regions, where each sequence corresponds to a single moving object. We refer to a sequence of regions corresponding to a single moving object as *region histories*. These occurrences of interest can be modelled by suitably combining qualitative spatial and temporal relationships between the corresponding region histories. This type combination is referred to as *qualitative spatio-temporal relationships*.

**Interactions.** In this work, interactions are regarded as a special subset of qualitative spatio-temporal relationships. They are used to model a sequence of distinct qualitative spatial states between a set of region histories. We represent interactions in a graph based structure called interaction graphs. Interactions facilitate well defined similarity measures and the application of machine learning techniques.

**Event Classes.** Intuitively, an event class represents spatio-temporally similar ways of performing some task. More formally, an event class is regarded as a probability distribution over a finite set of similar interactions or equivalently interaction graphs. For example, an unloading event class may represent spatio-temporally similar interactions between vehicles such as loaders, trolleys and planes for the task of transferring luggage from a plane to an airport.

**Events.** Events are regarded in this work as occurrences of event classes in space and time. More expressly, an event is a set of tracklets (part of tracks) such that they are an embedding of the spatio-temporal relationships modelled by an event class. Event classes may have many event occurrences in extended periods of activity. For example, several occurrences of the unloading event class may be expected to be observed over extended periods of aircraft apron activity.

**Complicating Factors.** Three factors are regarded as inducing complexities when activities are observed and these are referred to as complicating factors. First, events are allowed to *overlap* i.e. they may share tracklets. Second, there are often occurrences that do not embody any of the event classes that are associated with activities for a domain. We refer to these occurrences as *coincidences* in this work. Finally, when activities are observed and their video is processed, complexities in image processing tends to give rise to changes in spatio-temporal relationships. We refer to this as *observation noise*.

**Functional Object Classes.** Functional object classes are regarded as being closely associated with event classes. These are classes of objects whose instances have similar functionalities or functional roles across event occurrences of the same class. For example, objects of a functional class such as trolleys play a similar functional role (of carrying the bags from the loader to the airport) across event occurrences of the unloading event class.

**Functional Relationships.** Functional relationships are significant correlations between event classes and functional object classes. Functional relationships associate event classes, which are just spatio-temporal relationships without any object type, to particular functional object classes. For example, associating an unloading event classes to potential functional object classes such as loaders, trolleys and planes is the description of their functional relationships.

## 1.2 Goals and Challenges

Our goal of *unsupervised video activity understanding* is the task of learning event classes, functional object classes and their functional relationships assuming no prior knowledge about these entities.

We use the word *unsupervised* in the sense that the machine does not have any prior knowledge about the event and functional object classes for the domain of interest (or any

other domain).

The goals described above are challenging with regard to both the representation and learning, for the following reasons. First of all, a good representation should be able to abstract only the relevant information from video. The videos considered here are a sequence of millions of images and contain a lot of information in them. The abstraction of relevant information is the first step to learning from such complex data. Specifically, a good representation needs to be expressive enough to capture spatial and temporal dependencies that characterize interactions. At the same time, the representation needs to abstract just the relevant information and represent it in a way that interactions can be compared quantitatively. The representation should also be robust enough to variations in spatio-temporal relationships arising either due to the intrinsic variations amongst events of the same class or due to instabilities in image processing.

The second aspect, which is the task of unsupervised learning, is challenging because neither the segmentation of the tracks (that compose activities) into events nor the event classes are known. When the event classes are known in advance, as in the supervised learning setting, events may be recognized and segmented in activities. On the other hand, when the segmentation of activities into events is known in advance, event classes may be found by clustering event descriptions.

The task of jointly learning events and the event classes is difficult, especially due to the three complicating factors. The presence of overlap between events makes the search space of events more complex. The presence of coincidences adds further complexity as it becomes necessary to incorporate properties that could be used to differentiate events from coincidences. Finally, the task of jointly learning events and event classes is made further difficult due to the presence of noise.

The task of learning functional object classes and functional relationships is also challenging as it requires a formulation in which functional roles are discovered and then objects are represented in terms of their functional roles with respect to events. This representation has to be robust enough to deal with uncertainties in the event classes that are learned from potentially noisy video data.

All these aspects of representation and learning make the desired goal of activity understanding a challenging problem. The following section describes our approach in addressing these challenges.

### 1.3 Overview of the Framework

The approach for understanding activities is explained in three parts: (i) qualitative spatio-temporal relationships and interactions; (ii) event classes and a generative process; (iii) learning functional object classes and functional relationships.

**Qualitative Spatio-temporal Relationships and Interactions.** In order to represent abstract qualitatively interesting aspects of complex occurrences in video activities, we simplify continuous space using a few qualitative spatial relationships. This makes it possible to afford the representation of complex temporal dependencies using qualitative temporal relationships. We introduce the notion of an interaction to refer to a restricted subset of qualitative spatio-temporal relationships that facilitate well defined similarity measures. Interactions are represented in the form of graphs which facilitate the application of graph based learning mechanisms. Finally, in order to make this description robust to observation noise from the tracked output, a generative process is used to estimate the true occurrences of qualitative spatial relationships.

**Event Classes and a Generative Process.** We define our notion of event classes, in order to model spatio-temporally similar ways of performing some task. Accordingly an event class is defined as a probability distribution over a finite set of interaction graphs. According to this probability distribution, the more common ways of performing a task have higher probability than other ways. This idea has an intuitively appealing basis in real events across domains, for example, event classes such as making coffee, tackling and football tackles. There are usually some standard or prototypical ways in which these tasks are performed, while other ways are less probable.

We model activities for a domain as arising from a generative process. The first aspect of this generative process is a set of event classes for the activities of a domain. Event classes are characterized by certain desirable properties called event-like properties. These properties help distinguishing between events and coincidences. The second aspect of the generative process is an *activity graph* which captures the spatio-temporal relationships between all the tracks present in the activities. The activity graph captures spatio-temporal relationships that correspond to events and coincidences. The final aspect is that a set of tracks that are observed and are regarded as a possible embedding of an activity graph. The generative process models the joint probability distribution between these three aspects, namely the event classes, activity graph and a possible embedding.

In the unsupervised learning setting, a set of tracks is observed for a video from a certain domain. The generative process is given *a priori* as specified above. The goal is

to discover the latent configurations of this process (which includes event classes, activity graph, events and coincidences) i.e. to find the most likely *interpretation* of the video. The posterior probability for any candidate interpretation is a measure of how likely it is that the candidate interpretation could have generated the observed set of tracks. The Maximum a Posteriori (MAP) solution is found by efficiently sampling the space of posterior distributions of candidate interpretations using Markov Chain Monte Carlo (MCMC) techniques.

**Functional Object Classes and Functional Relationships.** Once event classes are learned using the approach described above, the framework proceeds to learning functional object classes. In order to learn functional object classes, sets of objects that have the same functional roles with respect to each of the event classes are discovered. Objects are then represented in terms of the functional role they play and then clustered into functional object classes. The final aspect of activity understanding is learning the associations between event classes and functional object classes. The functional relationships between the functional object classes and event classes are learned by considering only the significant correlations between them.

## 1.4 Novelty and Significance

This thesis introduces a novel framework for understanding activities from video. The framework is unsupervised and therefore does not assume knowledge about any particular domain. The following are the novel and significant contributions of this work.

1. This work represents activities in terms of spatio-temporal relationships. We define the notion of interactions, in order to performing learning tasks on activities.
2. A novel generative process for learning a compact representation of activities from complex video is formulated in this thesis. This involves learning the most likely configurations of the generative process which are assumed to have generated the activities.
3. Learning is posed as MAP estimation of the latent configuration of this generative process. An MCMC procedure for sampling and evaluating solutions fits well with the proposed generative process, and is described in detail.
4. Finally, functional object classes are introduced. A simple procedure for learning these classes from the learned event classes is another contribution. The functional

relationships between the functional object classes and the event classes represents the culmination of this work, as this relationship forms an essential aspect of activity understanding.

## 1.5 Thesis Overview

The rest of the thesis is organised as follows. Chapter 2 reviews the research context for work on activity understanding. This chapter starts by describing aspects that make the problem of unsupervised event analysis a complex task. Then area of event analysis is reviewed under various approaches such as pattern recognition, state space models, grammars and logic. The summary of each relevant technique is followed by a discussion relating to the complexity of the task. This chapter includes closely related work in graph based learning as this work focusses on representation and learning with graphs. This chapter also includes a literature survey on object classes. Finally this chapter describes the spatio-temporal relationships that are closely related to this work.

Chapter 3 focusses on representing and comparing interactions. The key idea of interactions are first introduced in this chapter. This is followed by a relational representation of interactions using a graph based structure called the interaction graph. In order to compare interactions, a similarity measure between these interaction graphs are defined. The idea of representing the interactions in an entire activity using an activity graph is also discussed in this chapter. A generative process for inducing interaction graphs from video data is then described. Finally, the notion of embedding a spatio-temporal description as tracks in space and time that embody these relationships is detailed.

Chapter 4 focusses on learning of event classes from activities in an unsupervised way. The key idea of a generative process for activities where event graphs are sampled from an event class distribution is described in this chapter. The problem of unsupervised learning is formulated as MAP estimation of the latent configuration of a generative process. An efficient procedure that uses MCMC to address this problem is described.

Chapter 5 details the proposed technique of learning object classes once the event classes have been learned. This chapter concludes with the description of the technique for learning functional relationships.

Chapter 6 describes the evaluation of the proposed framework in an aircraft domain. This chapter starts by evaluating the efficacy of the HMM based framework for obtaining stable qualitative relationships from video. Then the frameworks for learning event classes from video are evaluated both qualitatively and quantitatively. Finally the evaluation of the functional object classes and relationships are described.

The last chapter 6 offers describes the main contributions and novelties of this work. This is followed by limitations of the existing framework and insights into future directions. This thesis concludes with some general remarks about the work.

# Chapter 2

## Related Work

---

Event analysis is a research topic at the confluence of other areas such as computer vision, knowledge representation and machine learning. The last decade has seen considerable progress in this field for two reasons. Firstly, there has been an increasing demand for tasks such as recognizing events, retrieval of clips with similar semantic content and discovery of patterns for domains such as surveillance, television production and robotic vision. Secondly, there has been considerable progress in research in computer vision, knowledge representation and machine learning that can now effectively address these tasks.

It is apparent that this progress has been more dominant along certain directions of research than others. For example, with regard to learning, there has been considerable focus on the supervised learning setting for the task of event recognition. The development of well established theoretical frameworks and efficient algorithms in machine learning such as discriminative methods (e.g. support vector machines, boosting) and generative probabilistic methods (e.g. dynamic Bayesian networks, latent dirichlet analysis) and grammatical methods have been accompanied by a swift application to event analysis.

In contrast to the above, there has been relatively little work on the unsupervised learning setting. Moreover, there has been little work on the application of relational learning and qualitative spatio-temporal relationships to event analysis, despite the growing body of work in these areas.

This thesis focusses on learning events in an unsupervised setting. Here, qualitative



spatio-temporal relationships are regarded as being fundamental for characterizing activities. Besides, relational learning, especially in a graph based setting, is proposed as a natural formalism for learning from this representation.

The above considerations point towards a natural way of organizing this chapter. A review of the main trends in related research for event analysis such as pattern recognition methods (Section 2.2), state space models (Section 2.3), grammars (Section 2.4) and logic (Section 2.5) are first discussed. For each of these, supervised approaches are first described briefly followed by a more detailed treatment of unsupervised approaches, since unsupervised learning is the focus of this thesis. The literature on graph mining (Section 2.6) is reviewed as our approach applies and builds upon techniques developed in this research area. The survey of event analysis in this chapter starts with Section 2.1, which clarifies and defines some key words used to characterize each thread of related work.

The review of event analysis and graph mining is followed by a review of research on learning object categories in Section 2.7, which form a secondary but important aspect of the framework proposed in the thesis. This is followed by a review of qualitative spatio-temporal relationships in Section 2.8, as they are central for characterizing events in this work. However, since this work focusses on applying qualitative spatio-temporal relationships, a brief review of relevant techniques are discussed in this section.

Section 2.9 characterizes our approach using certain key terms that are introduced in the beginning of this chapter. These terms are used throughout the chapter to describe other related work.

Section 2.10 describes the relationship with other existing approaches in the literature. The applicability of existing approaches to the unsupervised activity understanding task addressed in this work is also discussed.

This chapter is concluded by Section 2.11.

## 2.1 Three Aspects of Event Analysis

This section defines key concepts and terms used in event analysis. These concepts and terms are necessary systematically describing the related work and contextualizing the proposed framework in relation to it. A summary of each related work is followed by a discussion that uses the key words described in this section. A secondary motivation for this section is to clarify these, given their highly ambiguous usage in the literature.

Some of these concepts have been clarified in previous surveys [Lavee et al., 2009, Xie et al., 2008] from which this chapter borrows ideas. However, a more detailed classification has been found necessary in order to present the related work as a context to ours.

These concepts are organized under three broad categories: (i) domain and task; (ii) learning; (iii) nature and complexity of videos. These terms are first described below, without explicit references to work that are characterized by them. After this description, they are used to classify different parts of related work described in the rest of the chapter.

### 2.1.1 Domain and Task

The increasing need to analyse videos automatically for certain *domains* has played a significant role in determining the direction of research in event analysis. One example is surveillance, where there has been an increasing need to semi-automate CCTV monitoring for domains such as airports, banks, and train stations. Another example is robotic vision where robots that can visually understand and participate in domains such as workshops, kitchens and elderly care homes can be potentially very useful. Other applications such as the analysis of television broadcasts, particularly sports and news have stimulated much research in activity analysis. Even though the domains are many, much of event analysis can be summarized by a handful of tasks of which event detection, retrieval and discovery are particularly prevalent. The goal of event detection is to detect events in a video stream. Event retrieval corresponds to retrieving videos that contain similar events given a query event from a video database. Unsupervised event discovery is the task of analysing videos and discovering events and event classes in them. This work focusses on this third problem of event discovery from videos.

### 2.1.2 Learning

Learning from data is a powerful paradigm that has permeated many fields including event analysis. The success of this paradigm for complex entities such as events is also because many recent approaches allow the combination of expert domain knowledge together with mechanisms for learning from data. The following are three key aspects of these machine learning based approaches [Lavee et al., 2009].

**Learning Setting.** The learning setting is used to refer to the nature and degree to which expert domain knowledge is encoded. While supervised approaches generally encode semantic knowledge about the type of events for a domain, unsupervised approaches tend to assume minimal knowledge about the type of events. However these two crisp categories are only useful for characterizing the two ends of a spectrum. The following paragraphs endeavour to characterize this spectrum in terms of sharper and well defined distinctions,

for the sake of bringing in greater precision in summarizing event analysis approaches, in terms of their learning settings.

- Supervised Settings

- At one end of the spectrum of learning setting is the case of a *manually defined setting*, where the events are described as hand crafted rules. A degree of confidence may be associated for each rule. In such a case, events are detected either by logical abduction or probabilistic inference.
- In the *standard supervised setting*, events are manually segmented and labelled. A model of the events is usually learned using supervised learning frameworks in machine learning. The events may be segmented into separate clips or the precise information (e.g. spatio-temporal extents together with participating objects) may be well specified for straightforward segmentation.
- We refer to a *deictic supervision setting* as the application of *multiple instance learning setting* [Dietterich et al., 1997] to event analysis, where only the spatial and temporal extents (called deictic intervals) of events are specified along with their respective class labels.

- Unsupervised Settings

- In the *clustering setting*, it is assumed that the events are segmented but unlabelled. In this setting, features may be extracted for each event and clustered using techniques in machine learning.
- We refer to a *transactional learning setting* when the deictic spatio-temporal regions are given but not labelled. This word is borrowed from the body of research in data mining [Cook and Holder, 2007].
- We refer to a *unsupervised discovery setting* when, neither any form of segmentation of the activities into events are given nor are the class labels for any part of the video are assumed. The primitive events are also assumed as not given.

**Features.** In the field of event analysis, features are used to refer to descriptions of salient aspects in a video. Features have been categorized in the survey paper of [Xie et al., 2008] as *low level*, *mid level* and *high level*. Low level features are usually properties of pixels in images such as colour, texture and shape or their properties in image sequences such as quantized location or direction of motion. Low level features have been found to

be particularly well suited for modelling the behaviour of groups in crowded scenes, especially because object detection and tracking are typically too hard for such scenes. Mid level features capture perceptual intuitions as well as semantically meaningful aspects of images such as tracked objects, segmented body parts etc. High level features aggregate low level or mid level features into high level data structures such as sequences, bags and relational entities such as logical formulae and graphs.

**Event Models.** Event models are a description of events in some formalism. These have been systematically categorized and discussed in the survey paper [Lavee et al., 2009] as follows: (i) pattern recognition techniques; (ii) state space models; (iii) grammars; (iv) logic. Both logic and graphs can be used to model relational data. This work introduces a framework in which graphs are used for relational representation and modelling of events.

### 2.1.3 Characterizing Complexity

A study of the factors that influence the complexity of the task of event analysis provides a good insight into the nature of different approaches that have been employed to this end. Moreover, these factors provide a paradigm for conceptually comparing different approaches that have been developed in event analysis.

**Nature of Video Input.** This chapter adopts the following categorization of video input from a previous survey [Xie et al., 2008]: (i) single stream from one take; (ii) multiple cameras, single take; (iii) single stream from multiple takes; (iv) media collectives.

**Complexity of Events.** In general, complexity of events is relative and may be characterized in several different ways, depending on the problem at hand. However, the following are some high level features for characterizing complexity. More importantly, as will be seen in the rest of the chapter, these features provide useful concepts for comparing different approaches in event analysis.

- *Atomic* (e.g. a hand scooping some butter) or *composite* (making breakfast). It is important to note that these definitions are not absolute and that the consideration of whether an event is atomic or composite depends on the level of abstraction that is chosen.
- *Single temporal process* (e.g. a hand scooping some butter) vs *multiple parallel processes* (e.g. a hand scooping some butter, while the other hand takes a slice of

bread). There are two aspects of multiple temporal processes which influence the complexity of events.

- *Independent processes* (e.g. two cooks making separate dishes without any dependencies) or *dependent processes* (e.g. two cooks cooking the same dish in a way that their actions depend on each other).
- *Sharing* of processes or objects (e.g. sharing a spoon while cooking two different dishes).
- *Structural complexity* characterizes the degree of complexity in the structure of events. One aspect of structural complexity is aptly characterized by a hierarchy of grammatical types in Chomsky's hierarchy such as the simpler regular grammar to the more complex context free grammars, context sensitive grammars and recursively enumerable grammars. These grammars correspond to recognition machines such as finite state automata, recursive finite state automata etc. While structurally simpler events are modelled by simpler models such as finite state automata, complex events which for example may have nested structure (e.g. *aaabbb*) require structurally more complex recognition machines.

Another aspect of structural complexity is temporal complexity. While simpler events may be modelled by assuming a first order Markov process, temporal complex events tend to have complex temporal dependencies that may require higher order Markov processes or Allen's temporal relations [Allen, 1983].

- *Semantic complexity* characterizes the degree of expressiveness required to describe events. For example while events with propositional descriptions (e.g. *abcd*) are simpler to model, events that are best described in a relational form (e.g.  $R_1(a)R_2(b, c)R_3(c, d)R_4(a)$ ) are more complex to model.
- *Spatially scene dependent* events (e.g. cars turning left at a particular junction in a particular scene) are easier to characterize as they are learned from a single scene. On the other hand, *spatially scene independent* events (e.g. cars turning left at any junction in any scene) are more complex to learn as they require generalization across different scenes.
- *Degree of variations* in the event classes. Event classes whose event instances have identical structure are easier to mine using standard techniques in data mining. On the other hand, event classes which have variations are harder to discover and model, especially in the unsupervised setting.

**Complexity of Learning Events from Activities.** The complexity of activities is related to the ease with which events can be detected or discovered from them. Three key aspects that characterize complexity of learning events from activities are detailed below.

- *Noise and co-incidental occurrences* make the problem of learning events from activities more complex. This is because the event patterns may become veiled in the presence of noise and coincidence, necessitating mechanisms to focus attention on event like structures, or ways of modelling noise and coincidences.
- *Sharing of objects and interactions across events* make the problem of learning more complex. These two factors make the problem of segmentation of activities into events less straightforward.
- *The learning setting* has a direct relationship to the complexity of learning. In the supervised setting both the segmentation and labelling of activities into events are assumed to be given. In the clustering setting, the events are segmented and given, but the labelling has to be inferred. In the deictic supervision setting, the learning task can be slightly more complex if the deictic regions contain noise or events from other classes. The unsupervised event discovery setting is generally more complex than the above settings, because event classes have to be learned while simultaneously searching the space of possible segmentations.

## 2.2 Pattern Recognition Techniques

Pattern recognition techniques represent events in a space with a suitably defined similarity metric. In this thesis, both classical supervised approaches such as event classification and unsupervised approaches such as clustering are generally applied.

### 2.2.1 Supervised Learning Approaches

Supervised approaches belonging to pattern recognition techniques are mostly used to recognize events that correspond to types of motion. These have been used in conjunction with low level and mid level features. Events such as walk, run and skip in [Cao et al., 2004] and events such as putting on sunglasses [Smith et al., 2005] are recognized using low level pixel based features with K-NN [Dasarathy, 1990], SVMs [Burges, 1998] and Boosting [Schapire, 1990] respectively. Events such as human walking and dog running in [Goldenberg et al., 2002] and person crossing the corridor in [Pittore et al., 1999] are recognized using mid level object based features with K-NN and SVMs respectively.

## 2.2.2 Unsupervised Learning Approaches

Unsupervised learning using pattern recognition approaches has been applied to event analysis to recognize different types of events and in conjunction with features at different levels of representation. A summary of these approaches are described below.

**Clustering Sequences of States.** In many domains *events* take the form of a certain sequence of states and the segmentation of events are known or relatively easy to extract. For example, in traffic scenes, *trajectories* of interest such as a car turning or a pedestrian crossing are events and their segmentation is known in advance. *Event classes* are regarded as clusters of similar state sequences.

Similarity between trajectories is often expressed as a combination of properties of both appearance and motion (behaviour). For example authors in [Wang et al., 2006] use these two aspects to infer the structure of the scene, as it is hypothesized that the scene structure (e.g. the location of the pedestrian crossing ) affects the behaviour of moving objects in the scene. Trajectories that represent candidate events are clustered in two stages. In the first stage, an appearance attribute, specifically the size, is used to separate vehicles and pedestrians in a road scene. The behavioural attributes, which are the spatial location and motion based features, are used to further separate these two clusters into semantic regions such as a one way road, u-turn exits etc. Thus a scene is automatically interpreted in terms of event classes, whose event instances correspond to trajectories in spatial regions, within which objects exhibit semantically similar appearance and behaviour, as illustrated in Fig. 2.1.

Several features and distance measures have been used to cluster trajectories. The work described above [Wang et al., 2006] uses spatial coordinates, object size and velocity with a modified Hausdorff distance. The authors in [Buzan et al., 2004] use spatial coordinates with the longest common subsequence (LCSS) [Vlachos et al., 2008] distance, while the authors in [Hervieu et al., 2008] use translation, rotation and scale invariant features with a HMM based distance measure to cluster trajectories.

Authors in [Oh and Hoogs, 2010] characterize behaviour of trajectories by using location independent features which capture the relationship of a trajectory to scene structures such as buildings, parking spots, roads etc. These features are used to produce location independent event classes such as vehicles passing through, people walking on side walks, people crossing the road etc. However, this technique relies on the knowledge of a particular scene and is in this sense scene dependent.

Many domains such as indoor (e.g. a kitchen) or cargo unloading involve *events* which may be expressed as a sequence of location based states in which an agent moves

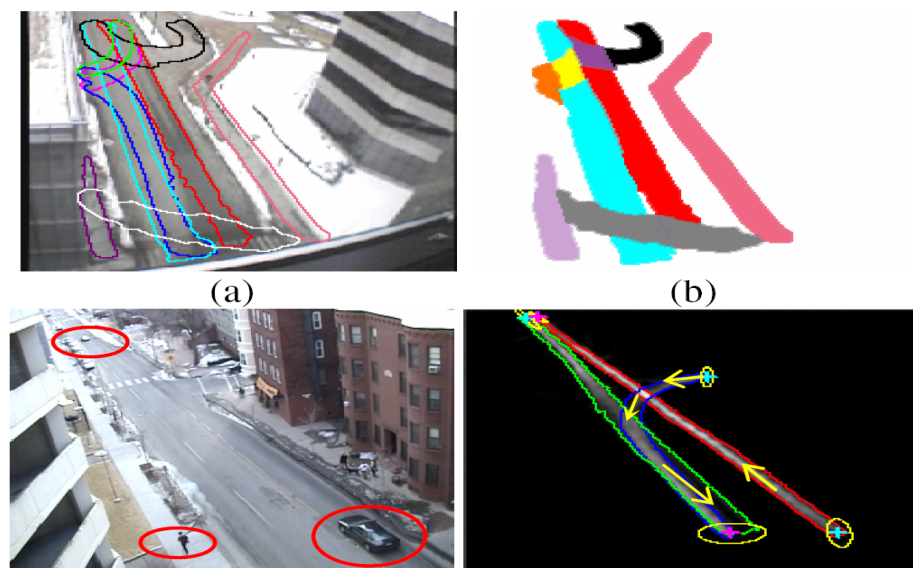


Figure 2.1: Clustering of events, each of which is a trajectory, in order to obtain event classes. Figure from [Wang et al., 2006].

around. For examples two events may be  $e_1 : (shelf, stove, washer, sink, washer)$ ,  $e_2 : (fridge, stove, washer, sink, table)$ . Events that are similar tend to share similar sub-sequences (e.g. *stove, washer, sink* for the above events  $e_1$  and  $e_2$ ) and therefore may be grouped into *event classes*.

This approach is undertaken in [Hamid et al., 2009]. In this work, events are initially represented as sequences of states, which are manually described using a predefined event vocabulary e.g. enter the kitchen, turn stove on, get eggs, as illustrated in Fig. 2.2. Then they are re-represented in terms of statistics of their local event subsequence, so that events that share similar statistics tend to be similar, and may hence belong to the same event class. The locations are not metrically represented ( $x, y$  coordinates), but qualitatively interesting locations are symbolically represented (fridge, table). Therefore, these events, in principle, are spatially scene independent. This aspect of the work highlights the advantage of qualitative representation.

**Discussion.** The approaches described above are a case of the *clustering setting*, since the segmentation of activities into events (which are just trajectories or sequences of states) is known a priori. Also, these events are often *scene dependent* as they are spatially related to a particular scene under consideration. The events are *atomic*, except in [Hamid et al., 2009], in contrast to compositional events such as *a person crossing the road after the car stops for him/her*, which are explored in [Wang et al., 2009]. Events are modelled as a *single temporal process*. It is worthwhile noting that the sequences are propositional



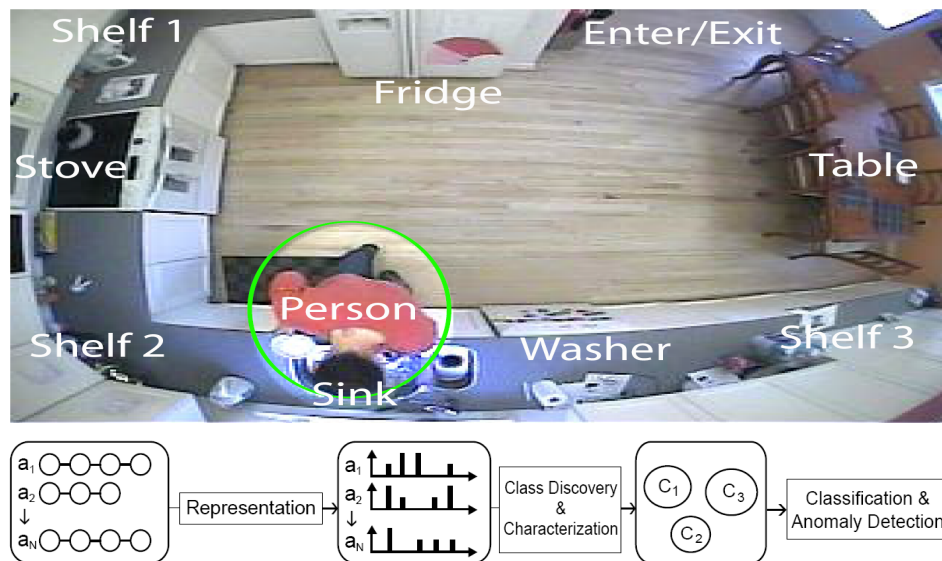


Figure 2.2: Clustering of events which are a sequence of qualitatively interesting states. Figure from [Hamid et al., 2009]

in nature and are therefore *semantically simple*.

**Mining Sequences of States.** When events take the form of state sequences, but their segmentation is unknown, frequent sequence mining may be applied to obtain frequent patterns that may correspond to events. In [Toshev et al., 2006], interactions between vehicles and predefined zones on a road gives rise to a propositional sequence of spatial states similar to work described above by authors in [Hamid et al., 2009]. However, unlike [Hamid et al., 2009], the event segments are assumed unknown and therefore a sequence mining approach is used to discover classes of frequent sequences, which are regarded as event classes.

The authors in [Wang et al., 2005b] transform a video into a sequence of symbols which are obtained by clustering low level video features. Repetitive sub-sequences are mined from this sequence to obtain event classes that capture similar high level concepts such as a “diver preparing to dive on a springboard” in broadcast videos.

**Discussion.** The approaches described above are cases of the *unsupervised discovery setting*, since the segmentation of activities into events (which are just trajectories or sequence of states) are not known a priori. The events considered are *composite* in nature. Events are modelled as a *single temporal process*. Also, since the sequences are propositional, they are *semantically simple*.

## 2.3 State Space Methods

State space models explicitly model the structure of the state space in an event domain [Lavee et al., 2009]. The following paragraphs discuss different types of state space models that have been used for event modelling. This is followed by more detailed descriptions of unsupervised learning approaches that use state space models.

### 2.3.1 Supervised Approaches

State-space models specify a joint probability distribution over state-space i.e. over observed and unobserved random variables that characterize the states of a system. Bayesian networks [Heckerman, 1999] are a way to make state space models tractable by factorizing the joint probability distribution into a simpler distribution by making it possible to model conditional independences between these variables. Bayesian networks are amongst the most popular directed state space models. They represent a set of random variables and their conditional independences via a directed acyclic graph (DAG). Bayesian networks have been used in different ways for modelling events. One classical way of using Bayesian networks is discussed in [Hongeng and Nevatia, 2001], where the *compositional* relationships between events, their sub-events and other observed features are represented at multiple levels of granularity. The higher levels usually correspond to composite events (a person approaching another), the mid level to *atomic events* (move towards, slow down) and the lower levels to observable features (distance to reference object, direction, bounding box), as illustrated in Fig. 2.3.

Using Bayesian networks, the probability of a composite event is inferred by propagating the probabilities upwards from the observation level to the atomic sub-event level and finally to the level of the composite event at the top of this. In general, Bayesian networks have been applied mostly to object based abstractions to recognize events such as *overtaking*, *following* as reported in [Buxton and Gong, 1995], [Lv et al., 2006]. While these Bayesian networks are well suited for modelling: (i) the hierarchical and semantic space of events; (ii) their ability to model uncertainty; (iii) their computational tractability, they are not ideally suited for temporal modelling.

Hidden Markov Models (HMM) [Rabiner, 1989] are a class of graphical models that are well suited to model the temporal aspect of events. A HMM is a statistical model in which the observations and states are assumed to be modelled by a Markov Process with states that are not observed. These hidden states emit a sequence of observations.

There are many variations of the classical HMM model, capturing different kinds of dependencies between random variables. These have been used to model *multiple par-*

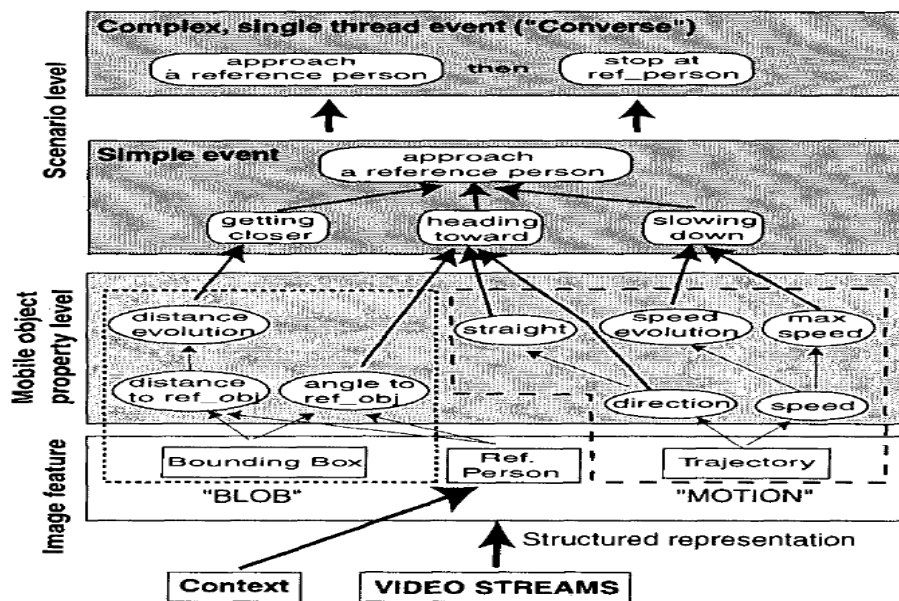


Figure 2.3: A Bayesian network for complex events. Detection is inference over the Bayesian network. Figure from [Hongeng and Nevatia, 2001]

*allel temporal processes* and varying degrees of *temporal dependencies* between these processes. The simplest of these are the Parallel HMMs (PaHMM) [Vogler and Metaxas, 2001] where the multiple temporal processes are modelled but are assumed to be *temporally independent* of each other. They have been used for sign language recognition [Vogler and Metaxas, 2001], where the signs from each hand represent a single temporal process. These two processes are assumed to be independent of each other based on the cited linguistic evidence that the two temporal process of the American Sign Language can be viewed as acting with a high degree of independence on the phoneme level.

In contrast to PaHMMs, Coupled HMMs (CHMM) [Oliver et al., 2000] are used to recognize events such as interactions between persons (e.g. follow, reach and walk together), where it is essential to model the *temporal dependencies* between the multiple temporal processes.

Hidden Semi-Markov Models (HSMM) have been used to model *duration* for recognizing events such as a car passing through a check point [Hongeng and Nevatia, 2003]. Another variation of the standard HMM is the Hierarchical Hidden Markov Model (HHMM) which has been used in [125] to model the natural hierarchy present in composite events. These are used to recognize events such as short meals, have snacks. Several hybrids of the above HMM variations have been used for various tasks. However greater expressiveness is often found to come at the cost of tractability and achieving this trade

off is a matter of design expertise.

### 2.3.2 Unsupervised Approaches

While much work with state space models focuses on a supervised learning setting, there has been some notable work for the case of unsupervised setting. The following paragraphs describe this work in detail.

**Hierarchical Hidden Markov Models.** In structured domains such as sports, hierarchically nested repetitive segments may occur. A HHMM is used in [Xie et al., 2003] to model recurring events in each in terms of simpler HMMs, and the higher-level transitions between these events as another level of a Markov chain. They are used to automatically discover high-level structures, for example, plays and breaks in soccer and baseball. Some parameters such as state connectivity, number of levels of the Markov chain and the time scale of the states are manually specified. The search for the optimum model is performed using Markov Chain Monte Carlo techniques with Bayesian Information Criteria (BIC) as the model posterior.

**Dynamic Bayesian Networks.** Domains such as aircraft aprons and shops, often consist of *composite events* which are composed of multiple parallel processes of atomic events. In these domains, it is often the case that some of these temporal processes are dependent while some others are independent of each other. Dynamic Bayesian Networks (DBNs) are a class of all Bayesian networks that have random variables associated with a time series. While classical HMMs, PaHMMs and CHMMs are all examples of DBNs, the generic DBN framework can model *multiple parallel processes* with *partial temporal dependencies* and thus strike a middle ground between PaHMMs and CHMMs, which are on either extremes in terms of modelling temporal dependencies.

In [Xiang and Gong, 2006], an event class is regarded as generating a sequence of atomic events, each of which is a group of significant pixel changes (over time) in a local image neighbourhood. Event classes are obtained by clustering events, which are represented in terms of certain properties of the respective group of pixels, such as their position, shape, visual change type and motion features. The temporal relations between events are modelled using a Dynamically Multi-Linked Hidden Markov Model (DML-HMM), which is a kind of DBN, based on the discovery of salient dynamic interlinks among multiple temporal processes corresponding to the events generated by multiple event classes. This technique is unsupervised because, first of all, the event classes are found by clustering, as illustrated in Fig. 2.4. The number of clusters is automatically

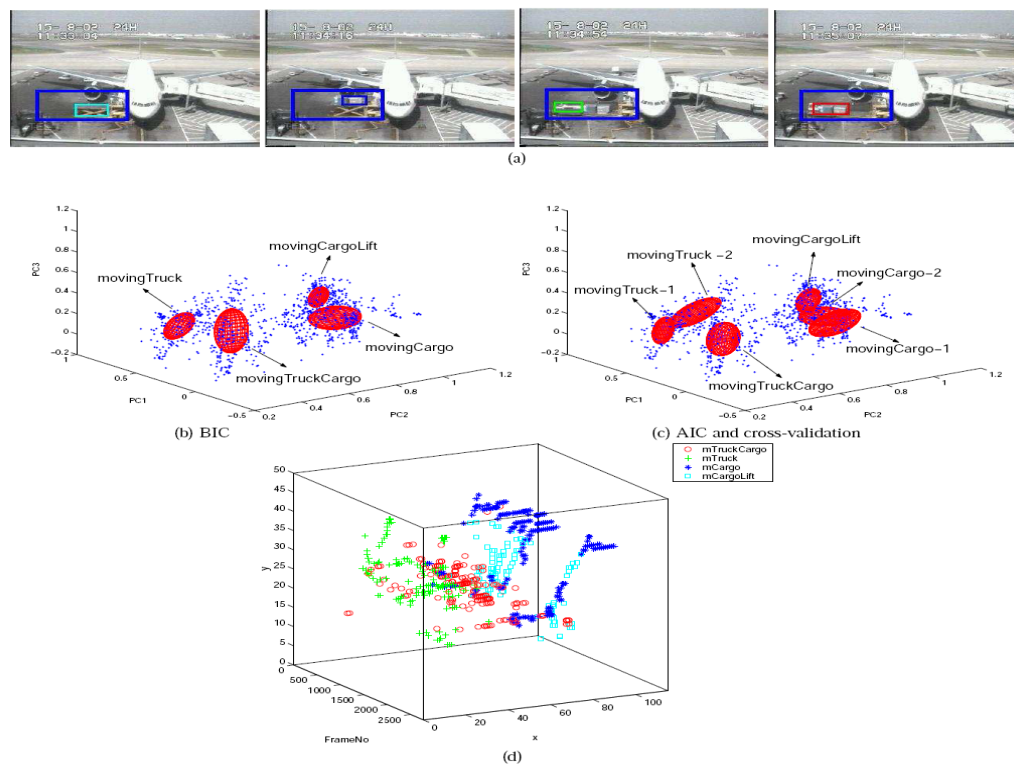


Figure 2.4: Clustering of events into event classes. Figure from [Xiang and Gong, 2006].

determined using Bayesian Information Criterion. Moreover, the DML-HMM is built using BIC based factorization resulting in its topology being intrinsically determined by the underlying causality and temporal order among events, as shown in Fig. 2.5. Experiments on a shop and aircraft scene confirm that the DML-HMM is superior compared to other Dynamic Probabilistic Networks such as Multi-Observation Hidden Markov Model (MOHMM), a Parallel Hidden Markov Model (PaHMM) and a Coupled Hidden Markov Model (CHMM) for the task of unsupervised learning.

**Discussion.** The approaches described above are a case of the *Unsupervised discovery setting*, since the segmentation of activities into events is not known in advance. The events in the latter work [Xiang and Gong, 2006] are *scene dependent* as they are spatially related to a particular scene under consideration. The events in [Xie et al., 2003] and [Xiang and Gong, 2006] are *composite*. The HHMM can model nested structures and are therefore not *structurally simple*, though the events in [Xie et al., 2003] are generally simpler. The events modelled in [Xiang and Gong, 2006] are *multiple temporal processes*, though the temporal dependencies that they model are essentially first order Markov. It is worthwhile noting that the sequences are propositional in nature and are

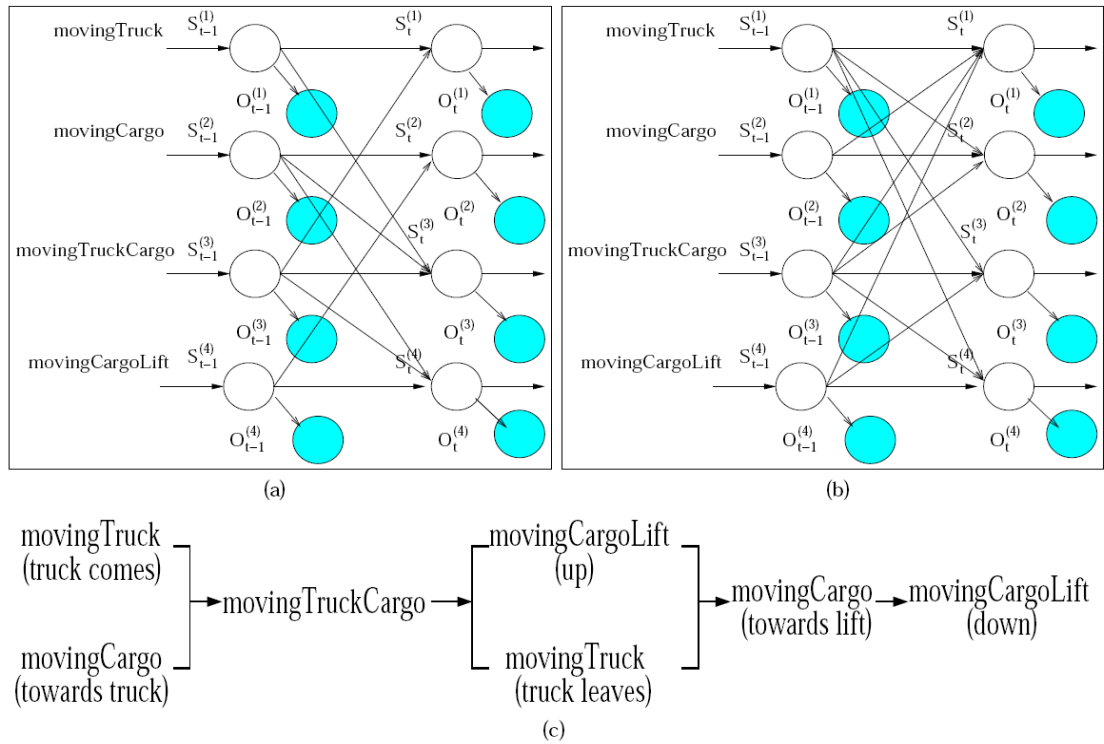


Figure 2.5: Dynamic Bayesian networks for modelling multiple parallel processes. Figure from [Xiang and Gong, 2006].

therefore *semantically simple*.

**Topic Models.** Topic models learn discrete latent variables from co-occurrence data or features by exploiting the conjugacy of Dirichlet and multinomial distributions [Blei et al., 2003]. In domains such as crowded traffic scenes, where groups of objects exhibit patterns of motions, topic models are well suited to *composite events* which are composed of *atomic events* such as a *car stopping* followed by *person crossing the road*. Topic models which are a class of Bayesian networks have been gaining popularity in computer vision after their success in document analysis research. In the extension to event analysis, the entity that is analogous to a text document is a video clip for a domain such as traffic in [Wang et al., 2009]. A text document is regarded as containing a mixture of several topics. Analogously a video clip (document) is regarded as containing several atomic events (topics) such as *car stopping* and *pedestrian crossing*. Thus a video clip contains a *composite event*. Just as a set of textual words are the building blocks of topics, a *video word*, which is the quantized positional and directional features for each pixel in the video clip, constitute atomic events. This idea is illustrated in Fig. 2.6.

The video clip is assumed to be generated by first assigning for each pixel an atomic

event (topic) and then assigning a visual word given this event in the following four steps. First, given a video clip, a distribution over atomic events is picked (from a distribution over distributions). For example, according to this distribution, the clips may be more likely to contain atomic events such as *car stopping* and *pedestrian crossing*. Second, given this distribution, one topic is sampled from this distribution for any particular pixel (word position in a document) in the video clip. Thirdly, for that particular pixel and for topic chosen for this pixel, a distribution over possible video words is picked (from a distribution over distributions). Finally, a visual word is sampled from this distribution over video words for those pixels. Note that the visual words are considered to be independent given the topics according to a standard bag of words model assumption.

In [Wang et al., 2009], this generative process is formulated as variants of Latent Dirichlet Allocations (LDA) and Hierarchical Dirichlet Processes (HDP). Given just video sampled into non-overlapping clips (analogously the set of documents), the atomic activities (topics as co-occurring words) and their interactions (co-occurring topics) are discovered in an unsupervised manner.

**Discussion.** The approaches described above are cases of the *unsupervised discovery setting*, since neither the segmentation of activities into events, nor the event models that generated these events are known in advance. The events are *scene dependent* as they are spatially related to a particular scene under consideration. These approaches model *composite events* in terms of co-occurring *atomic events*. While *multiple temporal process* are modelled using this technique, the bag of words assumption implies that the spatial and temporal relations between these process are not modelled. It is worthwhile noting that the sequences are propositional in nature and are therefore *semantically simple*.

## 2.4 Grammars

A grammar is a set of rules of a specific kind, that can be used to form strings in a formal language. The rules describe valid ways of forming strings according to the language's syntax. The following paragraphs describe related work on supervised approaches with grammars for event analysis. This is followed by a description of unsupervised approaches.

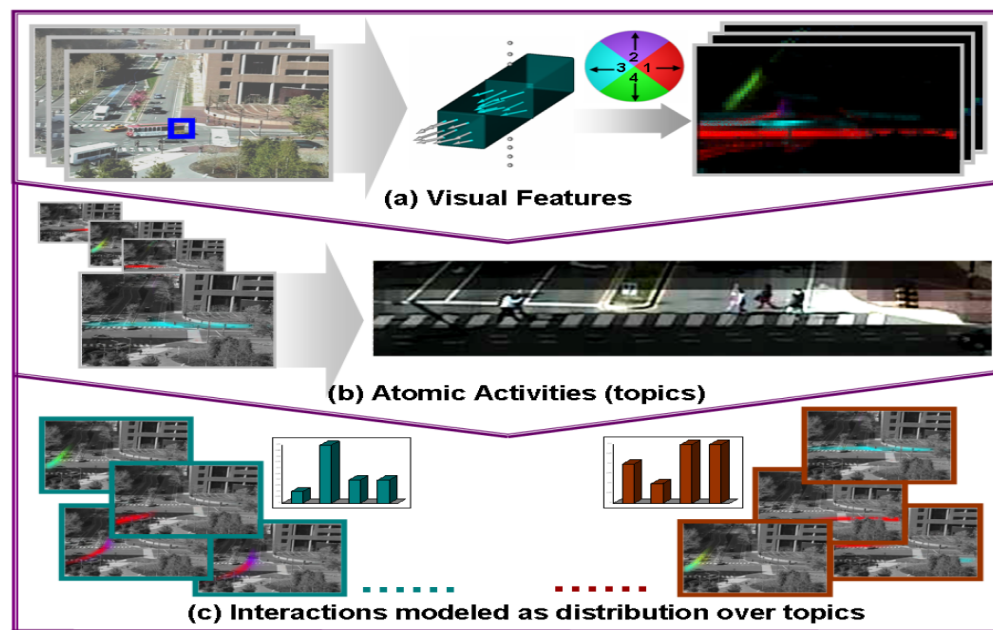


Figure 2.6: Topic models for discovering events in a scene. At the topmost level, the quantized positional and directional features for each pixel in the video clip constitute atomic events (e.g. car stopping) in the middle level. The atomic events constitute the composite events (e.g. car stopping and pedestrian crossing) in the bottom level. Figure from [Wang et al., 2009].

### 2.4.1 Supervised Approaches

In domains such as parking lots and train stations, events such as *drop off* and *pick up* tend to take the form of nested structures with long term pairwise correlations. Stochastic context free grammars [Stolcke, 1995] are aptly suited for modelling such events as they generate nested structure in an outside-in fashion rather than from left to right fashion of regular grammar based models such as the HMMs.

Context Free Grammars are expressed in terms of the following three components in event analysis. The first is a set of terminals (e.g. *person\_appear*, *person\_dissappear*, *car\_stop*, *car\_start* [Joo and Chellappa, 2006]) that correspond to *atomic events* that are detected using low level features. Second, a set of non-terminals (e.g. *CAR\_START*) which usually correspond to *composite events*. The set of production rules generate composite events using atomic events. Stochastic context free grammars allows probabilities to be associated with each production rule.

The key property of context free grammars are that a production rule can generate a correlated pair of terminal/non-terminal symbols, then another correlated pair inside that.



Rules such as the one given below can generate nested structures in an outside-in fashion.

$$\begin{aligned} DROP\_OFF &\rightarrow CAR\_STOP \textit{person\_appear} \textit{person\_disappear} CAR\_START \\ CAR\_START &\rightarrow \textit{car\_stop} \textit{car\_start} CAR\_START \end{aligned}$$

In the language of formal grammars, event recognition corresponds to determining whether a sequence of terminals i.e. primitive events (extracted from a video) constitutes an event. The technique used for determining this is called parsing which returns a parse tree – that shows how to generate the sequence via the application of production rules, or shows the composition of the sequence in terms of the production rules.

Context free grammars are used for interpreting a video of object manipulations [Brand, 1996] into coherent events such as *add*, *remove*, *move* objects using qualitative spatial primitives such as *attach*, *detach*, *motion* etc. Two-person interactions are modelled using simple context free grammars in [Ryoo and Aggarwal, 2006].

Grammars have been extended with probabilities (SCFG) [Moore and Essa, 2002] and used in conjunction with mid level features for detecting events such as parking events. An example of a rule with an associated probability (.7) is given below.

$$CAR\_START \rightarrow \textit{car\_stop} \textit{car\_start} CAR\_START .7$$

Attribute grammars [Joo and Chellappa, 2006] have been used in conjunction with mid level features for recognizing events such as parking, drop off and pick up. Attribute grammars formally associate attributes with each symbol (terminals and non-terminals) and rules for assigning values. The application of rules can also be restricted by constraints on attributes.

## 2.4.2 Unsupervised Approaches

In comparison to supervised approaches, there has been relatively much less work on unsupervised learning of grammars for activities. The following are two approaches that have been found in the literature.

**Learning Grammar using Compression.** Human gestures such as those that occur in musical conducting are instances of human activities that are regarded in [Wang et al., 2001] as consisting of gesture primitives corresponding to atomic events, with a high level structure controlling the temporal ordering of these events. Thus a grammar based model is adopted to model gestures. The atomic events which are the terminal symbols of the

grammar are obtained by over segmenting a original long gesture into short segments and then clustering them to obtain clusters of very similar and basic motion patterns. Over segmentation involves searching for natural inconsistent points that are usually accompanied by dips in velocity or abrupt changes in direction of movement. Clustering involves training a HMM on each segment and then grouping these segments on the basis of a distance metric between their respective HMMs (HMMs provide a distance metric for sequence comparison).

Thus each segment is an atomic event and is represented as a non-terminal symbol that corresponds to its cluster number. By replacing the segments with these symbols in the original long gesture, the gesture is expressed as a sequence of symbols or alphabets. A heuristic approach called COMPRESSIVE [Witten, 2000] is adopted to approximate a MDL (Minimum Description Length [Grünwald, 2005]) solution by iteratively substituting the current most compressive subsequence of the original sequence into a non-terminal and thus learns a grammar. The compressive ratio of a subsequence is expressed as a trade-off between its frequency in the original sequence and its length. The approach is applied to musical conducting and results in grammars for 2, 4 and 6 beat patterns.

Many complex activities may be represented by symbolic sequences whose underlying nested structures can be modelled by grammars as described above. However, when these sequences are extracted from real video data they often tend to be corrupted by interleaved noisy symbols arising from two sources which are considered in [Kitani et al., 2008]. First, due to errors<sup>1</sup> in observation. The other source is coincidental occurrences (that do not play any important purpose in the activities).

For such corrupted sequences, a grammar that explains the nested patterns of events can be learned (in principle) only when the noise is identified. Equivalently, once the non-noise symbols is identified, the grammar that describes the sequence can be recovered. This intuition is the basis for the approach in [Kitani et al., 2008]. The approach involves generating hypothetical context free grammars corresponding to a candidate set of non-noise symbols using the COMPRESSIVE algorithm [Witten, 2000]. This is converted into a SCFG by adding the production probabilities for each learned rule. The candidate SCFG that produces the minimum description length of the sequence of hypothetical non-noise symbols is obtained by searching in the space of candidate SCFGs and selecting the one with the lowest value of the description length. The description length of a candidate SCFG is expressed as a weighted sum of the description length of the grammar (considered by itself) and the description length of data (sequence of non-noise symbols) when encoded by the candidate SCFG. This approach is evaluated on activities

---

<sup>1</sup>such as insertion, deletion, substitution errors

from a convenience store and is able to learn a parse tree of common structure found in the data despite the presence of noise.

**Discussion.** The approaches described above is a case of the *Unsupervised discovery setting*, since the segmentation of activities into events is not known in advance. These approaches model *composite events* in terms of nested structures of *atomic events*. Thus the events are *structurally complex*. These techniques model *single temporal processes* and offer no mechanisms to separate multiple temporal processes. The event sequences are propositional in nature and are therefore *semantically simple*.

## 2.5 Logical and Relational Models

Many complex events involve relations between entities (*attach(loader, plane)*) and domain constraints that are expressed in terms of existential and universal statements such as *only one loader can attach to a plane during a turnover*. Logical and relational languages models are more expressive than string grammars as they can encode complex propositions, functions and quantification.

Atoms are the basic unit of a logical representation to express relationships between entities such as jeff is a parent of paul (e.g.  $\text{parent}(\text{jeff}, \text{paul})$ ), someone  $X$  is a parent of paul (e.g.  $\text{parent}(X, \text{paul})$ ), or some person  $X$  is a parent of someone else  $Y$  (e.g.  $\text{parent}(X, Y)$ ). Atoms consists of *predicates* (e.g.  $\text{parent}$ ) which relate *terms* that may be either called *constants* which are specific objects (e.g. jeff, paul) or *variables* (e.g.  $X, Y$ ) which stand for an arbitrary object in the universe of discourse.

Clauses are formulas that relate atoms to each other and express relationships such as  $X$  is the grandparent of  $Y$  if  $X$  is a parent of  $Z$  and  $Z$  is a parent of  $Y$ . This is expressed as a clause  $c$  which is  $\text{grandparent}(X, Y) : \neg \text{parent}(X, Z), \text{parent}(Z, Y)$ , where  $\text{grandparent}(X, Y)$  is the head( $c$ ) of this clause, and  $\text{parent}(X, Z), \text{parent}(Z, Y)$  the body( $c$ ). Clauses with an empty body, such as  $\text{parent}(\text{jeff}, \text{paul})$  are called *facts*.

A *substitution*  $\theta$  (e.g.  $\{sfX/\text{anne}\}$ ), is an assignment of term(s) (e.g. anne) to a variable(s)  $X$  in a clause  $c$ , and is denoted by  $c\theta$  which is

$$\text{daughter}(\text{anne}, Y) : \neg \text{female}(\text{anne}), \text{parent}(Y, \text{anne})$$

If  $c$  and  $c'$  are two clauses, then the clause  $c$   $\theta$ -subsumes  $c'$  if there exists a substitution  $\theta$  such that  $c\theta \subseteq c'$ . According to definition the clause  $c$   $\theta$ -subsumes all the following clauses (i)  $\text{daughter}(\text{anne}, \text{mary}) : \neg \text{female}(\text{anne}) \text{parent}(\text{mary}, \text{anne})$  obtained by substi-

tution of constants; (ii)  $\text{daughter}(X, Y) : \neg\text{female}(X)$ , obtained by removing a literal.

### 2.5.1 Manually Defined Setting

In supervised approaches using logical models, a set of logical predicates and their relationships in the form of inference rules are used to manually encode the knowledge about the events in a domain. Event recognition is performed using logical inference techniques.

Authors in [Shet et al., 2005] use a first order logic framework of Prolog in the domain of surveillance to detect events such as violations of security in the entrance of a building. Primitive events such as *a person swiping a card or entering a building* are first detected using low level features from video. These are then translated into observed facts, which a logic programming language (Prolog) uses in conjunction with manually encoded rules for events, to arrive at valid inferences regarding events observed in the video.

$$\text{theft}(P, B, T) = \text{human}(P), \text{package}(B), \text{possess}(P, B, T), \text{not}(\text{belongs}(B, P, T))$$

This framework is extended in later work [Tran and Davis, 2008] to account for inherent uncertainty in video events such as a parking lot where occlusions and missing detections offer additional challenges. A degree of uncertainty is used to extend the three components: (i) atomic events (e.g. *put a bag into a car*); (ii) rules encoding composite events (e.g. *a person enters some car*); (iii) common-sense knowledge (e.g. *people walking together usually enter the same car*). A Markov logic framework is used to combine these three components into one framework and perform probabilistic inference to detect complex events e.g. determine the probability of an event such as *a person enters some car*, given the input video sequences.

In all of the above approaches, events are usually encoded by a domain expert. Inductive Logic Programming (ILP) offers a paradigm for learning the descriptions of events themselves in a supervised setting where these are segmented and labelled, as described below.

### 2.5.2 Supervised Approaches using ILP

Inductive Logic Programming [Muggleton and De Raedt, 1994] is a framework for learning logical definitions of relationships from data. Even though ILP has been extended in the recent years to a whole spectrum of tasks such as regression, clustering, association analysis, the most common task has been to learn logical definitions for the positive class

Training examples		Background knowledge
$daughter(mary, ann).$	$\oplus$	$parent(ann, mary).$ $female(ann).$
$daughter(eve, tom).$	$\oplus$	$parent(ann, tom).$ $female(mary).$
$daughter(tom, ann).$	$\ominus$	$parent(tom, eve).$ $female(eve).$
$daughter(eve, ann).$	$\ominus$	$parent(tom, ian).$

Figure 2.7: Background knowledge of positive examples (+), negative examples (-) and facts.

in a binary classification task. For example, given the background knowledge in Fig. 2.7, the positive class is identified by a target relation such as  $daughter(X, Y)$ , and the goal is to learn a logical definition such as:

$$daughter(X, Y) \leftarrow female(X), parent(Y, X)$$

ILP searches the hypothesis space of candidate target definitions by starting from the most general hypothesis (an empty clause) and then specializes this by exploiting a natural order of clauses based on  $\theta$ -subsumption and adding additional predicates to the clause. If a candidate hypothesis satisfies some positive examples, it is removed, and in this manner proceeds until the hypothesis covers as many positive examples and as few negative examples, as possible. Additionally, an MDL term is used to prefer hypotheses of smaller sizes to larger ones.

The ILP approach constructs a clause satisfying some of the positive examples, adds this clause to the body of the target definition, and removes these positive examples that have been explained [Muggleton and De Raedt, 1994].

In domains such as card games, the events are subject to the protocols of the game. Identical event instances are subject to the same protocol, i.e. they have the same high level description and can therefore be regarded as being in the same event class. Thus, ILP is a suitable framework for learning these event classes i.e. protocols, once the positive examples of events are labelled. In [Needham et al., 2005], the events are not labelled directly by a domain expert but indirectly with audio utterances that accompany positive examples of the events. For instance, the positive examples of events such as placing of two cards with the same colour or shape are accompanied by spoken utterances such as *same colour, same shape* etc. Since these utterances are continuous audio signals, they are discretized and grouped into clusters. The cluster labels for each utterance are used as the class labels for the corresponding events. Once the positive examples for events are labelled, the target definitions for the positive classes, that are intended to correspond to

protocols of the game, are learned using ILP.

The use of ILP to learn descriptions of the events is extended in [Dubba et al., 2010] to learning events for more complex domains such as aircraft turnovers, once deictic supervision has been provided by a domain expert. This approach builds upon a equivalent logical representation of events proposed in [Sridhar et al., 2008] (described in chapter 3), where qualitative spatial relationships are integrated with temporal relationships in a graph based representation. Deictic supervision is used to characterize positive examples for each event class for the videos of an aircraft apron. ILP is used to learn the description of these events that covers as many of the positive examples and as few of the negative examples as possible.

### 2.5.3 Unsupervised Approaches

The following describes two main approaches in ILP for unsupervised learning from a logical database representation and a logical sequence representation respectively. This is followed by describing one single work that has been found to apply relational sequence learning to video event analysis.

**Relational Learning Approaches.** Unsupervised approaches for discovering patterns using a logical representation is studied under the topic of relational data mining (RDM). Many of the approaches here can be thought of as extending work from frequent itemset mining and ILP for the unsupervised scenario. A common theme in RDM is to use Datalog queries [Ceri et al., 1989] to find frequent patterns in a logical database. For example, the query  $? - \text{person}(X), \text{parent}(X, Y), \text{hasPet}(Y, Z)$  finds triples  $(x, y, z)$ , where child  $y$  of person  $x$  has pet  $z$ . In Datalog, an additional parameter called a key needs to be specified. If an atom such as  $\text{person}(X)$  is a key then this atom must be present in all queries considered during discovery and the query needs to be frequent with respect to this atom. The frequency of a query  $Q$  is the number of answer substitutions  $\theta$  for the variables in the key atom, when the query  $Q\theta$  is present in the logical database.

The WARMR [Dehaspe and Toivonen, 1999, King et al., 2001] system follows a similar approach taken by the APRIORI [Agrawal and Srikant, 1994b] algorithm for finding frequent item sets for a propositional database. However, unlike APRORI, the WARMR system uses techniques from ILP to upgrade a logical database. A level wise approach is adopted where a complex query  $Q_2$  at level  $l + 1$  is generated from a simpler frequent query  $Q_1$  at level  $l$ , in such a way that that  $Q_1$   $\theta$ -subsumes  $Q_2$ .

Several approaches that are in spirit similar to those of WARMR have been adopted for the more specialized case of discovering sequential logical patterns. In seqlog [Lee

and De Raedt, 2002], a relational sequence database may consist of sequences of unix commands such as  $\text{cd}(\text{april})$ ,  $\text{latex}(\text{par}, \text{tex})$ ,  $\text{dvi}(\text{par}, \text{dvi})$ ,  $\text{lpr}(\text{par}, \text{pdf})$ . Relational sequence mining aims at finding frequent patterns for e.g.  $\text{latex}(X, \text{tex})$ ,  $\text{dvi}(X, \text{dvi})$  which means that a user after compiling a latex file into a dvi file, converts it into a pdf file. Frequent patterns are found by extending the standard level-wise search to sequences, in a space structured using  $\theta$ -subsumption, such as in WARMR.

**Relational Sequence Learning for Video Events.** Very recently, an unsupervised approach to learning event classes for card games such as Uno is explored in [Antanas et al., 2009]. The primitive events correspond to the type of card that is being dealt and this is recognized in a simplified vision setting. The dealings are expressed as a relational sequence that captures various attributes of a card such as its colour, number etc. A relational sequence mining framework called  $r$ -grams, as formulated in [Kersting et al., 2008], is then used to discover compositional event classes in the relational sequence that correspond to protocols in the card games such as: *a card is followed by a card of the same colour or same number*.

**Discussion.** The above approach is a case of the *Unsupervised discovery setting*, since the segmentation of activities into events is not known in advance. It models *composite events* in terms of *atomic events*. These techniques model *single temporal processes* and offer no mechanisms to separate multiple temporal processes. The event sequences are relational in nature and are therefore *semantically complex*.

## 2.6 Learning with Relational Graphs

Graph based representations offer powerful mechanisms for analysing relational data. This work introduces a graph based representation for event analysis, where graphs are used to represent spatio-temporal relationships between interacting objects. The following sections survey work in graph mining that are closely related to the two approaches for learning with graphs. The first is a mining based approach where activities are represented by one large activity graph and the task of mining event graphs – subgraphs that correspond to events – is addressed. The second learning approach hypothesises that activities are generated by a compact representation of event classes, each of which is a similar set of graphs with a probability distribution. The goal of learning is to find the event classes and the process that generated the activity.

The next section discusses graph based learning approaches under three main headings: (i) *graph mining*; (ii) *graph classification and clustering*; (iii) *graph sampling*. These are very closely related to this work and also cover a large part of graph based learning.

### 2.6.1 Graph Mining

Much of the initial work on graph based learning [Washio and Motoda, 2003] focussed on efficient mining of frequent subgraphs, since the isomorphism of graphs is combinatorially expensive [Köbler et al., 1993]. Despite this restriction, many solutions that efficiently search the space of candidate frequent graphs have been developed. The focus of much graph mining research is a *transactional setting* rather than a *single graph setting*. In a transactional mining setting, a set of graphs are given, referred to as a *graph database*. The *support* of a subgraph in the graph database is the number of graphs in the database in which the subgraph occurs at least once. *Frequent subgraphs* are those whose support exceeds a preset threshold. The notion of support in a single graph setting is discussed in [Bringmann and Nijssen, 2008].

Approaches for graph mining can be categorized into the following groups: (i) greedy search using compression; (ii) apriori based breadth first; (iii) depth first ; (iv) constraint based mining.

**Greedy Search with Compression.** SUBDUE [Holder et al., 1994] is a constrained beam search based technique which uses the MDL principle for obtaining a compressed representation of an input graph(s). The technique generates a candidate graph  $G$  by adding a vertex at each iteration starting with a subgraph with a single vertex that is present in the input graph  $I$ . At each iteration, candidate graphs are evaluated using the total description length (DL) [Grünwald, 2005] defined as the sum of the description length of the subgraph,  $L(G)$ , and description length of the input graph,  $I(L|G)$ , in which all the instances of the subgraph  $G$  are replaced by single nodes. Once a subgraph  $\hat{G}$  that minimizes the total description length is found, the next iteration starts by using the rewritten graph as a new input. Two additional features of SUBDUE are that it allows incorporation of background knowledge and permits variations in the subgraphs using approximate matching. However, this approach is prone to getting stuck in local optimum as Subdue is based on a greedy search with no backtracking. Moreover, the maximum width of the beam is predetermined.



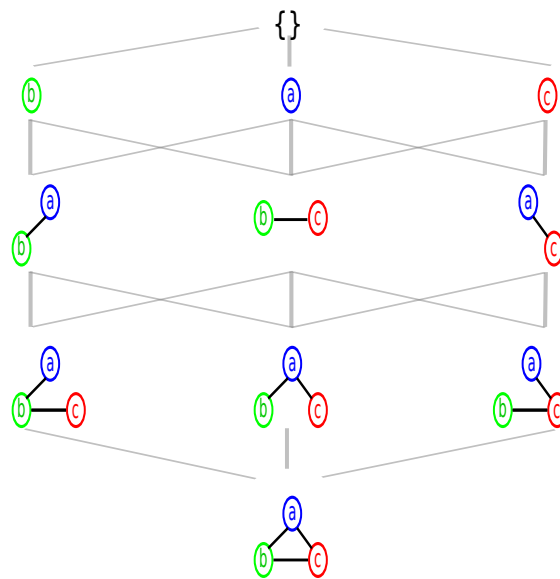


Figure 2.8: Edge expansion lattice, in which one edge is added at a time from the top in order to obtain the target graph at the bottom.

**Apriori based Breadth First Search.** The apriori based approaches perform a level wise search of the space to generate candidate  $k + 1$  sized graphs by combining *only* pairs of frequent  $k$  sized graphs, that share a common  $k - 1$  size graph. The frequency of a  $k + 1$  size candidate graph is computed by scanning the graph data base. By using only frequent graphs to generate candidates at the next level, the search space is kept under control. The two main techniques that are based on the apriori approach are AGM [Inokuchi et al., 2000] which adopts a vertex based approach where the size corresponds to the number of vertices and FSG [Kuramochi and Karypis, 2001], which adopts an edge based approach, where the size corresponds to the number of edges. The possible graphs are obtained by adding a vertex as in AGM and an edge as in FSG. The space of subgraphs that are obtained by adding one edge at a time, in order to obtain a target subgraph, is shown with an example edge expansion lattice [Hasan and Zaki, 2009], in Fig. 2.8.

In order to efficiently determine the frequency of graphs in AGM, the graphs data set are transformed into their canonical forms for computing subgraph isomorphisms.

FSG uses canonical labelling to efficiently check if a particular candidate subgraph has already been generated, since joining two subgraphs of size  $k$  can lead to multiple subgraphs of size  $k + 1$ . FSG increases the efficiency of deriving canonical labels by taking advantage of graphs which have subsets of vertices with the same labels and degree. The vertices are partitioned according to those with the same label and degree and only the possible permutations within each partition are exhaustively tested instead of testing all permutations. FSG also increases the efficiency of frequency counting of a  $k + 1$  size

candidate graph by computing the intersection of the TID lists [Dunkel and Soparkar, 1999] of its frequent  $k$  sized subgraphs. If the size of the intersection is below the support this  $k+1$  size candidate graph is pruned, otherwise the frequency of this graph is computed by limiting the search only to the transactions in the intersections of the TID lists.

**Depth First Search.** The above approaches suffer from a bottleneck that they are prone to generating the same graph many times, as there are  $n$  different  $n - 1$  edge graphs that can generate the same  $n$  edged graph. Moreover, sub-graph isomorphism is a costly operation and testing for false candidates degrades the performance. Gspan [Yan and Han, 2002] dramatically improves the performance by reorganizing the edge-expansion lattice into a DFS code tree, where the nodes at level  $k$  correspond to a candidate subgraph with  $k$  edges. The key characteristic of this tree is that a depth-first traversal can enumerate the tree in increasing lexicographical order. The algorithm proceeds with an outer loop which starts with frequent 1 edge graphs and for each such edge shrinks the graphs in the database by removing the edge, once all the descendants of this 1-edge graph have been searched. In the inner loop, the search is made efficient by pruning all descendants of a subgraph either if its not frequent or if its code is not minimum. The latter condition is a key feature of this approach since the lexicographical order ensures that if the code is minimum, then it has been discovered before and this in turn implies that all its descendants have been discovered before and so can be discarded.

**Constraint based Search.** The techniques above have been developed mainly for improving scalability on subgraph mining. However, when the graphs are dense the extraction is not always tractable and results in many uninteresting graphs outputted. CabGin [Wang et al., 2005a] and gprune [Zhu et al., 2007] introduced paradigms being to use constraints in order to reduce the cost of mining and increase the focus on interesting patterns. The size constraint is satisfied by a graph  $G$  if and only if size is greater than or lesser than a pre-defined threshold. Structural constraints on rigid structures such as the cycle and chain constraint were introduced due to the significance of chains and rings in molecular biology. The size of a cycle or a chain being greater than a pre-defined threshold is an example of a structural constraint. Other structural constraints are on girth and circumference. The girth of a graph is the length of a shortest simple cycle in the graph and the circumference is the length of a longest simple cycle. Another interesting class of constraints are constraints on the labels. These are used to express very specific constraints on the labels such as to mine substructures that consists of vertices labelled with atoms O and N with edges with not more than two double bonds and a ring of at least

size four. Another class of constraint is used to mine only those substructures that are a subgraph or a supergraph of a given graph. Finally, aggregate constraints such as *count*, *min*, *max*, *sum* are computed over labels or their attributes.

## 2.6.2 Graph Classification and Clustering

The task of classification is learning to classify separate individual graphs in a graph database into two or more categories. Both supervised classification and unsupervised approaches for graph clustering have been developed in the recent past.

The Subdue algorithm is extended for supervised learning as SubdueCL [Gonzalez et al., 2002]. Given graphs from positive and negative classes, this algorithm performs a beam search to generate candidate subgraphs which are evaluated according to classification accuracy or minimum description length. Once the optimum subgraph is found, the positive examples that are covered by this graph are removed and this process is repeated until all positive examples are covered. This procedure finds a set of connected graphs that is used to classify unseen graphs. An unseen graph is classified as positive if there is at least one subgraph in the set that is present in this graph and negative if it contains no graph in this set.

Another approach [Deshpande et al., 2005] first identifies frequent subgraphs in training examples using frequent graph mining and then represents each example in terms of a feature vector, where each feature is a function, such as frequency, that represents the membership of the frequent subgraph in that example. These features are used to train a support vector machine. During classification, an unseen example is represented in a similar manner and is given to the SVM for classification.

Another approach is to use graph kernels [Gärtner et al., 2003] which compute a similarity measure between a pair of graphs as the inner product of the feature vectors of these graphs over a high dimensional feature space. The advantage of using a kernel is that this similarity can be computed feasibly without explicitly generating the subgraph features. Intuitively, the direct product kernel between two graphs counts the identical walks that can be taken in both of them.

The approach in [Kudo et al., 2004] uses the DFS tree in gSpan to find subgraphs that distinguish between the positive and negative examples according to the boosting framework. A lower bound on the error is used to prune the search space in the DFS tree to prevent further expansion of candidate subgraphs.

### 2.6.3 Graph Sampling

Very recently there has been much interest in sampling a small set of frequent graphs that are frequent, interesting or representative, instead of mining for all subgraphs with respect to these criteria. Two interesting papers have been recently published that use Markov Chain Monte Carlo methods to sample interesting (frequent, discriminatory) subgraphs from a graph database [Hasan and Zaki, 2009] and representative graphs [Hübler et al., 2008] from a single graph respectively. When the direct sampling from a probability distribution is difficult, the Metropolis-Hastings (M-H) [Neal, 1993] algorithm approximates this distribution with a histogram by sequentially obtaining random samples. The key merit of the M-H algorithm is that it can draw samples from any probability distribution provided that a function proportional to the density can be calculated i.e. without having to compute the normalization constant that is required for obtaining the exact probability in Bayesian computations.

Such a sampling is performed with Markov chains which are a stochastic process that have the Markov property that future states depend only on the current and not the previous states. At each step in the Markov chain, a new sample is proposed by applying a move to the current sample. This sample is accepted or rejected according to an acceptance probability.

In [Hasan and Zaki, 2009], frequent or interesting graphs are sampled without enumerating the entire set of candidate subgraphs. The M-H algorithm performs a random walk on the partial order lattice of subgraphs using moves that either extend or remove edges. This results in generating candidates by moving up to a pattern super-neighbour or moving down to a pattern sub-neighbour in the partial order lattice. The acceptance probability for each candidate subgraph is evaluated as a ratio of their interestingness measures and the transition probabilities which are expressed in terms of the degrees in the edge-expansion lattice. In this manner, the Markov chain is used to sample interesting subgraphs till the random walk converges to a stationary distribution.

The authors in [Hübler et al., 2008] focus on finding representative subgraphs for a single large graph using the M-H algorithm. Representative graphs are regarded as smaller subgraphs of a large graph which approximates the properties of the original large graph. Starting with an initial random subgraph from the original graph, the M-H algorithm searches the space of subgraphs by generating candidate subgraphs with a move which corresponds to the composite operation of removing one node and adding a new node at each iteration. The acceptance probability for each candidate subgraph is evaluated as a ratio of their similarity (or equivalently an inverse distance function) to the original graph with respect to a pre identified set of properties. The graph with the highest similarity

score is chosen after the Markov chain converges to the stationary distribution.

**Discussion.** While the above approaches provide much insight into how graphs may be applied for the task of event analysis, they do not provide complete solutions to this problem. For example, in the mining task, the notion of interestingness of a subgraph as defined in the literature is not close to what is required for event analysis. For the generative model based approach, the goal is to find the best partition or cover an activity graph with subgraphs that are event like. The above approaches do not provide the framework for such an analysis.

However, there are ideas that are worth borrowing from the data association literature which has been employed often in tracking to assign detections to objects, and extending them to graphs. Recent solutions [Russell et al., 2006] use MCMC to find the optimal association by searching the space of possible associations. A Markov chain is constructed to sample the space of associations to search for the association that maximizes a posterior. The work proposed in this thesis is inspired by the ideas in data association and graph based learning literature.

## 2.7 Learning of Object Classes

The previous sections focussed on related work concerning event analysis. While event analysis is the main focus of the thesis, an additional contribution of this thesis is that of learning functional object classes using the learned event classes, as outlined in chapter 5. This part of the chapter describes related work on representing objects in various ways.

Learning models of object classes is a well researched and one of the most important topics in computer vision and robotics. While most traditional approaches have focussed on learning visual object classes that model the appearances of objects, recent work in the robotics community [Saxena et al., 2008] has focussed more on object affordances that model what actions can be performed on the object. Thus a chair is sittable, pushable, liftable etc. While affordance is primarily concerned with how an object may be used based on their physical properties, *functionality* focusses on *how the objects are actually used*, or more generally, the roles that they play with respect to events. That is, objects of a functional category tend to have a similar functional role with respect to events of the same event class, as explained further in chapter 5. The following paragraphs first review relevant literature on visual object classes in various learning settings, then work based on affordance based object classes and finally functional object classes.

### 2.7.1 Visual Appearance based Object Classes

Much of the work concerning visual object representation has focussed on the supervised setting, where a set of labelled object classes are given and a model trained on this is used to detect objects in unseen images. However, there has been growing interest in the last five years on the unsupervised learning setting due to the rapid proliferation of images containing various classes and the need to develop highly reliable object recognition systems, while minimizing the labour required in manual annotation of images for training. Moreover, unsupervised learning is closer to the goal of designing machines that observe and learn from the world with little guidance from humans.

There have been a plethora of techniques, both within the supervised and the unsupervised settings that vary in the nature and degree of supervision, as previously discussed in the context of event analysis in Section 2.1.2.

**Supervised Settings.** The completely supervised techniques (e.g. [Dalal and Triggs, 2005]) learn object class models when the position and the class labels for the objects of interest are specified. These models are used to recognize instances belonging to these classes in unseen images.

Other supervised approaches require a lesser degree of human guidance. The authors in [Duygulu et al., 2002] assume that the segmentations and the labels of more than one object class in each training images are given. However the correspondence between the segmentations and the labels are not given. A machine translation based approach is used to learn the correspondence and the object class models simultaneously. These models are used to predict the class labels of segments in unseen images.

Manual effort is further reduced in the work of [Fergus et al., 2005], where the condition that the position of the object needs to be specified is relaxed. Training images for object classes of interest are obtained with little effort using the Google search engine. The positions of the objects of interest are not specified and topic models are used to learn the distribution. A similar degree of supervision is adopted in [Todorovic and Ahuja, 2006], where it is assumed that instances of an object class of interest is frequent across several images. However, some images may contain multiple instances of the class of interest, objects of other classes and may not even contain the objects of interest. Image features are extracted and organized in a tree structure and frequently occurring trees are extracted and the object classes are modelled using a tree union between these trees.

**Object Discovery Settings.** A more unsupervised approach is the object discovery setting, where neither the object position/segmentation nor the object labels are assumed to

be given. The machine learns to discover the position/segmentation of the object, but not the class to which it belongs.

In the case of static images, image segmentation may result in object discovery in cases where the objects are homogeneous with respect to properties of their pixels such as colour, texture etc. Image segmentation approaches [Shi and Malik, 2000, Carreira and Sminchisescu, 2010] tend to segment images into regions which are homogeneous with respect to these properties.

In the case of video input, additional information such as homogeneity in motion is used to discover objects. The oldest and simplest approach that may be used to discover objects to an extent is foreground extraction. Some main issues with the foreground are that: (i) object blobs may be merged with the background; (ii) an object may be covered only partially by an object blob or may be even split into two blobs; (iii) two moving objects tend to be merged into the same foreground if they are moving spatially close to each other.

Object discovery techniques in [Southey and Little, 2006] address this problem by combining appearance, shape, and rigid object motion to discover and model multiple objects. Their approach involves finding a sequence of images and depth maps of a given scene. Appearance and shape features are used to oversegment the scene into smaller regions. Groups of moving features are identified and the movement of these features are used to determine which of these correspond to objects. Thus an object which may be oversegmented could be recovered by combining the segments that are likely to correspond to objects. Models of the discovered objects are formed with additional features. These additional features are extracted by taking snapshots of the discovered objects, as they move in time. In this way objects are automatically discovered and modelled.

**Unsupervised Settings.** At the other end of the spectrum are the unsupervised approaches whose input are the images or videos with no information about the position or the labels of the objects of interest. The output from these approaches are the position/segmentation along with the class labels that the system learned.

Objects are discovered and their classes learned simultaneously from a collection of static images [Russell et al., 2006] in two steps. First multiple segmentations for each image are produced, by varying the parameters of the normalized cut technique with the assumption that each object instance is correctly segmented at least by one segmentation. Then object classes which are groups of correctly segmented objects that are coherent in a large set of candidate segments, are learned.

Another approach by Parikh et. al. [Parikh and Chen, 2007] obtains a hierarchy

of object classes for static scenes by grouping image features which spatially co-occur across images for the same scene, under the same leaf of the hierarchy. In this manner, the technique learns to identify candidate objects such as keyboards, while also learning higher level object classes such as a desk area (consisting of a computer, desk etc).

Recent research in [Celik et al., 2009] learns object classes and detects instances of these in video in an unsupervised manner. Given a video, foreground blobs and their HOG features are first extracted. These blobs are clustered into object classes and a multi-class object detector is trained for these positive classes against negative samples automatically extracted from the background. The models learned on these object classes are used for detecting corresponding objects in the video. This work was first developed for a single dominant object class [Celik et al., 2008] in videos such as those of train stations, where objects of a single class such as persons are mostly present. Recently, this approach has been extended for multiple object classes for videos of domains such as traffic, where several classes such as cars, people etc. are expected to be seen.

**Discussion.** In this framework, where object interactions are modelled, the first step is the extraction of object blobs from video. While most of the work on extracting objects from a scene by modelling visual appearances are supervised, the object discovery [Southey and Little, 2006] and unsupervised settings [Russell et al., 2006, Celik et al., 2009] are showing increasing potential in recent years for this task. While this work uses a supervised approach for detecting and extracting object blobs for further processing using visual appearance modelling, their class information is not used further during the process of learning events and functional object classes.

## 2.7.2 Affordance Based Object Categories

Representing and modelling objects in terms of their affordances has in recent years gained increasing interest in the robotics and cognitive vision communities. In the context of cognitive agents such as robots, affordance based categories have been found more relevant than visual object categories, since robots interact with objects and the notion of how an object can be used is important for such interactions. Affordance properties are learned from the way a human or a robot interacts with objects.

In [Stark et al., 2008], simple prototypical actions between a human and an object (e.g. placing a cup with a handle) are used to learn the *affordance cues*, which correspond to the region of interaction on the object (e.g. the handle). Visual features are extracted for these interaction regions for affordance cue based object recognition. This approach is also extended to combining multiple affordance cues. In [Veloso et al., 2006], the



affordance properties of an object (e.g. chair) by are learned by detecting some pre-defined actions related to them (e.g. sitting). These are used to learn the visual properties of the object and are used for detecting the object in other parts of the video.

In [Ridge et al., 2009], a cognitive agent learns the affordances of objects by performing certain actions (e.g. push) on them and then seeing the result (e.g. slide, roll). Each object is represented in terms of these results in a result vector. The clustering of the result vector yields affordance classes. Then, both visual appearance vectors and the result vectors are taken and the best matching cluster (in the affordance classes) is found for each result vector using a nearest neighbour search. The best-matching clusters for each result vector are then used along with the object property vector for training and used for detecting unseen objects.

Another approach is followed by Needham et al. in [Needham et al., 2005], in which the actions associated with a particular configurations of objects in a table top game are learned. Objects are described by a tuple of features (colour, texture, position), and the rules learnt by the system induce a hierarchy on the objects, by grouping objects indistinguishable by virtue of resulting actions, into equivalence classes.

## 2.8 Qualitative Spatio-Temporal Relations

One of the main themes of this work is that qualitative spatio-temporal relationships offer a potential way for bridging the gap between low level event features and high level event descriptions. Qualitative primitives structure quantitative measurements into crisp equivalence classes, making distinctions which enable the abstraction of qualitatively interesting concepts from quantitative measurements [Cohn et al., 2003].

Qualitative spatial and temporal representations emerged in order to represent and reason with two important aspects of knowledge, namely time and space. Qualitative temporal reasoning is a part and parcel of commonsense knowledge and is reflected in natural language when we speak of certain things happening during, before etc. with respect to other things. Allen's Interval Algebra [Allen, 1983] was introduced as a calculus for temporal reasoning in 1983. This calculus defines 13 possible base relations between convex intervals on a directed line. The basic relations and a graphical depiction are given in Fig. 2.9. Allen's temporal algebra provides a composition table that can be used for reasoning about temporal descriptions of events. In this work, temporal relationships are used for representing and learning about events.

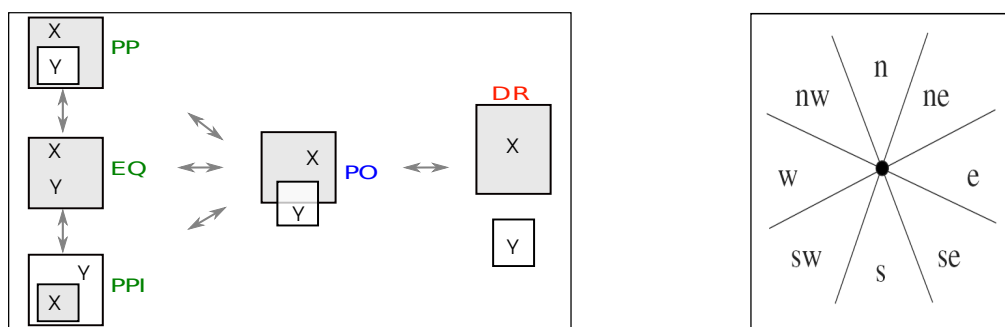
A number of qualitative spatial calculi were derived from the interval algebra for representing and reasoning about objects that are related to each other in space. The three

<p>Before(<math>X, Y</math>)</p>	<p>During(<math>X, Y</math>)</p>
<p>Meets(<math>X, Y</math>)</p>	<p>Finishes(<math>X, Y</math>)</p>
<p>Overlap(<math>X, Y</math>)</p>	<p>Equal(<math>X, Y</math>)</p>
<p>Starts(<math>X, Y</math>)</p>	

Figure 2.9: Allen’s temporal relationships between pairs of intervals. The intervals are given by  $X$  and  $Y$ . For each asymmetric relationship (Before), there is also an inverse relationship (BeforeI). Thus, there is a total of 13 base temporal relationships.

main aspects of spatial relationships are topology (e.g. *touch, inside*), orientation (e.g. *left of, above*) and distance (e.g. *near, far*). These relationships are used in natural language to describe the spatial aspects of the world qualitatively. Examples of topological and orientation relations are illustrated in Fig. 2.10(a) and 2.10(b) respectively.

In order to apply logical reasoning to spatial relationships, it is useful to have a set of qualitative binary *base relations* which have the property of being jointly exhaustive and pairwise disjoint i.e. between any two spatial entities exactly one of the base relations hold. These primitives allow compositional spatial relations that are exploited by reasoning engines which use compositional tables. The compositional tables constrain relations



(a) Conceptual Neighbourhood for the RCC-5 [Randell et al., 1992] [Cohn et al., 1997].

(b) Qualitative orientation relations.

Figure 2.10: Topological (RCC-5) and directional relationships are shown.

between two regions  $x$  and  $z$  when the relationships between regions  $x, y$  and  $y, z$  are known. Region Connection Calculus or specifically RCC-8 and RCC-5 [Randell et al., 1992, Cohn et al., 1997] are the most well known calculi for topological relationships. RCC-5 is a topology-based spatial theory which abstracts spatial configurations of two physical regions to a set of jointly exhaustive pairwise disjoint (jepd) qualitative relationships that can hold between this pair of regions. Five topological relations between two regions  $x$  and  $y$  are defined based on the parts they share. The conceptual neighbourhood graph (CNG) captures the continuity between these five relations as shown in Fig.2.10(a).

Qualitative spatial relationships may be manually specified or may be learned from the data. In [Fernyhough et al., 2000b], primitive spatial relations such as *right*, *ahead*, *behind*, which are typical on a road traffic scenario are manually defined. Composite events such as *following*, *pulling out*, *pulling in the front*, corresponding to sequences of primitive events, are learned from data. The approach in [Galata et al., 2002] is similar to that of [Fernyhough et al., 2000b], except that this approach automatically learns the qualitative spatial relationships from data and moreover explicitly computes probabilities associated with the composite events.

The approach in [Southey and Little, 2007] learn Qualitative Spatial Relations between objects by using a maximum entropy model and proximity features (Touching, Near, Mid, Far) to model the 'interactions' between those objects. One of their examples is that of an unknown object in the centre as well as a fork left to and a knife right to it. It is proposed that a model of qualitative spatial relationships between objects in a scene can provide scene contextual cues in order to detect objects such as a plate, which is surrounded by other objects like the fork and knife. They demonstrate its applicability for object recognition based on the learned scene contextual cues.

In this section, we characterize our approach using the key terms described in section 2.1.

## 2.9 Characterizing the Proposed Approach.

In this work, we address the problem of unsupervised activity understanding. This involves event learning and functional object learning. The event learning task that we address falls in the category of *unsupervised discovery setting*. In this setting, both event classes and the decomposition of activities in terms of events have to be learned simultaneously.

We use *high level features* in the form of relational structures. Specifically we propose a relational graph based representation in this work. This representation relies on the

extraction of mid-level features in the form of object tracks from video.

The events that we aim to learn are generally *compositional* and consist of *multiple parallel processes*, some of which can be *temporally dependent* on each other. Furthermore events may also *share objects*. The events can be *structurally complex*. For example, they can have nested structures. Also the events we characterise tend to have *complex temporal dependencies*.

The events considered here also tend to be *semantically complex*, as relational descriptions are needed to suitably describe interactions between objects. The events tend to be *spatially scene independent*. Therefore they are better characterized by spatial relationships rather than absolute position. Finally, event classes are allowed to have structural variations, i.e. they need not be structurally identical.

The activities that we address are *complex* to learn due to the presence of overlap between events and the presence of *coincidences*. Otherwise, the segmentation of activities into events is more straightforward and the events may be directly clustered to form event classes. Another source of complexity is the presence of *observation noise*, that arises due to complexities in detection and tracking.

## 2.10 Applicability of Related Approaches

The following paragraphs summarize the applicability of other approaches in literature to the unsupervised activity understanding task that we address. We focus on the approaches that have been classified under the unsupervised discovery setting, in this chapter.

**Pattern Recognition Techniques.** First, we consider previous work on the application of pattern recognition based techniques, described in Section 2.2. Sequence mining is used to extract patterns, when activities are represented as a single propositional sequence in [Toshev et al., 2006] and [Wang et al., 2005b]. However these approaches have very limited applicability to our task, as the sequences are propositional and activities are modelled as a single temporal process. In our case, activities are composed of events happening in parallel or with shared objects. When these are represented as a sequence, these parallel processes get interleaved and it becomes almost impossible to extract patterns using these approaches. Moreover, these approaches do not offer a principled way of handling coincidence and noise.

**State Space Models.** Second, we consider previous work on the application of state space models, described in Section 2.3. While the approach using a HHMM in [Xie

et al., 2003] models structurally complex events, their representation is propositional and they do not model multiple temporal processes. Therefore, it is hard to separate parallel interleaved events. For this reason this technique has limited applicability to our task.

On the other hand, the approach in [Xiang and Gong, 2006] uses a DML-HMM to model multiple temporal processes to model a single event. This technique required substantial data (more than 20 turnarounds at an airport apron) most of which contain only the event of interest, just to model a single composite event. This is due to the parametric generative model that requires a reasonable amount of training data. Their representation is also propositional, making it extremely hard to separate interleaved multiple parallel events. For these reasons, these approaches have limited applicability for activities that we address i.e where there are multiple events and coincidences occurring in parallel.

The approach in [Wang et al., 2009] model activities in terms of topic models. While approaches based on topic models offer a promising paradigm to model multiple parallel processes, their applicability to our task is limited by the bag of words assumption, that make it difficult to model complex spatial and temporal relationships between these process. Moreover, they are spatially scene dependent as they require that all the activities that are modelled have similar spatial configurations, e.g. the activities on the same road. Our approach models spatial relationships between objects and are therefore spatially scene independent.

The approaches described above are a case of the *Unsupervised discovery setting*, since the segmentation of activities into events is not known in advance. The events in the work [Xiang and Gong, 2006] are *scene dependent* as they are spatially related to a particular scene under consideration. The events in [Xiang and Gong, 2006] are *composite*. The HHMM can model nested structures and are therefore not *structurally simple*, though the events in [Xie et al., 2003] are generally simpler. The events modelled in [Xiang and Gong, 2006] are *multiple temporal processes*, though the temporal dependencies that they model are essentially first order Markov. It is worthwhile noting that the sequences are propositional in nature and are therefore *semantically simple*.

**Grammars.** We consider previous work on the application of grammars, described in Section 2.4. Grammars (particularly SCFGs) are interesting because they can model structurally complex events. The approach in [Kitani et al., 2008] uses SCFGs for the unsupervised discovery task. They also model the presence of observation noise. However, they model only *single temporal processes* and offer no mechanisms to separate multiple temporal processes. Their propositional representation makes it harder to apply to activities with multiple parallel processes. This limits the applicability to our task.

**Logic based Relational Learning.** We consider previous work on the application of logic, as described in Section 2.5. A relational sequence approach [Antanas et al., 2009] has been used for the unsupervised discovery task for activities. While the events that are modelled are semantically complex, the approach does not offer any mechanisms to separate multiple temporal processes. This aspect limits the applicability to our task.

**Graph Based Relational Learning.** Finally, we consider previous work on the application of graphs, as described in Section 2.6. To the best of our knowledge, we have not found any other work apart from our own, that applies graph based relational learning to the task of unsupervised activity understanding.

In the earlier phase of this work, we represented the entire activity as a graph structure and applied existing graph mining techniques to discover patterns. While the graphs that were generated were mostly uninteresting (i.e. did not represent any meaningful entity such as an interaction), they provided a starting point for exploring the possibility of incorporating additional constraints into the mining process.

We found that existing techniques do not allow a way of incorporating this knowledge, and thus could not be directly applied for our task. However, these initial explorations lead to a graph based relational learning framework that we use to model complex activities. The rest of the thesis describes this framework.

## 2.11 Conclusions

This chapter has reviewed several aspects of related work that is relevant to this thesis. The review first covered the main trends in the related work on event analysis, such as pattern recognition methods, state space models, grammars and logic. The literature on graph mining is reviewed as our approach applies and builds upon techniques developed in this research area. We have also reviewed relevant aspects of literature on learning object classes, since this forms a secondary but important aspect of the framework proposed in the thesis. Finally, we have reviewed related work on qualitative spatio-temporal relationships, as they are central for characterizing events in this work.

# Chapter 3

## Representation of Interactions

---

### 3.1 Introduction

Our supposition is that qualitative spatio-temporal relationships *between* objects are fundamental in characterizing many human activities, even more so than the behaviour of objects considered individually. Even though objects move by themselves through continuous space and time, the human mind tends to represent and process this data in terms of discrete states that are qualitatively interesting [Cohn et al., 2003]. For example, on an aircraft apron, even though a trolley, a loader and a plane may move through continuous space across a time frame of several thousand image frames, there are usually only a very small set of qualitatively interesting states between these objects, of which three are illustrated and explained in Fig. 3.1. Qualitatively interesting states between a set of objects are marked by a persistence of qualitative spatial relationships between all pairs of objects until there is a difference in qualitative spatial relationship between at least one pair of objects.

We regard an *interaction* as a sequence of distinct qualitatively interesting topological states<sup>1</sup> between a set of region histories. A region history is simply a temporal sequence of regions. Interactions are regarded as conceptual or schematic entities, that exist independently of observable objects in space and time. However, they are said to be *embedded* in space and time when a set of concrete objects exemplify an identical sequence of spa-

---

<sup>1</sup>We restrict ourselves to topology in this work. But the general idea can be assumed to be adaptable to other spatial aspects.

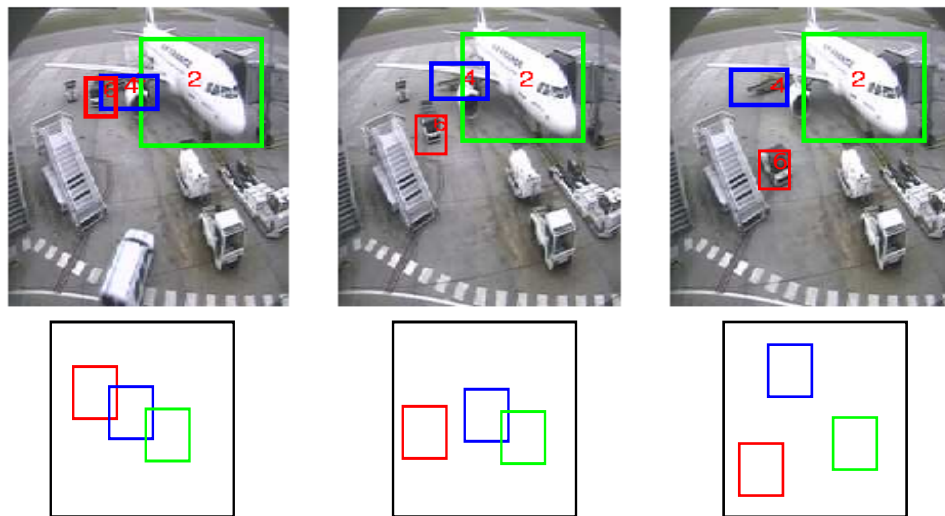


Figure 3.1: An interaction is depicted below in a schematic form, as a sequence of three distinct qualitative spatial states. An embedding of this sequence by concrete objects in space and time is shown above.

tial states that characterize the interaction. An example of an interaction between three abstract region histories and a corresponding embedding by real objects in space and time is shown in Fig. 3.1.

This chapter focusses on three aspects: (i) interactions and their relational descriptions; (ii) a similarity between these relational descriptions; (iii) relationship between interactions and their embeddings in space and time.

The first focus of this chapter is to provide a relational description of interactions in terms of qualitative spatial and temporal relationships between the corresponding region histories. When the relational description of an interaction is exemplified by region histories that correspond to a concrete set of observable objects, these objects are regarded as an *embedding* of the interaction in time and space. However, the interaction itself is an abstraction that has no reference to specific details of embeddings such as spatial location and temporal durations in which the corresponding objects are observed. The relational description takes the form of a tripartite graph structure called an *interaction graph*.

The second focus of this chapter is to define a similarity measure on relational descriptions in order to compare interactions spatio-temporally i.e. on the basis of their qualitative spatio-temporal relationships. This similarity measure is used for clustering interactions and will be useful for the purposes of learning from this relational representation in chapter 4.

The final focus of the chapter is the relationships between interactions and their embeddings in space and time. The first part deals with mapping a set of tracks from video



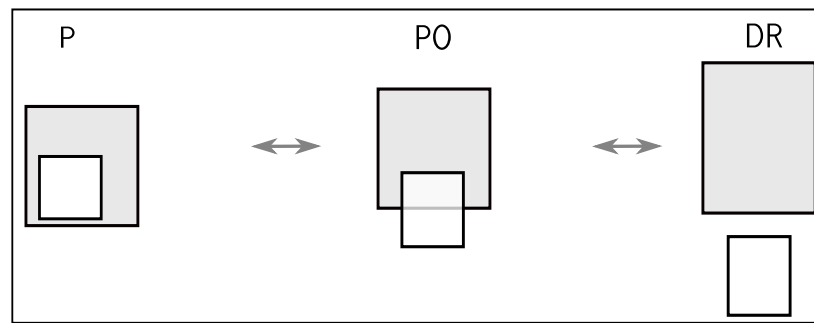


Figure 3.2: The Conceptual Neighbourhood Graph (CNG) for the three qualitative spatial relations –  $\{P, PO, DR\}$  – used in this work, is shown. The CNG captures the continuity between these three relations.

data to the corresponding interaction, by inducing qualitative spatial relationships that hold between them. These relationships have been traditionally defined [Cohn et al., 2003] in terms of sharp boundaries of change between regions. However, this can result in unstable relationships due to noise that arises from complexities of processing video data. Moreover, the traditional definitions are defined for a static scenario i.e. for a single frame and do not take advantage of the temporal history of spatial relationships that can be obtained from previous frames which are available in video data. Therefore, a Hidden Markov Model (HMM) based technique for computing a more stable sequence of qualitative spatial relationships from video data is proposed. The second part deals with how this HMM based approach can be used to determine how likely it is that a set of tracklets is an embedding of an interaction.

Section 3.2 introduces the qualitative spatio-temporal relationships used in this work. Section 3.3 describes interactions in terms of qualitative spatio-temporal relationships with some added constraints. This section also introduces interaction graphs, which are a graph based representation of interactions. Section 3.4 details the similarity measure between interaction graphs. A non-parametric clustering approach to form clusters of interaction graphs is also described in this section. Section 3.5 describes the relationships between an interaction and its embeddings. Finally, Section 3.6 concludes the chapter.

## 3.2 Qualitative Spatio-Temporal Relationships

This section describes the qualitative spatial and temporal relationships used in this work. This is followed by how these are combined to form spatio-temporal relationships, which are an important construct for describing the notion of interaction described in the next Section 3.3. This work focusses on three simple spatial relationship, for which the con-

ceptual neighbourhood graph is shown in Fig. 3.2:

$$\mathfrak{R} = \{P \text{ (Part Of) , PO (Partially Overlaps) , DR (Discrete)}\}$$

The choice of these spatial relationships is motivated by considerations arising from real video data, where interesting things tend to happen even when there is a change with respect to these simple spatial relationships. The illustration in Fig. 3.1 lends support to this motivation.

These three relationships are derived from the RCC-5 calculus [Cohn and Hazarika, 2001] (Fig. 2.10(a)) according to the following considerations. Firstly, different objects are unlikely to have equal spatial extents in real video data and therefore EQ is not needed. Moreover, the inverse PPI of PP (Proper Part) is not needed because the representation followed in this work takes into account this duality. Therefore the three original relations PP, PPI and EQ (Equals) in RCC-5 are combined into one state P. In this way, RCC-5 is transformed into a simpler calculus with just three qualitative spatial relations  $\mathfrak{R} = \{DR, PO, P\}$ .

The temporal relationships used in this work are derived from Allen's temporal relations [Allen, 1983] by taking the seven basic relations and omitting the respective inverses, since the representation followed in this work takes into account this duality. The temporal relationships are the following:

$$\mathfrak{N} = \{Before(<), Meets(m), Overlap(o), Starts(s), During(d), Finishes(f), Equal(=)\}$$

While the choice of spatial relationships in this work are specific and simple, the proposed framework for representing and learning events is more general. For example it allows the possibility of incorporating other types of spatial relationships (e.g. orientation and distance) respectively, or even learning them from data using approaches similar to [Galata et al., 2002]. These possibilities are discussed further in chapter 7 on future work.

**Spatial-Temporal Relationships.** Spatio-temporal relationships are a representation that combine spatial and temporal relationships and are useful when spatial relationships hold for a certain duration. For example when the spatial relationships between two pairs of region histories (either same or different pairs) hold for a certain duration, then the spatio-temporal relationships between them can be expressed as: (i) the fact that the spatial relationship between the first pair of region histories holds for a certain interval; (ii) the fact that the spatial relationship between the second pair of region histories hold for a certain interval; (iii) the temporal relationships between these respective intervals.

To formulate this notion more precisely, first of all, the predicate  $\text{Holds}(s, (r_i, r_j), I_k)$  represents the fact that the spatial relationship  $s \in \mathfrak{R}$  holds for a pair of region histories  $(r_i, r_j)$ , during the interval  $I_k$ .

The especially interesting case is when  $s$  holds *maximally* for an interval  $I_k$ . This can be expressed as  $\text{Holds}(s, (r_i, r_j), I_k)$  and a different pair of spatial relationships  $s', s'' \in \mathfrak{R}$  and  $s' \neq s, s'' \neq s$  holds immediately before and after this interval respectively. This special case is given by the predicate  $\text{MaximallyHolds}(s, (r_i, r_j), I_k)$ .

An *episode* is defined as  $e_k = (r_i, r_j, s, I_k)$  for compactly specifying spatial relationships that hold maximally for a certain duration  $I_k$ .

The following functors are defined for an episode  $e_k$ : (i)  $\text{Regions}(e_k)$  maps to the respective pair of region histories  $(r_i, r_j)$ ; (ii)  $\text{Interval}(e_k)$  maps to the temporal interval  $I_k$ ; (iii)  $\text{Spatial}(e_k)$  maps to the spatial relation  $s$ . Using these functors, the *truth value* that a spatial relationship  $\text{Spatial}(e_k)$  *maximally holds* between  $\text{Regions}(e_k)$  during the interval  $\text{Interval}(e_k)$ , for an episode  $e_k$  is expressed as a predicate:

$$\text{MaximallyHolds}(\text{Spatial}(e_k), \text{Regions}(e_k), \text{Interval}(e_k))$$

Two such episodes  $e_k$  and  $e_l$  for two different pairs of region histories are shown in Fig. 3.3.

For a pair of episodes  $e_k$  and  $e_l$ , their respective intervals  $\text{Interval}(e_k)$  and  $\text{Interval}(e_l)$  can be related temporally by specifying the corresponding temporal relation<sup>2</sup>

$$a = \text{Temporal}(\text{Interval}(e_k), \text{Interval}(e_l)), \text{ where } a \in \mathfrak{N}$$

The functor  $\text{Temporal}$  is defined in Appendix D. The qualitative spatio-temporal relationships are expressed as a conjunction of spatial and temporal relationships. For the example in Fig. 3.3, the qualitative spatio-temporal relationships are expressed as:

$$\begin{aligned} & \text{MaximallyHolds}(\text{Spatial}(e_k), \text{Regions}(e_k), \text{Interval}(e_k)) \wedge \\ & \text{MaximallyHolds}(\text{Spatial}(e_l), \text{Regions}(e_l), \text{Interval}(e_l)) \wedge \\ & \text{Overlaps}(\text{Interval}(e_l), \text{Interval}(e_k)) \end{aligned} \tag{3.1}$$

<sup>2</sup>Note that either  $\text{Temporal}(\text{Interval}(e_k), \text{Interval}(e_l))$  or  $\text{Temporal}(\text{Interval}(e_l), \text{Interval}(e_k))$  is well defined, but not both, unless  $\text{Temporal}(\text{Interval}(e_k)) = \text{Equal}$

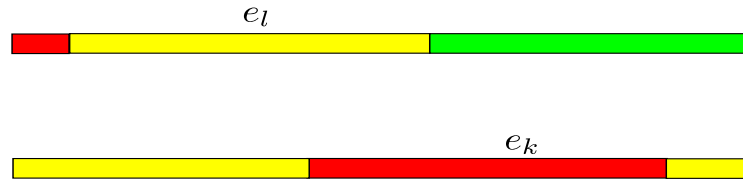


Figure 3.3: Co-temporal spatial relationships are illustrated. Each color band represents an episode. The colors green, yellow and red correspond to spatial relations P, PO, DR respectively. Particularly of interest in the text are the two episodes  $e_l$  and  $e_k$ , as they provide an example for describing spatio-temporal relationships.

### 3.3 Interactions and Interaction Graphs

This section describes the concepts of interaction and interaction graph proposed in this thesis. Informally, an *interaction* is regarded as a time sequence of distinct qualitative states between a set of region histories. The *qualitative spatial state* of a set of region histories  $R = \{r_1, r_2, \dots, r_k\}$  at any instant in time can be defined as the set of all pairwise spatial relationships  $s_{ij} \in \mathfrak{R}$  for each distinct pairs of region histories  $(r_i, r_j) \in R$ . Interactions arise when there is a change in spatial relationships between atleast one pair of region histories. An interaction is characterized by the bounding interval within which the changing qualitative states are considered to be of interest. An example of an interaction as a sequence of three different qualitative states is shown in Fig. 3.4 between the two solid vertical lines.

Interactions are schematic entities, where the region histories and the time intervals need not be necessarily *observed* in real space and time. While interactions can be expressed as a sequence of qualitative spatial states, in this work, we choose to represent interactions in terms of spatio-temporal relationships, as described above. This representation provides a compact and natural representation, particularly for spatial relationships that are co-temporal, as illustrated in Fig. 3.4. In contrast, representing as a global sequence of qualitative states would require artificially segmenting certain episodes along boundaries of other episodes. However, not all qualitative spatio-temporal relationships (e.g. as given in expression 3.1) qualify as interactions. The following paragraphs describe certain conditions with respect to spatial and temporal relationships for characterizing interaction.

**Spatial Relationships.** A set of episodes  $\mathcal{E}$  for a set of region histories  $R$  can be regarded as specifying the *spatial relationships* for an interaction if  $\mathcal{E}$  implies the existence of a sequence of qualitative states between  $R$  within a bounding interval, such that  $\mathcal{E}$  describes *all and only* those spatial relationships, between  $R$  during this bounding interval.

For example, this set of episodes  $\mathcal{E} = \{e_2, e_3, e_4, e_5, e_6, e_7\}$  qualify in characterizing the interaction in Fig. 3.4 for the following reason. Consider the set of *initial episodes*  $\mathcal{E}_I$  (i.e.  $\{e_2, e_4, e_5\}$ ) and the set of *final episodes*  $\mathcal{E}_F$  (i.e.  $\{e_3, e_4, e_7\}$ ) for this set  $\mathcal{E}$ . The first reason is that the bounding interval for this interaction is given by the *starting time instants* of one or more of the *last starting initial episodes* (e.g. as given by the solid vertical line at the start of  $e_2$ ) and by the *ending time instants* of one or more of the *first ending final episodes* (e.g. as given by the solid vertical line at the end of  $e_7$ ). The second reason is that this set of episodes  $\mathcal{E}$  characterizes *all and only* those spatial relationships, between the region histories  $r_1, r_2, r_3$  during this bounding interval.

More formally, the set of initial episodes  $\mathcal{E}_I$  and the set of final episodes  $\mathcal{E}_F$  for a set of episodes  $\mathcal{E}$ , are defined as follows.

$$\begin{aligned} \mathcal{E}_I &= \{e : e \in \mathcal{E} \wedge \forall e' \in \mathcal{E}, [\text{Regions}(e') = \text{Regions}(e)] \Rightarrow [\text{Meets}(\text{Interval}(e), \text{Interval}(e')) \\ &\quad \vee \text{Before}(\text{Interval}(e), \text{Interval}(e'))]\} \\ \mathcal{E}_F &= \{e : e \in \mathcal{E} \wedge \forall e' \in \mathcal{E}, [\text{Regions}(e') = \text{Regions}(e)] \Rightarrow [\text{Meets}(\text{Interval}(e'), \text{Interval}(e)) \\ &\quad \vee \text{After}(\text{Interval}(e), \text{Interval}(e'))]\} \end{aligned}$$

A set of episodes  $\mathcal{E}$  is regarded as describing an interaction if the following two conditions are satisfied.

1. The set  $\mathcal{E}$  guarantees the existence of an interval (as shown between the two solid vertical lines in Fig. 3.4) that characterizes the interaction.
2. The set  $\mathcal{E}$  should contain all the episodes (e.g.  $e_2 - e_7$  in Fig. 3.4) between any two episodes for the same pair of region histories considered in  $\mathcal{E}$ .

These two conditions are elaborated below. Firstly, the existence of the starting and ending point, for the interval that is used to characterize an interaction, is guaranteed for a set of episodes  $\mathcal{E}$ , if for a set of episodes  $\mathcal{E}$ : (i) every pair of *initial episodes*  $\mathcal{E}_I$  share an overlapping interval; (ii) every pair of *final episodes*  $\mathcal{E}_F$  share an overlapping interval. The condition on the initial episodes would ensure that they all share the *starting time*

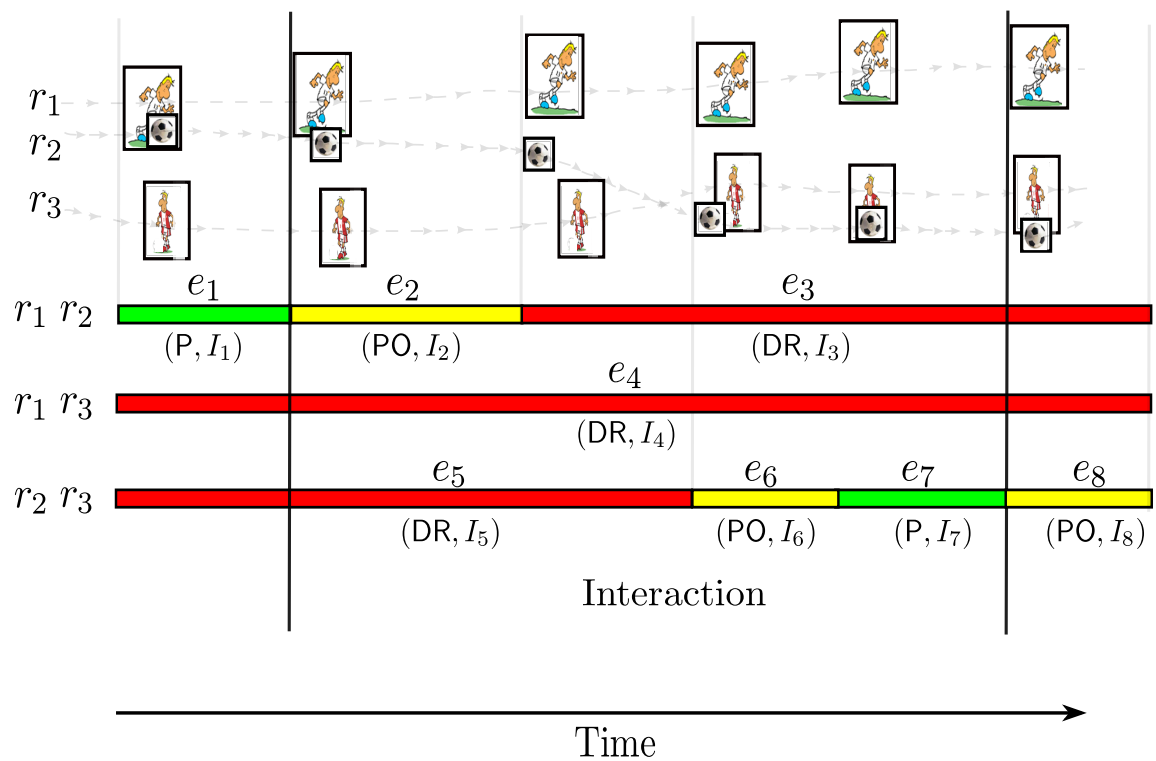


Figure 3.4: Our idea of an interaction is shown to be represented in two ways. *Above*: an interaction between three region histories  $r_1, r_2, r_3$  is shown as a temporal sequence of four distinct qualitative spatial states. These states are bounded by a temporal interval shown by two solid vertical lines. This interaction is related to a schematic notion of passing a ball i.e. where the first player (with region history  $r_1$ ) passes the ball ( $r_2$ ) to the second player ( $r_3$ ), who keeps it. *Below*: the episodes ( $e_2$  to  $e_7$ ) that correspond to this interaction are shown. The second way of representing an interaction involves specifying *qualitative spatio-temporal relationships* in terms of these episodes, as described in the main text.

*instant of the last starting initial episode(s)*. The condition on the final episodes would ensure that they all share the *ending time instant of the first finishing final episode(s)*. In this manner, these two conditions imply the existence of the starting and ending time instants of the *interval* which contains (or overlaps with) *only* those episodes in  $\mathcal{E}$ . The existence of such an interval, as implied by these two conditions, is the first requirement in characterizing an interaction. This is expressed as:

$$\begin{aligned} & \forall (e_i, e_j) \in \mathcal{E}_I [\neg \text{Before}(\text{Interval}(e_i), \text{Interval}(e_j)) \wedge \neg \text{After}(\text{Interval}(e_i), \text{Interval}(e_j))] \wedge \\ & \forall (e_i, e_j) \in \mathcal{E}_F [\neg \text{Before}(\text{Interval}(e_i), \text{Interval}(e_j)) \wedge \neg \text{After}(\text{Interval}(e_i), \text{Interval}(e_j))] \end{aligned} \quad (3.2)$$

Secondly, in order that  $\mathcal{E}$  completely describes an interaction, it should contain all episodes between any two episodes for the same pair of region histories considered in  $\mathcal{E}$ , i.e. the set of episodes  $\mathcal{E}$  should be gapless. A description without gaps ensures that it completely and uniquely represents an interaction. That is:

$$\begin{aligned} & \forall e_i, e_j \in \mathcal{E} : [\text{Regions}(e_i) = \text{Regions}(e_j) \wedge \text{Before}(\text{Interval}(e_i), \text{Interval}(e_j))] \\ & \Rightarrow \exists e_k \in \mathcal{E} : [\text{Meets}(\text{Interval}(e_i), \text{Interval}(e_k)) \wedge \text{Regions}(e_k) = \text{Regions}(e_i)] \end{aligned} \quad (3.3)$$

The set of episodes  $\mathcal{E} = \{e_2, e_3, e_4, e_5, e_6, e_7\}$  in the example interaction in Fig 3.4 satisfy conditions 3.2 and 3.3.

**Temporal Relationships.** Episodes are naturally related to each other in time and these dependencies are suitably captured using Allen's temporal relationships between the respective intervals for pairs of episodes. For the interaction in Fig. 3.4, the temporal relation  $\text{Meets}(\text{Interval}(e_2), \text{Interval}(e_3))$  expresses the temporal dependency between the intervals corresponding to episodes  $e_2$  and  $e_3$ .

Given a set of episodes  $\mathcal{E}$ , a set  $\mathcal{A}$  is regarded as a consistent and complete description of the temporal relations that correspond to an interaction if and only if all and only those temporal relationships that are *germane* to that interaction are contained in  $\mathcal{A}$ . The temporal relationships between all pairs of episodes in  $\mathcal{E}$  are considered germane to the corresponding interaction, *except* those which are between either (i) two initial episodes  $\mathcal{E}_I$ , or (ii) two final episodes  $\mathcal{E}_F$ , as these non-germane temporal relationships may be

determined outside the bounding interval for the corresponding interaction.

The inclusion of these non-germane temporal relationships would imply that that even though two interactions may be identical with respect to their spatio-temporal relationships within an interval, the spatio-temporal relationships describing these two interactions may be *non-isomorphic* (in a sense to be made precise in Section 3.4), as these relations can vary depending on when the respective initial episodes start or when the respective final episodes end.

More formally, the set of temporal relationships  $\mathcal{A}$  for episodes  $\mathcal{E}$  is defined as follows<sup>3</sup>:

$$\mathcal{A} = \{a(\text{Interval}(e_k), \text{Interval}(e_l)) : a = \text{Temporal}(\text{Interval}(e_k), \text{Interval}(e_l)) \wedge \{e_k, e_l\} \subseteq \mathcal{E} \wedge \neg [\{e_k, e_l\} \subseteq \mathcal{E}_I \vee \{e_k, e_l\} \subseteq \mathcal{E}_F]\} \quad (3.4)$$

The corresponding set of temporal relationships  $\mathcal{A}$  for  $\mathcal{E} = \{e_2, e_3, e_4, e_5, e_6, e_7\}$  in the example interaction in Fig 3.4 is given by considering only germane temporal relationships that are obtained by temporally relating the intervals corresponding to the following episode pairs:

$$\{(e_2, e_3), (e_2, e_6), (e_2, e_7), (e_3, e_5), (e_3, e_6), (e_4, e_6), (e_5, e_6), (e_5, e_7), (e_6, e_7)\}$$

**Spatio-Temporal Relationships for an Interaction.** The qualitative spatial and temporal relationships together describe an interaction. Given a set of episodes  $\mathcal{E}$  that satisfy conditions 3.2 and 3.3 and a corresponding set of temporal relationships  $\mathcal{A}$  given by equation 3.4, an interaction is represented as follows:

$$\bigwedge_{e \in \mathcal{E}} \text{MaximallyHolds}(\text{Spatial}(e), \text{Regions}(e), \text{Interval}(e)) \wedge \bigwedge_{a \in \mathcal{A}} a \quad (3.5)$$

### 3.3.1 Interaction Graph

We choose to represent interactions in a relational form with a tripartite graph based structure called an *interaction graph*. This has several computational advantages over a purely textual logical notation. These advantages are discussed further below in this section.

<sup>3</sup>The following clarification needs to be made, given the way the predicate *Temporal* has been defined. If  $\text{Temporal}(\text{Interval}(e_k), \text{Interval}(e_l))$  is undefined, then there is no  $a(\text{Interval}(e_l), \text{Interval}(e_k))$  relationship in  $a$ , but there may be a dual  $a(\text{Interval}(e_k), \text{Interval}(e_l))$ .



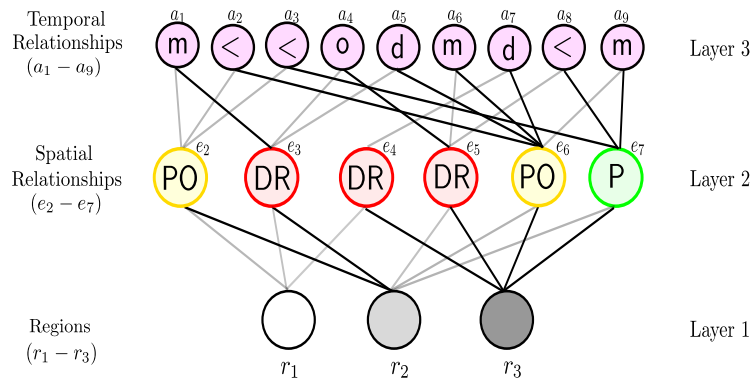


Figure 3.5: The interaction graph for the interaction in Fig. 3.4 is shown. The layer 1 nodes at the bottom, correspond to region histories  $r_1, r_2, r_3$  in Fig. 3.4. The layer 2 nodes at the middle correspond to the episodes  $e_2$  to  $e_7$  for the interaction shown in Fig. 3.4. These nodes are labelled with corresponding spatial relationships. The layer 3 nodes at the top are labelled with corresponding the core temporal relationships between certain pairs of episodes (as clarified in the main text). The lighter arrows point upwards, while the darker arrows point downwards, in order to represent the asymmetry in the qualitative spatial and temporal relationships respectively.

The nodes of three layers are as follows (refer to Fig. 3.5 for the examples): (i) *layer 1 nodes* map to the region histories (e.g.  $r_1, r_2, r_3$ ); (ii) *layer 2 nodes* represent the germane set of episodes  $\mathcal{E}$  (e.g.  $e_2 - e_7$ ) and are labelled with their respective spatial relationships, such that the episode node (e.g.  $e_2$ ), the corresponding pair of region history nodes ( $r_1, r_2$ ) together with the connecting edges express the logical relationship implied by the episode node (e.g. the nodes corresponding to  $e_2, r_1, r_3$  together with the edges that connect them represents the proposition  $\text{MaximallyHolds}(\text{PO}, (r_1, r_2), I_2)$ ); (iii) *layer 3 nodes* (e.g.  $a_1 - a_9$ ) are labelled with germane temporal relationships expressed in equation 3.4, such that a layer 3 node (e.g.  $a_1$ ) and the corresponding pair of episode nodes (e.g.  $e_2, e_3$ ), together with the connecting edges, express the temporal relationships in  $\mathcal{A}$  (e.g.  $\text{Meets}(\text{Interval}(e_2), \text{Interval}(e_3))$ ).

An interaction graph  $g = (\mathcal{V}, \mathcal{D}, \psi_2, \psi_3, \mathfrak{R}, \mathfrak{N})$  is a directed edge-labelled *layered graph* with three layers. The vertices  $\mathcal{V}$  of this graph are divided into three sets corresponding to the three layers and is given by  $\mathcal{V}^1 \cup \mathcal{V}^2 \cup \mathcal{V}^3$ . The directed edges  $\mathcal{D}$  exist only between adjacent layers. All the relations are binary, so there are exactly two edges from each node to nodes in the layer below. The function  $\psi_2$  maps the nodes in the second layer to labels which are spatial relations in  $\mathfrak{R}$ . The function  $\psi_3$  maps nodes in the third layer to labels which are temporal relations in  $\mathfrak{N}$ .

1. *Layer 1 nodes* (e.g.  $r_1, r_2, r_3$  in Fig. 3.5) correspond to the interacting region histo-

ries  $R$ . There is a 1-1 mapping  $\varphi_1 : \mathcal{V}^1 \leftrightarrow R$ .

2. *Layer 2 nodes* (e.g.  $e_2$  to  $e_7$  in Fig. 3.5) correspond to a set of episodes  $\mathcal{E}$  that satisfy conditions 3.2 and 3.3 and are labelled with the respective spatial relationships. That is, there is a 1-1 mapping  $\varphi_2 : \mathcal{V}^2 \leftrightarrow \mathcal{E}$  where  $\mathcal{E}$  is the set of episodes generated by the region histories  $R = \varphi_1(\mathcal{V}^1)$ . These nodes are labelled with spatial relations between the respective pairs of region histories pointed to in layer 1 as shown in Fig. 3.5. A mapping  $\psi_2 : \mathcal{V}^2 \rightarrow \mathfrak{R}$  is defined such that for  $v \in \mathcal{V}^2$ ,  $\psi_2(v) = s \in \mathfrak{R}$  if and only if:

$$\begin{aligned} \exists v', v'' \in \mathcal{V}^1 \wedge s = \text{Spatial}(\varphi_2(v)) \wedge \\ \text{Regions}(\varphi_2(v)) = \langle \varphi_1(v'), \varphi_1(v'') \rangle \wedge \langle v : v', v'' \rangle \in \mathcal{D} \end{aligned}$$

where  $\langle v : v', v'' \rangle \in \mathcal{D}$  is defined as follows.

$$\langle v : v', v'' \rangle \in \mathcal{D} \equiv [\langle v, v' \rangle \wedge \langle v'', v \rangle \in \mathcal{D}] \vee [\langle v', v \rangle \wedge \langle v, v'' \rangle \in \mathcal{D}]$$

This notation enables us to handle the convention introduced above for *asymmetric* binary spatial and temporal relations. The asymmetry in spatial relationships is given by the direction of the edges. For example, when the relationship between  $r_2$  and  $r_3$  is P for the interaction in Fig. 3.4, this is given by an upward arrow (light shade in Fig. 3.5) from the node corresponding to  $r_2$  to the second layer node labelled with P and a downward arrow (dark shade in Fig. 3.5) from this node to the the node corresponding to  $r_3$ . In the above equation, the direction of the edges is given by the order of the vertices, for example in the above  $\langle v, v' \rangle \in \mathcal{D}$  is used to indicate that there is a directed edge from node  $v$  to  $v'$ . The same scheme is followed for representing the *asymmetry* in temporal relationships.

3. *Layer 3 nodes* are labelled with the set of *germane* temporal relationships  $\mathcal{A}$  as given in equation 3.4 between the durations corresponding to pairs of episode nodes, in the second layer. The layer 3 nodes are labelled with Allen's temporal relation between intervals corresponding to the episodes for the pair of layer 2 nodes, pointed to from the layer 3 node. A mapping  $\psi_3 : \mathcal{V}^3 \rightarrow \mathfrak{N}$  is defined such that for any  $v \in \mathcal{V}^3$ ,  $\psi_3(v) = a \in \mathfrak{N}$  if and only if:

$$\begin{aligned} \exists v', v'' \in \mathcal{V}^2 \wedge a = \text{Temporal}(\text{Interval}(\varphi_2(v'), \text{Interval}(\varphi_2(v'')))) \wedge \\ \varphi_3(v) = (v', v'') \wedge \langle v : v', v'' \rangle \in \mathcal{D} \end{aligned}$$

In order to ensure isomorphism of spatio-temporally identical interactions, which only differ in when the initial and final episodes respectively start and end, it is necessary to represent all and only the germane relationships as given in equation 3.4. In terms of the interaction graph, this is expressed in terms of the above specification of the layer 3 nodes together with the following condition:

$$\forall v', v'' \in \mathcal{V}^2 : \{v', v''\} \notin \mathcal{E}_I \wedge \{v', v''\} \notin \mathcal{E}_F \Leftrightarrow \exists v \in \mathcal{V}^3 : \langle v : v', v'' \rangle \in \mathcal{D}$$

The above equation expresses the condition that whenever a pair of layer 2 nodes  $v', v''$  are both not corresponding to initial episodes or both not corresponding to final episodes, then the temporal relationship between them is specified by a layer 3 node  $v$ . Conversely, whenever a temporal relationship is specified by a layer 3 node  $v$ , where  $v$  relates a pair of layer 2 nodes  $v', v''$ , then  $v', v''$  are both not corresponding to initial episodes or both not corresponding to final episodes.

The graph based representation provides a computationally efficient alternative to the logic based representation expressed in the statement 3.5. While there is a repetition of region histories and episode variables in the logic based representation, the interaction graph avoids this repetition by uniquely representing each region history and episode just once. The graph based representation also enables a well defined and computationally efficient comparison of interactions by means of a similarity measure defined in Section 3.4. In the rest of this thesis, the terms interaction and interaction graphs are used interchangeably.

### 3.4 Comparing Interactions Spatio-Temporally.

Interactions may be related to each other with respect to their spatio-temporal relationships in several ways. Two interactions may be identical or even similar to each other.

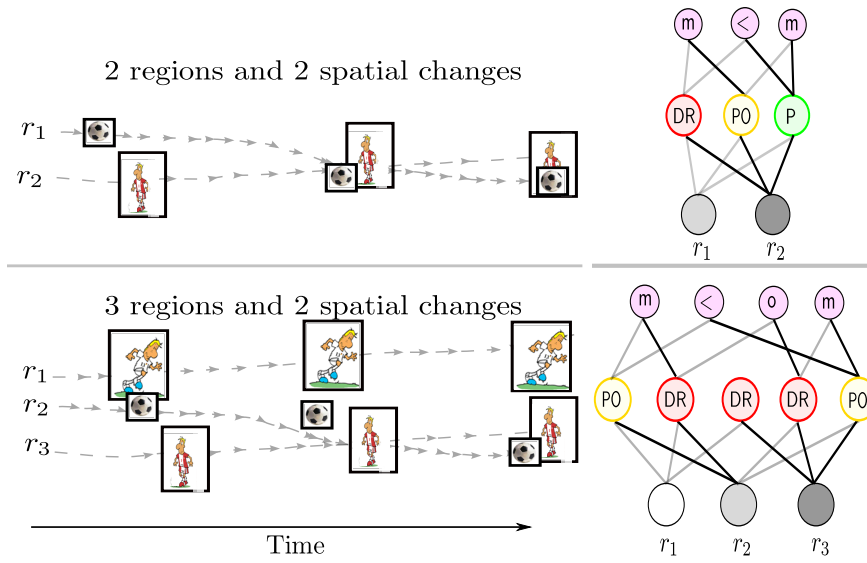


Figure 3.6: Two sub-interactions of the interaction in Fig. 3.4 are shown on the left. The corresponding interaction graphs for these sub-interactions are shown on the right. Note that these are sub-graphs of the interaction graph shown in Fig. 3.5.

An interaction may be a sub-interaction or a super-interaction of another. The following paragraphs describe the concepts involved in comparing interactions using interaction graphs.

**Identical Interactions.** Let  $g = (\mathcal{V}, \mathcal{D}, \psi_2, \psi_3, \mathfrak{R}, \mathfrak{N})$  and  $g' = (\mathcal{V}', \mathcal{D}', \psi'_2, \psi'_3, \mathfrak{R}, \mathfrak{N})$  be two interaction graphs. They are spatio-temporally identical if and only if there is a bijection  $\Psi : g \leftrightarrow g'$  such that

1. Correspondence wrt. edges:

$$\forall (v, w) \in \mathcal{V} : \langle v, w \rangle \in \mathcal{D} \Rightarrow \langle \Psi(v), \Psi(w) \rangle \in \mathcal{D}'$$

$$\forall (v', w') \in \mathcal{V}' : \langle v', w' \rangle \in \mathcal{D}' \Rightarrow \langle \Psi^{-1}(v'), \Psi^{-1}(w') \rangle \in \mathcal{D}$$

2. Correspondence wrt. node labels:

$$\forall v \in \mathcal{V}^2, \psi_2(v) = \psi'_2(\Psi(v)) \wedge \forall w \in \mathcal{V}^3, \psi_3(w) = \psi'_3(\Psi(w))$$

$$\forall v' \in \mathcal{V}'^2, \psi'_2(v') = \psi_2(\Psi^{-1}(v')) \wedge \forall w' \in \mathcal{V}'^3, \psi'_3(w') = \psi_3(\Psi^{-1}(w'))$$

In the rest of the thesis, we use  $g \simeq g'$  to say that two interaction graphs  $g$  and  $g'$  are isomorphic to each other.

**Sub-interactions.** The notion of sub-interactions is useful for defining a similarity measure between relations and for defining events and sub-events below. An interaction  $g'$  is a sub-interaction of another interaction  $g$ , if there is a sub-graph  $h \subseteq g$  such that  $g'$  and  $h$  are isomorphic.

In general, there are two kinds of sub-interactions along with their combinations that are possible: (i) one interaction can have fewer region histories; (ii) fewer qualitative spatial changes, or both. For example, consider the interaction in Fig. 3.4 with 3 region histories and 3 spatial changes. The corresponding interaction graph shown in Fig. 3.5. An interaction sub-graph and the corresponding interaction with one less region history (i.e. 2 region histories, 2 spatial changes) is shown at the top in Fig. 3.6. An interaction sub-graph and the corresponding interaction with one less spatial change (i.e. 3 region histories, 2 spatial changes) is shown at the bottom in Fig. 3.6.

**Frequency.** Another useful notion is the *frequency* of  $g'$  in  $g$  and is given by the number of unique sub-graphs  $h$  in  $g$  that are isomorphic to  $g'$ . An interaction  $g'$  is *sufficiently frequent* in  $g$  if its frequency is greater than a pre-defined threshold.

**Similarity between Interactions.** In order to compare any two interactions  $g$  and  $g'$  on the basis of their spatio-temporal relationships, an appropriate similarity measure between  $g$  and  $g'$  needs to be defined. If two interactions are spatio-temporally identical, then their respective interaction graphs will be isomorphic.

However, in order to compare spatio-temporally similar interactions, a suitable measure of similarity between their respective interaction graphs is needed. One way of measuring similarity between any two graphs (or more generally structures) is to represent them in terms of bag of sub-graphs (or substructures), so that if two graphs contain a similar set of sub-graphs, then we can expect the two graphs to be similar.

While this is a suitable approach in general [Deshpande et al., 2005], in the case of interaction graphs, most sub-graphs do not represent interactions and therefore may not form good features. Also, these redundant sub-graphs increase the dimensionality of the feature space, potentially add noise and are computationally inefficient as sub-graph isomorphism needs to be examined individually for each sub-graph.

Thus an efficient representation is adopted in this work, where each interaction is represented in terms of a fixed dictionary  $Gr = (v_1, \dots, v_i, \dots, v_n)$  of interaction graphs<sup>4</sup>.

<sup>4</sup>This dictionary is prepared offline by constructing the interaction graphs for all possible interactions between region histories, such that the number of region histories and the number of spatial changes are within a pre-determined bound.

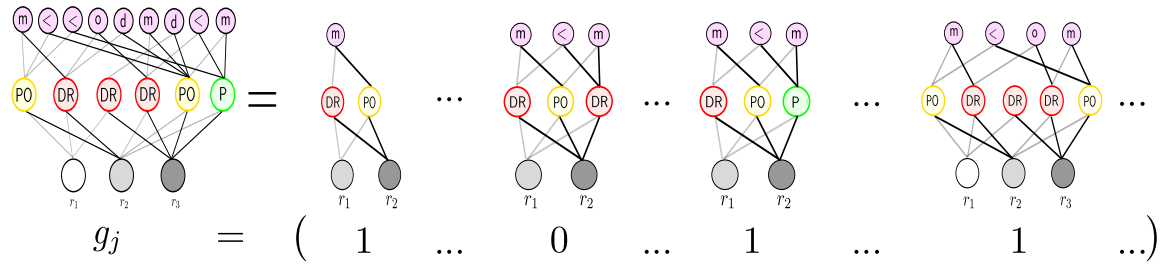


Figure 3.7: An interaction graph  $g_j$  (the same as in Fig. 3.5) is shown to be represented as a vector. An entry in this vector corresponds to the frequency with which a grapheme in  $Gr$  (clarified below) occurs in  $g_j$ . Just a few graphemes are shown for the sake of clarity. The sequences of dots are used to convey the presence of the rest of the graphemes and the corresponding entries in the vector.

These graphs in this dictionary are called graphemes. The representation of the interaction graph for the interaction in Fig. 3.4 in terms of graphemes is shown in Fig. 3.7.

Another way in which the dimensionality of the feature space is restricted is to constrain the number of region histories and the number of changes in spatial relationships by respective upper bounds. It is natural to expect that these numbers would typically be constrained by the domain under consideration. Thus by restricting the dictionary to those graphs that represent interactions and constraining the number of region histories and changes in spatial relationships, the dimensionality of the representation is reduced substantially.

Using this dictionary, an interaction graph  $g$  is re-represented as a histogram of length  $n$ :

$$\Phi(g) = [w_1 f_1, \dots, w_i f_i, \dots, w_n f_n]$$

The term  $f_i$  is the frequency with which a grapheme  $v_i \in Gr$  occurs in the interaction graph  $g$ . The weight  $w_i$  is an exponential function  $\exp(v_i)$  of the mean of the number of region histories and number of spatial changes represented in  $v_i$ . This serves the purpose of giving greater weights to larger graphs, as two graphs are more similar if they share a larger common sub-graph than a smaller one.

Using the representation described above, any two interaction graphs  $g_j$  and  $g_k$  may be re-represented as a bag of graphemes (BoG). The BoG kernel  $\mathcal{K}(g_j, g_k)$  measures the similarity between two graphs  $(g_j, g_k)$ , in terms of the extent to which they share common graphemes.

$$\mathcal{K}(g_j, g_k) = \langle \Phi(g_j), \Phi(g_k) \rangle = \frac{\sum_{i=1}^n w_i^2 f_{ji} f_{ki}}{(\sum_{i=1}^n (w_i f_{ji})^2 \sum_{i=1}^n (w_i f_{ki})^2)^{\frac{1}{2}}} \quad (3.6)$$

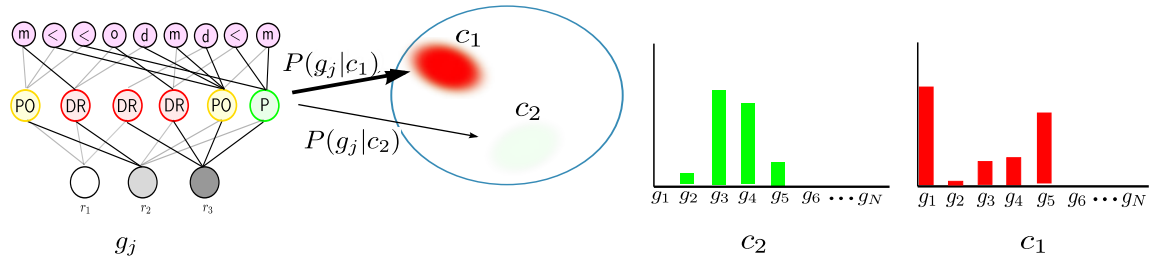


Figure 3.8: This figure illustrates that an interaction graph (from Fig. 3.5) referred to as  $g_j$  can belong to multiple classes with different probabilities  $P(g_j|c_1)$  and  $P(g_j|c_2)$ . The relative thickness of the arrows pointing from  $g_j$  to the two classes, is used to convey that the  $g_j$  is more likely given  $c_1$  than  $c_2$ . The class conditional probabilities for the graphs in a given bag  $\mathcal{B}$  are illustrated on the right.

**Density Estimation using the Similarity Measure.** The BoG based representation together with the BoG Kernel makes it possible to represent an interaction in a constrained high dimensional space, where similar interactions occupy a similar position in this high dimensional space. This representation naturally leads to the notion of clusters of similar interactions. When these clusters are infused with certain event-like properties, they are regarded as event classes, an idea that is central to the next chapter.

In this work, we assume that each cluster is generated by its own density function. We have explored parametric finite mixture modelling techniques such as the Gaussian Mixture Model (GMM) [Figueiredo and Jain, 2002], as they have been shown to be successful in many applications. However, we regard a non-parametric model as being more suitable for the following reason: when interaction graphs are represented as a vector, the mean of a set of vectors may not correspond to a valid interaction graph<sup>5</sup>.

Therefore, we model each cluster density using a non-parametric kernel density estimate. We apply a recent approach described in [Mallapragada et al., 2010] since this constructs an explicit probabilistic model for each cluster, unlike other non-parametric approaches, for example spectral clustering [Ng et al., 2001]. In this approach, given a bag  $\mathcal{B}$  of interaction graphs  $\{\dots, g_j, \dots\}$ , the kernel density estimate  $P(g_j|c_i)$  for a cluster  $c_i$  (belonging to a set of clusters  $\mathcal{C}$ ) is given by

$$P(g_j|c_i) = \sum_{g_k \in \mathcal{B}} q_k^i \mathcal{K}_d(g_k, g_j) \text{ where } \sum_{k=1}^{|\mathcal{B}|} q_k^i = 1$$

In the above equation,  $q^i = (q_1^i, \dots, q_{|\mathcal{B}|}^i)$  is regarded as the profile vector for cluster  $c_i$  and

<sup>5</sup>Since only a small subset of possible vectors actually correspond to meaningful interaction graphs.

$Q = (q_1, \dots, q_{|C|})$  as the profile matrix. We apply the procedure presented in [Mallapragada et al., 2010] to learn the profile matrix  $Q$  for a bag of interaction graphs  $\mathcal{B}$ . The RBF kernel  $\mathcal{K}_d$  [Scholkopf, 2001] in the above equation, is defined in terms of the BOG kernel in equation 3.6 as follows:

$$\mathcal{K}_d(g_j, g_k) = \exp\left(-\frac{(\mathcal{K}(g_j, g_j) + \mathcal{K}(g_k, g_k) - 2\mathcal{K}(g_j, g_k))}{2\sigma^2}\right) \quad (3.7)$$

This idea is illustrated in Fig. 3.8, where an event graph is shown as belonging to two different classes with different probabilities. The distribution for each class is also shown. We will see in chapter 4 that having an explicit probabilistic model for each cluster without a parametric form is useful for defining and learning event classes.

### 3.5 Interactions Embedded in Space and Time

This chapter has so far described interactions as schematic entities i.e. without any reference to any observed objects occupying regions in space and time. This section focusses on the relationship between interactions and their embeddings in space and time.

When an object travels through space and time, the entire trajectory traced by the regions occupied by the object through time is called its track. *Tracklets* are regarded as contiguous parts of tracks. Any set of tracklets that are likely to have the spatio-temporal relationships characterizing an interaction is regarded as an *embedding* of this interaction. More formally, an embedding  $\varepsilon$  of an interaction graph  $g$  with a particular subset of tracklets is given by the mapping  $\xi(g) = \varepsilon$ .

The following two questions are particularly of interest in exploring the relationship between interactions and their embeddings (especially for the sake of learning described in chapter 4).

1. Given a set of tracklets, what is the most likely interaction characterizing it?
2. Given an interaction, what is the likelihood of a particular embedding ?

#### 3.5.1 Most Likely Interaction Graph for a Set of Tracks

It is assumed that a set of tracks for a video are given by  $\mathcal{T} = \{\tau_1, \tau_2, \dots, \tau_n\}$ , where each track  $\tau_i$  consists of a sequence of blobs  $\{o_i^1, \dots, o_i^p\}$ . Three tracks  $\tau_1, \tau_2, \tau_3$  and the interaction between their corresponding region histories  $r_1, r_2, r_3$  are illustrated in Fig.



3.4<sup>6</sup>.

In order to hypothesize the most likely interaction for this set of tracks  $\mathcal{T}$ , it is required to obtain the most likely qualitative spatial states between those tracks at any time instant. In other words, for each pair of tracks, a likely sequence of qualitative spatial relationships needs to be induced.

The qualitative spatial relationships described above have traditionally been computed by using techniques that are defined in a Boolean fashion solely in terms of the connectivity of the spatial extents [Cohn and Hazarika, 2001]. The main limitation of using connectivity is the rapid flipping of spatial relationships, as a result of the slightest possible movement, at the boundaries of spatial change. The application of connectivity for video data, where the jitter of bounding boxes is not uncommon, results in unstable spatial relationships between blobs, as confirmed by experiments in chapter 6.

In order to address this problem, firstly a simple Bayesian model is proposed to compute the probability of a spatial relationship given a pair of bounding boxes in a particular frame. This model is then incorporated into a HMM which overlays a temporal model for smoothing the outcome of the Bayesian Model. These two solutions are described below.

### Qualitative Relationships at Each Frame

In line with common practice, we abstract the spatial extents of objects to their bounding boxes. Let  $o_i^t, o_j^t$  be the bounding boxes for a pair of blobs at time  $t$ , belonging to the respective pair of tracks  $\tau_i = \{o_i^{t^h}, \dots, o_i^{t^H}\} \in \mathcal{T}$  and  $\tau_j = \{o_j^{t^h}, \dots, o_j^{t^H}\} \in \mathcal{T}$ , where it is assumed that both blobs are observed together for a certain time interval. Let  $s$  be any one of the spatial relations  $\{DR, PO, P\}$ . The following paragraphs define the probability  $P(s|\delta(o_i^t, o_j^t))$ , of a spatial relationship  $s$ , given the dissimilarity  $\delta(o_i^t, o_j^t)$  between a pair of bounding boxes  $o_i^t, o_j^t$ , at time  $t$ . This dissimilarity measure  $\delta(o_i^t, o_j^t)$  is defined as follows:

$$\delta(o_i^t, o_j^t) = \frac{d(c_2) - d(c_1)}{\min(d(o_i^t), d(o_j^t))} \quad (3.8)$$

This measure is defined with the help of two circles  $c_1$  and  $c_2$ , that are illustrated in Fig. 3.9. The first circle  $c_1$  is the largest circle in the intersection of two regions  $o_i^t$  and  $o_j^t$ . The second circle  $c_2$  is the smallest circle that connects two disconnected regions  $o_i^t$  and  $o_j^t$ . The diameters of the two circles are given by  $d(c_1)$  and  $d(c_2)$  respectively. Finally  $d(o_i^t)$  and  $d(o_j^t)$  are the diagonals of the bounding boxes  $o_i^t$  and  $o_j^t$  respectively.

The dissimilarity measure  $\delta(o_i^t, o_j^t)$  is positive when  $o_i^t, o_j^t$  are disconnected, zero when

<sup>6</sup>Note that the object type is not used to represent the interaction between these three tracks. The object types are used only for illustrating the interaction with a real life example.

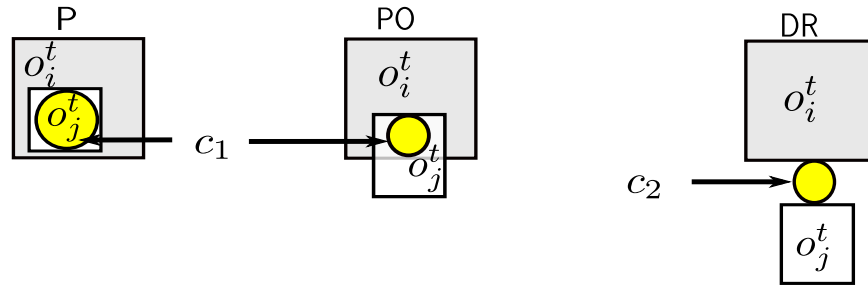


Figure 3.9: Two circles  $c_1$  and  $c_2$  for formulating the notion of region based distance between two objects  $o_i^t$  and  $o_j^t$  at a time instant  $t$ . The first circle  $c_1$  is the largest circle in the intersection of two regions  $o_i^t$  and  $o_j^t$ . The second circle  $c_2$  is the smallest circle that connects two disconnected regions  $o_i^t$  and  $o_j^t$ .

they touch and goes further negative as the relations change from DR to PO towards P, as shown in Fig. 3.10. The denominator  $\min(d(o_i^t), d(o_j^t))$  makes the dissimilarity  $\delta(o_i^t, o_j^t)$  independent of the sizes of  $o_i^t, o_j^t$ , since the maximum value of  $d(c_1)$ , is equal to  $\min(d(o_i^t), d(o_j^t))$  multiplied by a constant, which is the square root of two.

In order to compute the probabilities  $P(s|\delta(o_i^t, o_j^t))$  for each relationship  $s$ , Bayes rule is applied:

$$P(s|\delta(o_i^t, o_j^t)) = P(\delta(o_i^t, o_j^t)|s)P(s)$$

The prior probabilities  $P(s)$  for each spatial relationship is the proportion of the occurrences of that spatial relationship, to that of all the spatial relationships, in the data. The probabilities  $P(\delta(o_i^t, o_j^t)|s)$  for each of the spatial states {DR, PO, P} are modelled by the corresponding logistic functions given below. The three states and their logistic functions are illustrated in Fig. 3.10.

$$P(\delta(o_i^t, o_j^t)|\text{DR}) = \beta_{11}(1 - (1 + e^{\beta_{12}(\delta(o_i^t, o_j^t) - \beta_{13})})^{-1}), \beta_{12} > 0 \quad (3.9)$$

$$P(\delta(o_i^t, o_j^t)|\text{PO}) = \beta_{21}(1 - (1 + e^{\beta_{22}\|\delta(o_i^t, o_j^t) - \beta_{23}\|})^{-1}), \beta_{22} < 0 \quad (3.10)$$

$$P(\delta(o_i^t, o_j^t)|\text{P}) = \beta_{31}(1 - (1 + e^{\beta_{32}(\delta(o_i^t, o_j^t) - \beta_{33})})^{-1}), \beta_{32} < 0 \quad (3.11)$$

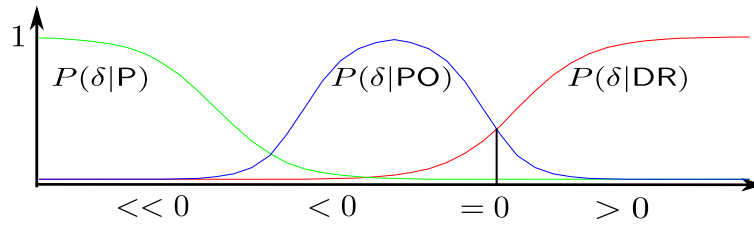


Figure 3.10: Logistic functions for the three spatial relationships  $\{P, PO, DR\}$  respectively. The *horizontal axis* is the region based dissimilarity measure (given simply by  $\delta$ ) and the probability distribution for each of the three spatial states are shown above this axis.

### A HMM for Smoothing Sequences of Spatial Relationships

While the above Bayesian formulation is superior to traditional point set intersection techniques in that it incorporates priors on spatial relations, it still does not use the temporal information from other nearby frames that are available in videos. The following HMM based formulation augments the Bayesian model above by overlaying a temporal model that incorporates additional information available in a video sequence, in order to make better predictions of qualitative spatial relationships for each video frame. In this manner the temporal model can be used to smooth the rapidly flipping transitions between spatial relations that arise from visual noise and predict a more stable sequence of qualitative spatial relationships between the corresponding blobs for a pair of observed tracks.

For each pair of blobs, an observed sequence of dissimilarities between them is assumed to be generated by a corresponding Markov chain of hidden qualitative spatial states and their respective observation models. More formally, let  $\delta(\tau_i, \tau_j) = (\delta_{t'}, \dots, \delta_t, \dots, \delta_{t'+k})$  be an observed sequence of dissimilarities between a pair of tracks  $\tau_i \in \mathcal{T}$  and  $\tau_j \in \mathcal{T}$  during the maximal interval  $(t', t' + k)$  (where  $t' = \max(t_i^h, t_j^h)$  and  $t' + k = \max(t_i^H, t_j^H)$ ) in which they both are present. Here  $\delta_t$  stands for the dissimilarity  $\delta_t(o_i^t, o_j^t)$  between the corresponding pair of blobs  $o_i^t \in \tau_i$  and  $o_j^t \in \tau_j$ , at time  $t$ . Let  $S(\tau_i, \tau_j) = (s_{t'}, \dots, s_t, \dots, s_{t'+k})$  be the corresponding hidden sequence of spatial states between  $\tau_i, \tau_j$ . Here  $s_t$  stands for the spatial relationship  $s_t(o_i^t, o_j^t) \in \mathfrak{R}$  at time  $t$ . The HMM that models the joint probability distribution of the observed and hidden states is given by the tuple  $\theta = (\mathfrak{R}, A, B, \pi)$ , where

1.  $\mathfrak{R} = \{DR, PO, P\}$  are the states of the HMM as illustrated by circles in Fig. 3.11.
2.  $A = \{\dots, a_{ij}, \dots\}$  is the state transition matrix in which each entry  $a_{ij}$  represents the probability of transition from state  $st_t = s_i \in \mathfrak{R}$  to  $st_{t+1} = s_j \in \mathfrak{R}$ . Only those transitions that are physically possible (as shown by the arrows in Fig. 3.11) have



Figure 3.11: The states of the HMM are shown. The allowable state transitions are constrained by the conceptual neighbourhood graph for the spatial states  $\{P, PO, DR\}$ .

non-zero transition probabilities.

3.  $B = \{b_i(\delta_t)\}$  is the observation model, where  $b_i(\delta_t)$  represents the probability  $P(\delta_t | st_t = s_i)$  of observing a region based distance  $\delta_t$  while being in state  $st_t = s_i \in \mathfrak{R}$ . The observation models for DR, PO, P are the logistic functions defined in equations 3.9, 3.10 and 3.11 respectively. These are illustrated in Fig. 3.10.
4.  $\pi = \{\pi_i\}$  is the initial state distribution, where  $\pi_i$  represents the probability of state  $st_t = s_i \in \mathfrak{R}$  being the initial state.

The above model is trained with a dataset of sequences of region based distances (between tracks) that are manually annotated with the subjectively correct spatial relations. The transition probabilities are learned from the statistics of bi-grams of spatial relations in the annotation. Finally, the parameters of the observation models are learned using maximum likelihood estimation using only the data corresponding to each of the states.

With a trained HMM, it is possible to predict the most likely sequence of spatial relationships  $\hat{S}(\tau_i, \tau_j)$ , given a sequence of observed distances  $\delta(\tau_i, \tau_j)$  and the HMM model  $\theta$  for any pair of tracks  $\tau_i, \tau_j$ . For each pair of tracks<sup>7</sup>, a Viterbi decoder is used to find the most likely sequence of spatial relationships.

$$\hat{S}(\tau_i, \tau_j) = \arg \max_{S(\tau_i, \tau_j)} P(S(\tau_i, \tau_j) | \delta(\tau_i, \tau_j), \theta)$$

The standard implementation of a Viterbi decoder [Rabiner, 1989] provides the 1-best Viterbi path corresponding to presumably the most likely sequence of spatial relationships. However, an alternate way is to obtain  $k$ -best Viterbi paths and use other contextual information to disambiguate between them. This idea will be used later on in chapter

<sup>7</sup>A reasonable assumption to make is that the spatial relationships for each pair of states can be computed independently of other pairs.

4. The  $k$ -best paths are obtained by finding all possible ways to reach a state, and then picking those states corresponding to the  $k$  highest probabilities.

While this idea has been explored as early as 1993 [Rabiner and Juang, 1993], standard implementations do not encode this idea. We therefore used our own naive implementation to obtain the  $k$ -best sequences of spatial relationships. Though this required  $k^2$  times the original cost, we maintain efficiency by using only the top 2 results. This aspect is described in chapter 4.

Once the  $k$  most likely sequences of qualitative spatial relationships between each pair of tracks in the set  $\mathcal{T}$  are induced, the corresponding episodes can be obtained and the interaction and its corresponding interaction graph can be obtained as described in Section 3.3.

### 3.5.2 Likely Embeddings of an Interaction Graph

The previous section described a way of inducing the  $k$  most likely interactions (or interaction graphs) from an observed set of tracks. This section considers the inverse problem of determining how likely it is that a set of tracks form an embedding of a given interaction.

It is assumed that embeddings should be more likely if they are prototypical of the spatial relations i.e. pairs of tracks for which the sequence of distances are perceptually clearer given the sequence of spatial relationships. For example, consider the interaction graph at the top right in Fig. 3.6 and the two possible embeddings in Fig. 3.12. The embedding at the top is more prototypical of the spatial relations in Fig. 3.6, than the one at the bottom, because in the sequence of region based distances for the one at the bottom, the spatial relations hold more clearly, than the one at the top.

The measure of typicality is given by the probability  $P(\delta(\tau_i, \tau_j) | S(\tau_i, \tau_j), \theta)$  of a sequence of region based distances  $\delta(\tau_i, \tau_j)$  given a sequence of qualitative spatial relationships  $S(\tau_i, \tau_j)$  and the HMM model  $\theta$ . The probability  $P(\delta_t(\tau_i, \tau_j) | S_t(\tau_i, \tau_j), \theta)$  at time  $t$  is given by the respective observation models depending on the particular spatial state at this time  $t$ , as given in equations 3.9, 3.10 and 3.11. From the conditional independence assumptions encoded in the HMM, it follows that:

$$P(\delta(\tau_i, \tau_j) | S(\tau_i, \tau_j), \theta) = \prod_t P(\delta_t(\tau_i, \tau_j) | S_t(\tau_i, \tau_j), \theta) \quad (3.12)$$

The probability of a particular embedding  $\varepsilon$  of  $g$  measures the degree to which each pair of tracks in this embedding is prototypical of the spatial relationships given by the

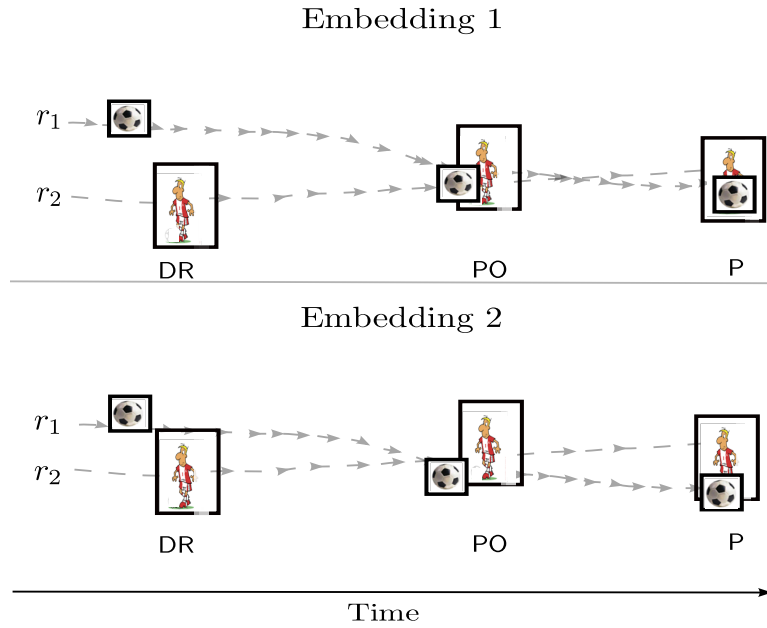


Figure 3.12: Two embeddings for the interaction graph at the top right of Fig. 3.6 are shown. Embedding 1 is more prototypical than embedding 2 for this graph, since relationships between the region histories  $r_1$  and  $r_2$  are *perceptually prototypical* of the sequence of spatial relationships (DR, PO, P), as represented by this interaction graph.

interaction graph. Assuming independence for the sake of tractability, this is expressed as a product of the observation probabilities given in equation 3.12 for the respective pair of tracks.

$$P(\varepsilon|g) = \prod_{(\tau_i, \tau_j) \in \varepsilon} P(\delta(\tau_i, \tau_j) | S(\tau_i, \tau_j), \theta) \quad (3.13)$$

### 3.6 Conclusion

In this chapter, we introduce our definition of an interaction, which is in return used to describe all the fundamental spatio-temporal relationships between a set of objects (represented as regions). We represent interactions by interaction graphs. Interaction graphs provide a computationally efficient and a well defined way way of comparing interactions and clustering them. Chapter 3 also introduces probabilistic relationships between interactions and the embedding of such, in a set of tracks.

# Chapter 4

## Learning Events from Activities

---

### 4.1 Introduction

We think of activities for a domain as being composed of semantically significant occurrences that we refer to as *events*. We think of an event as being composed of a set of tracklets. We regard events as belonging to a finite set of *event classes*. Thus activities are closely associated with a set of event classes. For example kitchen activities are closely associated with event classes such as making hot drinks, cakes and aircraft apron activities with unloading, refuelling etc.

We address our unsupervised event learning task in this chapter. In this task, activities are observed for an extended period and recorded as a video. The goal is to learn the associated event classes and events that are most likely given the observations in the video.

This task is challenging due to two reasons. The first is the complexity of the solution space which involves a combination of all possible candidate event classes and all possible candidate events. The second is due to the presence of three *complicating factors*. The first of these factors is the possibility of *overlap* between events. The second is the possible presence of *coincidences*. These are occurrences that *do not* belong to event classes. The third is possibility of noise in observation and we refer to this as *observation noise*. They arise due to complexities in image processing.

In order to address these challenges, we formulate a *probabilistic generative process* which characterizes likely solutions of event classes and events using *three event-like*

*properties*. These properties are made precise in the rest of the chapter. The generative process also incorporates a way of modelling the three complicating factors. An efficient search procedure is used to search for the most probable configurations of the generative process which could have given rise to the observations in a video.

The following Section 4.1.1 starts with an informal introduction of the framework with some illustrations. Section 4.2 describes the three *event-like* properties that can be used to characterize the original set of event classes and events. Section 4.3 models these three event-like properties and the complicating factors within the framework of a probabilistic generative process.

An efficient search procedure for the most likely configurations of the generative process is discussed in Section 4.4. Section 4.5 describes the concept of *interactivity*, which is an important property that is used to characterize event classes. The final Section 4.6 concludes this chapter.

### 4.1.1 Overview of the Unsupervised Event Learning Framework

This section presents an overview of various aspects of our unsupervised event learning framework. These are presented in an informal manner with the help of illustrations. These notions are defined more formally in later sections of this chapter.

**Event Classes.** We assume that human activities for a certain domain are strongly associated with significant occurrences that can be classified into one or more semantically meaningful classes. These significant occurrences are referred to as *events* and the semantically meaningful classes as *event classes*. Several events for each of the classes are expected to occur over an extended period of activities for a domain.

Some event classes are regarded as being more probable than others. Moreover, some interactions are more probable than others for an event class. For example, coffee making may be regarded as a more likely event class than washing up dishes for activities in a certain kitchen. Certain types of interactions (between cups, spoons etc.) may be considered more likely than others, given a coffee making event class.

We model an *event class* as a discrete probability distribution over a finite set  $\mathcal{G}$  of interaction graphs. The set  $\mathcal{G}$  represents *all* possible interactions up to a fixed number of region histories and changes in spatial relationships<sup>1</sup>. Two event classes  $c_1$  and  $c_2$

---

<sup>1</sup>The upper bound on the number of region histories and changes in spatial relationships is fixed as 7 and 15 respectively in the experimental section of this thesis. The reason is that typically the size of event graphs in real world activities are limited by these numbers.



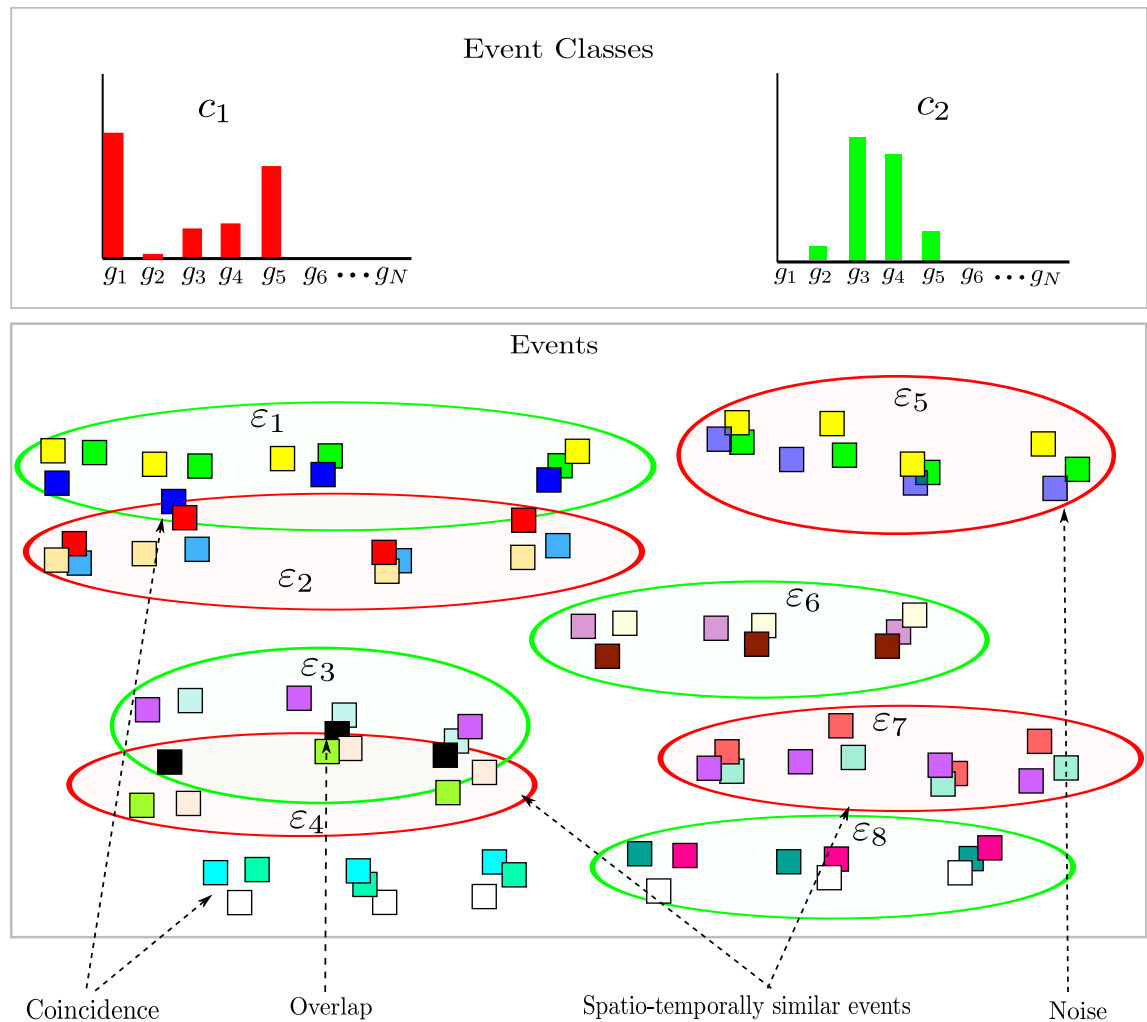


Figure 4.1: Activities for a domain over an extended period of observation is illustrated. The two key aspects of our model – event classes and event cover – are shown. Event classes  $c_1$  and  $c_2$  are modelled as probability distributions over a finite set of interaction graphs  $\mathcal{G} = \{g_1, \dots, g_N\}$ . Since it is not easy to display the interacting tracks, we show it schematically. Each event (e.g.  $\epsilon_1$ ) is represented as an interaction between a set of tracklets. Each tracklet is depicted by a sequence of regions with a distinct colour.

are shown in Fig. 4.1 with red and green colour respectively. They are illustrated as specifying a probability distribution over the set of event graphs  $\mathcal{G} = \{g_1, \dots, g_N\}$ .

**Events.** While event classes are abstractions, events are concrete occurrences in space and time. An event is characterized in three ways. First of all, it is simply a set of tracklets. Secondly, it is also an embedding of an interaction graph in  $\mathcal{G}$ . Finally, this interaction graph has to be likely with respect to an event class.

Events  $\varepsilon_1$  to  $\varepsilon_8$ , corresponding to the two example event classes  $c_1$  and  $c_2$  are illustrated in Fig. 4.1. The class membership of the events are given by associated colours which are either red or green. The same set of tracks may be split into two different sets of tracklets corresponding to two different events. This is illustrated in Fig. 4.1, where  $\varepsilon_1$  and  $\varepsilon_5$  share the same set of tracks.

**Event Cover.** The composition of the tracks of an activity in terms of its constituent events is given by what we refer to as its *event cover*. An event cover is a set of events. An example of an event cover with constituent events  $\varepsilon_1$  to  $\varepsilon_8$  is shown in Fig. 4.1.

**Overlap.** In order to model activities in the real world, it is assumed that events within an event cover may *overlap* with one another. This is shown between events  $\varepsilon_3$  and  $\varepsilon_4$ , which share an overlapping tracklet. In a real world example of aircraft apron activities, events belonging to the unloading and bridge-on-off event classes, share the tracklet that corresponds to a plane.

**Coincidences.** In the case that the union of all tracklets that belong to events is equal to the set of tracks, then the event cover is *complete* i.e. all parts of the all tracks are in the cover. Otherwise, we call the rest of the *interacting* uncovered tracks *coincidences*. We emphasize that the occurrences of these interactions are called coincidences, because they happen by chance, *in contrast to events that belong to event classes*. In other words, events are planned for activities and coincidences are not.

Coincidences can be of two types. First, they can occur between tracklets belonging to the same or different events. A coincidence that is the result of a change of spatial relationship between two different events  $\varepsilon_1$  and  $\varepsilon_2$  is illustrated in Fig. 4.1. Second, they can occur between other tracklets. This is illustrated in Fig. 4.1, by tracklets around which there is no ellipse. A real world example of a coincidence is an occasional change of spatial relationships that some times happens between a plane puller and a trolley.

**Observation Noise.** Finally, it is assumed that an event could be corrupted with *observation noise*, which involves a random change in spatio-temporal relationships. We emphasize that observation noise arises *only* due to complexities in processing the observed data. A noisy embedding of an event class *tends* not to be prototypical (in the sense described in Section 3.5), of any of the interaction graphs that are likely for an event class. Note that while a coincidence does not belong to the event cover, a noisy event is considered to be a part of the event cover.

A noisy event  $\varepsilon_5$  is shown in Fig. 4.1. Even though  $\varepsilon_5$  belongs to class  $c_1$ , the final spatial state between the tracklets in  $\varepsilon_5$  is shown to be different from other prototypes ( $\varepsilon_2, \varepsilon_4, \varepsilon_7$ ) of the same class  $c_1$ , as it has been corrupted by noise. For video data, noise arises due to complexities in image processing e.g. misplaced detections or jitter of bounding boxes.

**Unsupervised Event Learning.** In the *unsupervised event learning* framework considered here, a set of tracks that correspond to extended periods of observation of activities for a certain domain are assumed to be given. The original event classes and event cover associated with the activities for a domain are assumed to be unknown. The aim is to discover the most likely event classes and the event cover for the given set of tracks.

The following section elucidates the three event-like properties used to characterize a likely set of event classes and an event cover respectively. These are first described in a simple setting, where an event class is represented by a single prototypical event graph and the three complicating factors are absent. Section 4.3 extends these three properties and also models the complicating factors in a probabilistic generative process.

## 4.2 Event-like Properties in a Simple Setting

In a simple setting, event classes are very simply represented with just one interaction. Moreover, the three complicating factors are assumed to be absent. A simple setting is illustrated in Fig. 4.2.

We characterize event classes in the simple setting with three *event-like properties*. The first two – size and interactivity – are desirable properties of event classes. The third – frequency – is a desirable property of an event cover of the observed data given a set of event classes.<sup>2</sup>

<sup>2</sup>In the probabilistic extension described later in Section 4.3, the first two properties are used to characterize a prior distribution over event classes. The third property is used to characterize a likelihood term.

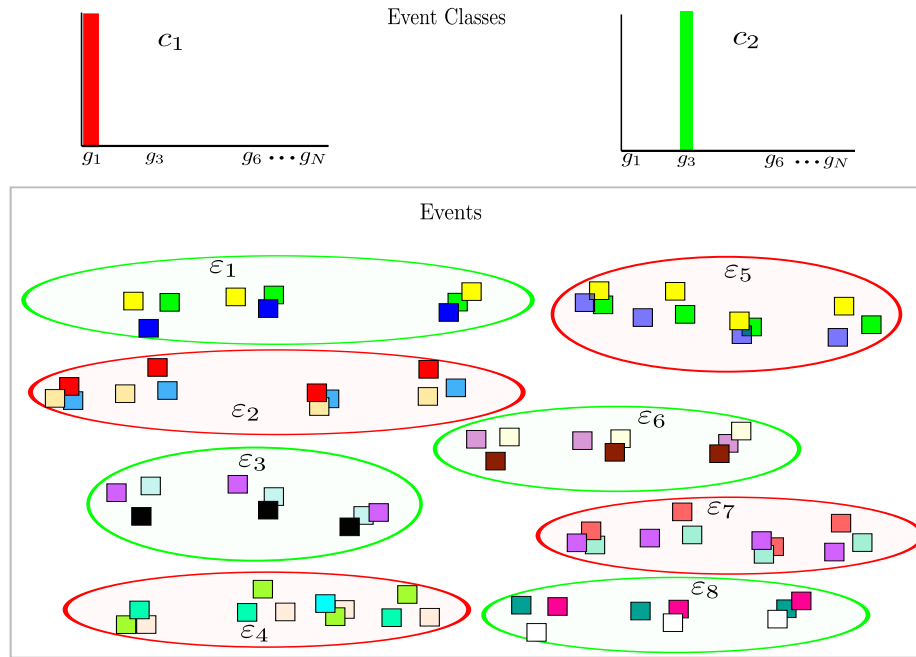


Figure 4.2: A simplified setting where event classes are represented as a single prototype. For the above illustration,  $P(g_1|c_1) = 1$  and  $P(g_3|c_2) = 1$ . The three complicating factors – overlap, coincidence and noise – are absent. These aspects stand out on comparison with the general setting as illustrated in Fig. 4.1.

**Size of the Interactions.** We characterize event classes with larger interaction graphs than smaller ones. The size of an interaction graph is the mean of the number of region histories and changes in spatial relationships.

We regard size as a desirable property of event classes, since real world activities (e.g. aircraft apron) are generally planned in terms of larger units (e.g. unloading), than smaller ones (e.g. trolley attaches to a loader). This property is intuitive from a Minimum Description Length (MDL) [Grünwald, 2005] perspective, since larger interactions would tend to explain more of the observations than smaller ones.

**Interactivity.** We characterize event classes with interaction graphs that represent actively engaging objects<sup>3</sup>. *Interactivity* is a measure of how actively a set of objects are engaged with each other, as further motivated and made precise in Section 4.5.

We regard interactivity as a desirable property of event classes, because many interesting events are composed of actively engaging objects, e.g. a subset of players who are passing the ball in a football game, rather than all other players. This property is also intu-

<sup>3</sup>Rather than some objects being passive while others are actively changing their spatio-temporal relationships with respect to each other.

itive from an MDL perspective, since the event classes with higher interactivity and their associated event covers would tend to describe the more dense part of the data (with a lot of spatio-temporal relationships caused by actively engaging objects). That is, they tend to have a higher descriptive power, rather than those event classes with low interactivity, which would tend to describe the sparser parts of the activities.

**Frequency in the Event Cover.** The third property is of a covering of the observed data, given a set of event classes. An event cover of the observed data is regarded in this work, as being well characterized by a set of event classes, if the event cover has frequent embeddings of the event classes (i.e. their respective prototypical interactions) in the data.

We regard frequency as a desirable property of a covering of the data given a set of event classes, because events are expected to re-occur frequently for extended periods of observation<sup>4</sup>. This property is also intuitive from an MDL perspective since an event cover of the observed data is more likely given event classes if the regularities (in this case through frequent occurrence) in the data are explained by the model.

**Combination of the Event-Like Properties.** While each of these three event-like properties can be used to characterize likely event classes and event covers, we consider them more effective in combination. This is due to the following reason. It is expected that embeddings of interactions, which are large and interactive, also tend to be frequent, only if they are events. In other words, it is assumed that coincidences do not tend to have the property implied by the combination of these three event-like properties.

### 4.3 A Generative Process for a Complex Setting

In a complex setting, it is assumed that event classes represent spatio-temporally similar ways of performing some task<sup>5</sup>. Moreover, the three complicating factors are assumed to be present. A complex setting was illustrated in Fig. 4.1.

We introduce a probabilistic generative process in order to model a more general notion of event classes and the three complicating factors in one integrated framework. The generative process consists of two components. The first is an *event model* that generalizes the notion of event classes as a probability distribution over interaction graphs. The second component is an *activity graph* used to model the three complicating factors.

---

<sup>4</sup>This follows from the assumption that many of the interactions in activities are associated with a set of event classes.

<sup>5</sup>In contrast to a simple setting, where events belonging to an event class were assumed to be spatio-temporally identical.

The complex setting with an event model and an activity graph is illustrated in Fig. 4.3. Throughout this chapter, we will use Fig. 4.3 as an illustration of our framework.

## Event Model

An event model is defined in terms of event classes, where each class is a probability distribution over  $\mathcal{G}$ , which is a finite set of interaction graphs. Two event classes  $c_1$  and  $c_2$  and their respective distributions over a set of graphs  $\mathcal{G} = \{g_1, \dots, g_N\}$  are shown in Fig. 4.3.

More formally, it is supposed that for activities of a domain, there is an underlying *event model*. We refer to this event model as  $\Theta$ . This model specifies a probability distribution  $P(X = g_j)$  or simply  $P(g_j)$ . Here  $X$  is a random variable which can be assigned to an interaction graph  $g_j \in \mathcal{G}$ , where  $\mathcal{G}$  is a finite set of interaction graphs, as introduced above.

The probability distribution  $P(g_j)$  is further expressed as a mixture of distributions. We adopt the non-parametric model described in Section 3.4 to model the mixture densities. This mixture is associated with a finite set of event classes  $\mathcal{C} = \{c_1, \dots, c_p\}$ . Each event class  $c_i \in \mathcal{C}$  is associated with a probability  $P(c_i)$ . Each interaction graph  $g_j \in \mathcal{G}$  is associated with an event class  $c_i \in \mathcal{C}$  with probability  $P(g_j|c_i)$ .

Thus  $P(c_i)$  and  $P(g_j|c_i)$  can be interpreted as the parameters of the event model  $\Theta$ . Henceforth, we will express them as  $P(c_i|\Theta)$  and  $P(g_j|c_i, \Theta)$ . This is because, we would like to explicitly characterize the dependency of the mixture distribution on  $\Theta$ , *since the underlying model  $\Theta$  is itself allowed to vary in this work, as given below*<sup>6</sup>.

## Activity Graph

An activity graph is an intermediate representation between an event model and an observed set of tracks, as shown in Fig. 4.3. The activity graph is shown to be composed of component sub-graphs ( $\lambda_1$  to  $\lambda_8$  in Fig. 4.3), each of which maps to an interaction graph in  $\mathcal{G}$ . Moreover, they represent interactions between region histories, which correspond to tracks. The correspondence is shown by the use of the same colour for a track and the layer 1 node of the activity graph, as shown in Fig. 4.3.

More formally, it is supposed that for a given set of tracks  $\mathcal{T}$ , there is an underlying three layered activity graph  $\Lambda$ . This graph specifies the spatio-temporal relationships

<sup>6</sup>As described further below, the goal of unsupervised event learning is to find the optimal event model and activity graph given a set of tracks, by searching through possible event models and activity graphs.

between a set of region histories  $\mathcal{R}_{\mathcal{T}}$ , that are represented in the layer 1 nodes, such that there is a bijective mapping between  $\mathcal{R}_{\mathcal{T}}$  and  $\mathcal{T}$ .

The activity graph  $\Lambda$  is composed of component sub-graphs  $\{\dots, \lambda_k, \dots\}$ , where each  $\lambda_k$  is isomorphic to an interaction graph in  $\mathcal{G}$ . Moreover  $\lambda_k$  represents an interaction between a subset of region histories in  $\mathcal{R}_{\mathcal{T}}$ .

Finally, the component sub graphs of the activity graph are partitioned into two parts:  $\Lambda = \Lambda_1 \cup \Lambda_2$ . The precise reason for partitioning the activity graph is described further below. Henceforth, we use *activity graph* to mean a partitioned activity graph. We refer to  $\Delta$  as the set of all possible activity graphs for tracks  $\mathcal{T}$ .

The above specification of an activity graph  $\Lambda$  with partitions  $\Lambda_1$  and  $\Lambda_2$  is expressed as:

$$\Lambda \equiv \{\lambda : \exists g_j \in \mathcal{G} \wedge \lambda \simeq g_j\} \text{ such that } \{r : \exists \lambda \in \Lambda \wedge r \text{ is a layer 1 node of } \lambda\} = \mathcal{R}_{\mathcal{T}}$$

$$\Lambda = \Lambda_1 \cup \Lambda_2 \text{ and } \Lambda_1 \cap \Lambda_2 = \emptyset$$

In the above expressions,  $\lambda \simeq g_j$  is used to denote that  $\lambda$  is isomorphic to  $g_j$ .

**Relationship to Event Classes.** An activity graph is closely related to an event model. Given an event model  $\Theta$ , the two parts  $\Lambda_1$  and  $\Lambda_2$  become semantically significant in the following way.

We regard the first part  $\Lambda_1$  as the set of *event graphs*. An event graph  $\lambda \in \Lambda_1$  is isomorphic to an interaction graph  $g_j \in \mathcal{G}$ . We regard the second part  $\Lambda_2$  as the set of *coincidental interaction graphs*. Event graphs are more likely with respect to an event model than the coincidental interaction graphs. The likelihood of an activity graph given an event model is expressed in terms of the event graphs  $\Lambda_1$ , further below. According to this formulation, those partitions  $(\Lambda_1, \Lambda_2)$  that results in more likely  $\Lambda_1$ , are considered as more likely activity graphs  $\Lambda$ .

**Relationship to Tracks.** An activity graph is related to a set of tracks in the following way. First of all, each layer 1 node of the activity graph maps to a corresponding set of tracks (from the above definition of an activity graph).

Secondly, each component graph  $\lambda$  of the activity graph  $\Lambda$ , can be mapped to a subset of tracklets, such that these tracklets represent a possible embedding of  $\lambda$ . This relationship is illustrated in Fig. 4.3.

Finally, the partition of the activity graph into event graphs and coincidental interaction graphs induces a corresponding partition of the tracks into sets of tracklets represent-

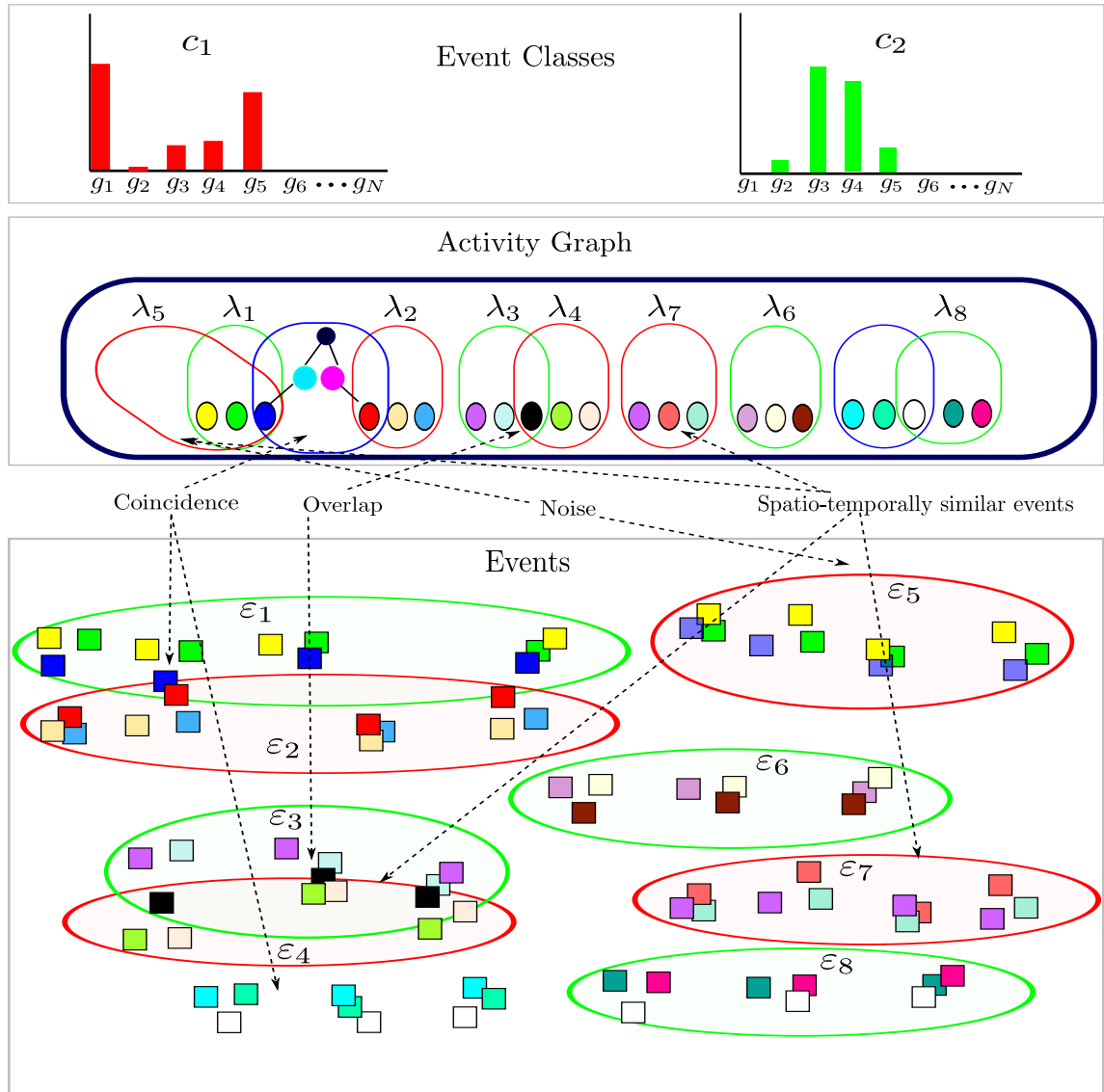


Figure 4.3: An activity graph is shown as an intermediate representation between event classes and a set of tracks. The event graphs  $\lambda_1$  to  $\lambda_8$  are the event graphs whose corresponding embeddings are events from  $\epsilon_1$  to  $\epsilon_8$ . The two coincidental interaction graphs are surrounded by blue ovals and they correspond to the two coincidences as shown. The coincidental interaction between a region history in  $\lambda_1$  and another region history in  $\lambda_2$  is shown with the help of two layer 2 nodes and one layer 3 node respectively. The layer 1 nodes of the event graphs are coloured with the colours of the tracklets for the corresponding events, in order to illustrate this mapping. The event graphs  $\{\lambda_2, \lambda_4, \lambda_5, \lambda_7\}$  are likely given class  $c_1$  and are therefore surrounded by red ovals. The event graphs  $\{\lambda_1, \lambda_3, \lambda_6, \lambda_8\}$  are likely given class  $c_2$  and are therefore surrounded by green ovals. The structure of the activity graph reflects those of the tracklets with respect to properties such as overlap, coincidence and noise. These aspects are shown by dashed lines with arrows.



ing *events* and others representing *coincidences*.

More formally, an event  $\varepsilon_k$  is an embedding of an event graph  $\lambda_k \in \Lambda_1$  with a particular subset of tracklets belonging to  $\mathcal{T}$ , and is given by the mapping  $\xi(\lambda_k)$  described in Section 3.5. An event cover  $E$  is defined as the set of all events. This is expressed as:

$$E \equiv \{\varepsilon_k : \lambda_k \in \Lambda_1 \wedge \xi(\lambda_k) = \varepsilon_k\}$$

Note that according to the above definition of the activity graph  $\Lambda$ , the event graphs  $\lambda \in \Lambda_1$  can overlap by sharing region histories in  $\mathcal{R}_{\mathcal{T}}$ . Since there is a one to one correspondence between  $\mathcal{R}_{\mathcal{T}}$  and  $\mathcal{T}$ , sharing of region histories between a pair of event graphs implies that the corresponding events (given by  $\xi$ ) will *overlap* with respect to the respective tracklets.

## Unsupervised Event Learning Task

In our unsupervised event learning task, neither the original event model  $\Theta$ , nor the original activity graph (whose embeddings are the tracks)  $\Lambda$  are known. The goal is to learn  $\Theta$  and the activity graph  $\Lambda$  that are most likely given the set of tracks  $\mathcal{T}$  is given for a video of a domain.

In order to address this task we introduce two other models apart from the event model  $\Theta$ . These are: (i) an exponential model of the proportion of overlap  $\Omega$  favouring minimum overlap; (ii) an exponential model  $\Upsilon$  of the proportion of coincidences, favouring minimum coincidences. Whilst these properties are made more precise below, we now continue with the description of the unsupervised learning task.

A candidate *interpretation*  $\mathfrak{S}$  is defined as a tuple  $\mathfrak{S} = \langle \Theta, \Lambda, \Omega, \Upsilon \rangle$  that is most likely for a given set of tracks  $\mathcal{T}$ .

Our goal in unsupervised event learning is to find the most likely interpretation  $\hat{\mathfrak{S}}$  given a set of tracks. Accordingly, the existence of a *target distribution* of possible interpretations is assumed, according to which each interpretation  $\mathfrak{S}$  has a posterior probability (given tracks) and the optimal interpretation  $\hat{\mathfrak{S}}$  has the highest posterior probability.

$$\hat{\mathfrak{S}} = \arg \max_{\mathfrak{S}} P(\mathfrak{S}|\mathcal{T}) \quad (4.1)$$

For tractability we assume that  $\mathcal{T}$  is conditionally independent of  $\Theta, \Omega, \Upsilon$  given  $\Lambda$ . Since  $\Omega$  and  $\Upsilon$  are given, their prior probability is equal to one. The probability  $P(\mathcal{T})$  is not represented in the factorization as the set of tracks  $\mathcal{T}$  is given and so does not influence

the comparison between the interpretations. Thus equation 4.1 is transformed to:

$$\hat{\mathcal{S}} = \arg \max_{\Theta, \Lambda} P(\Theta)P(\Lambda|\Theta, \Omega, \Upsilon)P(\mathcal{T}|\Lambda)$$

We continue explaining each of the factors  $P(\Theta)$ ,  $P(\Lambda|\Theta, \Omega, \Upsilon)$  and  $P(\mathcal{T}|\Lambda)$ , under the following sections entitled event classes and activity graphs respectively.

## Event Classes

In our unsupervised event learning framework, the underlying event model  $\Theta$  for the activities for a domain are unknown. However, some event models are regarded as being more likely to characterize an observed video of activities than others. We model *our a priori belief* about an event model  $\Theta$  in the form of a prior probability distribution  $P(\Theta)$ .

### $P(\Theta)$ : Prior Distribution over Event Classes

We characterize this prior probability in terms of two properties – *expected size* and *expected interactivity*. These are probabilistic extensions of the two event-like properties – size<sup>7</sup> and interactivity<sup>8</sup>. The reason for using these two properties was motivated in Section 4.2. Here, we present their extension to the more general notion of event classes, that is assumed for the complex setting.

The expected size  $\mathbb{E}(S(X)|\Theta)$  for the random variable  $X$  defined over  $\mathcal{G}$ , given an event model  $\Theta$  is:

$$\mathbb{E}(S(X)|\Theta) = \sum_{g_j \in \mathcal{G}} S(g_j) \sum_{c_i \in \mathcal{C}} P(g_j|c_i, \Theta)P(c_i|\Theta)$$

The expected interactivity  $\mathbb{E}(I(X)|\Theta)$  for the random variable  $X$  defined over  $\mathcal{G}$ , given an event model  $\Theta$  is:

$$\mathbb{E}(I(X)|\Theta) = \sum_{g_j \in \mathcal{G}} I(g_j) \sum_{c_i \in \mathcal{C}} P(g_j|c_i, \Theta)P(c_i|\Theta)$$

We now define the *prior distribution*  $P(\Theta)$ . The prior distribution  $P(\Theta)$  is expressed

<sup>7</sup>Size  $S(g_j)$  for an interaction graph  $g_j$  is the mean of the number of region histories and number of spatial changes in  $g_j$ , as introduced in Section 4.2.

<sup>8</sup>Interactivity  $I(g_j)$  for an interaction graph  $g_j$  measures the extent to which the region histories represented by  $g_j$  actively engage with each other. Interactivity is made precise in Section 4.5

as a normalized exponential model of a weighted combination of expected interactivity and expected size, since the higher values of these properties implies higher prior probability for the event model  $\Theta$ .

$$P(\Theta) = \frac{1}{z_1} \exp \left( \gamma_1 \mathbf{E}(I(X)|\Theta) + \gamma_2 \mathbf{E}(S(X)|\Theta) \right) \quad (4.2)$$

In the above equation,  $z_1$  is a normalization term ensuring that  $P(\Theta)$  sums to one over the range of all possible event models  $\Theta$ 's. The setting of the parameters  $\gamma_1$  and  $\gamma_2$  (also  $\gamma_3$  to  $\gamma_5$  used below) are described in chapter 6.

## Likelihoods with respect to Activity Graphs

In our unsupervised event learning framework, the underlying activity graph is unknown. However, some activity graphs  $\Lambda \in \Delta$  are regarded as being more likely in characterizing the observed activities than others. In order to characterize likely activity graphs, we first take note of the *dual role* of the activity graph.

The activity graph is an intermediate representation between an event model and a set of tracks. Therefore, it possess what we regard as a *dual characteristic of data and model*. With respect to an event model, it behaves like data, in the sense that it can be regarded as an abstracted representation of the data, i.e. the spatio-temporal relationships between a given set of tracks. Thus it is reasonable to formulate the likelihood of the activity graph given a model of events, overlap and coincidence.

With respect to a set of tracks, it behaves like a model, since the activity graph could generate many possible embeddings. Thus, it is reasonable to formulate the likelihood of a set of tracks, given an activity graph. These two dual aspects are formulated more precisely below.

### $P(\Lambda|\Theta, \Omega, \Upsilon)$ : Likelihood of an Activity Graph

The conditional probability  $P(\Lambda|\Theta, \Omega, \Upsilon)$  of an activity graph given an event model  $\Theta$ , a model of overlap  $\Omega$  and a model of coincidences  $\Upsilon$  is expressed as an exponential model consisting of three terms as follows:

$$P(\Lambda|\Theta, \Omega, \Upsilon) = \frac{1}{z_2} \exp \left( \gamma_3 \mathbf{E}(F(X, \Lambda|\Theta)) - \gamma_4 \mathbf{O}(\Lambda_1) - \gamma_5 \mathbf{N}(\Lambda) \right) \quad (4.3)$$

In the above equation,  $z_2$  is the normalization term. The first term  $\mathbf{E}(F(X, \Lambda_1|\Theta))$  stands for the *expected frequency*<sup>9</sup> of the random variable  $X$  in  $\Lambda_1$ , given an event model

<sup>9</sup>The frequency of an interaction graph in another interaction graph is defined in Section 3.4.

$\Theta$ . Expected frequency is defined in terms of the function  $F(g_j, \Lambda_1)$  that maps a graph  $g_j$  to its frequency in  $\Lambda_1$  and is defined as:

$$\mathbb{E}(F(X, \Lambda_1 | \Theta)) = \sum_{g_j \in \mathcal{G}} F(g_j, \Lambda_1) \sum_{c_i \in \mathcal{C}} P(g_j | c_i, \Theta) P(c_i | \Theta)$$

The second term  $O(\Lambda_1)$  is a measure of the proportion of overlap of region histories between event graphs in  $\Lambda_1$ . This measure is computed by the total number of region histories represented in  $\Lambda_1$ , divided by the sum total of the number of region histories for each  $\lambda \in \Lambda_1$ . The negative sign for this term is used to model the belief that activities tend to be composed of events that overlap minimally. The parameter  $\lambda_4$  is the parameter of the model of overlap  $\Omega$ .

The third term  $N(\Lambda)$  is a measure of the proportion of coincidental spatial changes in  $\Lambda_2$  divided by the total number of spatial changes in  $\Lambda_1$ . The total number of spatial changes is computed by counting the number of *meets* relationships in layer 3 nodes of  $\Lambda_1$  or  $\Lambda_2$  respectively. The negative sign for this term is used to model the belief that activities tend to be composed of a minimal proportion of interactions in which coincidental spatial changes are present. The parameter  $\lambda_5$  is the parameter of the model of coincidence  $\Upsilon$ .

### $P(\mathcal{T}|\Lambda)$ : Likelihood of an Embedding

Certain sets of tracks are regarded as being better embeddings of the activity graph in space-time than others if they are more prototypical of the spatial relations. The degree of typicality is measured using equation 3.12 described in chapter 3. Embeddings which are perceptually prototypical of the spatial relations should be generated with a high probability, according to this equation. The probability  $P(\mathcal{T}|\Lambda)$  of a set of tracks  $\mathcal{T}$  given an activity graph  $\Lambda$  is simplified by decomposing the  $\Lambda$  into independent HMMs for each pair of tracks  $\{\tau_i, \tau_j\}$ . Thus  $P(\mathcal{T}|\Lambda)$  can be expressed as the following product<sup>10</sup>, which we convert into an exponential form in order to be in-line with equations 4.3 and 4.2:

$$\begin{aligned} P(\mathcal{T}|\Lambda) &= \prod_{(\tau_i, \tau_j)} P(\delta(\tau_i, \tau_j) | S(\tau_i, \tau_j), \theta) \\ &= \exp\left(\sum_{(\tau_i, \tau_j)} \log(P(\delta(\tau_i, \tau_j) | S(\tau_i, \tau_j), \theta))\right) \end{aligned}$$

<sup>10</sup>The terms in this equation are explained in chapter 3.

**Modelling Observation Noise.** The likelihood term defined above makes it possible to model *observation noise*. As introduced before, observation noise arises due to complexities in image processing.

In order to model observation noise as a part of the generative process, it is assumed that when an event graph  $\lambda \in \Lambda$  is embedded as a set  $\varepsilon$  of tracklets, there may be *random* changes in spatial relationships. When these changes occur, the embedded tracklets  $\varepsilon$  are no longer perceptually prototypical<sup>11</sup> of this event graph  $\lambda$ . The set of tracklets  $\varepsilon$  is more likely to correspond to some other event graph  $\lambda' \in \Lambda'$ , where  $\Lambda'$  is some other activity graph.

Consequently, the likelihood  $P(\mathcal{T}|\Lambda)$  would be smaller than the likelihood  $P(\mathcal{T}|\Lambda')$ . However, the likelihood  $P(\Lambda|\Theta, \Omega, \Upsilon)$  is expected to be greater than  $P(\Lambda'|\Theta, \Omega, \Upsilon)$ , because the presence of  $\lambda \in \Lambda$  would be expected to be more frequent than  $\lambda' \in \Lambda'$ . As a result there is a chance that the overall posterior probability for  $\Lambda$  is higher than for  $\Lambda'$ , even though noise has caused the event  $\varepsilon$  to be perceptually more prototypical of  $\lambda' \in \Lambda'$  than  $\lambda \in \Lambda$ .

The above reasoning suggests that it is possible to propose different possible activity graphs (e.g.  $\Lambda, \Lambda'$ ) for the same set of tracks  $\mathcal{T}$  and then allow the combination of the two likelihood terms of the posterior, as given in equations 4.3 and 4.4, to favour the appropriate activity graph.

This combination may be used to recover the appropriate activity graph (in this case  $\Lambda$ ), despite the presence of observation noise. This idea is applied by the use of *moves*, that switch between alternative sequence of spatial states. This aspect is described further in Section 4.4.

## 4.4 Search for the Most Likely Interpretation

Our generative process for activities provides a probabilistic framework for formulating the posterior probability for an interpretation given the set of tracks. The posterior probability for a candidate interpretation is a measure of how likely it is that the candidate interpretation could have generated the observed set of tracks. In principle, the Maximum a Posteriori (MAP) solution could be found by exhaustively searching the space of candidate interpretations. However, in practice, enumerating all possible interpretations is infeasible. Therefore MCMC with simulated annealing [Kirkpatrick et al., 1983] is used to sample this distribution of interpretations and choose the maximum on convergence. A review of MCMC with simulated annealing is described in Appendix A.

<sup>11</sup>In the sense described in Section 3.5

### 4.4.1 Searching for the Optimal Interpretation

In this work, the Metropolis Hastings (M-H) algorithm [Neal, 1993] is used to sample the posterior distribution  $P(\mathfrak{S}|\mathcal{T})$  of interpretations  $\mathfrak{S}$  given tracks  $\mathcal{T}$ . The posterior distribution can be expressed in a compact form, as follows:

$$P(\mathfrak{S}|\mathcal{T}) = \frac{1}{z_1 z_2} \exp \left( \gamma_1 \mathbf{E}(\mathbf{I}(X)|\Theta) + \gamma_2 \mathbf{E}(\mathbf{S}(X)|\Theta) + \gamma_3 \mathbf{E}(\mathbf{F}(X, \Lambda|\Theta)) \right. \\ \left. - \gamma_4 \mathbf{O}(\Lambda_1) - \gamma_5 \mathbf{N}(\Lambda) + \sum_{(\tau_i, \tau_j)} \log(P(\delta(\tau_i, \tau_j)|S(\tau_i, \tau_j), \theta)) \right)$$

The normalization terms  $z_1 z_2$  is hard to compute as it involves a summation through each sample in the space of all possible interpretations. The M-H algorithm addresses the problem of drawing samples from a sample space with a posterior probability, when the normalization term for the posterior probability is hard to compute. This idea is explained in detail in Appendix A. In such cases, the posterior probability is given only in an un-normalized form, that is, without  $z_1 z_2$  in the above equation.

Thus M-H algorithm is used to sample the un-normalized density by simulating a Markov chain of interpretations  $(\mathfrak{S}^1, \dots, \mathfrak{S}^t, \dots)$ . Each interpretation  $\mathfrak{S}^t$  at the  $t$ 'th iteration of the Markov chain corresponds to an event model  $\Theta^t$  and an activity graph  $\Lambda^t$ . We do not make  $\Omega$  and  $\Upsilon$  explicit, since they are fixed for all iterations. The activity graph  $\Lambda^t$  at each iteration induces a corresponding event cover  $E^t$  for the given set of tracks  $\mathcal{T}$ .

We first present an overview of the algorithm and then discuss the details. A Markov chain is simulated by sampling a move  $m$  from a set of moves  $M$  (line 5). The move  $m$  is applied to the activity graph  $\Lambda^t$  of the current interpretation  $\mathfrak{S}^t$  (line 6). This move transforms it to a new activity graph  $\Lambda^{t+1}$ . For this activity graph, a new event model  $\Theta_{t+1}$  is obtained (line 9) by finding that event model  $\Theta^{t+1}$  given this new activity graph  $\Lambda^{t+1}$ , which maximizes the posterior probability:

$$\Theta^{t+1} = \arg \max_{\Theta} P(\Lambda^{t+1}, \Theta, \Omega, \Upsilon|\mathcal{T})$$

The activity graph  $\Lambda^{t+1}$  and the new event model  $\Theta^{t+1}$  form the new interpretation  $\mathfrak{S}^{t+1}$  (line 11). The candidate interpretation  $\mathfrak{S}^{t+1}$  of the Markov chain is accepted or rejected as given by the acceptance probability (line number 13):

$$\zeta(\mathfrak{S}^t, \mathfrak{S}^{t+1}) = \min \left( 1, \left( \frac{Q(\mathfrak{S}^{t+1}, \mathfrak{S}^t) P(\mathfrak{S}^{t+1}|\mathcal{T})}{Q(\mathfrak{S}^t, \mathfrak{S}^{t+1}) P(\mathfrak{S}^t|\mathcal{T})} \right)^{\frac{1}{T}} \right)$$

---

**Algorithm 1:** The M-H algorithm for finding the optimal interpretation for a set of tracks.

---

**Input:**  $\mathcal{T}$ : Tracks  
**Result:** Optimal Interpretation  $\hat{\mathfrak{S}} = \langle \hat{\Theta}, \hat{\Lambda} \rangle$

- 1 Initialize  $\Lambda^0$  ;
- 2 Compute  $\Theta^0: \Theta^0 \leftarrow \arg \max_{\Theta} P(\Lambda^0, \Theta, \Omega, \Upsilon | \mathcal{T})$ ;
- 3 **repeat**
- 5      $m \leftarrow \text{SampleMove}(M)$ ;
- 7      $\Lambda^{t+1} \leftarrow \text{ApplyMove}(m, \Lambda^t)$ ;
- 9      $\Theta^{t+1} \leftarrow \arg \max_{\Theta} P(\Lambda^{t+1}, \Theta, \Omega, \Upsilon | \mathcal{T})$ ;
- 11     $\mathfrak{S}^{t+1} = \langle \Theta^{t+1}, \Lambda^{t+1} \rangle$ ;
- 13     $\zeta = \min \left( 1, \left( \frac{Q(\mathfrak{S}^{t+1}, \mathfrak{S}^t) P(\mathfrak{S}^{t+1} | \mathcal{T})}{Q(\mathfrak{S}^t, \mathfrak{S}^{t+1}) P(\mathfrak{S}^t | \mathcal{T})} \right)^{\frac{1}{T}} \right)$ ;
- 14     $\varsigma \leftarrow \text{Rand}(0, 1)$ ;
- 16    **if**  $\zeta > \varsigma$  **then**
- 17        $\mathfrak{S}^{t+1} \leftarrow \mathfrak{S}^t$ ;
- 18    **else**
- 19        $\mathfrak{S}^{all} \leftarrow \mathfrak{S}^{all} \cup \mathfrak{S}^{t+1}$ ;
- 20 **until** convergence ;
- 21  $\hat{\mathfrak{S}} \leftarrow \max(\mathfrak{S}^{all})$ ;
- 22 **return**  $\hat{\mathfrak{S}}$

---

In the above equation,  $Q(\cdot)$  represents proposal probabilities and is described below, in the context of the moves that can be applied. Simulated annealing is incorporated with the introduction of a temperature  $T$  with a geometric annealing schedule with parameter  $\kappa$ . This is in order to speed up the convergence of the M-H algorithm towards the global optimum. In this manner the posterior distribution of the candidate interpretations is sampled and the optimal interpretation is chosen from this sample after convergence (line 21). The set of moves, the respective proposal probabilities and the process of obtaining an optimal event model at each iteration is described below.

### Moves and Proposal Probabilities

The proposal probabilities  $Q(\mathfrak{S}^t, \mathfrak{S}^{t+1})$ ,  $Q(\mathfrak{S}^{t+1}, \mathfrak{S}^t)$  for each move  $\mathfrak{S}^t \rightarrow \mathfrak{S}^{t+1}$  and its reverse move  $\mathfrak{S}^{t+1} \rightarrow \mathfrak{S}^t$  are computed as a fraction of the number of moves that result in state  $\mathfrak{S}^{t+1}$  from  $\mathfrak{S}^t$  to the total number of moves that can be applied to  $\mathfrak{S}^t$ . The proposal probabilities for each of the moves are described below:

**Birth and Death.** For a birth move, we pick a coincidental interaction graph  $\lambda_j \in \Lambda_2$  uniformly at random (u.a.r), and include it as an event graph in  $\Lambda_1$ . The respective proposal probability  $Q(\mathfrak{S}^t, \mathfrak{S}^{t+1})$  for a birth move is 1 divided by the total number of coincidental interaction graphs in  $\Lambda_2$ . A death move is the reverse procedure. We pick

an event graph u.a.r from  $\Lambda_1$  and include it as a coincidental interaction graph in  $\Lambda_2$ . The respective proposal probability  $Q(\mathfrak{S}^{t+1}, \mathfrak{S}^t)$  for a death move is 1 divided by the total number of coincidental interaction graphs in  $\Lambda_1$ . This move is shown schematically with the corresponding interactions in Fig. 4.4.

**Merge/Split Temporally.** The first type of move *merges* or *splits* interactions by increasing or decreasing the number of spatial states, while keeping the number of tracks constant. In a merge move, a pair of event graphs  $\lambda_j, \lambda_k$  is selected u.a.r from the set of all possible merge pairs in  $\Lambda_1^t$  and they are merged. This move is shown schematically with the corresponding interactions in Fig. 4.4. For a split move, an event graph  $g_k$  is selected u.a.r. For this event graph with qualitative spatial states  $s_1, \dots, s_{n_s}$ , a break point is chosen u.a.r from  $\{1, \dots, n_s\}$  and then split as illustrated schematically with the corresponding interaction in Fig. 4.4.

The respective proposal probability for a merge move is 1 divided by the number of possible merges possible. In principle, any two event graphs can be merged. However, to make the sampling more efficient, we define the notion of neighbourhood in space and time in which these event graphs are embedded as events. Two events are regarded as neighbours if their temporal distance and spatial distance are smaller than a threshold. The spatial distance between two events is regarded as the shortest distance between their respective tracks. The temporal distance is regarded as the proportion of the time interval where the two events intersect divided by the sum of the time intervals of both events. These two thresholds are fixed manually and discussed in chapter 6.

The proposal probability for the split move is 1 divided by  $n_s - 1$ , which is the number of possible ways an event graph, corresponding to an interaction, with  $n_s$  qualitative states can be split into two sub-event graphs.

**Merge/Split Region Histories.** The second type of move merges or splits interactions by increasing or decreasing the number of interacting objects. For a merge move, a pair of event graphs  $\lambda_j, \lambda_k$  is selected u.a.r from the set of all possible merge pairs and are merged as shown schematically with the corresponding interactions in Fig. 4.4. However, events may also be merged so that their objects overlap with each other. The probability that two merged events overlap is set to a small value<sup>12</sup>. For a split move, an event graph  $g_k$  is selected u.a.r and is partitioned u.a.r into event graphs with different sets of objects.

<sup>12</sup>This value is set heuristically as .1 to allow for a relatively small proportion of overlap.



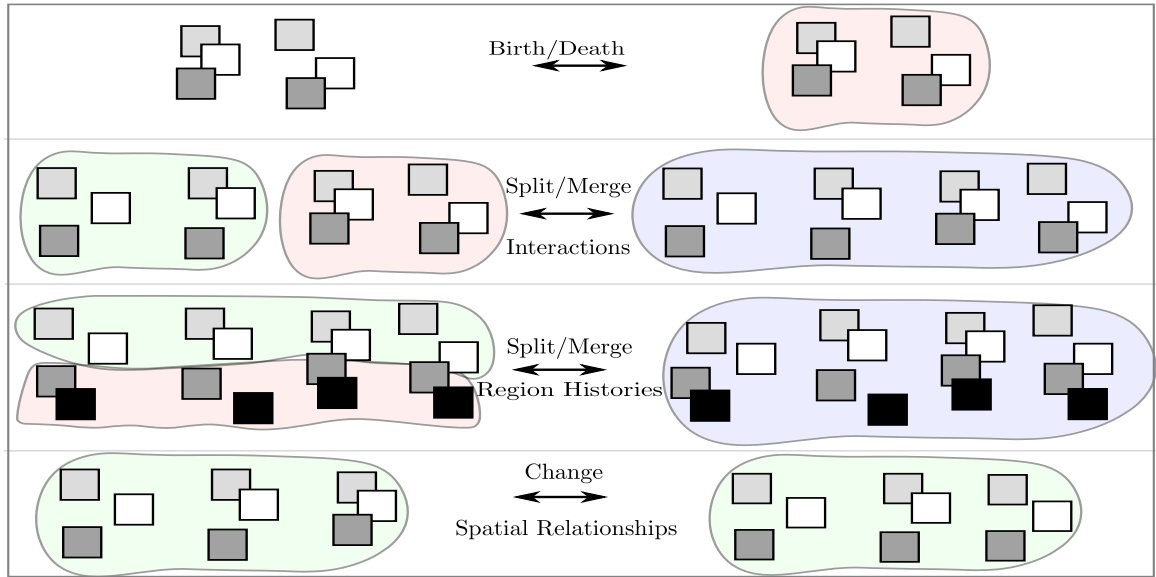


Figure 4.4: Illustrates the four types of moves for the MCMC search. The first type of move changes labels of a coincidental interaction graph as an event graph or vice versa. The second type of move splits or merges event graphs based on interactions. The third type of move splits or merges event graphs based on objects. The fourth switches between the two most likely spatio-temporal relationships for the same set of tracks.

**Change Spatial Relationships.** For the same set of tracklets of an event, the third move switches between the two most likely interaction graphs. The top two interaction graphs are obtained by choosing a pair of tracklets in an event. We first obtain the two most likely sequence of spatial relationships for this pair, as described in Section 3.5.1. Then we switch between these two sequences i.e. if the current sequence is the most likely then we switch to the second most likely sequence, and vice versa. Since there is only one possibility for the move and its reverse move, both  $Q(\mathfrak{S}^{t+1}, \mathfrak{S}^t)$  and  $Q(\mathfrak{S}^t, \mathfrak{S}^{t+1})$  are equal to 1.

**Obtaining an Optimal Event Model at every Iteration.** For any activity graph,  $\Lambda^{t+1}$  that is generated, a corresponding bag of event graphs  $\mathcal{B} = \{\dots, \lambda_i, \dots\}$  such that  $\lambda_i \in \Lambda_1^{t+1}$  are induced. In order to find the optimal event model, we enumerate all possible class labellings for this set of graphs. Accordingly, we try all possible class assignments by varying the number of classes and for each number, we explore all possible assignments of the graphs in this set  $\mathcal{B}$  into event classes. We model the class conditional probabilities  $P(g_j|\mathcal{C}, c_i)$  and  $P(c_i|\mathcal{C})$  in a non-parametric form, as described in Section 3.4. For each candidate event model  $\Theta$  we compute the posterior  $P(\Lambda^{t+1}, \Theta, \Omega, \Upsilon|\mathcal{T})$  and assign the  $\Theta$

that maximizes the posterior to  $\Theta^{t+1}$ .

## 4.5 Interactivity

As already identified in the introduction, those candidate events in which all participating tracks are actively engaged uniformly over time are considered more event-like. More precisely, preference is given to those candidate event graphs in which spatial relations are distributed uniformly (i) *across all subsets of tracks* (e.g. Fig. 4.5 (a) rather than Fig. 4.5 (b)) (ii) *temporally* (for e.g. Fig. 4.5 (a) rather than Fig. 4.5 (c)). Preference (i) means preferring candidate events with fewer tracks involved ignoring extraneous ones. (ii) means preferring candidate events in which interactions between pairs of tracks are tightly interleaved.

Pointwise mutual information (PMI) [Watanabe, 1960] is a well suited measure to model the degree of association between a subset of outcomes belonging to random variables. The degree of interaction between a subset of tracks for a candidate event is modelled in terms of the PMI between them. Then *interactivity* is expressed in terms of pointwise total correlation [Watanabe, 1960], which is just a weighted sum of PMIs over all subsets of tracks for a candidate event graph. Pointwise total correlation is highest when interactions between the tracks for the candidate event graph  $H$  are well distributed, both temporally and amongst subsets of tracks.

Let  $H_1$  be the layer 1 nodes (corresponding to tracks) of a candidate event graph  $H$  and let  $\omega \subseteq H_1$  be a subset of these nodes. PMI measures the strength of association between a set of tracks  $\omega$ , by comparing the joint probability of interaction  $P(\omega)$  between tracks in  $\omega$ , to the joint probabilities of interactions  $P(\omega')$ , of all its respective subsets<sup>13</sup> of tracks  $\omega' \subseteq \omega$ .

$$PMI(\omega) := \log \left( \prod_{\omega': \omega' \subseteq \omega} P(\omega')^{q_{\omega'}} \right) \text{ where } q_{\omega'} = (-1)^{\|\omega'\|} \quad (4.4)$$

We adopt a well known procedure for estimating the joint probabilities  $P(\omega')$  in equation 4.4, by measuring the proportion of contexts (which is appropriately defined below) in which the interaction between all tracks in the subset  $\omega'$  are observed, to the total num-

<sup>13</sup>When  $\omega$  is a set of two outcomes  $\{x, y\}$ , we have the well known form  $PMI(x, y) = \log(P(x, y)P(x)^{-1}P(y)^{-1})$ . This form is generalized to more than 2 variables in equation 4.4. Note that joint probabilities of subsets  $\omega' \subseteq \omega$  with odd cardinality (e.g.  $P(x), P(x, y, z)$ ) are in the denominator since  $q_{\omega'} = -1$ , while those of even cardinality (e.g.  $P(x, y)$ ) are in the numerator, since  $q_{\omega'} = 1$ .

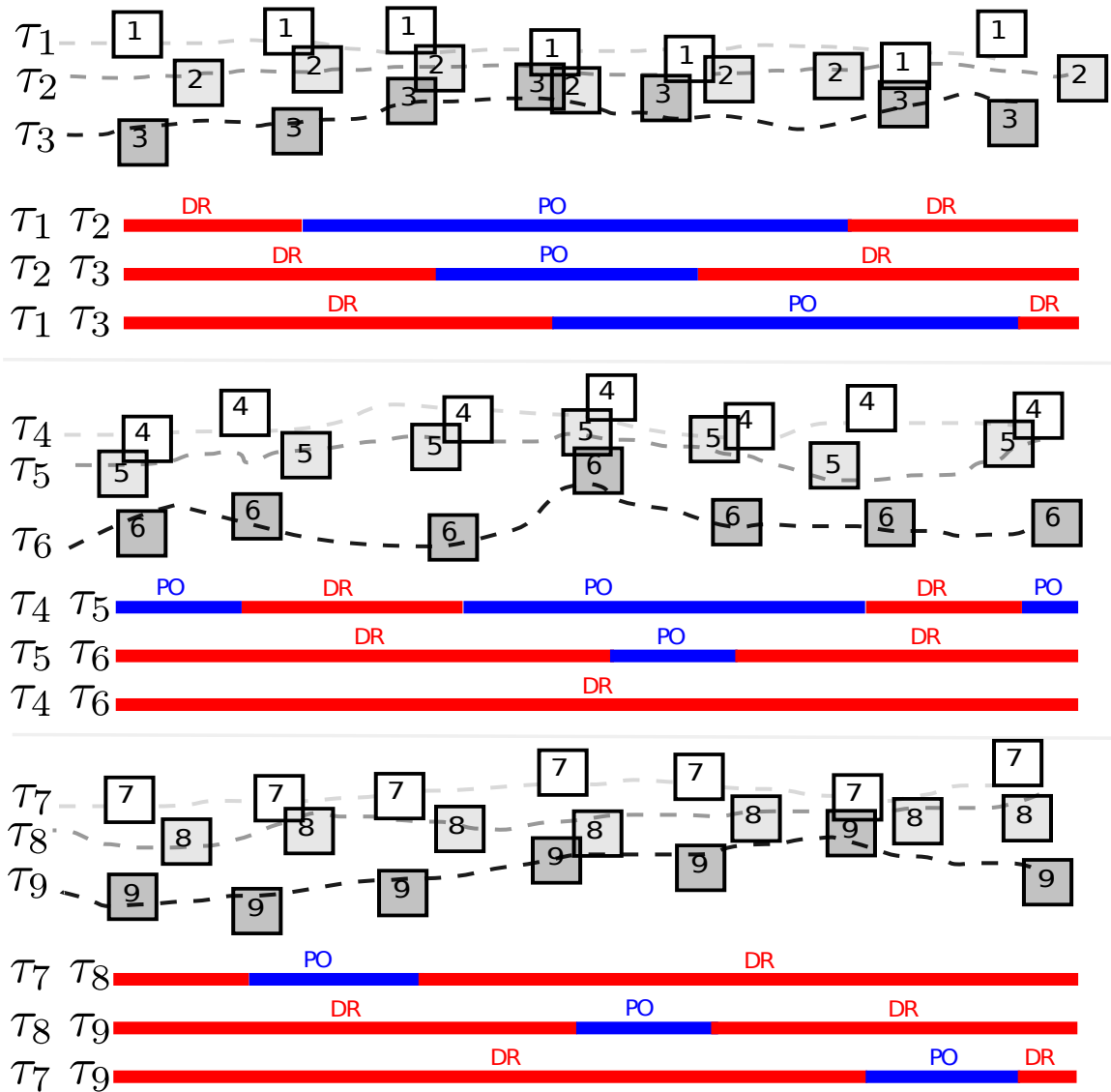


Figure 4.5: The notion of interactivity is illustrated with three scenarios. (a) Interactions between all three tracks  $\tau_1, \tau_2, \tau_3$  are uniformly distributed between all subsets of tracks and over the temporal period. (b) Interactions between tracks  $\tau_4, \tau_5$  are far more than between the other subsets  $\{\tau_5, \tau_6\}$  and  $\{\tau_4, \tau_6\}$ . (c) While interactions are evenly distributed between subsets of tracks, they are less evenly distributed temporally. Here,  $\tau_7, \tau_8$  interact initially, while  $\tau_9$  is a *bystander*, and then  $\tau_7, \tau_9$  interact while  $\tau_8$  is a *bystander*.

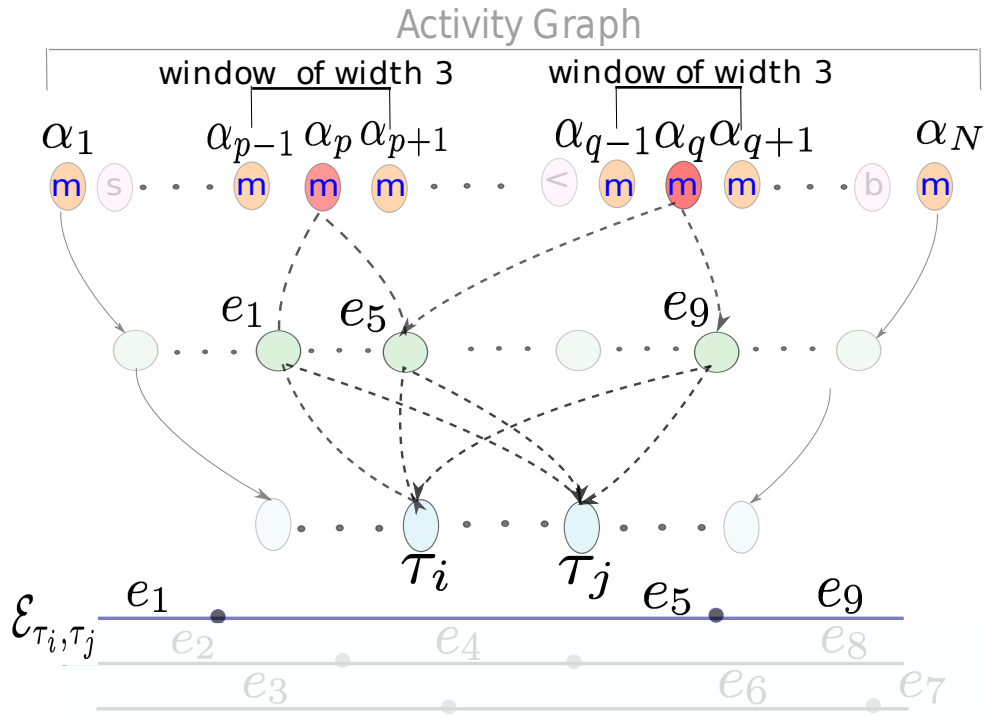


Figure 4.6: Windows on a sequence of layer 3 meets nodes  $\alpha_1, \dots, \alpha_N$  of an  $\Lambda$  are used to measure the joint probabilities for a set of tracks - here just  $\tau_i$  and  $\tau_j$ .

ber ( $N$ ) of all possible contexts. We have found that a window of width  $w$  that captures  $w$  consecutive interactions of the activity is an appropriate context for our purpose<sup>14</sup>.

We formulate the window in terms of the activity graph  $\Lambda$  by first noting that the layer 3 nodes labelled by *meets* capture (points of) interactions between all pair of tracks for the entire activity. We order these nodes temporally (by the end of each initial episode of the *meets* relation) to get a sequence  $(\alpha_1, \dots, \alpha_N)$  as shown in Fig 4.6, where  $N$  is the total number of interactions for an entire activity. A window of width  $w$  is simply defined as a subsequence  $(\alpha_k, \dots, \alpha_{k+w-1})$  of length  $w$ .

The probability of interaction  $P(\tau_i)$  for a single track  $\tau_i \in \mathcal{E}$ , with respect to a candidate event graph  $H$ , is just the fraction of the total number of windows  $N - w + 1$ , in which  $\tau_i$  interacts with any other track in  $H_1$ .

We estimate  $P(\tau_i)$  from  $\Lambda$ , by counting the number windows, which contain layer 3 nodes labelled by *meets* in  $H$ , such that:

1. Descendants in layer 1 contain  $\tau_i$ .
2. All the layer 2 nodes are in  $H$ .

<sup>14</sup>A sliding *window* of a fixed width  $w$  (e.g. a window of  $w$  words) has been regarded as a good context in the statistical natural language processing community, where it is used to compute the association between co-occurring words (e.g. *bread and butter*) by computing their co-occurrence within such windows.

3. Normalizing by  $N - w + 1$ .

In a similar manner, the joint probability  $P(\omega)$  for any set of tracks  $\omega$ , with respect to a candidate event graph  $H$ , is estimated by:

1. Counting the number of windows of width  $w$ , which contain layer 3 *meets* nodes in  $H$ .
2. The set of all layer 1 descendants of which are equal to  $\omega$  and all the descendants in layer 2 are in  $H$ .
3. Normalizing by  $N - w + 1$ .

In Fig. 4.6, the leftmost sample window of width 3 captures the interaction  $meets(e_1, e_5)$  between  $\tau_i, \tau_j$  as shown below the activity graph, as represented by the layer 3 node  $\alpha_p$ . Similarly, the rightmost window captures the interaction and  $meets(e_5, e_9)$  between  $\tau_i, \tau_j$  as shown below the activity graph, and as represented by the layer 3 node  $\alpha_q$ . All the descendants of  $\alpha_p$  and  $\alpha_q$  in layer 1 are equal to  $\{\tau_i, \tau_j\}$ .

We insert the probabilities of interaction for all subsets  $\omega' \subseteq \omega$  in equation 4.4 to obtain  $PMI(\omega)$ , and thus measure the PMI for all subsets  $\omega \subseteq H_1$  of layer 1 nodes (tracks) for a candidate event graph  $H$ . The PMI for scenarios such as shown in Fig. 4.5 (a), in which interactions are temporally well interleaved, tend to be higher than scenarios such as Figs. 4.5 (b),(c), since the former kind are likely to induce more windows, for larger subsets of tracks (and therefore a greater PMI score).

We now compute pointwise total correlation  $\xi(H)$  [Watanabe, 1960], which is the sum of all the pointwise mutual information  $PMI(\omega)$  over all subsets  $\omega$  of tracks  $H_1$  of a candidate event, weighted by their respective joint probabilities  $P(\omega)$ .

$$\xi(H) := \sum_{\omega: \omega \subseteq H_1} P(\omega) PMI(\omega)$$

The value of  $\xi(H)$  is highest when interactions between the tracks for the candidate event graph  $H$  are well distributed, both temporally and amongst subsets of tracks. Therefore  $\xi(H)$  is regarded as a good measure of *interactivity*. Finally, as smaller intervals measure interaction more significantly than larger intervals, we define the interactivity  $l(\omega)$  for a candidate event  $\omega$  by weighting the pointwise total correlation with an exponentially decreasing function of window width  $\eta$ ,

$$l(H) = \sum_{\sigma} e^{-\eta} \xi_{\eta}(H)$$

In this manner, the interactivity measure has been defined in order to give higher scores to interactions that represent actively engaging objects.

## **4.6 Conclusion**

This chapter describes a procedure for learning event classes as well as events from domain-specific video data. We represent activities of a given domain by the events, which are modelled by an event model. The event model itself consists of event classes. An event class is regarded as a probability distributions over a finite set of interactions and represents similar spatio-temporal relationships between objects.

In an unsupervised event learning task, neither the original event classes nor the original events that compose a given video of activities are known in advance and both have to be learned simultaneously. In order to address this challenge, this chapter also introduces a mathematical formulation of a generative process. This process incorporates certain event-like properties that can be used to characterize the original set of event classes and events. We further generalize our formulation to model overlap, observation noise and coincidences of events. Finally, it is incorporated into an algorithm that efficiently searches for the most likely interpretation for a given video.

# Chapter 5

## Learning Functional Classes

---

### 5.1 Introduction

The previous chapter described a procedure for learning event classes from video in an unsupervised way. These event classes were described just in terms of interactions between objects. The object type was regarded as being irrelevant. However, a more intuitive understanding of activities takes into account the relationship between object classes and event classes. The close association between event classes and object classes for three different domains are illustrated in Fig. 5.1.

The work described in this chapter is based on the idea that there are certain classes of objects whose instances (e.g. a trolley in an airport apron) tend to play similar functional roles (e.g. transporting bags) with respect to specific event classes (e.g. unloading). They tend to have similar functionality with respect to events and are hence called *functional object classes*. The relationship between the functional object classes and event classes are regarded as *functional relationships*.

This chapter firstly focusses on learning functional object classes. These are learned from the event classes that have themselves been learned using the techniques described in chapter 4. The learning procedure involves firstly inducing functional roles for each object from the learned event classes. Then, each object is represented in terms these roles. Finally, the objects are clustered to form functional object classes. It is expected that objects that have similar functional roles are likely to belong to the same functional

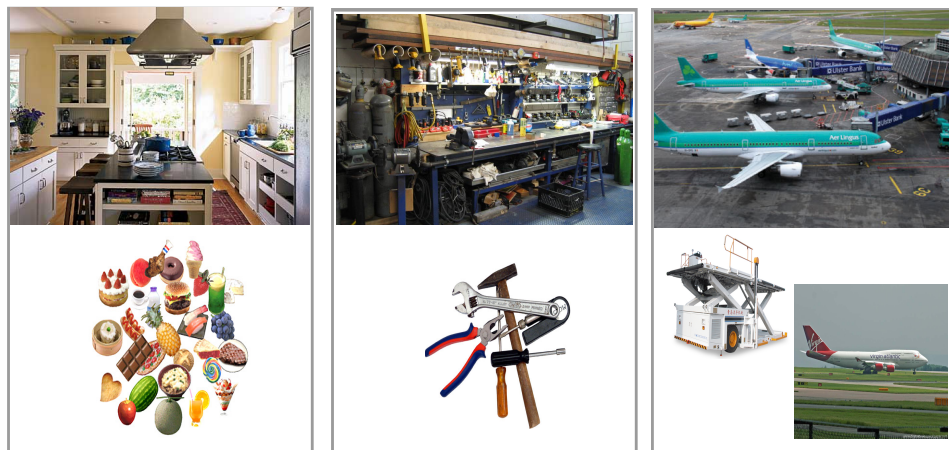


Figure 5.1: Three different domains – kitchen, workshop and airport apron – are shown above. Objects belonging to various object classes that are associated with the events in these domains are shown below.

object class.

The second focus of this chapter is to learn the functional relationships between the learned functional object classes and the event classes. Functional relationships between events and objects are regarded as the key for understanding and describing human activities. For example, in aircraft apron activities, an unloading operation in an aircraft domain is intuitively well described by a certain kind of interaction (given by the event class) between certain types of objects (given by the functional object classes such as loaders, trolleys and planes). Functional relationships are learned by mining association rules for strong associations between functional object classes and event classes.

Section 5.2 describes a procedure for learning functional object classes. Section 5.3 describes how functional relationships are discovered by mining for significant patterns of associations between event classes and functional object classes. Finally Section 5.4 presents some concluding remarks for this chapter.

## 5.2 Unsupervised Learning of Functional Object Classes

Object classes are an important cognitive construct for comprehending the diversity of objects that are perceived at each moment of wakeful existence. While objects in the world are perceived in terms of visual features i.e. what they look like, they are often comprehended in terms of their *function* i.e. what they are used for. Even a short reflection suffices to convince one that objects that we encounter in day-to-day activities such as knives, cars, pens etc. are all perceived according to their visual features and understood according to what they do i.e. cutting, driving, writing respectively.



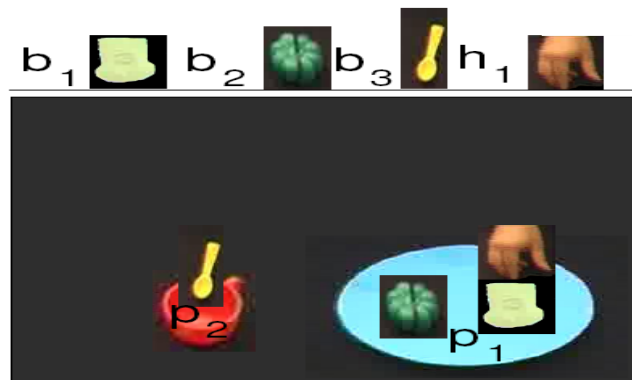


Figure 5.2: A scenario where several occurrences of the *taking away* event classes can be found. It is supposed that three occurrences of this event class have been found. Specifically  $h_1$  takes away  $b_1$  from  $p_1$ ,  $b_2$  from  $p_1$  and  $b_3$  from  $p_2$  respectively.

The following paragraphs introduce the idea of *functional object classes* as object classes whose instances play *similar functional roles* across occurrences of the same event class. We use the following running example in the rest of the chapter. We therefore introduce the example first and then describe the functional object learning procedure with the help of this example.

Consider an event class which represents interactions where an object takes away another object from a third object. A scenario where such events may occur is illustrated in Fig. 5.2. Let us suppose that three occurrences of this event has been observed and that they are spatio-temporally identical<sup>1</sup>. These three occurrences are enumerated below.

$h_1$  takes  $b_1$  away from  $p_1$

$h_1$  takes  $b_2$  away from  $p_1$

$h_1$  takes  $b_3$  away from  $p_2$

We say that the object instances are naturally classified into three obvious functional roles<sup>2</sup>:

1. The first object(s)  $\{h_1\}$  play the functional role of *an agent that takes away*, in typical dining activities, this could typically be a hand, as shown in Fig. 5.2.
2. The second object(s)  $\{b_1, b_2, b_3\}$  play the functional role of *being taken away*. This

<sup>1</sup>In the case, when an event class is given by more than one prototype, which are spatio-temporally similar, we consider the occurrences for each of these prototypes separately.

<sup>2</sup>We informally describe functional roles here. A more formal definition of functional roles is described further ahead in the chapter.

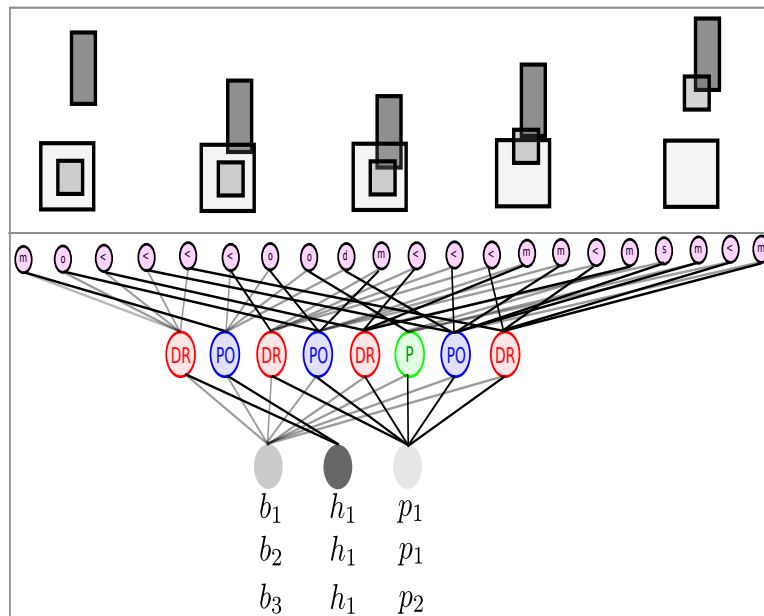


Figure 5.3: The interaction and an interaction graph for the *taking-away* event class is shown. The interaction graph represents the spatio-temporal relationships between three region histories  $r_1, r_2, r_3$ , where  $r_1$  is *taking away*  $r_2$  from  $r_3$ . At the bottom three occurrences of the taking away event is represented and their equivalent representation in the form of predicates  $\{F(h_1, b_1, p_1), F(h_1, b_2, p_1), F(h_1, b_3, p_2)\}$  are shown.

could typically be a food item such as a vegetable, bread or even cutlery such as a spoon as shown.

3. The third object(s)  $\{p_1, p_2\}$  play the function role of an object *from which other objects are being taken away*. These could typically be a *container* such as a plate or a cup, as shown.

The above example provides a way of classifying objects into meaningful functional roles, provided an alignment (e.g.  $b_1, b_2, b_3$ ) were somehow given. The *key hypothesis of this chapter* is that the mapping between the layer 1 nodes of an event graph and the corresponding object occurrences induces an alignment of the objects in such a way that this alignment results in intuitive functional roles.

We believe that this is a reasonable hypothesis for the reason that across several occurrences of an event class (e.g. taking away), objects sharing a particular functional role (e.g. instances of forks, knives) and objects sharing another functional role (e.g. instances of vegetables or instances of plates) tend to have similar or event identical spatio-temporal relationships with each other e.g. vegetables may interact in an identical way with forks and plates i.e. they are initially surrounded by the plate, then get attached to fork and are finally disconnected with the plate.

For the taking-away event class, the interaction between three region histories that represent taking-away are shown in Fig. 5.3. The corresponding interaction graph has three occurrences in the data, as enumerated below with corresponding object labels<sup>3</sup>.

The above hypothesis provides a way of inducing functional roles by exploiting the alignment induced by occurrences of event classes. The following paragraphs describe a procedure for inducing functional roles and then learning functional object classes.

**A Predicate based Representation.** We would like to abstractly represent an event graph that represents interactions such as  $X_1$  takes  $X_2$  away from  $X_3$  by the predicate  $F(X_1, X_2, X_3)$ . More formally, given an event graph  $g$ , we can represent this abstractly by a predicate  $F(X_1, \dots, X_n)$ , where  $n$  is the number of layer 1 nodes of  $g$  and  $X_1, \dots, X_n$  are ordered in some canonical order. We call  $F$  an event class predicate and  $F(X_1, \dots, X_n)$  an *uninstantiated event class*.

We would like to represent instantiated event classes of the form  $F(X_1, X_2, p_1)$ , where the third variable  $X_3$  has been instantiated by the object label  $p_1$ . This represents the event class of any object  $X_1$  taking away another object  $X_2$  from  $p_1$ . More formally, by  $F_i^\alpha$ , we denote the instance of  $F(X_1, \dots, X_n)$  obtained by instantiating  $x_i$  with  $\alpha$  i.e.  $F(X_1, \dots, X_{i-1}, \alpha, X_{i+1}, \dots, X_n)$ .

**A Lattice for General to Specific Event Classes.** We would like to represent event classes with different degrees of instantiations and learn from this representation. Accordingly, we construct an upper semi lattice  $L$  called an *event class lattice*, with  $n + 1$  levels such that the top node (at level 0) is  $F(X_1, \dots, X_n)$  and the level  $n$  nodes are all the instances of  $F(X_1, \dots, X_n)$  that correspond to all the event occurrences of  $g$  in the data.

This lattice is partially ordered by instantiation such that at level  $k$ ,  $k$  variables of the event predicate are instantiated, and the immediate descendants of any node in  $L$  are instances of it (with one more variable instantiated). All the nodes in levels  $L^1, \dots, L^{n-1}$ , we call *specific* event classes. This is because, they all have some, but, not all, of their arguments instantiated.

The lattice for the *taking away* event class is shown in Fig. 5.4. It can be seen that bottom most level 4 of the lattice are all the instances  $F(h_1, b_1, p_1), F(h_1, b_2, p_1), F(h_1, b_3, p_2)$ , that correspond to all the event occurrences of an interaction graph for the *taking away* event class.

---

<sup>3</sup>These labels have nothing to do with their type.

**Functional Roles.** This lattice facilitates a mechanism to take advantage of the spectrum of event classes and define functional roles for each of these classes.

Let  $l$  be a node at level  $k$  of  $L$ , such that the  $i$ 'th argument of  $l$  is an uninstantiated variable. i.e. if  $L^k$  be the set of all nodes at level  $k$  of  $L$ , then  $l \in L^k$  there exists  $\alpha$  such that  $l_i^\alpha \in L^{k+1}$ .

### Level

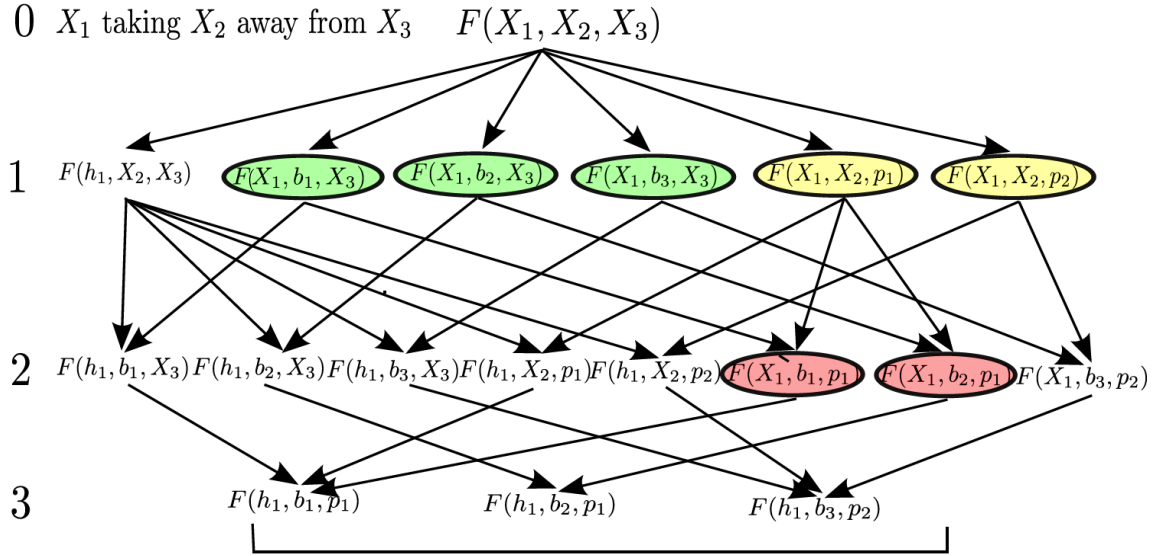


Figure 5.4: A lattice that represents event classes with different degrees of instantiations – from general to specific. The lattice shown above is constructed for the the three example occurrences of the *taking-away* event class. This lattice  $L$  is partially ordered by instantiation such that at level  $k$ ,  $k$  variables of the event predicate are instantiated, and the immediate descendants of any node in  $L$  are instances of it (with one more variable instantiated).

For example, with respect to the lattice in Fig. 5.4,  $l = F(h_1, X_2, X_3)$  is a node at the first level of  $L$  such that the second argument ( $i = 2$ ) is uninstantiated.

A *functional role* is characterized by *all instantiations* of a variable  $i$  of a node  $l \in L_k$  at the *next level*  $L_{k+1}$ . Thus we define a functional role  $R_i^l$

$$R_i^l = \{\alpha : l_i^\alpha \in L\} \text{ where the } i\text{'th argument of } l \text{ is an uninstantiated variable.}$$

In the above example, the functional role  $R_1^{l_1} = \{h_1\}$  induced by  $l_1 = F(X_1, X_2, X_3) \in L^0$  for the variable  $X_1$  has only one element, since there is only one substitution  $h_1$ , such that  $l_1$  belongs to nodes  $L^1$  of the lattice  $L$  as shown in in Fig. 5.4.

Interestingly the more specific event class  $l_6 = F(X_1, X_2, p_1) \in L^1$  induces a specialisation of the functional role  $R_2^{l_6} = \{b_1, b_2\}$  for the variable  $X_2$  i.e. objects that are taken

away from the *plate*  $p_1$  ( $X_3$  takes the value  $p_1$ ).

Five functional roles are summarized below<sup>4</sup>.

1.  $R_1^{l_1} = \{h_1\}$  (hand): *objects that take way.*
2.  $R_2^{l_1} = \{b_1, b_2, b_3\}$  (bread, vegetable, spoon): *objects that are taken away.*
3.  $R_3^{l_1} = \{p_1, p_2\}$  (plate, cup): *objects from which are taken away.*
4.  $R_2^{l_6} = \{b_1, b_2\}$  (bread, vegetable): *objects that are taken away from the plate  $p_1$ .*
5.  $R_2^{l_7} = \{b_3\}$  (spoon): *objects that are taken away from the cup  $p_2$ .*

These specific functional roles are interesting because objects that have similar interaction with the same object are often functionally more similar. In this example, objects that are taken away from a plate are usually food items and are more likely to be similar to each other than objects that are taken away from an arbitrary object. While this argument may be generalized by considering objects that are taken away from instances of the same functional object class (class of plates) are also more similar to each other, this is beyond the scope of this thesis, and is therefore discussed in chapter 7.

The functional roles described above suggests a natural categorization of objects. First we would like to put  $R_1^{l_1}, R_2^{l_1}, R_3^{l_1}$  into three separate functional classes, as the objects in each of these three sets play the same functional role with respect to this event class. Furthermore, we would like to sub-classify  $R_2^{l_1}$  into two more specific functional object classes  $R_2^{l_6} = \{b_1, b_2\}$  that are taken away from the plate and functional object class  $R_2^{l_7} = \{b_3\}$  that is taken away from the cup. Such a nested relationship between classes can be used to derive a functional object taxonomy.

---

**Algorithm 2:** An algorithm for mining functional roles from an event class lattice.

---

**Input:** Instances of  $F(X_1, \dots, X_n)$  that correspond to all the  $p$  occurrences of  $g$  in the data

**Result:** A set  $\mathcal{L}$  of functional object roles

- 1 Initialize  $\mathcal{L} \leftarrow \emptyset$ ;
  - 2 Form the lattice  $L$  with  $n + 1$  levels for  $F$  and the  $p$  occurrences;
  - 4 **foreach**  $k = 0 : n - 1$  **do**
  - 6     **foreach**  $l_j \in L^k$  **do**
  - 8         **if**  $i$ 'th variable is *uninstantiated* **then**
  - 10              $\mathcal{L} \leftarrow \mathcal{L} \cup R_i^{l_j}$ ;
- 

<sup>4</sup>Note that the mention of object categories such as hand, vegetable in brackets are only meant to make the presentation more intuitive. These are not known to the system and the goal is to be able to learn categories that are close to these.

	$R_2^{l_1}$	$R_3^{l_1}$	$R_2^{l_6}$
$b_1$	1	0	1
$b_2$	1	0	1
$b_3$	1	0	0
$p_1$	0	1	0
$p_2$	0	1	0

Table 5.1: An object by event role matrix is illustrated. If an object (e.g.  $b_1$ ) participates in an event role  $R_2^{l_1}$ , then the corresponding entry in the matrix is 1. Otherwise it is zero.

### 5.2.1 A Procedure for Mining Functional Roles

A general procedure for mining functional roles is shown in algorithm 2. For each learned event class and for each of the event graphs  $g$  with  $p$  corresponding occurrences, the lattice as described above, is formed. The input for this algorithm is all the instances of  $F(X_1, \dots, X_n)$  that correspond to all the occurrences of  $g$  in the data. The desired output of this procedure is the set  $\mathcal{L}$  of all functional roles for the nodes between levels  $L^0$  to  $L^{n-1}$ .

The procedure described in algorithm 2 first initializes  $\mathcal{L}$  to a null set. Then it scans this lattice in a level-wise manner from level  $L^0$  the  $L^{n-1}$ . For each node  $l_j$  at the level  $L^k$  if the  $i$ 'th variable is not already instantiated, then the functional role  $R_i^{l_j}$  is found. The discovered functional role is added to  $\mathcal{L}$ . This procedure continues till all nodes in  $L$  have been explored.

The procedure described above is used to mine for functional roles for each event class. The next stage involves representing objects in terms of functional roles so that objects that play similar functional roles may be clustered into functional object classes. Accordingly, each object is represented as a row vector of an *object by functional roles* matrix  $M$ , in which  $M_{i,j}$  equals 1 if the object  $o_i$  occurs in a particular functional role  $R_i^{l_j}$  and 0 otherwise. This matrix is illustrated in table 5.1 for three example event roles.

As each functional role corresponds to an event role, the row vectors of this matrix summarize each object in terms of the role it plays in all the event-roles and thus induce a multidimensional *object space*. In this space, objects that have a similar functional role with respect to similar sets of events are expected have a high similarity measure. We therefore perform  $k$ -means clustering using a cluster partition index to determine  $p$ . Thus a set of functional object classes  $c_1, c_2, \dots, c_p$  are learned. The object classes obtained using this approach are shown in the experimental chapter 6.

### 5.3 Functional Relationships

The association between events and objects is essential for understanding and describing human activities. For example an unloading operation in an aircraft domain is well described by a certain kind of interaction between objects of functional classes such as loaders, trolleys and planes. Thus human activities are suitably described by the functional relationships between event classes which describe interactions and functional object categories.

The approach described above results in a set of functional object classes  $c_1, \dots, c_p$ . In order to discover functional relationships between event classes and functional object classes, the object instance  $o_i$  for each predicate  $F_q(\dots, o_i, \dots)$  is instantiated with the corresponding learned functional class  $c_m$  and this yields :  $F_q(\dots, c_m, \dots)$ . Suppose for example that the following instantiations are obtained and represented without the use of respective brackets:

$$F_1, c_1, c_2, c_3$$

$$F_1, c_1, c_2, c_3$$

$$F_1, c_1, c_2, c_3$$

$$F_1, c_1, c_4, c_3$$

$$F_1, c_1, c_4, c_3$$

These can be regarded as a database of item-sets, where the first item (e.g.  $F_1$ ) is always the symbol for the event class and the others ( $c_1, c_2, c_3$ ) are the instantiated classes for the objects. The above database suggest it may possible to mine for the significant associations between event classes and object classes. For the above example, it can be seen that  $F_1, c_1$  is a very strong association between the event class  $F_1$  and the object class  $c_1$ , since this occurs relatively frequently in the above database of item sets. Similarly, the association given by  $F_1, c_1, c_2, c_3$  can be regarded as quite strong from the above database, since its frequency (equals 3) is again relatively high.

A simple apriori technique [Agrawal and Srikant, 1994a] from the area of frequent item set mining is used in order to discover significant associations event classes and functional object classes in this work. An outline of this procedure is shown in algorithm 3. This procedure involves forming candidate functional relationships (e.g.  $Can = F_1, c_1, c_2$ ) which consists of two parts: an event class label (e.g.  $F_1$ ) and a sequence of ob-

---

**Algorithm 3:** An apriori based algorithm for finding functional relationships.

---

**Input:**  $\{\dots, \{F_i, c_{k_1}, c_{k_2}, \dots, c_{k_l}\}, \dots\}$  Event class - object classes item sets for an event class  $F_i$

**Result:**  $FR$  Functional relationships in the form of frequent item sets

```

2 foreach  $o = 1 : o_{max}$  do
3    $Can \leftarrow GenerateCandidate;$ 
5 if  $\|Can\| > k_{fr}$  then
6    $FR \leftarrow FR \cup Can;$ 
7 return  $FR$ 

```

---

ject classes (e.g.  $c_1, c_2$ ). Candidate generation (given in line 2 of algorithm 3) is followed by counting the frequency  $Can$  in the item set based description for that event class  $F_1$  (line 5 of algorithm 3). The apriori search proceeds in a breadth-wise fashion, where item sets of each level are merged to form those of the next level, as described in [Agrawal and Srikant, 1994a]. However, only those item sets whose frequency is above a threshold  $k_{fr}$  (line 5 of algorithm 3). Once the level-wise search terminates, all item sets which are *non-maximal* are eliminated. This way significant associations between event classes and object classes are discovered.

## 5.4 Conclusions

This chapter described a procedure for learning functional object classes, given the event classes learned using the procedure in Chapter 4. The objects of a given domain are regarded as being modelled by functional object classes, such that those of the same class play a similar functional role whenever an event belonging to the associated event class occurs. In order to learn functional object classes, each object is represented in terms of a vector of functional roles it plays in each of the event classes. Clustering with this representation results in clusters that represent the desired functional object classes. The final focus of this chapter is the learning of functional relationships between the learned event classes and the functional object classes. We learn functional relationships by using a simple, yet effective procedure based on frequent item set mining.



# Chapter 6

## Evaluation of Activity Understanding

---

### 6.1 Introduction

Our framework for unsupervised activity understanding was presented in two stages. The first stage was event learning, where the aim was to learn event classes and an event cover, given a video that captures extended period of activities, for a certain domain. A procedure for event learning was described in chapter 4. The second stage is functional object class learning, which uses these learned event classes to induce functional object classes and functional relationships. Procedures for learning functional object classes and relationships were described in chapter 5. Both these aspects are based on the representation of event classes in terms of interactions, as described in chapter 3. This chapter evaluates our framework for unsupervised activity understanding presented in chapter 3, 4 and 5.

The HMM based procedure for obtaining a regularized sequence of spatial relationships despite noise from video is first evaluated. Synthetic data sets are used to evaluate event learning in a more controlled environment. A real data set – an aircraft apron – is used to evaluate the techniques for learning event classes, functional object classes and functional relationships respectively.

Section 6.2 describes the synthetic and real data set respectively. Section 6.3 describes the evaluation of the HMM based procedure. Section 6.4 describes the experimental settings, parameter estimation and evaluation scores for the event learning procedure. Section 6.5 describes our experiments on real data. Section 6.6 describes the evaluation of the procedures for functional object class learning and discovering functional relationships.

Section 6.7 summarizes the overall conclusions of the experimental evaluation.

## 6.2 Data Sets, Ground Truth and Evaluation Procedures

We use both synthetic and real datasets to evaluate our framework for activity understanding. While synthetic datasets are used for evaluating only the event learning aspect of the framework, real datasets are used to evaluate the entire framework. This section describes the two datasets.

### 6.2.1 Synthetic Datasets

Synthetic data provides a controlled environment where it is possible to simulate the generation of real-world-like activities, while being able to analyse the role of event-like properties – expected size, frequency and interactivity – and also the effect of the complicating factors – overlap, noise and coincidence. Synthetic data is produced by a simulation by a technique described in Appendix B. A simulation results in an event model  $\Theta$  with event classes  $\hat{C} = \{\hat{c}_1, \dots, \hat{c}_{nc}\}$  and an associated event cover  $\hat{E} = \{\hat{e}_1, \dots, \hat{e}_{ne}\}$ . The association between the event classes and the events in the event cover is given by a labelling of the events  $\hat{L} = \{\hat{c}_{k_1}, \dots, \hat{c}_{k_{ne}}\}$  into their respective classes. In addition to these events, the synthetic data is used to capture a certain proportion of: (i) coincidences  $n_{co}$ ; (ii) overlaps  $n_{ov}$ ; (iii) events  $n_{no}$  that are corrupted by noise. A simulation  $\mathcal{S}$  is expressed as  $\mathcal{S} = (\hat{\mathcal{E}}, \hat{\mathcal{L}}, n_{co}, n_{ov}, n_{no})$ .

In the following experiments, *two types* of synthetic datasets are simulated, one simple and the second complex. In the first type, the three complicating factors – coincidence (c), overlap (o) and noise (n) – are absent i.e  $\mathcal{S} = (\hat{\mathcal{E}}, \hat{\mathcal{L}}, 0, 0, 0)$ . Therefore this dataset is called *synthetic data(-con)*. In the second type of dataset, the three complicating factors are all present i.e.  $\mathcal{S} = (\hat{\mathcal{E}}, \hat{\mathcal{L}}, n_{co}, n_{ov}, n_{no})$ , where  $n_{co} > 0, n_{ov} > 0, n_{no} > 0$ . This is referred to as just *synthetic data*.

### 6.2.2 Real Datasets

A real dataset, which is a video of activities at an aircraft airport apron is used to evaluate the event and object learning procedures. This dataset was chosen as it embodies the kind of structure and complexities that tend to be present in several other datasets (e.g. workshops, football matches). Structure manifests itself as spatio-temporally similar events over extended periods of observation. These events are embeddings of certain semantically interesting event classes such as bridge-on, unloading etc. Complexity manifests



Figure 6.1: An image from the airport apron dataset.

itself in the form of factors such as noise, overlap and coincidences. Another reason for the choice of this dataset was data availability.

This dataset comprises of approximately 24 turnarounds totally spanning around 37 hours of video footage, showing servicing of aircraft between flights. The camera positioning for all the turnarounds is the same, so the same view as shown in Fig. 6.1 is obtained. It is important to note that we deliberately chose not to provide any information about the start and end of each turnaround. That is all the turnarounds are concatenated temporally and given as one large video, and no additional preprocessing is carried out to separate the video into turnarounds. Providing the learning system with the additional information about segments corresponding to turnarounds would imply a certain amount of supervision (corresponding to a transactional setting), that we intend to avoid.

In the rest of the work, the airport dataset is divided into two parts. The first part consists of 4 representative turnarounds<sup>1</sup> and is referred to as *part 1 apron data*. The second part consists of 20 turnarounds is referred to as *part 2 apron data*<sup>2</sup>.

### 6.2.3 Processing the Video Data Sets to Obtain Tracks

The processing of the real data sets involved two stages – detection and tracking. These two stage are described below.

---

<sup>1</sup>These turnarounds were chosen in a way that they represent the variances in appearance for the other 20 turnarounds.

<sup>2</sup>The event and object learning framework is only applied to part 2 apron data only.

**Detection.** For the first stage, a multi-class object detector [Ott and Everingham, 2009] based on HOG features was trained on *part 1 apron data* and applied to each frame of *part 2 apron data*<sup>3</sup>. The five classes considered were: (i) plane; (ii) trolley; (iii) loader; (iv) bridge; (v) plane-puller. The choice of these object classes were based on the consideration that the three IATA (The Air Transport Association) events, namely *unloading*, *bridge-on-off* and *plane-puller-on* described interactions between these objects. Pedestrians are not detected for this scene for the reason that their size reduces substantially, making detections complex. Detection becomes even more complex, when pedestrians merge together and split, especially at this distance.

Approximately 5000 training samples were collected for each of these classes from *part 1 apron data*. In order to simplify the task of data collection, a semi-automated technique was used. This technique was based on the work reported by authors in [Celik et al., 2009]. It involved clustering of foreground blobs based on their appearance features into clusters. These clusters were used to prepare the training data for each class. In order to obtain good performance, repeated bootstrapping and manual removal of noisy samples was used to improve the quality of the detector on *part 1 apron data*. The trained multi-class object detector is applied on each frame of the entire *part 2 apron data*. This gives rise to a set of detections with associated membership probabilities for each of the five object classes described above.

**Tracking.** The second stage involves applying our implementation of the tracking technique reported in [Yu and Medioni, 2008] to the detected blobs in *part 2 apron data*. The MCMC data association tracker searches for a covering of the detected blobs into tracks, such that this covering has the maximum posterior probability, given a set of detected objects. The likelihood term of the posterior probability favours a set of tracks that are likely with respect to continuity of appearance, motion and class IDs (output from the detector). The prior term that favours larger tracks which have less overlap and less noise. We chose this technique since it performs global optimization to obtain the most likely set of tracks.

**Note:** Although the tracked objects have types associated with appearance based object classes, the *event learning procedure deliberately ignores these in order not to be dependent on them. Thus in principle, it could work equally with untyped tracks.*

---

<sup>3</sup>The choice of this technique was the result of preliminary experiments with different features, after which it was found that HOG features are well suited to characterize the objects in the scene.

### 6.3 Evaluation of the HMM based Procedure

A HMM was proposed in Section 3.5, as a means of obtaining a more stable sequence of qualitative spatial relationships from video, in comparison to the existing point intersection technique which is defined in a Boolean fashion solely in terms of connectivity of the spatial extents, as surveyed in [Cohn and Hazarika, 2001]. The following experiment evaluates the HMM based procedure and compares it with the point set intersection technique, which is regarded as the baseline.

In order to train and test the HMM, the *part 1 apron data* is randomly divided into two parts. The first consisting of two thirds is used for training, and the rest one third for testing. Ten such random partitions are created for evaluation. The training data is hand annotated by associating pairs of tracks in the training set with a corresponding sequence of spatial relationships<sup>4</sup>. These annotations are subjectively assigned by the annotators.

Instead of labelling the entire data of several thousand frames, only those segments, where there are changes in spatial relationships are considered for training and testing the HMM. This is because, the main purpose of the HMM is to learn a stable transition between the spatial states, rather than parts where there is a considerably high certainty of the spatial states.

A total of 27 training segments and 14 test segments were prepared. In these segments, those pairs of tracks for which the spatial relationships change are first identified. These pairs are subjectively labelled with the appropriate spatial relationship for each frame in which both the tracks are observed. The segments are also provided with respective episodes for this sequence of spatial relationship. One such segment is illustrated in Fig. 6.2.

The HMM is trained on the training segments for each random partition using the procedure described in Section 3.5.1. The trained HMM is then applied on the test segments for the corresponding random partition. This gives rise to a sequence of spatial relationships between pairs of tracks on the test segments. A corresponding sequence of episodes are constructed from the inferred sequence of spatial relationships, for the sake of evaluation which is described below.

#### Results

The performance of the HMM is evaluated in two ways. The first involves evaluating the extent to which the HMM outputs a correct sequence of episodes. It was observed

---

<sup>4</sup>This is because, the purpose of the HMM is to learn a mapping from a pair of tracks to a corresponding sequence of spatial relationships.

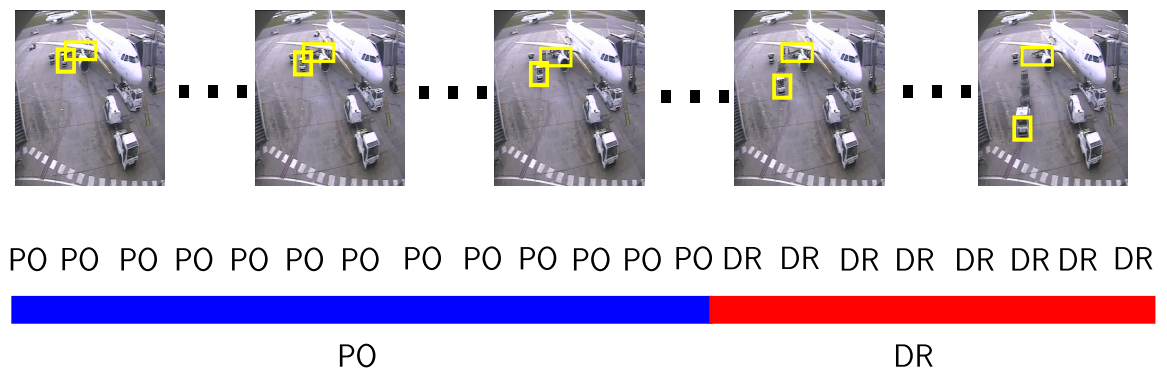


Figure 6.2: A segment with which the HMM based procedure for obtaining spatial relationships is trained and evaluated. Some images from a segment that has been manually annotated for training and evaluating the HMM are shown. An example of an annotation in the form of a sequence of spatial relationships is shown for a pair of tracks corresponding to a loader and trolley respectively.

across the 10 random partitions, the sequence of episodes for 85.7% of the test segments, with a variance of 8.3%, exactly corresponded to the ground truth. However the point intersection based technique only resulted in an accuracy of 42.8% with a variance of 17.2%.

The second evaluation involves evaluating the extent to which the outputted episodes temporally align with those of the ground truth. This evaluation is restricted only to those those segments whose sequence of episodes obtained from the HMM matches the ground truth. This is because, the purpose is to understand the extent of deviation in temporal alignment, despite the fact that the episodes have been matched correctly. A good alignment ensures a reduced chance of structural difference in temporal relationships (amongst the episodes) between the ground truth and the output of the HMM.

It was observed that across the 10 random partitions, there was an average overlap of 68.1% temporal overlap with a variance of 4.3% between the outcome of the HMM and the ground truth. The point intersection based technique resulted in an average overlap of 33.4% with a variance of 14.8%.

## Conclusion

It can be concluded that the HMM significantly outperforms the traditional point intersection based technique. In particular, the potential advantage of using the HMM described in Section 3.5, for inducing stable sequence of qualitative spatial relationships from video data, has been demonstrated.

## 6.4 Experimental Settings, Parameter Estimation and Evaluation Scores

The following paragraphs describe the two kinds of experimental settings for evaluating our event learning procedure, a method for parameter determination and two types of evaluation scores.

**Experimental Settings.** This section describes four experiments to evaluate the event learning procedure proposed in this work, on synthetic data. Synthetic data provides a controlled experimental environment in which the proposed framework can be evaluated and analysed. Three experiments are performed with simulations of the type *synthetic data (-con)*. The goal is to understand the significance of the three event-like properties and the effect of their combination in the absence of complicating factors. The absence of these factors makes it easier to analyse the role of these properties in isolation. For these three experiments, the factors that model these additional complexities (con) are turned off and this is called as *generative process (-con)*. The fourth experiment is performed with simulations of type *synthetic data*, in order to understand the effect of increasing the presence of these factors, on the performance of the event learning procedure. In this experiment, the event learning procedure is referred to as *generative process*.

**Parameter Determination.** The generative process is expressed as an exponential function of a linear combination of event like properties and complicating factors in 4.4. The parameters  $\gamma_1$  to  $\gamma_5$ , involved in the linear combination, are regarded as the *meta-parameters* of the generative process.

The estimation of meta-parameters is often a difficult problem, especially for an unsupervised case where a ground truth is not provided for training. We therefore use a separate synthetic data set in order to determine the parameters. We use the ground truth for this synthetic dataset and determine the meta-parameters. For the rest of the experiments, both on real and synthetic data, we re-use these parameters. We believe that reusing the parameters is reasonable, since most datasets encode similar assumptions (e.g. minimum overlap, coincidence etc.) to those of the synthetic datasets. This saves us from providing supervision in the form of knowledge of the events and event classes, for a new dataset, in order to determine parameters.

We adopt the approach reported in [Yu and Medioni, 2008] for determining parameters. The choice of the approach is motivated by the consideration that their technique shares certain similarity with ours. Their approach involves the use of MCMC for searching for an optimal covering of object blobs in the form of tracks. Our approach involves the use of MCMC for searching for an optimal covering of tracks in the form of an event

	E. Size	E. Int.	E. Freq.	Combination
F-Measure	13.5	67.7	32.4	100.0
Rand Index	7.1	31.6	21.9	100.0

Table 6.1: The table shows the accuracies of the generative process (-con), when each of the three event like properties are evaluated individually and also when they are combined. The individual evaluations for each property is given by the the following abbreviations (i)expected size (E.Size); (ii) expected interactivity (E.Int); (iii) expected frequency (E.Freq). The final column shows the accuracies with their combinations.

cover, together with finding the associated event classes.

Their approach makes use of the fact that the posterior density is a linear combination of the parameters. Since the posterior for the ground truth is known the posterior for all other sub-optimal interpretations is expressed as being lesser than that of the ground truth. This constraint is expressed as a linear equation. After collecting multiple constraints, we use Linear Programming to find a solution of positive parameters with a maximum sum of the parameters, given the constraints. We have found that the 1000 constraints are enough to be able to obtain an accurate estimate of the parameters.

**Evaluation Scores.** Two types of evaluation scores are used to quantitatively evaluate the outcome of the event learning procedures on synthetic data. The first measures the accuracy of a learned event cover  $E$  with respect to the ground truth event cover  $\hat{E}$ , using the F-measure [Baeza-Yates and Ribeiro-Neto, 1999]. The second evaluation score namely the Rand Index [Rand, 1971], measures the accuracy of the class assignments  $C$  of a learned event cover with respect to the ground truth event classes  $\hat{C}$ . Appendix C describes the use and suitability of F-measure and Rand Index respectively.

## Experiment 1: Contribution of the Event-Like Properties

The purpose of this experiment is to evaluate the contribution of each event-like property and their combined contribution. Accordingly, a total of 20 simulations were performed to generate synthetic datasets of the type synthetic data(-con). For each simulation, generative process(-con) is applied by turning off each of the three properties i.e. size, interactivity and frequency respectively. Then the generative process(-con) is applied by with a combination of these properties, with the parameters that were found using the approach described above.



## Results

The results of this experiment are tabulated in table 6.1. For all 20 simulations, the combination gave a 100% accuracy with respect to both F-measure and Rand index. However, each factor, when taken individually resulted in much lower accuracies as shown.

## Conclusions

This experiment demonstrates the significance of combining the three event like properties in order to recover the original event classes and events. This experiment also demonstrates that the method for parameter determination described above gives rise to parameters that facilitates this recovery.

## Experiment 2: An Analysis of the Trace of Event-Like Properties

While the above experiment supports the hypothesis about the significance of combining the event-like properties, what is more interesting is to gain a precise understanding of the role that these properties play at different stages during the learning process i.e. during the progression of the Markov chain.

Accordingly, the respective values of the posterior together with expected values of size, interactivity and frequency at each iteration of the Markov chain were plotted against the iteration number. These plots for one out of the 20 simulation, are shown in Fig. 6.3<sup>5</sup>. In addition, the F-measure and Rand index values are also plotted against the number of iterations in Fig. 6.3, in order to analyse the relationships between these event-like properties, posterior, F-measure and Rand index respectively.

## Results

An inspection of the results in Fig. 6.3 suggest three distinct phases in the Markov chain. These are described below:

1. In the first phase, within the first 200 iterations, expected size and interactivity increases steeply, while there is hardly any increase in frequency. During this phase, the single tracklets aggregate into two or more tracklets such that expected size and expected interactivity increases with a corresponding increase in the posterior. This is also reflected in an increase in the F-Measure and Rand Index. That is, moves that favour an increase in size and interactivity are favoured. However, the lack of

---

<sup>5</sup>The plots for the rest of the 19 simulations showed a similar behaviour as described below.

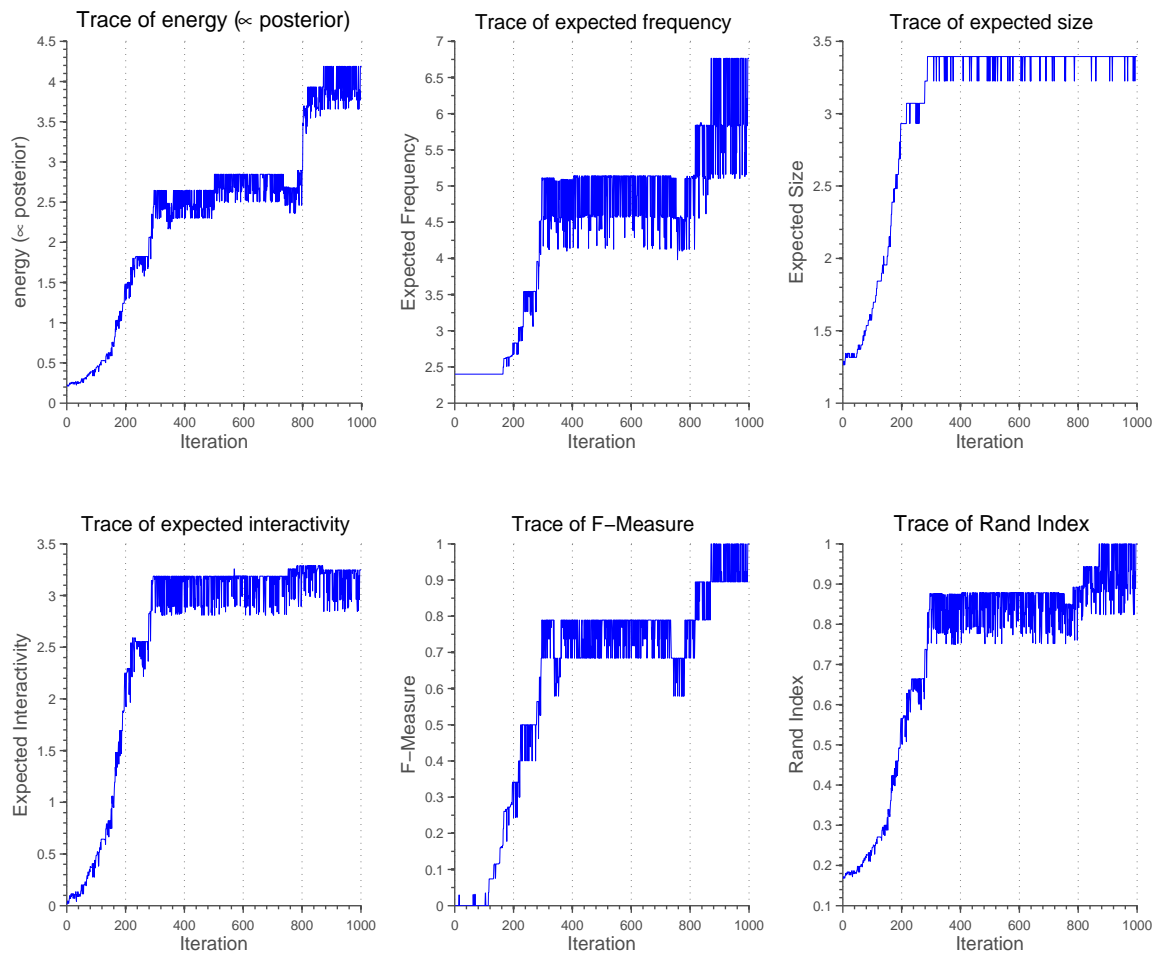


Figure 6.3: Trace of (i) energy (which is proportional to the posterior); (ii) expected frequency; (iii) expected size; (iv) expected interactivity; (v) F-measure; (vi) Rand index, with respect to the number of iterations of the Markov chain.

significant increase in frequency indicates that the patterns of similar events have not started to emerge.

2. In the second phase, between 200 and 300 iterations, whilst there is still a sharp increase in size and interactivity, expected frequency evidences a substantial rise, with corresponding rise in the posterior. This is also reflected in an increase in the F-Measure and Rand Index. This indicates that patterns have started to surface and moves that give rise to higher expected frequency are favoured.
3. In the third phase, between 300 and 400 iterations, expected size and interactivity show variations but no substantial increase. However, during this phase, there is a sharp rise in increase for expected frequency, indicating that more and more patterns are found, while exploring within the same range for interactivity and size. A similar behaviour is observed till 1000 iterations, where it can be seen that it is increases in expected frequency that cause an increase in the posterior. This is also reflected in an increases in the F-measure and Rand index respectively. In this phase, the moves that cause a substantial increase in expected frequency are favoured and it is these moves that drive the search toward the optimal event classes and events.

### **Conclusions**

To summarize, this analysis gives an insight into how the three event like properties combine together in recovering the original set of event classes and the event cover respectively. Initially, expected size and expected interactivity together aggregate potential events, albeit with an error rate of around 50%. This aggregation gives rise to patterns of similar or identical events in the event cover. Then the expected frequency starts to dominate while still acting in tandem with the other two properties to resolve these errors.

### **Experiment 3: Sensitivity to Variation in Parameters**

In this work, we use a linear programming based approach [Yu and Medioni, 2008] to determine the meta-parameters, as described in Section 6.4. However, it has been observed from preliminary experiments on synthetic data that under a wide range of values for the parameters, the MCMC optimization converges towards the optimal solution. Moreover the technique for parameter determination in Section 6.4 estimates values well within this range. It was observed that for parameter values within this range, MCMC converges relatively fast. Outside this range, it takes longer to converge and some times the Markov chain does not converge after several thousand iterations.

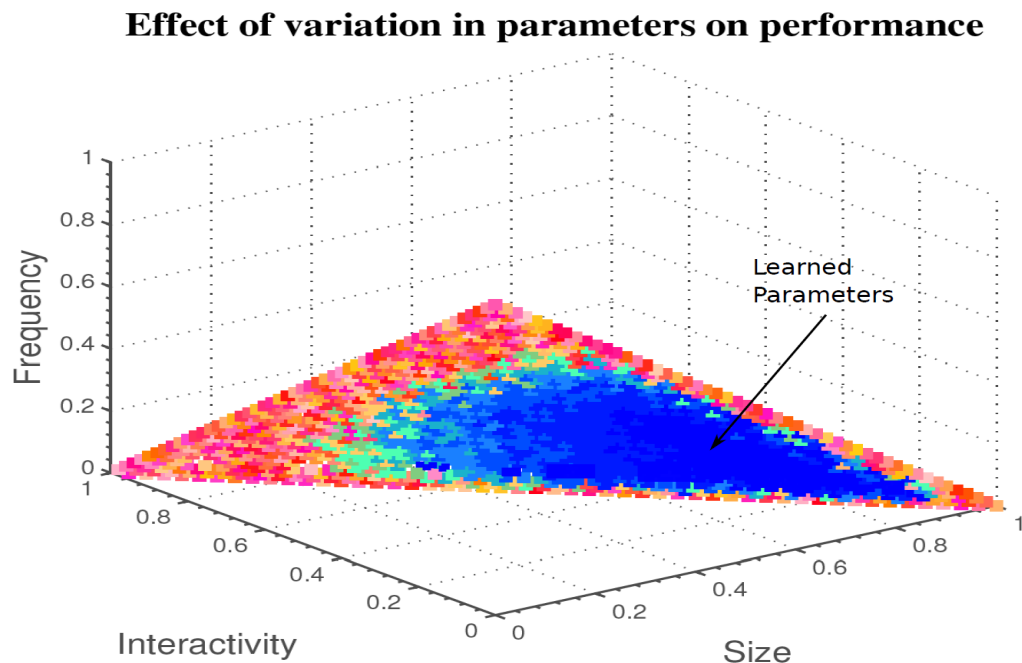


Figure 6.4: A plot of accuracy with respect to variation of meta-parameters for expected size, interactivity and frequency is shown. Each point in the coordinate system is a tuple of parameters for these three properties. The accuracy for this point is plotted with a colour coding, where red is for 0% accuracy, and blue represents 100% accuracy.

In order to analyse the performance of the system with respect to variations in parameters in a systematic manner, the following experiment was performed. A total of 20 simulations were performed in order to capture the variance in the data. For each simulation, the parameters for size, interactivity and frequency were varied such that they sum to one. The accuracies are plotted with a colour coding where red is for 0% accuracy and blue represents 100% accuracy. The number of simulations were restricted to 5000, since it was observed during the preliminary explorations that for a large range of parameter values, the Markov chain converges well within 5000 iterations.

A very similar behaviour was observed across all 20 simulations. Therefore a plot for one of the simulations is shown in Fig 6.4.

## Results

The following observations were made. First, there is a fairly large range of values (as shown in blue) for which the accuracy is 100% and beyond this range, the accuracy declines quite sharply. Second the estimated values are roughly in interior of this region, implying that there is a fairly good range within which the variation of the parameters

does not cause a decrease in accuracy.

### Conclusions

These two observations suggest that the method for parameter estimation described in 6.4 gives rise to parameters that are stable i.e. only a considerable deviation from these parameters can cause a decrease in accuracy.

## Experiment 4: Effect of Increase in Complicating Factors on Performance

The generative process was formulated in order to be robust to the three complicating factors. In order to experimentally validate our formalism, we perform the following experiment to evaluate its robustness to each of these three properties.

Accordingly, three experiments are performed to evaluate the effect of increase in overlap, coincidence and noise respectively. For each of these experiments, the three corresponding factors are increased gradually, such that at the first stage, the factor under consideration is absent and in the 11'th stage, its presence is significant. For each factor and for each stage, two sets of 10 simulations are performed. The first 10 simulations are performed with *generative process (-con)* which is regarded as the baseline, and the next 10 with *generative process*. The average accuracy with an error-band are plotted for each of the 11 stages and for increase in each factors respectively in Fig. 6.5.

### Results

It can be seen that the performance of the *generative process (-con)* degrades rapidly with increase in noise, coincidences and overlap. However, the performance of the *generative process* degrades very slowly till a certain point, and beyond this point degrades more gracefully than the baseline technique.

### Conclusions

To summarize, this experiment has demonstrated that for synthetic data, the generalized model (*generative process*) is far more robust to increase in complicating factors, as compared to the baseline (*generative process (-con)*) technique. Experiments with real data in Section 6.5 further demonstrates the robustness of the framework to these factors.

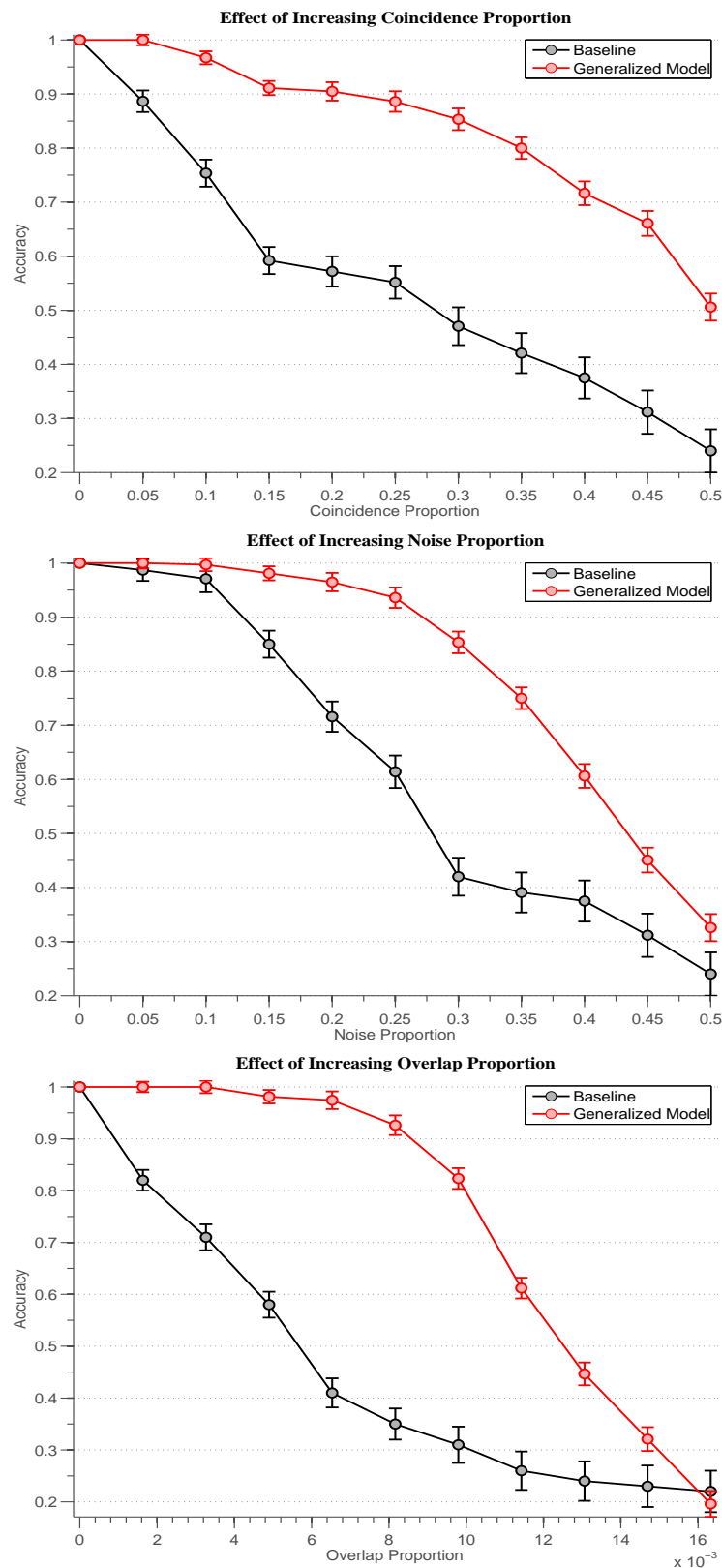


Figure 6.5: Evaluation of the robustness of the generative process to an increase in the presence of each complicating factor. The three plots show the accuracies of the baseline (generative process (-con)) and the generalized model (generative process) with increase in proportion of (i) coincidence; (ii) noise; (iii) overlap, respectively.

## 6.5 Evaluation of Event Learning on Real Data

The framework for learning event classes is evaluated on the airport apron dataset, as described in Section 6.2. An event cover ground truth for real data sets is obtained by manually annotating the set of tracks in terms of subsets which consists of tracklets. For the aircraft apron data set, an event cover ground truth consisting of event occurrences for three predetermined event classes is prepared.

These event classes are: (i) unloading; (ii) bridge/loader attaches (on) and detaches (off) from the plane; (iii) plane-puller attaches to the plane. These classes were predefined as *interesting* with respect to monitoring tasks that were prescribed by independent domain experts.

A ground truth consisting of the event covers for the aircraft apron video for these three classes was defined by other domain experts. This ground truth consists of an event cover, where an event occurrence of the cover is specified by a set of objects and the interval during which they interact. In addition, the event class for each occurrence is made available.

The framework for learning event classes is evaluated on the two real data sets – airport apron – as described in Section 6.2. Both the baseline approach and the generative process were evaluated. The parameters  $\gamma_1$  to  $\gamma_5$  in equation 4.4 are set to values, that were determined for synthetic data, using the approach described in Section 6.4.

Even though we could have used the ground-truth for real data and determined the parameters using the approach in Section 6.4, we deliberately chose not to use any knowledge about the events in the particular domain, for setting the parameters. This choice is motivated by the consideration that we would like our event learning framework to be *unsupervised*, where supervision means providing knowledge about the nature of the event classes and events for a domain.

Using these parameters, the procedure outlined in Section 4.4 for searching for the optimal interpretation is applied. With a geometric scheduling starting after 3000 iterations, it was observed that this gives sufficient time for the technique to perform initial exploration of the distribution. The temperature  $T$ , subject to a geometric scheduling with a fairly high value for  $\kappa = .9999$ , is used to ensure that the Markov chain does not get stuck at local maxima during the initial phase.

### Results

The generative process for the airport apron dataset gave rise to two event classes. The qualitative and quantitative evaluation of these event classes and the event cover are dis-

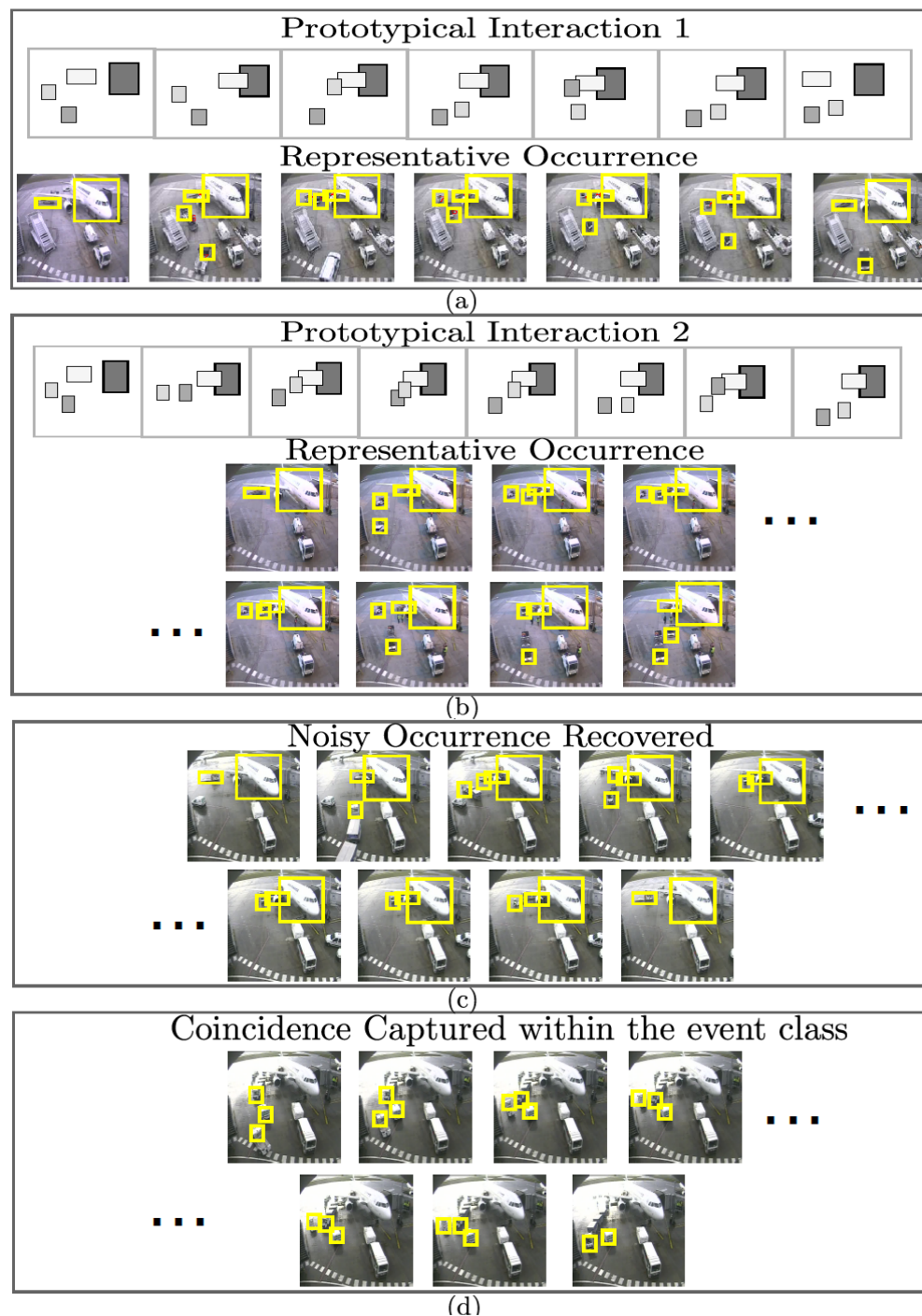


Figure 6.6: Representative occurrences of prototypes, coincidence and recovered noisy event are shown for event class 1. Note that in order to be able to display a sequence of over 7 images, we split them into two rows and use three dots (...) to represent the continuity. The two representative interactions that characterize the first event class are shown in (a) and (b) along with two of the respective occurrences, whose spatio-temporal relationships are identical to these prototypes. One of the noisy occurrences shown in (c), whose spatio-temporal relationships are not identical to (a) or (b), has been recovered by the framework and is considered as belonging to the first class. The first class has also erroneously captured coincidental interactions, which does not correspond to the unloading, one instance being shown in (d).



cussed below for this dataset.

1. *True Positives*: Class 1 is represented by two prototypical interactions with 18 occurrences in total. A total of 14 occurrences of these two prototypical interactions capture spatio-temporal relationships between instances of trolleys, loaders and planes, that correspond to what is semantically understood as *unloading*, according to the IATA definition. One representative occurrence out of these 14, for each prototypical interaction, is shown in Fig. 6.6(a) and (b) respectively.

Class 2 is represented by one prototypical interaction with 23 occurrences in total. A total of 16 occurrences of this prototypical interaction capture spatio-temporal relationships between instances of trolleys, loaders and planes, that correspond to what is semantically understood as a combination of two IATA events - bridge-on-off and plane-puller-on<sup>6</sup> One representative occurrence out of these 14, for this prototypical interaction, is shown in Fig. 6.7(a).

The generative process has been shown to have some robustness to observation noise. A representative noisy occurrence out of 5 for class 1, which is *not* spatio-temporally identical to either of the two prototypical interactions representing class 1 is shown in Fig. 6.6 (c). A representative noisy occurrence out of 6 for class 2 is shown in Fig. 6.7 (b).

2. *True Negatives*: The event learning procedure has been able to label several coincidental interactions correctly. Two examples of coincidental interactions are shown in Fig. 6.8.
3. *False Positives*: The generative process has erroneously captured some coincidental occurrences. A total of 4 coincidental occurrences which *do not* correspond to the semantic notion of unloading, were erroneously captured by class 1. One such representative occurrence is shown in Fig. 6.6 (d). A total of 7 coincidental occurrences which *do not* correspond to the semantic notion of bridge-on-off and plane-puller-on, were erroneously captured by class 1. One such representative occurrence is shown in Fig. 6.7 (c).
4. *False Negatives*: The generative process has not captured some occurrences that correspond to the unloading and bridge-on-off and plane-puller on class respec-

---

<sup>6</sup>It is reasonable to compare against this combination, since the ground truth event classes are specified at a certain level of granularity and it is unreasonable to expect the learned event classes to have the same granularity so that an exact comparison may be possible. On the other hand it is required to check whether whether the learned events are closely associated with the *official IATA* events determined by the domain experts.

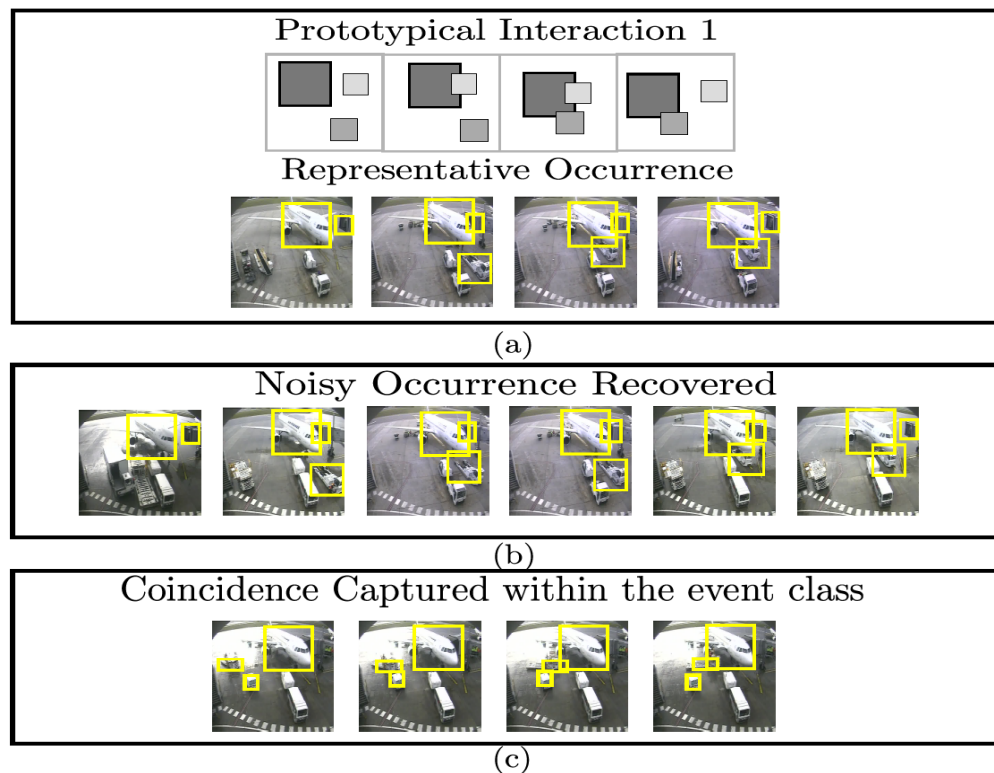


Figure 6.7: Representative occurrences of prototypes, coincidence and recovered noisy event are shown for event class 2. The one representative interaction that characterizes the second event class is shown in (a) along with one of the respective occurrence, whose spatio-temporal relationships are identical to these prototypes. One of the noisy occurrences shown in (b), whose spatio-temporal relationships is not identical to (a), has been recovered by the framework and is considered as belonging to the second class. The second class has also erroneously captured coincidental interactions, which does not correspond to the unloading, one instance being shown in (c).

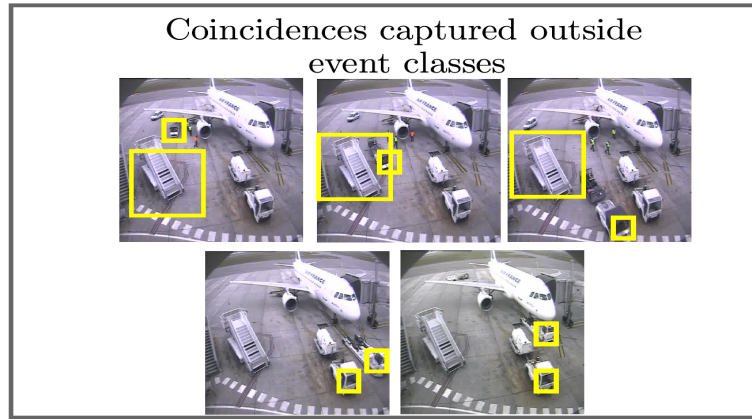


Figure 6.8: Example of two coincidental interactions, which are not classified as belonging to any of the learned event classes.

	Baseline	Generalized Model
Unloading	69.0,22.5, <b>39.9</b>	77.8,70.0, <b>73.7</b>
Bridge-on-off & plane-puller-on	58.5,50.0, <b>54.4</b>	69.6,80.0, <b>74.4</b>

Table 6.2: Evaluation of the baseline (generative process -(con)) and the generalized model (generative process) on real data, with respect to a predefined set of event classes: (i) unloading; (ii) bridge-on-off & plane-puller-on. The three entries are the precision, recall and the F-Measure respectively.

tively. Class 1 has not captured 6 occurrences of corresponding to unloading. Class 2 has not captured 6 occurrences of bridge-on-off and plane-puller-on.

It has been observed that these occurrences are not captured either because they are corrupted with noise that the framework has not been able to handle or because they are not frequent and similar enough to the prototypical interactions of the event classes.

The quantitative aspects of the above observations are summarized in table 6.2. The columns of table 6.2 are the two event classes in which these three predefined classes are present. The *first* of the pair of entries of table 6.2 is the precision, the second recall and the third is the F-measure, for each class.

## Conclusions

The following conclusions can be made from the above observations:

1. The two event classes represent distinct interactions that semantically correspond to IATA defined events.

2. These two event classes together describe a large part of what goes on in the scene between the type of vehicles under consideration.
3. These event classes capture similar ways of performing the same task and several of these similar ways correspond to the semantics captured by the respective event class.
4. However, the event classes have also erroneously captured some coincidences. Chapter 7 on future work discusses ways of addressing this problem.
5. The event learning framework has shown robustness to noise by capturing semantically relevant occurrences even though they are spatio-temporally different from the representative prototypical interaction for the classes.

## **6.6 Evaluating Functional Object Classes and Functional Relationships**

This section describes the evaluation of the procedure for learning functional object classes and functional relationships from the learned event classes, as described in Section 5.2. A total of 18 occurrences for class 1 and 23 occurrences for class 2 were input to the procedure for learning functional object classes.

The next step involved using the learned event classes and functional object classes to discover significant functional relationships, as described in Section 5.3. The object IDs are replaced by the respective functional classes, in the format given in Section 5.3. Then the frequent item set mining based procedure described in this section is applied to mine significant functional relationships between the object classes and event classes.

### **Results**

The procedure for learning functional object clustered the object tracks into five object classes object classes 1 to object classes 5, each of which are shown in Fig 6.9. From the results, it can be observed that the proposed scheme has been able to classify the objects in the scene into five functional categories that largely correspond to the commonly held categories i.e. trolley, loader, bridge, plane-puller and plane. In order to quantitatively assess the performance, the accuracy of these classes with respect to ground truth are summarized in table 6.3.

The procedure for learning functional relationships learned two functional relationships. The first one was the association between event class 1, object class 2, object class



Figure 6.9: The five functional object classes that were learned are shown. These five classes largely correspond to (i) trolley; (ii) bridge; (iii) loader; (iv) plane-puller; (v) plane. Table 6.3 displays the accuracies with respect to each of these five classes.

	Trolley	Loader	Bridge	Plane-puller	Plane
GenProc	82.7%	73.6%	78.2%	70.0%	93.75%

Table 6.3: Evaluation of functional object classes with respect to a predefined set of object classes.

5. The second is the association between event class 2, object class 3, object class 4 and object class 5.

Since most occurrences of event class 1 correspond to the semantic category of unloading, and most instances of object classes 1, 2 and 5 correspond to trolleys, loaders and planes, the following functional relationship has been learned: *Unloading takes place according to interactions (given by the event class 1) between object classes which mostly are loaders, trolleys and planes.*

Since most occurrences of event class 2 correspond to the semantic category, which is a combination of bridge-on-off and plan-puller-on, and most instances of object classes 3, 4 and 5 correspond to bridges, plane-pullers and planes, the following functional relationship has been learned: *bridge-on-off and plan-puller-on events take place according to interactions (given by the event class 2) between object classes which mostly are bridges, plane-pullers and planes.*

## **6.7 Overall Conclusion**

This chapter described the experimental evaluation of the framework for activity understanding described in chapters 3, 4 and 5 of this thesis. The HMM based procedure for obtaining a sequence of spatial relationships from video was evaluated. This experiment demonstrated that the HMM substantially outperforms existing techniques by exploiting the temporal information available in a video.

The procedure for learning event classes were evaluated both on synthetic data and a real dataset. The experiments on the synthetic datasets demonstrated the efficacy of event-like properties and its robustness to complicating factors. The experiments on the real datasets have demonstrated the framework's ability to discover semantically meaningful events from a real video dataset.

Finally, the procedures for learning functional object classes and functional relationships were evaluated. These experiments have demonstrated the framework's ability to learn semantically meaningful object functional object classes and functional relationships for real data.

# Chapter 7

## Summary and Future Work

---

This thesis proposed a framework for unsupervised learning of activities from video data. We introduced a representation which is able to model the changing qualitative relationships between a set of objects using the notion of interaction. Additionally we introduced an unsupervised learning scheme based on this representation. It involves the learning of a compact model for activities in terms of event classes and an activity graph. Finally, we introduced functional object classes, functional relationships and procedures for learning them.

In this chapter, we summarize the main contributions of our framework in section 7.1. We then discuss some limitations and possible ways of addressing these in the future, in section 7.2. We conclude this chapter with some final remarks in section 7.3.

### 7.1 Contributions

In this section we discuss the novel contributions of this thesis, focusing on three individual aspects. These are: (i) representation of activities; (ii) the generative process for modelling activities and (iii) the framework for unsupervised event learning.

#### 7.1.1 Representation of Activities

We consider the following as novel contributions with regards to the representation of activities, event classes and events.

**Application of Qualitative Spatio-temporal Relationships to Activity Understanding.**

This work contributes to the rather sparse research [Galata et al., 1999, Fernyhough et al., 2000a, Southey and Little, 2007] on the application of qualitative spatial relationships to activity understanding, by modelling spatial relations between objects instead of their individual behaviours, for learning about activities. This is in contrast to much previous work [Hamid et al., 2009, Oh and Hoogs, 2010, Wang et al., 2006] in the area of activity analysis that have modelled behaviour of individual objects in the form of sequences of locations, or as groups of pixels with certain motion characteristics [Wang et al., 2009, Xiang and Gong, 2006].

Another contribution is the modelling of complex temporal dependencies by using Allen's temporal relationships [Allen, 1983]. In contrast, previous work that use qualitative spatial relationships for learning about activities [Galata et al., 1999, Fernyhough et al., 2000a], assume a simple sequence-based representation.

Those approaches that do not simplify space using qualitative spatial relations are able to afford only a simple model of time. Thus, much previous work [Hongeng and Nevatia, 2001, Oliver et al., 2000, Vogler and Metaxas, 2001, Wang et al., 2009, Xiang and Gong, 2006, Xie et al., 2003] assume a low-order (mostly first order) Markovian temporal dependency.

To summarize, this work simplifies space by abstracting only qualitatively interesting spatial relationships. Simplifying space makes it possible to be able to afford the representation of complex temporal dependencies (in form of Allen's temporal algebra). This makes it possible to have a rich representation of very complex interactions that characterize human activities.

**Representing Interactions.** Second is the concept of an *interaction* as a distinct sequence of qualitative states between a set of regions, and a way of computing similarity between interactions. This work defines interactions in a novel way, as a subset of qualitative spatio-temporal relationships. Such a definition makes it possible to have a well defined similarity measure. The similarity measure enables the application of learning algorithms to interactions. Furthermore, the interaction graph introduced is another novelty that provides a computationally efficient way of performing learning with interactions.

**Modelling Interactions using a HMM.** Third is the use of a Hidden Markov Model (HMM) to model a probabilistic relationship between an interaction and its embedding. This probabilistic relationship makes it possible to learn events despite the presence of noise. A related contribution is that the HMM provides a way of obtaining stable qualita-



tive spatial relationships from video. In contrast, previous work as reviewed in [Cohn and Hazarika, 2001], has only provides a way of obtaining a single crisp representation from a static image, without incorporating temporal information available in a video.

### 7.1.2 A Generative Process for Modelling Activities

This thesis directly contributes to the field of learning about activities utilizing a generative process. We consider the following aspects as novel:

**Even Classes as Distributions over Interactions Graphs.** The representation of event classes as probability distributions over interaction graphs is novel. To our knowledge, there has been no previous work that defines a probability distribution over spatio-temporal relationships or interactions. We have also not come across any technique that characterizes likely event models with event-like properties to distinguish from coincidences, as introduced in this work.

**Concept of Activity Graphs.** Second is the concept of an activity graph for representing an entire stretch of real-world activity. The activity graph provides an elegant way of modelling the entire spatio-temporal layout of the activity in one graph structure. It makes it possible to separate the modelling of its relationships with an event model, a model of overlap and coincidence, from its relationships with tracks (in order to model observation noise). To the best of our knowledge, we have not come across any work that models these aspects of activities, in an explicit manner.

### 7.1.3 Unsupervised Event Learning

We contribute to the topic of unsupervised event learning in the following three ways.

**Unsupervised Setting.** The definition of an unsupervised setting whose aim is to learn the event classes and the event cover associated with activities is novel. To the best of our knowledge, we have not come across any work that searches simultaneously for an optimal event cover and an optimal event model.

**Optimal Interpretation via MCMC.** A contribution of this work is the use of MCMC for activity understanding. Previous work such as [Yu and Medioni, 2008] have used MCMC to find an optimal covering of detected blobs with a set of tracks. This work uses MCMC to simultaneously find an optimal event cover of tracks and an event model. It

is possible that such a procedure could also be potentially be extended to other areas of application.

### 7.1.4 Functional Object Classes and Functional Relationships

A further novel aspect of this work is a technique for learning functional object classes and functional relationships from the learned event classes.

While much work has focussed on learning and recognizing objects based on affordances, this work introduces the idea of functional object categories that relate to how objects are actually used with respect to events. The key insight to that part of the thesis establishes is that objects can be represented in terms of the functional roles they play with respect to the events that have been learned. This representation facilitates measuring the functional similarity of objects that play a similar role with respect to event instances in the same class. The emergent idea is that of functional object classes whose object instances are functionally similar with respect to the learned events.

Functional object classes are an important notion as their association with event classes leads to a holistic definition of activities, where events are not just interactions between any set of objects, but between instances of certain functional object classes. This idea is explored in chapter 5 and provides an intuitive framework for describing activities in the real world.

## 7.2 Future Work

Throughout this thesis, we have presented a framework for the unsupervised understanding of activities from video data. Possible future work extends into various directions. This section serves to summarize these directions.

**Additional Modelling Capabilities.** In our framework, qualitative spatial relations are prescribed manually to specific datasets. Thus each time the framework is applied to a new data set, an appropriate set of spatial relations would need to be manually prescribed. Generally speaking, the possibility exists that these manually assigned relations may not be optimal for characterizing the events in that dataset.

This problem can possibly be resolved by replacing the manually specified qualitative relationships by a set of automatically discretized relationships, which are learnt from training data. A similar idea has been explored by Galata et. al. in [Galata et al., 1999] by utilizing a HMM.

**Distinction of Events.** Our work strictly focuses on a spatio-temporal distinction of events. However, other possible types of distinctions can be considered, e.g. the types of objects that are engaged in the events or the location of the events.

We can address this issue by including additional qualitative relationships or features such as direction, position (zones), objects types into the framework.

**Distribution of Events.** The activity graphs introduced in this thesis assume that events are independent of each other and therefore randomly distributed with respect to each other in space and time. The proposed notions of minimal overlap and coincidental interactions between them can be looked at as a set of constraints of these activity graphs. However, in many real-world scenarios activities are often composed of events and sub-events that are structurally related to each other.

Future work therefore can consider a more complex modelling of events in an activity graph to account for these aspects.

**Characterizations of Events.** Our system favours only those events that have certain properties such as being relatively large, interactive and frequent. We do realize that, outside of the scope of this thesis, there may very well be domains where events have to be considered that do not fulfil these requirements. One way of addressing this limitation is to explore alternative characterizations of events, which might very well be domain specific.

**Fixed Tracks.** Our current implementation assumes a given and fixed set of tracks of objects. Additionally we either detect objects by using detectors trained prior to event discovery or by utilizing much simpler, yet somewhat supervised, colour segmentation techniques. Even though we do not use any information provided by the detectors about the class membership of each object, it would be desirable to point a camera at a video without providing any additional information about the types of objects involved.

Unsupervised object discovery from video [Southey and Little, 2006] is the obvious choice to resolve this situation. Note also that the proposed framework does not offer the ability to influence the tracking itself. Interweaving unsupervised object discovery with object tracking and the proposed framework therefore seems to be a very challenging yet desirable goal of future work.

**Functional Similarity.** In our framework two objects, such as a piece of vegetable and a slice of bread, are regarded as being functionally similar if they are taken away from the

same plate. We realize that this is overly restrictive.

It therefore seems desirable to learn functional roles of objects in a scene as well as to tightly integrate such a model of roles into our system. A possibly outline of a solution is the inclusion of the learning of functional object classes (Chapter 5) into the event discovery process.

### **7.3 Concluding Remarks**

This thesis introduced a framework for obtaining events from video data in an unsupervised fashion as well as modelling function object classes. We have experimentally validated that we can indeed discover events from video data and verified a relative strong robustness to noise introduced by low-level image processing techniques. Therefore we regard the proposed framework as a small but significant step towards activity understanding.

# Appendix A

## The Metropolis Hastings Algorithm

---

The Metropolis Hastings algorithm addresses the problem of drawing samples  $S$  from a sample space with a posterior probability  $\varrho(S)$ , when the normalization term for the posterior probability is hard to compute. In such cases the posterior probability is given only in an unnormalized form  $\varrho^*(S)$ , where the normalized density is

$$\varrho(S) = \frac{\varrho^*(S)}{\sum_{S' \in \Psi} \varrho^*(S')}$$

A sampling of this nature is accomplished by the use of Markov chains with state transition matrix  $\Pi$  where  $\Pi(S_n, S_{n+1}) = P(S_{n+1} | S_0, \dots, S_n)$  has the property that the future state  $S_{n+1}$  depend only on the current state  $S_n$  and not on past states. The unnormalized density  $\varrho^*$  is sufficient for the simulation of the Markov chain to converge to the stationary density if and only if both the following conditions hold [Neal, 1993]: (i) The Markov chain is ergodic under certain conditions [Neal, 1993], that is any state is reachable from any other state in a finite number of transitions; (ii) the normalized density follows the detailed balance condition, that is

$$\varrho(S) \Pi(S, S') = \varrho(S') \Pi(S', S)$$

Since the normalization constant cancels out in condition (ii), it implies that as long as  $\varrho^*(S)$  satisfies this condition, the Metropolis algorithm will converge to the stationary distribution. The transition probability  $\Pi(S, S')$  is expressed as the product of: (i) a proposal distribution  $Q(S, S')$  that describes the probability of moving from state  $S$  to  $S'$ ; (ii)  $\zeta(S, S')$  which is the probability of accepting the move from state  $S$  to  $S'$ . In order that the detailed balance is ensured, the acceptance probability is given by

$$\zeta(S, S') = \min \left( 1, \frac{Q(S', S)\varrho^*(S')}{Q(S, S')\varrho^*(S)} \right)$$

In the case when the M-H algorithm is used for optimization by means of sampling, the Markov chain may be prone to get stuck at local maxima and can be extremely slow. Simulated annealing was introduced to address this problem by changing the density in such a manner so that more transitions are accepted initially, till later on, only those with improvements in the density are accepted. This change in the density is designed to try and ensure that regions of low density are explored in the beginning without getting stuck in local maxima, and as the number of iterations increase, the tendency to move towards regions of high density increases. Such a change in density can be brought about by the following modification in the acceptance probability with the introduction of a temperature term  $T$  which is gradually reduced by a geometric schedule  $T = \kappa^{numiter}$  with  $0 < \kappa < 1$ , and  $numiter$  is the number of iterations. The geometric schedule is initiated after a certain minimum number of iterations, to allow the Markov chain to explore regions of low density at the beginning.

$$\zeta(S, S') = \min \left( 1, \left( \frac{Q(S', S)\varrho^*(S')}{Q(S, S')\varrho^*(S)} \right)^{\frac{1}{T}} \right)$$

# Appendix B

## Generation of Synthetic Datasets

---

Synthetic activities are simulated by the following procedure<sup>1</sup>. The following are the steps undertaken to generate synthetic data:

1. The number of event classes  $nc$  is first ascertained. This number is sampled uniformly at random (u.a.r) from the set  $\{4 - 10\}$ .
2. The event classes  $\mathcal{C} = \{c_1, c_2, \dots, c_{nc}\}$  are randomly assigned prior probabilities. Activities are modelled in terms of some event classes being very common and others less and less common. Therefore, the prior probability is modelled as a normalized geometric distribution  $P(c_i) = \frac{p(1-p)^i}{\sum_i^{nc} p(1-p)^i}$  where  $c_i \in \mathcal{C}$  and  $p$  is fixed to be equal to .5.
3. The total number of event graphs and events  $n_e$  is sampled u.a.r from the set  $\{40 - 70\}$ .
4. The number of event graphs  $np_i$  for each class  $c_i$  is given by  $n_e P(c_i)$ .
5. A bag of event graphs  $\mathcal{B}$  for each event class  $c_i$  is produced in the following steps. The first step is to assign a geometric distribution for each class  $c_i$ . Accordingly, a parameter  $p_i$  assigning a geometric distribution to each class  $c_i$ , so that for the  $np_i$  event graphs, the number of event graphs  $g_j$  is equal to  $(p_i(1 - p_i)^j)np_i$ .
6. Having obtained these statistics, the next step involves sampling the event graphs. This is carried out by choosing one distinct representative prototype for each class  $c_i$  as follows.

---

<sup>1</sup>This is similar in spirit to the generative process outlined in section 4.3 of chapter 4

Since event classes are well characterized by large interaction graphs as motivated in section 4.2, the number of region histories  $nr$  is sampled u.a.r from  $\{3 - 6\}$  and the number fo spatial changes  $ns$  is sampled u.a.r from  $\{4 - 12\}$ .

Another desirable property of event classes is that the their representative interaction graphs have high interactivity scores. Accordingly an interaction graph with size  $(nr, ns)$  is sampled from a distribution over  $\Gamma$  (described in section 4.3) that is proportional to their interactivity scores <sup>2</sup>. In this manner, the first representative prototype for the first event class is obtained.

7. Subsequently, one representative prototype for each class is obtained in the same manner with the added requirement that it is farthest (as queried by the pairwise distance matrix between the graphs in  $\mathcal{B}$ ) from the representative prototypes for all previous classes.
8. Other representative prototypes for each class are obtained by sampling the number of prototypes for class in such a way that they are relatively close to the representative prototype (again by querying the distance matrix between the graphs in  $\mathcal{B}$ ). For each representative prototype  $g_j$  belonging to  $c_i$ 'th class,  $(p_i(1 - p_i)^j)np_i$  copies of this prototype are obtained.
9. The next stage is the construction of an activity graph. The proportion of overlap  $n_{ov}$  is sampled from a geometric distribution
10. A small proportion of coincidental interactions  $n_{co}$  are obtained by sampling interaction graphs such that they are not large, and interactive at the same time. For size,  $nr$  and  $ns$  are sampled u.a.r from the sets  $\{2 - 3\}$  and  $\{2 - 4\}$ .
11. Once the event graphs and the coincidental interactions have been sampled, they are randomly dispersed in space and time such that the embeddings encode the proportion of overlap  $n_{ov}$  and coincidental interactions  $n_{co}$ .
12. Noise is obtained by randomly distorting a small proportion  $n_{no}$  of events. This is done in such a way that there is a single valid change in spatial relationship between a pair of tracklets.

The procedure described above generates a synthetic data of activities. This process is referred to as a *simulation* and is given by  $\mathcal{S} = (\hat{\mathcal{E}}, \hat{\mathcal{L}}, n_{co}, n_{ov}, n_{no})$ , where  $\hat{\mathcal{E}}$  is the original event cover,  $\hat{\mathcal{L}}$  the labelling of the event cover. The other three terms  $n_{co}, n_{ov}, n_{no}$  stand for the proportion of coincidence, overlaps and noise.

<sup>2</sup>so that interaction graphs with higher values of interactivity are more likely to be sampled



# Appendix C

## Evaluation Scores for Event Covers and Event Classes

---

The following paragraphs describe a way of evaluating an event cover. The second aspect is an evaluation score for event classes.

### Evaluation of an Event Cover

Let  $E = \{e_1, \dots, e_p\}$  and be the set of events in the event cover  $E$ , and the ground-truth  $GT = \{gt_1, \dots, gt_q\}$ , respectively. Let  $E \cap GT$  be the number of events in  $E$  that completely overlap with those in  $GT$ , where  $e_i \in E$  and  $gt_j \in GT$  are regarded as completely overlapping if they are composed of the same set of tracklets.

The  $F$  measure [Baeza-Yates and Ribeiro-Neto, 1999] is well suited for computing the accuracy of  $E$  with respect to  $GT$  and is defined as the harmonic mean of precision  $Pr$  and recall  $Re$ , as given below.

$$F = \frac{2PrRe}{Pr + Re}, \quad Pr = \frac{\|E \cap GT\|}{\|E\|}, \quad Re = \frac{\|E \cap GT\|}{\|GT\|}$$

Recall measures how much of the  $GT$  is covered by  $E$ , while precision measures how much of the  $E$  covers the  $GT$  and both of these take values between 0 and 1. While  $Pr$  can be low,  $Re$  can be high and vice versa, the value of  $F$ -measure is high when the values of both  $Pr$  and  $Re$  are high.

## Evaluation of Event Classes

The F-measure described above does not evaluate the accuracy of the event cover with respect to the assignment of events to a set of event classes. The Rand index [Rand, 1971] is well suited evaluation score for this purpose. A true positive  $TP$  decision assigns two events belonging to the same class in the ground truth to the same class in the learned event classes. A true negative  $TN$  decision assigns two events belonging to different classes in the ground truth to different event classes in the learned set. There are two types of errors. A  $FP$  decision assigns two events belonging to different clusters in the ground truth to the same event class in the learned set. A  $FN$  decision assigns two events belonging to the same event class in the ground truth to different event classes. The Rand index  $RI$  is a measure of accuracy i.e. the percentage of decisions that are correct:

$$RI = \frac{TP + TN}{TP + FP + FN + TN} \quad (\text{C.1})$$

# Appendix D

## Temporal Relationships between two Intervals

---

The temporal relationship  $\text{Temporal}(i_1, i_2)$  between two intervals  $i_1, i_2$  are given by:

$$\text{Before} \iff \text{End}(i_1) < \text{Start}(i_2) - 1$$

$$\text{Meets} \iff \text{End}(i_1) < \text{Start}(i_2) - 1$$

$$\text{Overlap} \iff \text{End}(i_1) < \text{Start}(i_2) \wedge \text{Start}(i_2) < \text{End}(i_1)$$

$$\text{Starts} \iff \text{Start}(i_1) < \text{Start}(i_2) \wedge \text{End}(i_1) < \text{End}(i_2)$$

$$\text{During} \iff \text{Start}(i_2) < \text{Start}(i_1) \wedge \text{End}(i_1) < \text{End}(i_2)$$

$$\text{Finishes} \iff \text{Start}(i_1) < \text{Start}(i_2) \wedge \text{End}(i_1) < \text{End}(i_2)$$

$$\text{Equal if } i_1 = i_2$$

# Bibliography

- [Agrawal and Srikant, 1994a] Agrawal, R. and Srikant, R. (1994a). Fast algorithms for mining association rules. In *Proceedings of the International Conference on Very Large Data Bases (VLDB)*.
- [Agrawal and Srikant, 1994b] Agrawal, R. and Srikant, R. (1994b). Fast algorithms for mining association rules in large databases. In *Proceedings of the International Conference on Very Large Data Bases (VLDB)*.
- [Allen, 1983] Allen, J. (1983). Maintaining knowledge about temporal intervals. *Commun. ACM*, 26(11):832–843.
- [Antanas et al., 2009] Antanas, L., Thon, I., van Otterlo, M., Landwehr, N., and De Raedt, L. (2009). Probabilistic logical sequence learning for video. In *Proceedings of the International Conference on Inductive Logic Programming (ILP)*.
- [Baeza-Yates and Ribeiro-Neto, 1999] Baeza-Yates, R. and Ribeiro-Neto, B. (1999). *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc.
- [Blei et al., 2003] Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *Machine Learning*, 3:993–1022.
- [Brand, 1996] Brand, M. (1996). Understanding manipulation in video. In *IEEE International Conference on Automatic Face and Gesture Recognition*.
- [Bringmann and Nijssen, 2008] Bringmann, B. and Nijssen, S. (2008). What is frequent in a single graph?. In *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*.
- [Burges, 1998] Burges, C. J. C. (1998). A tutorial on support vector machines for pattern recognition. *Proceedings of Data Mining and Knowledge Discovery (DMKD)*, 2(2):121–167.

- [Buxton and Gong, 1995] Buxton, H. and Gong, S. (1995). Visual surveillance in a dynamic and uncertain world. *Artificial Intelligence*, 78(1-2):431–459.
- [Buzan et al., 2004] Buzan, D., Sclaroff, S., and Kollios, G. (2004). Extraction and clustering of motion trajectories in video. In *Proceedings of the International Conference on Pattern Recognition (ICPR)*.
- [Cao et al., 2004] Cao, D., Masoud, O., Boley, D., and Papanikolopoulos, N. (2004). Online motion classification using support vector machines. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.
- [Carreira and Sminchisescu, 2010] Carreira, J. and Sminchisescu, C. (2010). Constrained parametric min-cuts for automatic object segmentation. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*.
- [Celik et al., 2008] Celik, H., Hanjalic, A., and Hendriks, B. S. (2008). Online training of object detectors from unlabeled surveillance video. In *Workshop on Online Learning for Computer Vision, CVPR*.
- [Celik et al., 2009] Celik, H., Hanjalic, A., and Hendriks, E. A. (2009). Unsupervised and simultaneous training of multiple object detectors from unlabeled surveillance video. *Computer Vision and Image Understanding (CVIU)*, 113(10):1076–1094.
- [Ceri et al., 1989] Ceri, S., Gottlob, G., and Tanca, L. (1989). What you always wanted to know about datalog (and never dared to ask). *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 1(1):146–166.
- [Cohn et al., 1997] Cohn, A. G., Bennett, B., Gooday, J., and Gotts, N. M. (1997). Qualitative spatial representation and reasoning with the region connection calculus. *Geoinformatics*, 1(3):275–316.
- [Cohn and Hazarika, 2001] Cohn, A. G. and Hazarika, S. M. (2001). Qualitative spatial representation and reasoning: An overview. *Fundamenta Informaticae*, 46(1-2):1–29.
- [Cohn et al., 2003] Cohn, A. G., Magee, D., Galata, A., Hogg, D. C., and Hazarika, S. (2003). Towards an architecture for cognitive vision using qualitative spatio-temporal representations and abduction. In Freksa, C., Habel, C., and Wender, K., editors, *Spatial Cognition III*, pages 232–248. Springer.
- [Cook and Holder, 2007] Cook, D. J. and Holder, L. B. (2007). *Mining Graph Data*. Wiley-Interscience.

- [Dalal and Triggs, 2005] Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*.
- [Dasarathy, 1990] Dasarathy, B.-V. (1990). *Nearest neighbor (NN) norms: NN pattern classification techniques*.
- [Dehaspe and Toivonen, 1999] Dehaspe, L. and Toivonen, H. (1999). Discovery of frequent datalog patterns. *Proceedings of Data Mining and Knowledge Discovery (DMKD)*, 3(1):7–36.
- [Deshpande et al., 2005] Deshpande, M., Kuramochi, M., Wale, N., and Karypis, G. (2005). Frequent substructure-based approaches for classifying chemical compounds. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 17(8):1036–1050.
- [Dietterich et al., 1997] Dietterich, T. G., Lathrop, R. H., Lozano-Perez, T., and Pharmaceutical, A. (1997). Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1-2):31–71.
- [Dubba et al., 2010] Dubba, K. S. R., Cohn, A. G., and Hogg, D. C. (2010). Event model learning from complex videos using ilp. In *Proceedings of the European Conference on Artificial Intelligence (ECAI)*.
- [Dunkel and Soparkar, 1999] Dunkel, D. and Soparkar, N. (1999). Data organization and access for efficient data mining. In *Proceedings of the IEEE International Conference on Data Engineering (ICDE)*.
- [Duygulu et al., 2002] Duygulu, P., Barnard, K., de Freitas, J. F. G., and Forsyth, D. A. (2002). Object recognition as machine translation: learning a lexicon for a fixed image vocabulary. In *Proceedings of the European Conference on Computer Vision (ECCV)*.
- [Fergus et al., 2005] Fergus, R., Fei-Fei, L., Perona, P., and Zisserman, A. (2005). Learning object categories from google’s image search. In *Proceedings of the International Conference on Computer Vision (ICCV)*.
- [Fernyhough et al., 2000a] Fernyhough, J., Cohn, A., and Hogg, D. (2000a). Constructing qualitative event models automatically from video input. *Image and Vision Computing*, 18:81–103.
- [Fernyhough et al., 2000b] Fernyhough, J. H., Cohn, A. G., and Hogg, D. C. (2000b). Constructing qualitative event models automatically from video input. *Image and Vision Computing*, 18(2):81–103.

- [Figueiredo and Jain, 2002] Figueiredo, M. and Jain, A. (2002). Unsupervised learning of finite mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 24(3):381–396.
- [Galata et al., 2002] Galata, A., Cohn, A. G., Magee, D. R., and Hogg, D. C. (2002). Modeling interaction using learnt qualitative spatio-temporal relations and variable length markov models. In *Proceedings of the European Conference on Artificial Intelligence (ECAI)*.
- [Galata et al., 1999] Galata, A., Johnson, N., and Hogg, D. C. (1999). Learning structured behaviour models using variable length markov models. In *Proceedings of the IEEE International Workshop on Modelling People*.
- [Gärtner et al., 2003] Gärtner, T., Flach, P. A., and Wrobel, S. (2003). On graph kernels: Hardness results and efficient alternatives. In *Proceedings of the Conference On Learning Theory (COLT)*, pages 129–143.
- [Goldenberg et al., 2002] Goldenberg, R., Kimmel, R., Rivlin, E., and Rudzsky, M. (2002). ‘dynamism of a dog on a leash’ or behavior classification by eigendecomposition of periodic motions. In *Proceedings of the European Conference on Computer Vision (ECCV)*.
- [Gonzalez et al., 2002] Gonzalez, J. A., Holder, L. B., and Cook, D. J. (2002). Experimental comparison of graph-based relational concept learning with inductive logic programming systems. In *Proceedings of the International Conference on Inductive Logic Programming (ILP)*.
- [Grünwald, 2005] Grünwald, P. (2005). A tutorial introduction to the minimum description length principle. In *Advances in Minimum Description Length: Theory and Applications*.
- [Hamid et al., 2009] Hamid, R., Maddi, S., Johnson, A., Bobick, A., Essa, I., and Isbell, C. (2009). A novel sequence representation for unsupervised analysis of human activities. *Art. Int. Journal*.
- [Hasan and Zaki, 2009] Hasan, M. A. and Zaki, M. J. (2009). Output space sampling for graph patterns. In *Proceedings of the International Conference on Very Large Data Bases (VLDB)*.
- [Heckerman, 1999] Heckerman, D. (1999). *A tutorial on learning with Bayesian networks*. MIT Press.

- [Hervieu et al., 2008] Hervieu, A., Bouthemy, P., and Le Cadre, J.-P. (2008). Video event classification and detection using 2d trajectories. In *Proceedings of the International Conference on Computer Vision Theory and Applications (VISAPP)*.
- [Holder et al., 1994] Holder, L. B., Cook, D. J., and Djoko, S. (1994). Substructure discovery in the SUBDUE system. In *Proceedings of the AAAI Workshop on Knowledge Discovery in Databases*.
- [Hongeng and Nevatia, 2001] Hongeng, S. and Nevatia, R. (2001). Multi-agent event recognition. In *Proceedings of the International Conference on Computer Vision (ICCV)*.
- [Hongeng and Nevatia, 2003] Hongeng, S. and Nevatia, R. (2003). Large-scale event detection using semi-hidden markov models. In *Proceedings of the International Conference on Computer Vision (ICCV)*.
- [Hübler et al., 2008] Hübler, C., Kriegel, H.-P., Borgwardt, K. M., and Ghahramani, Z. (2008). Metropolis algorithms for representative subgraph sampling. In *Proceedings of the International Conference on Data Mining (ICDM)*.
- [Inokuchi et al., 2000] Inokuchi, A., Washio, T., and Motoda, H. (2000). An apriori-based algorithm for mining frequent substructures from graph data. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD)*.
- [Joo and Chellappa, 2006] Joo, S.-W. and Chellappa, R. (2006). Attribute grammar-based event recognition and anomaly detection. In *Proceedings of Semantic Learning Applications in Multimedia, CVPR*.
- [Kersting et al., 2008] Kersting, K., De Raedt, L., Gutmann, B., Karwath, A., and Landwehr, N. (2008). Relational sequence learning. In *Probabilistic Inductive Logic Programming*, pages 28–55. Springer.
- [King et al., 2001] King, R. D., Srinivasan, A., and Dehaspe, L. (2001). Warmr: a data mining tool for chemical data. *Journal of Computer-Aided Molecular Design*, 15(2):173–181.
- [Kirkpatrick et al., 1983] Kirkpatrick, S., Gelatt, C. D. J., and Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220:671–680.



- [Kitani et al., 2008] Kitani, K. M., Sato, Y., and Sugimoto, A. (2008). Recovering the basic structure of human activities from noisy video-based symbol strings. *International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI)*, 22(8):1621–1646.
- [Köbler et al., 1993] Köbler, J., Schöning, U., and Torán, J. (1993). *The Graph Isomorphism Problem: Its Structural Complexity*. Birkhauser Boston.
- [Kudo et al., 2004] Kudo, T., Maeda, E., and Matsumoto, Y. (2004). An application of boosting to graph classification. In *Proceedings of Neural Information Processing Systems (NIPS)*.
- [Kuramochi and Karypis, 2001] Kuramochi, M. and Karypis, G. (2001). Frequent sub-graph discovery. In *Proceedings of the International Conference on Data Mining (ICDM)*.
- [Lavee et al., 2009] Lavee, G., Rivlin, E., and Rudzsky, M. (2009). Understanding video events: a survey of methods for automatic interpretation of semantic occurrences in video. *IEEE Transactions on Systems, Man, and Cybernetics*, 39(5):489–504.
- [Lee and De Raedt, 2002] Lee, S. D. and De Raedt, L. (2002). Constraint based mining of first order sequences in SeqLog. In *SIGKDD Workshop on Multi-Relational Data Mining*.
- [Lv et al., 2006] Lv, F., Song, X., Wu, B., Singh, V., and Nevatia, R. (2006). Left-luggage detection using bayesian inference. In *Proceedings of the International Workshop on Performance Evaluation of Tracking and Surveillance, CVPR*.
- [Mallapragada et al., 2010] Mallapragada, P. K., Jin, R., and Jain, A. K. (2010). Non-parametric mixture models for clustering. In *Structural, Syntactic, and Statistical Pattern Recognition, Joint IAPR International Workshop*.
- [Moore and Essa, 2002] Moore, D. and Essa, I. (2002). Recognizing multitasked activities from video using stochastic context-free grammar. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.
- [Muggleton and De Raedt, 1994] Muggleton, S. and De Raedt, L. (1994). Inductive logic programming: Theory and methods. *Journal of Logical Programming*, 19/20:629–679.
- [Neal, 1993] Neal, R. M. (1993). Probabilistic inference using markov chain monte carlo methods. Technical Report CRG-TR-93-1, University of Toronto.

- [Needham et al., 2005] Needham, C. J., Santos, P. E., Magee, D. R., Devin, V. E., Hogg, D. C., and Cohn, A. G. (2005). Protocols from perceptual observations. *Artificial Intelligence*, 167(1-2):103–136.
- [Ng et al., 2001] Ng, A. Y., Jordan, M. I., and Weiss, Y. (2001). On spectral clustering: Analysis and an algorithm. In *Proceedings of Neural Information Processing Systems (NIPS)*.
- [Oh and Hoogs, 2010] Oh, S. and Hoogs, A. (2010). Unsupervised learning of activities in video using scene context. In *Proceedings of the International Conference on Pattern Recognition (ICPR)*.
- [Oliver et al., 2000] Oliver, N. M., Rosario, B., and Pentland, A. P. (2000). A bayesian computer vision system for modeling human interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 22:831–843.
- [Ott and Everingham, 2009] Ott, P. and Everingham, M. (2009). Implicit color segmentation features for pedestrian and object detection. In *Proceedings of the International Conference on Computer Vision (ICCV)*.
- [Parikh and Chen, 2007] Parikh, D. and Chen, T. (2007). Unsupervised learning of hierarchical semantics of objects (hSOs). In *Beyond Patches Workshop, CVPR*.
- [Pittore et al., 1999] Pittore, M., Basso, C., and Verri, A. (1999). Representing and recognizing visual dynamic events with support vector machines. In *Proceedings of the International Conference on Image Analysis and Processing (ICIAP)*.
- [Rabiner and Juang, 1993] Rabiner, L. and Juang, B. H. (1993). *Fundamentals of Speech Recognition*. Prentice Hall, Englewood Cliffs, NJ.
- [Rabiner, 1989] Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings IEEE*, pages 257–286.
- [Rand, 1971] Rand, W. M. (1971). Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850.
- [Randell et al., 1992] Randell, D. A., Cui, Z., and Cohn, A. G. (1992). A spatial logic based on regions and connection. In *Proceedings of the Conference on Knowledge Representation and Reasoning (KR)*.

- [Ridge et al., 2009] Ridge, B., Skočaj, D., and Leonardis, A. (2009). Unsupervised learning of basic object affordances from object properties. In *Proceedings of the Fourteenth Computer Vision Winter Workshop*.
- [Russell et al., 2006] Russell, B. C., Efros, A. A., Sivic, J., Freeman, W. T., and Zisserman, A. (2006). Using multiple segmentations to discover objects and their extent in image collections. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*.
- [Ryoo and Aggarwal, 2006] Ryoo, M. S. and Aggarwal, J. K. (2006). Recognition of composite human activities through context-free grammar based representation. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*.
- [Saxena et al., 2008] Saxena, A., Driemeyer, J., and Ng, A. Y. (2008). Robotic grasping of novel objects using vision. *International Journal of Robotics Research*, 27(2):157–173.
- [Schapire, 1990] Schapire, R. E. (1990). The strength of weak learnability. *Machine Learning*, 5:197–227.
- [Scholkopf, 2001] Scholkopf, B. (2001). The kernel trick for distances. In *Proceedings of Neural Information Processing Systems (NIPS)*.
- [Shet et al., 2005] Shet, V., Harwood, D., and Davis, L. (2005). Vidmap: Video monitoring of activity with prolog. In *Proceedings of the IEEE International Conference on Advanced Video and Signal based Surveillance (AVSS)*.
- [Shi and Malik, 2000] Shi, J. and Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 22(8):888–905.
- [Smith et al., 2005] Smith, P., Lobo, N., and Shah, M. (2005). Temporalboost for event recognition. In *Proceedings of the International Conference on Computer Vision (ICCV)*.
- [Southey and Little, 2006] Southey, T. and Little, J. J. (2006). Object discovery through motion, appearance and shape. In *AAAI Workshop on Cognitive Robotics*.
- [Southey and Little, 2007] Southey, T. and Little, J. J. (2007). Learning qualitative spatial relations for object classification. In *From Sensors to Human Spatial Concepts, IROS Workshop*.

- [Sridhar et al., 2008] Sridhar, M., Cohn, A. G., and Hogg, D. C. (2008). Learning functional object-categories from a relational spatio-temporal representation. In *Proceedings of the European Conference on Artificial Intelligence (ECAI)*.
- [Stark et al., 2008] Stark, M., Lies, P., Zillich, M., Wyatt, J., and Schiele, B. (2008). Functional object class detection based on learned affordance cues. In *Proceedings of the International Conference on Computer Vision Systems (ICVS)*.
- [Stolcke, 1995] Stolcke, A. (1995). An efficient probabilistic context-free parsing algorithm that computes prefix probabilities. *Computational Linguistics*, 21(2):165–201.
- [Todorovic and Ahuja, 2006] Todorovic, S. and Ahuja, N. (2006). Extracting subimages of an unknown category from a set of images. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*.
- [Toshev et al., 2006] Toshev, A., Bremond, F., and Thonnat, M. (2006). An apriori-based method for frequent composite event discovery in videos. In *Proceedings of the International Conference on Computer Vision Systems (ICVS)*, page 10.
- [Tran and Davis, 2008] Tran, S. D. and Davis, L. S. (2008). Event modeling and recognition using markov logic networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*.
- [Veloso et al., 2006] Veloso, M., Rybski, P., and von Hundelshausen, F. (2006). Focus: A generalized method for object discovery for robots that observe and interact with humans. In *Human-Robot Interaction*.
- [Vlachos et al., 2008] Vlachos, M., Gunopoulos, D., and Kollios, G. (2008). Discovering similar multidimensional trajectories. In *Proceedings of the IEEE International Conference on Data Engineering (ICDE)*.
- [Vogler and Metaxas, 2001] Vogler, C. and Metaxas, D. (2001). A framework for recognizing the simultaneous aspects of american sign language. *Computer Vision and Image Understanding (CVIU)*, 81:358–384.
- [Wang et al., 2005a] Wang, C., Zhu, Y., Wu, T., Wang, W., and Shi, B. (2005a). Constraint-based graph mining in large database. In *Asia-Pacific Web Conference*.
- [Wang et al., 2005b] Wang, P., Liu, Z.-Q., and Yang, S.-Q. (2005b). A probabilistic template-based approach to discovering repetitive patterns in broadcast videos. In *Proceedings of the ACM International Conference on Multimedia*.

- [Wang et al., 2001] Wang, T., Shum, H., Xu, Y., and Zheng, N. (2001). Unsupervised analysis of human gestures. In *Proceedings of the IEEE Pacific Rim Conference on Multimedia*.
- [Wang et al., 2009] Wang, X., Ma, X., and Grimson, E. (2009). Unsupervised activity perception in crowded and complicated scenes using hierarchical bayesian models. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 31(3):539–555.
- [Wang et al., 2006] Wang, X., Tieu, K., and Grimson, E. (2006). Learning semantic scene models by trajectory analysis. In *Proceedings of the European Conference on Computer Vision (ECCV)*.
- [Washio and Motoda, 2003] Washio, T. and Motoda, H. (2003). State of the art of graph-based data mining. *SIGKDD Explorations*, 5(1):59–68.
- [Watanabe, 1960] Watanabe, S. (1960). Information theoretical analysis of multivariate correlation. *IBM Journal of Research and Development*, 4:66–82.
- [Witten, 2000] Witten, I. H. (2000). Adaptive text mining: Inferring structure from sequences. *Journal of Discrete Algorithms*, 2(2):137–159.
- [Xiang and Gong, 2006] Xiang, T. and Gong, S. (2006). Beyond tracking: Modelling activity and understanding behaviour. *International Journal of Computer Vision (IJCV)*, 67(1):21–51.
- [Xie et al., 2003] Xie, L., Chang, S.-F., Divakaran, A., and Sun, H. (2003). Unsupervised mining of statistical temporal structures in video. In Rosenfeld, A., Doremann, D., and Dementhon, D., editors, *Video Mining*, chapter 10. Kluwer Academic Publishers.
- [Xie et al., 2008] Xie, L., Sundaram, H., and Campbell, M. (2008). Event mining in multimedia streams. *Proceedings IEEE*, 96(4):623–647.
- [Yan and Han, 2002] Yan, X. and Han, J. (2002). gspan: Graph-based substructure pattern mining. In *Proceedings of the International Conference on Data Mining (ICDM)*.
- [Yu and Medioni, 2008] Yu, Q. and Medioni, G. (2008). Integrated detection and tracking for multiple moving objects using data-driven mcmc data association. In *IEEE Workshop on Motion and Video Computing*.

- [Zhu et al., 2007] Zhu, F., Yan, X., Han, J., and Yu, P. S. (2007). gprune: A constraint pushing framework for graph pattern mining. In *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*.