博士論文

# Modeling and Recognizing Human Activities from Video

## 映像にもとづく人物行動のモデリングと認識

東京大学大学院
情報理工学系研究科
電子情報学専攻

48-57415 Kris M. Kitani

指導教員 佐藤 洋一 准教授

平成20年9月

# Modeling and Recognizing Human Activities from Video

by

Kris M. Kitani

Bachelor of Science
*University of Southern California*
1999

Master of Science
*The University of Tokyo*
2005

A dissertation submitted in partial satisfaction
of the requirements for the degree of

Doctor of Philosophy

in

Information Science and Technology

in the

GRADUATE SCHOOL

of

THE UNIVERSITY OF TOKYO

September 2008

Modeling and Recognizing Human Activities from Video

by

Kris M. Kitani

# Abstract

Modeling and Recognizing Human Activities from Video

by

Kris M. Kitani
Doctor of Philosophy in Information Science and Technology

The University of Tokyo, Institute of Industrial Science

Associate Professor Yoichi Sato, Advisor

This thesis presents a complete computational framework for discovering human actions and modeling human activities from video, to enable intelligent computer systems to effectively recognize human activities. This work is motivated by a desire to create an intelligent computer system that can understand high-level activities of people, thus allowing computer systems to efficiently interact with people. A bottom-up computational framework for learning and modeling human activities is presented in three parts. First, a method for learning primitive actions units is presented. It is shown that by utilizing local motion features and visual context (the appearance of the actor, interactive objects and related background features), the proposed method can effectively discover action categories from a video database without supervision. Second, an algorithm for recovering the basic structure of human activities from a noisy video sequence of actions is presented. The basic structure of an activity is represented by a stochastic context-free grammar, which is obtained by finding the best set of relevant action units in a way that minimizes the description length of a video database of human activities. Experiments with synthetic data examine the validity of the algorithm, while experiments with real data reveals the robustness of the algorithm to action sequences corrupted with action noise. Third, a computational methodology for recognizing human activities from a video sequence of actions is presented. The method uses a Bayesian network, encoded by a stochastic context-free grammar, to parse an input video sequence and compute the posterior probability over all activities. It is shown how the use of deleted interpolation with the posterior probability of activities can be used to recognize overlapping activities. While the theoretical justification and experimental validation of each algorithm is given independently, this work taken as a whole lays the necessary groundwork for designing intelligent systems to automatically learn, model and recognize human activities from a video sequence of actions.

To Momoko and Aika

## Acknowledgements

> *Trust in the Lord with all your heart and lean not on your own understanding.*
> *In all your ways acknowledge him and he will make your path straight.*
>
> *–Proverbs*

I would like to extend my deepest thanks to my advisor, Yoichi Sato, for his discussions, guidance and encouragement over the past five years. His comments and suggestions were invaluable to my work. I would also like to thank my collaborators, Takahiro Okabe and Akihiro Sugimoto for their input during our periodic meetings and for being such a good source of objective criticism.

I would like to express my thanks to Sadao Kurohashi for his perspective on natural language processing and for still meeting with me after moving to Kyoto University. I also thank Mitsuru Ishizuka for giving me valuable feedback on my work during my doctoral studies.

I am indebted to many of my lab members who were often a source of inspiration to my research and life during my studies. In more or less chronological order: I thank Yasuhiro Ono and Dong Wang, for struggling with me as we prepared for the entrance exams together. I would like to thank Kenji Oka for answering all the questions I had about programming languages and the Japanese language during my masters studies. I thank Tetsuro Kito for answering my questions about C++ and pretty much anything else related to computers. I thank Imari Sato for her constant encouragement and eagerness to learn. Special thanks to Yoshinori Kobayashi for patiently and always very thoroughly helping with the many programming and computer related questions I brought to him. I thank Yusuke Sugano for all the philosophical debates we had in the cafeteria and getting me to think differently about a lot of issues. I also thank Gabriel Pablo Nava, Michihiro Kobayashi, Daisuke Sugimura, Shiro Kumano and Mark Ashdown for spending the time to help me work through different ideas.

I would also like to thank the leaders of the Tokyo evening mens BSF class for being such good role models of placing faith before work, and making it work.

I thank my parents for their prayers and encouragement. I thank my wife Momoko for proofreading my papers, making me lunch everyday and letting me stay so late at the lab to finish my work. I would not have been able to finish this work without her support. Lastly, I give thanks to my Creator, for giving me the abilities and resources to accomplish this work.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

The ultimate goal of this work is to create machines that can mimic the complex learning process of humans when learning about the physical world. However, the challenge of synthesizing the learning process is an overwhelming task. We have been endowed with the amazing capacity to autonomously learn about both the physical and immaterial world around us through constant observation and interaction. Through the continuous stimulation of our sensory organs, say vision, we can grow to recognize thousands of general object categories, identify a specific person face in an instant and perceive three dimensional shape with only a glance. What is more, we have been given the unique mental ability to understand these physical observations as more than physical phenomena but as discrete mental concepts. While the computing power of computers and the number of sophisticated algorithms has grown, current research is far from closing the so called *semantic gap* between physical observations and semantic expressions. Despite the seemly unsurmountable task of teaching a computer to learn as we do, the technical aim in this thesis is to propose a computational framework for modeling, recognizing and discovering human activities in a way that that resembles the human framework for perceiving activities.

## 1.1 Motivation

It is my hope that this work will be one of many endeavors to learn human actions from video for the purpose of creating intelligent systems are more *useful* to people because they are better at *understanding* them. The term “useful” is used to refer to systems that can perform high-level reasoning tasks and can therefore aid people in processing large amounts of data by understanding and responding to high-level commands.

It is expected that the results of this work will cover a wide range of applications. By proposing a system that can automatically learn normal actions ( and consequently detect abnormal behavior) from a surveillance video, a single dedicated watchman would be enabled to monitor a wider area. For home care applications, a single elderly person could comfortably remain at home, while an intelligent surveillance system could be used to monitor daily activities and notify a nurse at a medical center in the case an abnormal activity (e.g. falling down) is detected. For home use, an intelligent video search system could be implemented to search home videos for context by using simple key words like, *soccer game* or *birthday party*. These examples give only a small picture of both the potential applications of this work and the ways in which understanding human activity could enable intelligent computer systems to be more helpful to people.

## 1.2 Overview

In this thesis, the process of understanding human activities is simplified by dividing the task into three sub-tasks: (1) learning primitive actions, (2) discovering the structure of activities and (3) recognizing structured activities. As such, this thesis presents a bottom-up computational framework activity analysis by presenting the entire system in consecutive stages, where the output of each task provides the input of the proceeding task. The overview of this thesis is given as follows:

1. **Learning primitive action using motion and visual context in video**
   (Chapter 3)

   Previous work on primitive action learning has primarily used motion as the principle feature for representing actions. While in recent work the use of appearance information has been implemented, the use of appearance has been limited to pre-defined object appearance or object categories and has not taken into consideration the visual appearance of the entire visual context of the action. Based on the fact that the perception of an action is determined by a mix of motion features as well as key features produced by the visual context (relevant foreground and background features), a multi-modal latent variable model that utilizes both motion features as well as visual features is presented. Through experiments, a set of actions performed by hands on a desktop that include common actions like typing on a keyboard or flipping the pages of a book, are analyzed. Furthermore, while recent work on learning human actions has been limited to experiments on videos with plain backgrounds (i.e. very little background variation) this work deals with videos with cluttered backgrounds. It is shown that the proposed method for primitive action learning is able to robustly leverage relevant visual features for action discovery despite the presence of irrelevant background objects.

2. **Unsupervised learning of the high-level syntactic structure of human activities from video**
   (Chapter 4)

   The aim of this stage is to recover the basic hierarchical structure of a human activity from an action sequence that is prone to noise (i.e. random actions of people). In previous work, the underlying human activity grammar was typically created manually and the input sequences were assume to be without noise. In this work, a framework for identifying noise and recovering the basic activity grammar from a noisy symbol string produced by video without supervision is proposed. To acquire a video based symbol string, simple image processing techniques (color thresholds, contour extraction) are used to detect hands, money and other objects to produce a string of primitive action symbols (e.g. take money, give receipt, etc.). Noise symbols are identified by finding the set of non-noise symbols that optimally compress the training data, where the optimality of compression is measured using a minimum description length (MDL) criterion. Experiments with artificial data and a real video sequence from a local convenience store show the robustness of the system to noise and its effectiveness in learning the basic structure of human activity. Specifically, the proposed method is shown to be able to recover the basic patterns of a purchase transaction by watching continuous footage of interactions at a cash register between an employee and multiple customers.

3. **Probabilistic syntactic modeling and recognition of human activities from video**
   (Chapter 5)

   The final stage uses a grammar model (defined by a user or learned with an algorithm) to recognize human activities (i.e. being idle or leaving an object in the scene) that are captured by a video surveillance camera. Based on the observation that humans understand activities as discrete concepts with hierarchical structure, a probabilistic model based on a stochastic context-free grammar of human activity is implemented to analyze a video sequence. First, the algorithm converts a stochastic context-free grammar into a Bayesian network, which in turn is used to analyze a symbol string produced by a video sequence. A smoothing technique from language processing called deleted interpolation is used to recognize overlapped activities (e.g. a new activity begins while another is in progress). Through experiments, the proposed model is tested on a database of video surveillance images from a lobby of a building. Results show that the proposed method is able to correctly detect high-level activities as well as overlapped activities.

# Chapter 2

# Preliminaries

## 2.1 Video surveillance

Research on activity analysis needs first to be understood in light of the bigger picture, namely, video surveillance. Video surveillance is the real-time extraction and detection of single agent activities and multi-agent interactions from a temporal sequence of video images. It can be positioned as an integration of the fields of computer vision, pattern recognition, artificial intelligence and perceptual psychology.

Drawing from the categorizations used by Collins [CLK00] in a *IEEE transactions on Pattern Analysis and Machine Intelligence* special section on video surveillance, research foci can be divided into three categories: (1) detection and tracking, (2) human motion analysis and (3) activity analysis. Detection and tracking involves the identification of a moving object in a video sequence over time. Human motion analysis goes further by investigating the human body and its physical change in pose. It is interesting to note that since the publication of [CLK00], a new paradigm using a *bag of features* (BoF) approach, that represents human movement as groups of local features, forgoes some of the complexities of a stricter model and has been shown to be effective in roughly describing human actions [NWFF06]. Activity analysis strives to extract a high-level description of humans and other agents in a scene. While the first two parts deal with more or less the low-level descriptions of the agent themselves, activity analysis looks for semantic descriptions of agent behavior over time in a given environment.

## 2.2 Events, activities and actions

In the field of computer vision, words like action, activity, behavior and event have been used to mean different things. Various key terms are defined here to clarify what is meant by these words when used in this context and also to make clear the domain of this work.

### 2.2.1 Events

Events are things that happen [CV02]. Unlike objects which exist, events are said to occur, happen or take place. Events are understood and recognized by their temporal structure and not purely by their spatial structure. In contrast to objects that persist through time (continuants), events take up time and usually have relatively clear temporal boundaries. Some examples of an event would be, a person going to the bathroom or the sun setting.

### 2.2.2 Activity

An activity is a subset of events which involves an actor. For example, a person going to the bathroom is an activity while the sun setting is not an activity because it does not involve an actor. Activities are usually general abstract concepts of a sequence of shorter actions and as such not all activities have the same level of abstraction. That is, some activities can include other activities.

### 2.2.3 Primitive actions

Similar to activities, actions also involve an actor. However, a primitive action refers strictly to physically primitive actions. Primitive actions are usually short in temporal duration and are describe in terms of primitive motions. According to Casati and Varzi [CV96], an action is an objective happening that is performed at the will of the actor whereas, events (and activities) are more general and arise only in the perception of the observer.

A helpful definition of a primitive action defined as a first cause is given as: $\mathrm{a}$ *is a basic action of* $\mathbf{A}$ *if and only if (i)* $\mathrm{a}$ *is an action and (ii) whenever* $\mathbf{A}$ *performs* $\mathrm{a}$*, there is no other action* $\mathrm{a}'$ *performed by* $\mathbf{A}$ *such that* $\mathrm{a}$ *is caused by* $\mathrm{a}'$. [Dan63] (cited in [ZT01]).

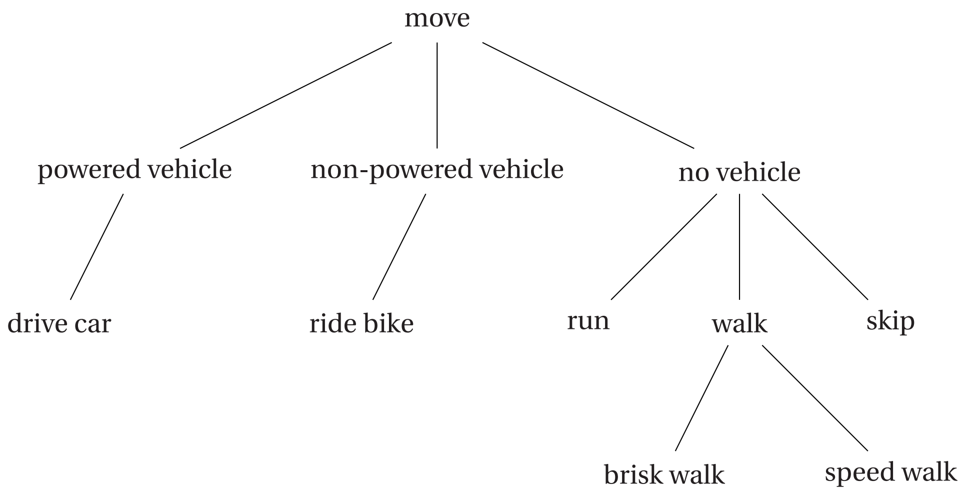


Figure 2.1. Taxonomic hierarchy for *move*.

### 2.2.4 Structure of human activity

**Taxonomic hierarchy**

According to insights from perceptual psychology [ZT01] actions and activities can be taxonomically organized. A taxonomic hierarchy is one that views an object with a hierarchical framework of "type of" relationships. For example, a car or airplane is a type of transportation vehicle. In the case of human activity, *running* or *walking* is a type of moving around. A simple taxonomic hierarchy for "move" is shown in Figure 2.1. While in this work, the taxonomy of activities and actions are not used for learning or recognition, taxonomy can be a powerful tool for discovering the semantic meaning associated to a specific physical motion.

**Partonomic hierarchy**

Activities are also partonomical and exhibit temporal relationships between its parts. A partonomic hierarchy views an object as a sum of essential parts. A table, for example, has a top and four legs. In the case of human activity, eating at a restaurant is a temporal sequence of an entering, ordering, eating and leaving. In the same way, the parts themselves may also be broken down in to even smaller sequential parts. A possible partonomic hierarchy for dining at a restaurant is given in 2.2. This partonomic nature of activities (especially goal oriented activities) is an important concept that allows one to give a syntactic interpretation to an activity.

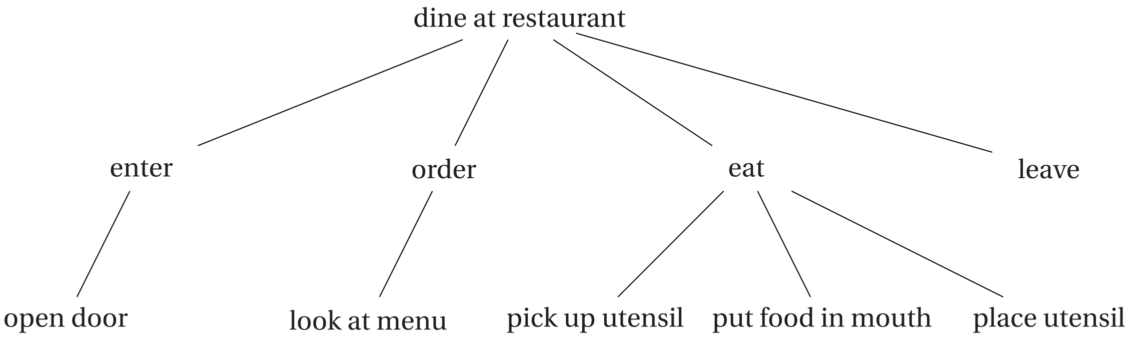


Figure 2.2. Partonomic hierarchy for *dining at a restaurant*.

### Primitive temporal relationships

Activities and actions display a series of basic temporal relationships. 13 basic temporal relationships were defined by Allen in [All84]. The different types of temporal primitives are given in Figure 2.3. Given two actions A and B there are 7 basic temporal relationships, 6 of which can be inverted.

1. A before B means that the action B will occur some time after A has completed.
2. A meets B means that B will occur at the same time A ends.
3. A overlap B means that B will start some time after A has started and before A is completed.
4. A during B means that A starts after B starts and A ends before B ends.
5. A start B means that A and B start at the same time and A ends before B ends.
6. A finish B means that A and B finish as the same time and A starts after B starts.
7. A equal B means that A and B start and end at the same time.

Concepts like A parallel B can be defined, for example, as a superset that includes A equal B, A during B, A start B and A finish B. Phinanez [PB98] has also shown that these temporal relationship can also be expressed using a Past, Now, Future space.

It is noted here, that for most of the work that follows, it is assumed that actions are discrete and occur in serial order (relationship 1 and 2). This is another assumption that is critical to a syntactic approach to activity analysis. The creation of a complete activity analysis system will most likely require that all of these relationships are modeled by the system.

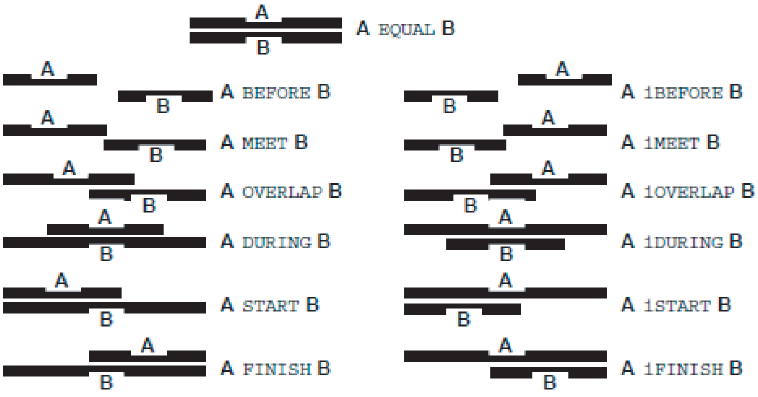


Figure 2.3. Different types of temporal primitives. Taken from [PB98, All84].

# Chapter 3

# Learning action primitives

In order to formulate a framework for recognizing human activities, one needs a means of first learning those activities. Likewise, to be able to learn human activities it is then necessary to learn the basic components (primitive actions) of an activity. In this chapter, an unsupervised method for learning primitive actions from a corpus of actions is proposed. It is shown that action categories can be discovered effectively when both motion and visual appearance are used to represent primitive action.

## 3.1 Introduction

Since the analysis of activities requires that primitive actions are first extracted from video, this chapter presents a novel framework for the unsupervised learning of human actions from a video corpus by leveraging relevant visual context. Considering the fact that actions can be understood at various temporal resolutions, the focus is placed discovering what is called *primitive actions*. Primitive actions are humans actions that can be defined over a very short period of time (a few seconds). For example, grabbing a cup, typing on a keyboard or flipping the page of a book can be recognized within a few seconds of observing the action. Learning primitive actions are important because they are the basic building blocks of many high-level activities [ME02, HJB$^+$05, KSS07].

Supervised learning techniques using such models as HMMs [SP95, IB00], Bayesian classifiers [SHM$^+$04] and temporal dynamics [Sis00] have been successful in describing primitive actions but need labeled data or a considerable amount of prior knowledge. While most of

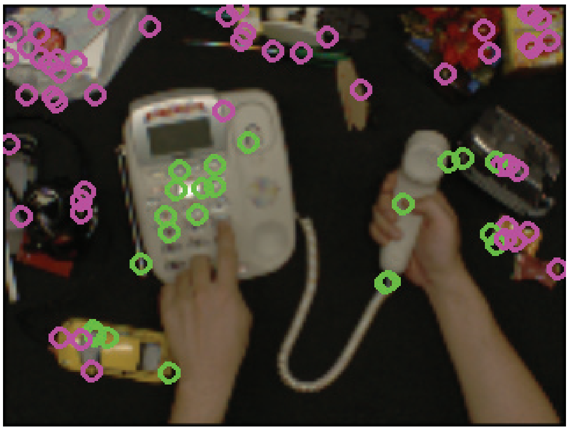


Figure 3.1. Leveraging visual features for action recognition: Relevant visual features (green) induced by using the telephone and irrelevant features (purple) produced by unrelated background objects.

these past works have used supervised approaches to learn primitive actions, there have also been several recent works focused on the unsupervised learning of actions.

An approach of growing interest for unsupervised action discovery is the use of *generative latent variable models* (mixture models [NMTM99], PLSA [Hof99], LDA [BNJ03], HDP [TJBB06]) based on the bag-of-words paradigm. Most of these methods were originally developed to learn topics from text in documents. Given the document analogy, a typical language model factorizes the observed documents (corpus items) and words (features) to discover a distribution over a set of hidden topics.

Niebles [NWFF06] proposed the application of a generative model to video to learn action categories (topics) from a bag-of-features. They used exactly the same framework as [Hof99] by simply replacing document indices with video indices, and words with spatial-temporal (ST) volumes. Their approach showed that similar to text, the local features of an action can be treated as though they were *exchangeable* (an action can be treated as a bag of uncorrelated features) to learn action categories. However, the conceptual problem with a straightforward use of a language model for action discovery is that the models are uni-modal (e.g. use only words).

It is known from experience that actions are composed of motions and visual appearance. For example, the hands of a person playing a piano and typing on a keyboard might have very similar motions but can easily be differentiated using the visual context of a piano or a keyboard. In fact, findings from neural science make it clear that actions are mentally perceived as a mix of motions and visual features of present objects [FA98]. In the light of this fact, many previous approaches to action discovery are limited by the fact that they only consider one mode, namely, motion.

For example, the ST volumes used to describe dynamic human actions in [NWFF06] do not explicitly account for important visual features of objects in the scene. Similarly, the approach proposed by Wang [WMG07] to automatically classify actions of cars and pedestrians at an intersection uses the change in pixel intensity between two frames as the input feature, making their system robust to the varying color and shape of automobiles and pedestrians. However, to apply their current system to more complex interactions between actors and objects will most likely necessitate the incorporation of more complex motion features and appearance features to improve performance.

While the joint use of appearance and motion to describe action is not entirely new, this work differs from previous work in that the proposed method does not use *a priori* information about the category, shape, size or color of actors or objects in the scene. For example, work using the appearance of related objects to recognize actions has depended on *a priori* knowledge of the appearance of related objects [GD07] or pre-defined object categories [MEH99]. Work leveraging the appearance of the actor, such as Fanti [FZMP05] and Niebles [NFF07], have proposed modified generative models that model the human body and account for both the appearance and motion of body parts. However, explicitly modeling the human body comes at the cost of losing the ability to apply the model to other types of actors. The more important distinction with this work however, is that the use of visual information was limited to pre-defined body parts and therefore could not explicitly take into account other relevant visual information possibly generated by co-occurring objects or scenic context.

Presented in this chapter is a robust framework for primitive action discovery by leveraging both motion and relevant visual context without the use of *a priori* information (e.g. an explicit shape model or pre-defined object categories). Experiments show that the proposed method properly leverages relevant visual appearance and is robust against irrelevant visual features (Figure 3.1) when learning action categories.

## 3.2 Proposed method for learning action primitives

The goal is to learn the primitive action categories that occur within a video corpus. First temporal features and spatial features are extracted from each video segment, under the assumption that actions are defined by both temporal motion and visual context (Section 3.2.1). Then a description of a dimension reduction scheme is given to create a codebook for each feature type (Section 3.2.2). Finally, an explanation of a bi-modal generative model is presented, that uses the histograms produced from a video corpus to learn the latent action categories (Section 3.2.3).

### 3.2.1 Extracting spatial and temporal features

The extraction process is explained here for spatial (visual) features and temporal (motion) features. For each frame in the training corpus, a sparse set of spatial features is extracted by finding SIFT key points [Low99]. These key points are then represented with a normalized 128 dimensional SIFT descriptor. Other combinations of key point detectors [MCUP02] and descriptors can be used as well.

Using the same temporal gradient descriptor as [BI05], a sparse set of temporal features are extracted from the video frames by extracting a $7 \times 7 \times 4$ (a $7 \times 7$ spatial window over $4$ frames) spatio-temporal volume for pixels that detected as a good feature to track [ST94] and are tracked by optical flow [Bou02] for two consecutive frames. The frontal face of the volume is centered at the location of the tracked feature in the second tracked frame and each element of the volume contains the temporal gradient magnitude. The descriptor is a normalized 196 dimensional vector containing the elements of the volume. More complex temporal keypoints can also be used, such as spatio-temporal cuboids [DRCB05] or space-time interest points [Lap05].

### 3.2.2 Two-stage feature clustering

Compared to documents or images, the number of features that can be extracted from a video sequence can be very large (e.g. about $20$ million temporal features for 7 minutes of video). Therefore, an efficient two stage clustering process that combines an online and offline algorithm is implemented to process the descriptors generated by the video corpus. The first stage simultaneously identifies key clusters (learns a codebook) and generates histograms in one pass over the database. In the second state a more holistic clustering (dimension reduction) algorithm is implemented to further reduce the dimensions of the histograms.

#### Nearest representative point clustering

An online clustering algorithm termed *nearest representative point clustering* (NRPC) is used to cluster descriptors and generate a histogram for all the videos in one pass. An online scheme is selected over offline clustering algorithms (e.g. K-means clustering) that are commonly used for bag-of-features based approaches because of the enormous computational

**Algorithm 1 –** Nearest Representative Point Clustering

```
for every video segment $d$ in the corpus $\mathbf{d}$ do
  Initialize segment histogram $\mathbf{v}_d = \mathbf{0}$
  for every descriptor $\mathbf{x}_{di}$ extracted from segment $d$ do
    Find nearest representative point $\mathbf{c}_j$ to $\mathbf{x}_{di}$
    if $L_2(\mathbf{x}_{di}, \mathbf{c}_j) > \theta$ then
      Create new representative point $\mathbf{c}_k \leftarrow \mathbf{x}_{di}$
      Initialize count of centroid $v_{dk} = 1$
    else
      Increment count $v_{dj}$ of nearest representative point $\mathbf{c}_j$
    end if
  end for
end for
```

cost involved with iteratively processing a large data set extracted from video. It is noted that in contrast to basic leader-follower clustering [DHS00], this algorithm represents a cluster with only one representative data point and the centroid is never updated.

An outline of the NRPC algorithm is given in Algorithm 1. The NRPC algorithm takes a single descriptor $\mathbf{x}_{di}$ from the set of all descriptors extracted from segment $d$ and decides whether to update the count of a pre-existing cluster or create a new cluster, depending on a threshold $\theta$. After processing all descriptors, a set of $n$ clusters $\mathbf{c}_1, \ldots, \mathbf{c}_n$ and a corresponding $n$ dimensional histogram vector of counts $\mathbf{v}_d = (v_{d1}, \cdots, v_{dn})^T$ for the video segment $d$ are obtained.

Each video segment $d \in \mathbf{d}$ is processed in the same way to produce the set of $m = |\mathbf{d}|$ histogram vectors of the histogram matrix $\mathbf{V} = (\mathbf{v}_1, \cdots, \mathbf{v}_m)$. Notice that the number of clusters $n$ can potentially increase each time a new video is processed (i.e. new clusters are created). The histograms of previously processed videos are simply padded with zeros to keep the same dimensionality $n$. This clustering process is done once for each feature modality (i.e. spatial features and temporal features). This type of online clustering is effective for video because many nearly identical features are produced by a single action. As a speed-up technique, an incrementally trained nearest neighbor classifier is used to find the nearest representative point.

### Non-negative matrix factorization

The first stage of clustering was done online and therefore did not take into account any of the global statistics of the data. In the second stage, the dimensionality of the training data is further reduced using a more holistic approach. Non-negative matrix factorization

(NMF) [LS99] is implemented to project the set of histograms $\mathbf{V}$ onto a lower dimensional non-negative subspace $\mathbf{H}$. Since it is assume that actions are additive (actions only produce features and never delete features), NMF is used over other projection techniques like PCA or ICA because NMF decomposes the data $\mathbf{V}$ as an additive (non-negative) combination of a lower dimensional basis subspace. It is interesting to note that both PLSA and NMF have been shown to be instances of multinomial PCA [Bun02]. In fact in the proposed framework, the projected dimensions of NMF $r$ and the number of hidden categories $k$ are set to be equivalent and as a consequence, this second stage can also be interpreted to be the pre-discovery of hidden actions categories for each independent modality of input.

Formally, NMF decomposes the $n \times m$ histogram matrix $\mathbf{V}$ (each column is a histogram of descriptors for a video) into a $n \times r$ basis matrix $\mathbf{W}$ and the $r \times m$ encoding matrix $\mathbf{H}$,

$$\mathbf{V} \approx \mathbf{W}\mathbf{H}. \tag{3.1}$$

The projected gradient method [Lin07] is used to factorize $\mathbf{V}$ and the resulting columns of the encoding matrix $\mathbf{H}$ contain the reduced (encoded) version of $\mathbf{V}$. To allow the iterative process to converge near an optimal solution, $\mathbf{H}$ is initialized by a randomized matrix weighted by the results of K-means clustering with the top $l$ principle components of the video segment histograms.

NMF is executed twice independently, once for spatial features and once for temporal features, by projecting the spatial and temporal descriptor histogram matrices $\mathbf{V}_s$ and $\mathbf{V}_t$ onto the reduced dimensional spaces $\mathbf{H}_s$ and $\mathbf{H}_t$, respectively. NMF maps the histogram from each video segment to a reduced dimensional histogram. As a result, the values of $\mathbf{H}_s$ and $\mathbf{H}_t$ yield an approximation to the term-by-document frequency matrix. The term-by-document frequency matrix is commonly used as the input for learning language models, where each element of the matrix $n(w, d)$ represents the number of times a feature $w$ is observed in the corpus item $d$.

### 3.2.3 Merging motion and visual context via the action model

**Parameter learning**

Under the framework of Bayesian networks, the joint probability of a set of random variables can be simplified by defining the conditional independence between variables. In the proposed action model (Fig. 3.2), it is assumed that temporal features $t$ are conditionally independent of spatial features $s$ given a latent action category $z$. That is, the proposed model is a bi-modal expansion of the standard mixture of unigrams model [NMTM99] that defines the

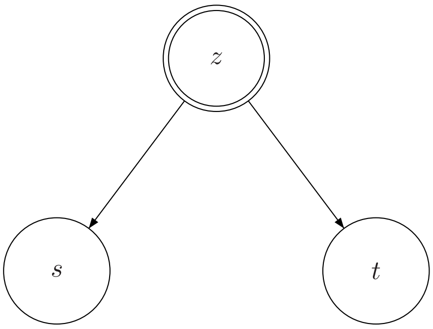


Figure 3.2. Bi-modal latent variable model defined by the latent topic $z$, a spatial feature $s$ and a temporal feature $t$.

probability of a video segment $d \in \mathbf{d}$ as below.

$$p(d) = \sum_z p(d|z)p(z) \tag{3.2}$$

$$p(d|z) \propto \prod_{s \in d} p(s|z) \prod_{t \in d} p(t|z) \tag{3.3}$$

$$= \prod_s p(s|z)^{n(s,d)} \prod_t p(t|z)^{n(t,d)} \tag{3.4}$$

Based on the conditional independence of the spatial feature $s$ and the temporal feature $t$ given the latent topic $z$, the conditional probability of a video segment can be computed as the product of the conditional probabilities of all spatial and temporal features in the video segment. The term $n(s,d)$ represents the number of times a spatial feature $s$ has occurred in a video segment $d$. The term $n(t,d)$ is interpreted similarly for temporal features.

To learn the parameters of the bi-modal mixture model, the desired goal is to find values for the parameters $p(s|z)$, $p(t|z)$ and $p(z)$, such that the log-likelihood of the entire video corpus $\mathbf{d}$ is maximized.

$$\log p(\mathbf{d}) = \sum_d \log \sum_z \Big[ \prod_s p(s|z)^{n(s,d)} \prod_t p(t|z)^{n(t,d)} \Big] p(z) \tag{3.5}$$

The expectation maximization (EM) algorithm is implemented to find a locally optimal set of parameters with respect to the likelihood. In general it can be shown that the data likelihood $p(\mathbf{x};\Theta)$ over a training set $\mathbf{x} = \{x^{(i)}\}$ can be decomposed in to the expectation of the complete data likelihood $E[\mathcal{L}^c]$ and the Kullback-Leibler distance $KL(q||p)$ between an arbitrary probability distribution $q(z)$ over the latent variable $z$ and the posterior distribution $p(z|\mathbf{x})$.

The derivation is as follows

$$
\begin{align}
\log p(\mathbf{x}) &= \sum_z q(z) \log p(\mathbf{x}) \tag{3.6} \\
&= \sum_z q(z) \log \frac{p(\mathbf{x}, z)}{p(z|\mathbf{x})} \tag{3.7} \\
&= \sum_z q(z) \log \frac{p(\mathbf{x}, z) q(z)}{p(z|\mathbf{x}) q(z)} \tag{3.8} \\
&= \sum_z q(z) \left\{ \log \frac{p(\mathbf{x}, z)}{q(z)} - \log \frac{p(z|\mathbf{x})}{q(z)} \right\} \tag{3.9} \\
&= \sum_z q(z) \log \frac{p(\mathbf{x}, z)}{q(z)} - \sum_z q(z) \log \frac{p(z|\mathbf{x})}{q(z)} \tag{3.10} \\
&= \sum_z q(z) \log p(\mathbf{x}, z) - \sum_z q(z) \log \frac{p(z|\mathbf{x})}{q(z)} - \sum_z q(z) \log \frac{1}{q(z)} \tag{3.11} \\
&= E[\mathcal{L}^c] + KL(q||p) + \mathcal{H}(q) \tag{3.12}
\end{align}
$$

This derivation shows that the lower bound of the data likelihood can be maximized by setting $q(z) = p(z|d)$ (expectation step) and maximizing the expectation of the complete log-likelihood $E[\mathcal{L}^c]$ (maximization step), where the entropy term can be ignored for maximization since it is a constant. To summarize, the function to be maximized is given as:

$$
E[\mathcal{L}^c] = \sum_d \sum_z p(z|d) \log \Big[ \prod_s p(s|z)^{n(s,d)} \prod_t p(t|z)^{n(t,d)} \Big] p(z). \tag{3.13}
$$

In the expectation step, the posterior of the latent variable is computed using Bayes' rule.

$$
p(z|d) = \frac{p(d|z) p(z)}{\sum_{z'} p(d|z') p(z')} \tag{3.14}
$$

While the values of the parameters are typically initialized randomly, a normalized encoding matrix $H$ is used as the initial values of $p(d|z)$. This is possible because the reduced dimensions of NMF $r$ is equivalent to the number of latent states $k$. Otherwise, the initial posteriors must be initialized randomly.

$$
L = E[\mathcal{L}^c] + \sigma_z \left[ 1 - \sum_s p(s|z) \right] + \theta_t \left[ 1 - \sum_t p(t|z) \right] + \zeta \left[ 1 - \sum_s p(z) \right] \tag{3.15}
$$

Solving for the maxima (taking the partial derivatives) of the Lagrangian function (3.15), which is composed of the complete data log-likelihood and standard conditions on the parameters (i.e. probabilities add to one), results in the following stationary equations that lead to the maximization equations.

$$\frac{\delta L}{\delta p(s|z)} = \frac{1}{p(s|z)} \sum_d n(s,d)p(z|d) - \sigma_z = 0 \tag{3.16}$$

$$= \sum_d n(s,d)p(z|d) - \sigma_z p(s|z) = 0 \tag{3.17}$$

$$\frac{\delta L}{\delta p(t|z)} = \frac{1}{p(t|z)} \sum_d n(t,d)p(z|d) - \theta_z = 0 \tag{3.18}$$

$$= \sum_d n(t,d)p(z|d) - \theta_z p(t|z) = 0 \tag{3.19}$$

$$\frac{\delta L}{\delta p(z)} = \frac{1}{p(z)} \sum_d p(z|d) - \zeta_z = 0 \tag{3.20}$$

$$= \sum_d p(z|d) - \zeta_z p(z) = 0 \tag{3.21}$$

$$\frac{\delta L}{\delta \sigma_z} = 1 - \sum_s p(s|z) = 0 \tag{3.22}$$

$$\frac{\delta L}{\delta \theta_z} = 1 - \sum_t p(t|z) = 0 \tag{3.23}$$

$$\frac{\delta L}{\delta \zeta} = 1 - \sum_z p(z) = 0 \tag{3.24}$$

Using system of stationary equations to solve for the optimal parameters, the re-estimation equations that maximize the likelihood of the data given the current posterior are given as below.

$$\hat{p}(s|z) = \frac{\sum_d n(s,d)p(z|d)}{\sum_s \sum_d n(s,d)p(z|d)} \tag{3.25}$$

$$\hat{p}(t|z) = \frac{\sum_d n(t,d)p(z|d)}{\sum_t \sum_d n(t,d)p(z|d)} \tag{3.26}$$

$$\hat{p}(z) = \frac{\sum_d p(z|d)}{|\mathbf{d}|} \tag{3.27}$$

This process between the expectation step and the maximization step is repeated until the log-likelihood function converges at a maximum.

### Inference and recognition

Once the parameters of the model have been learned, the naive Bayes model can also be used to recognize primitive actions. Specifically, given a test video segment $d$, a set of temporal and spatial features are extracted and binned to create a histogram of temporal and spatial features,$\mathbf{v}_d^t$ and $\mathbf{v}_d^s$ respectively. Then the histograms are projected onto the respective encoding spaces to obtain the encoding vectors $\mathbf{h}_d^t$ and $\mathbf{h}_d^s$ using the zeroed least-square solution

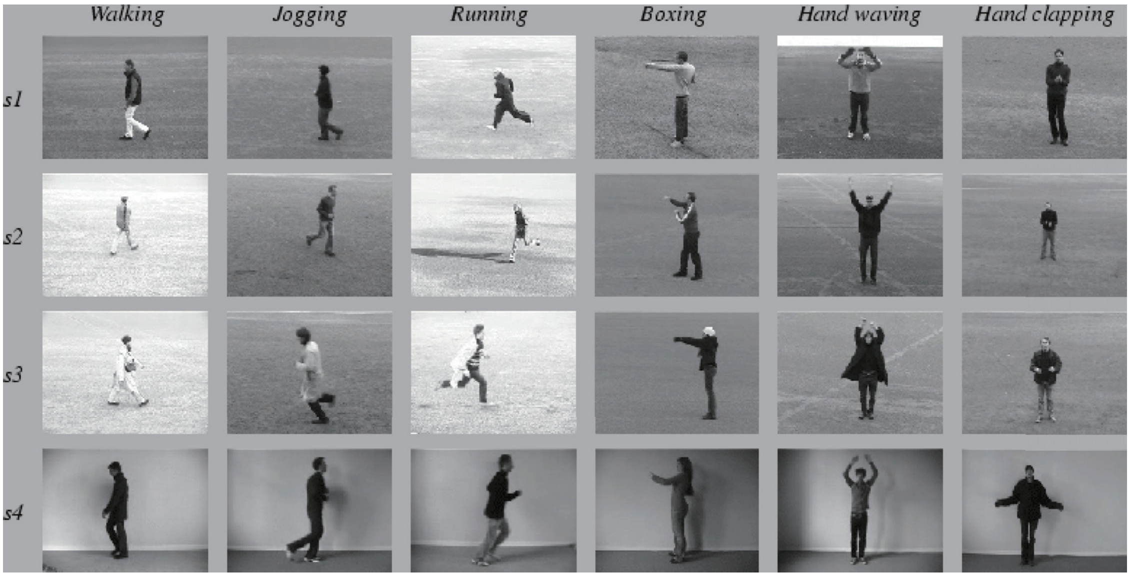
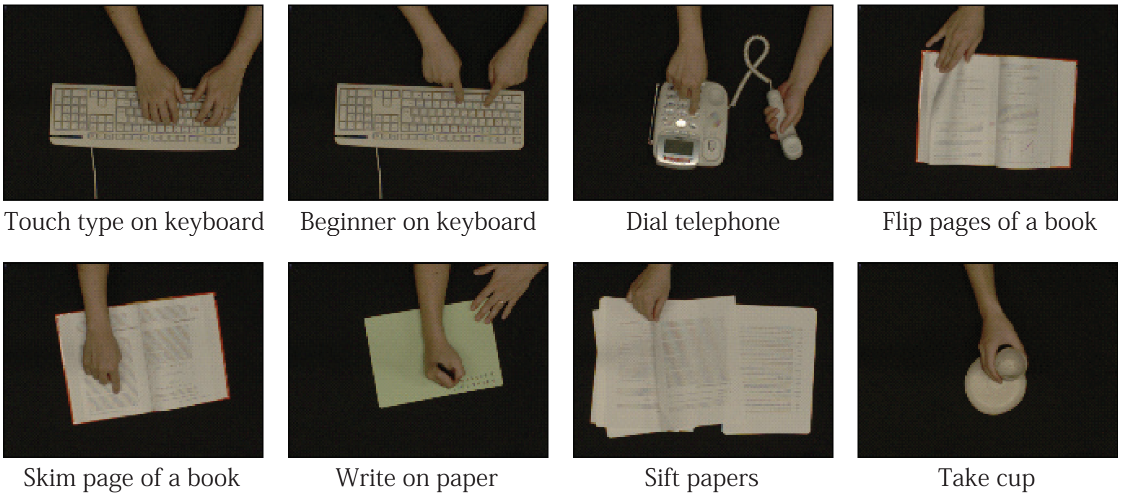


Figure 3.3. Examples from the KTH action dataset taken from [SLC04].

[OP07]. Normalizing the vectors $\mathbf{h}_d^t$ and $\mathbf{h}_d^s$ yields the distribution over the features $\mathbf{t}_d$ and $\mathbf{s}_d$ for the test video segment. These distributions are then passed on as likelihood evidence for the naive Bayes model to infer the distribution over the hidden actions $p(z|d)$ using belief propagation [Pea88].

## 3.3 Action datasets

Publicly available datasets used for human action recognition, like the KTH dataset [SLC04] (Figure 3.3), have very little background variation (i.e. a wall or a field) and usually only involves an actor with no interactive objects [DRCB05, BGS$^+$05, WKC07]. In contrast, it is completely reasonable to assume that many other objects will be visible in real world videos of human actions, especially important visual features that help define the actions being performed. This section presents four new primitive action video datasets, created to include various backgrounds and interactive objects. These datasets are needed to show how the proposed method is able to leverage relevant visual context along with motion information to effectively discover action categories.



Touch type on keyboard

Beginner on keyboard

Dial telephone

Flip pages of a book

Skim page of a book

Write on paper

Sift papers

Take cup

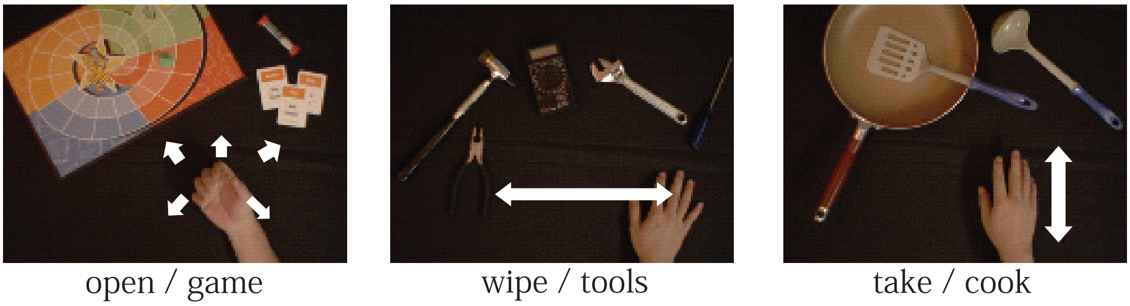
Figure 3.4. Key frames for the corpus $C_{OBJ}$ with 8 desktop actions involving objects.

### 3.3.1 Actions with objects corpus

The first motion and object corpus $C_{OBJ}$ consists of eight different primitive actions that involve a related physical object. A list is given below:

1. Touch typing on keyboard
2. Beginner on a keyboard
3. Dialing a phone
4. Flipping the pages of a book
5. Skimming the page of a book with finger
6. Writing with a pen on a piece of paper
7. Sifting through a stack of papers
8. Take a cup

Each action video was spliced into three second intervals. Using the first five segments per action yielded a total of 40 video segments. Each video segment was 90 frames long and all videos were created at a resolution of $160 \times 120$. Notice that some of the actions involve the same object. Key frames from the corpus are given in Figure 3.4.



open / game

wipe / tools

take / cook

Figure 3.5. Examples from corpus $C_{BG}$ with 9 actions including 3 different motions and 3 different background objects. Direction of motion is shown in white.

### 3.3.2 Actions with backgrounds corpus

The second motion and background corpus $C_{BG}$ consists of three different motions and three different visual scenes (backgrounds). The three motions are:

1. Take (vertical movement)
2. Wipe (horizontal movement)
3. Open (hand opens and closes in place)

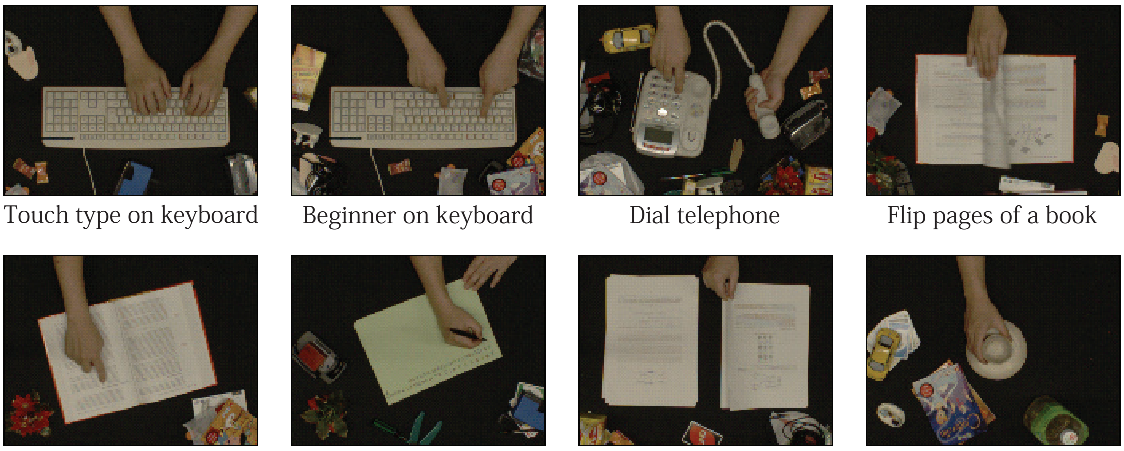
The three background scenes are:

1. Board game scene
2. Tools scene
3. Cooking scene

The combination of movement and scenes yields nine different actions. Five videos segments for each action resulted in a corpus of 45 videos. Each video segment was 90 frames long and all videos were created at a resolution of $160 \times 120$. Key frames of several combinations of motions and visual contexts are given in Figure 3.5.

### 3.3.3 Actions with objects and backgrounds corpus

The third motion with objects and background (messy desktop) corpus $C_{BGOB}$ contains the same actions as the first corpus $C_{OBJ}$ but also includes random backgrounds (Figure 3.6)



Touch type on keyboard

Beginner on keyboard

Dial telephone

Flip pages of a book

Skim page of book

Write on paper

Sift stack of paper

Take cup

Figure 3.6. Examples from corpus $C_{BGOB}$ with 8 different actions with objects and varied random background objects for each video segment.

which act as visual noise. The visual characteristic of each video segment was varied by including different random static objects in the background. This corpus used the same number of video segments and the same resolution as the first corpus $C_{OBJ}$.

### 3.3.4 Actions with objects and extraneous motion

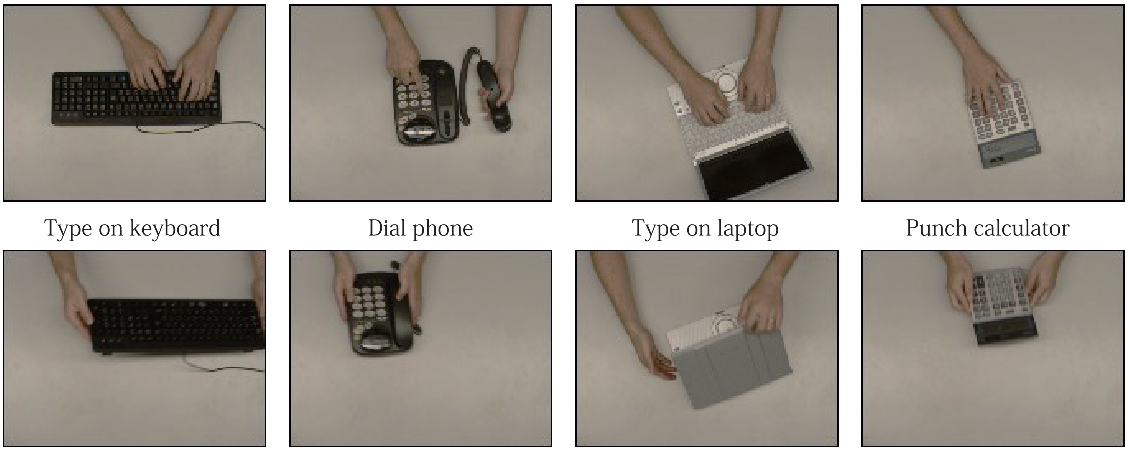
In the three datasets introduced earlier, each segment consisted of a single action taken from one repetitive sequence of the action and therefore did not include any extraneous movements that are not produced by one of the action categories. Since it is conceivable that a typical video corpus will include some extraneous motion, the dataset used in the next set of experiments utilizes a more challenging corpus that includes the transitions between actions (see Figure 3.7). This fourth extraneous motion $C_{EXMO}$ consists of four different primitive actions involving a related physical object. Furthermore, the same set of actions are repeated twice at different time instances to introduce more variations within each action category. Sample images from the dataset are given in Figure 3.7 and the action categories are:

1. Type on keyboard

2. Dial phone

3. Type on laptop

4. Punch numbers on calculator



Type on keyboard

Dial phone

Type on laptop

Punch calculator

Figure 3.7. Dataset $C_{EXMO}$ of four desktop actions (top row) including extraneous movements (bottom row)

A three minute video sequence of the four actions was indiscriminately segmented into 90 frame intervals and the video was created at a resolution of $160 \times 120$. As a results of the segmentation, this corpus had a total of 68 segments. Since the frames are cut indiscriminately some segments do not contain motions produced by an action category. These segments act as noise during the learning process.

## 3.4 Experiments with hand action datasets

First a baseline experiment is performed using only temporal features with NMF. Since it has been shown that NMF is essentially equivalent to PLSA [GG05], this baseline experiment is very similar to the approached use in [NWFF06]. Then four experiments are performed using the proposed framework and it is shown how leveraging visual context improves learning performance. For each experiment, the values of $r$ and $k$ are provided as prior knowledge but can also be learned using a model selection criteria as in [VG02]. To initialize the encoding matrix $\mathbf{H}$ for NMF near an optimal solution, the data is clustered into $r$ clusters to compute the initial values. That is, the set of histograms $\mathbf{V}$ were clustered with $k$-means clustering using the top eight principle components (via PCA) of the histograms. For each column in the initialization matrix $\mathbf{H}_{init}$, the element representing the nearest cluster was set to one and all other entries were set to zero (binarization). Then a random non-negative proper fractions was added to every element (random noise). This initialization scheme sets the initialization matrix to a value near a optimal solution while retaining some randomness. Likewise, to initialized the EM process of the bi-modal mixture model near an optimal solution, the initial values of $p(z|d)$ are set using the results of NMF, namely, the normalized values of the encoding matrix $\mathbf{H}$ of one of the modes (usually spatial features). For each experiment, the initial codebook is generated by heuristically setting the NRPC distance thresholds for both temporal features $\theta_t$ and spatial features $\theta_s$ to $0.015$.

### 3.4.1 Baseline experiment: Results using only motion features

The result of the first baseline experiment using only temporal gradient features is given in the form of a bar graph of the posterior probabilities $p(z|d)$ (the probability of a latent action category $z$ given the video segment $d$) in Figure 3.8. The average of the posterior probabilities for each action category is also given in Table 3.1. A single row of the table is created by taking the average of the posterior probability distributions of a group of video segments (indices of the segments indicated in the leftmost column) from the same action. The columns (latent states $z$) are labeled based on the element that has the greatest average probability. The probabilities marked in bold indicate the average probability matched to the correct action.

#### Performance metric

For the experiments in this section, the average of the bold values are used as a measure of performance, which is termed here as the probability of correct categorization (PCC). The higher the PCC value, the greater the certainty that a video segment is associated to the correct category. This measure is used in tandem with the standard AUC (area under the (ROC) curve) measure because the AUC values do not express the degree to which a dataset is properly categorized. That is, the AUC of a set of results will yield a score of $1.000$ if there exists a single threshold at which all of the items can be correctly classified. That means that the AUC only describes the ability of a classifier to categorize data and does not describe the margin (or degree) by which the data is classified when there exists an ideal threshold value. For example, a binary classifier $A$ that gives an average score of $49\%$ for class 0 and $51\%$ for class 1 and classifier $B$ that gives an average score of $1\%$ to class 0 and $99\%$ to class 1, both receive an AUC score of $1.000$ if the data can be split at $50\%$. However, one would prefer to use the second classifier $B$ because it has a smaller margin of error.

### Results

The PCC of the motion and object corpus $C_{OBJ}$ using only motion features was $85.12\%$ and the AUC was $1.000$. Since the simple temporal gradient descriptors only capture motion and they are not invariant to scale or rotation, there is some uncertainty about the classification, despite the high AUC score. The point to be made here is that the descriptive power of the temporal features is not strong enough to categorize the primitive actions with high confidence. The next experiment will show an increase in classification confidence when visual context is also used as an input.

Table 3.1. Average posterior probabilities for each action category using only temporal features.

| | Discovered Actions ($z$) | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | touch keyboard | write paper | begin keyboard | flip page | sift paper | take cup | skim page | dial phone |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 1-5 | **0.8724** | 0.0605 | 0.0065 | 0.0135 | 0.0007 | 0.0443 | 0.0012 | 0.0009 |
| 6-10 | 0.0686 | 0.0192 | **0.8508** | 0.0009 | 0.0077 | 0.0173 | 0.0340 | 0.0016 |
| 11-15 | 0.0290 | 0.0383 | 0.0705 | 0.0371 | 0.0073 | 0.0058 | 0.0103 | **0.8018** |
| 16-20 | 0.0159 | 0.0092 | 0.0055 | **0.9159** | 0.0374 | 0.0116 | 0.0044 | 0 |
| 21-25 | 0.0315 | 0.0176 | 0.0180 | 0.0382 | 0.0021 | 0.0298 | **0.7619** | 0.1007 |
| 26-30 | 0.0258 | **0.8148** | 0.0210 | 0.0349 | 0.0019 | 0.0544 | 0.0306 | 0.0165 |
| 31-35 | 0.0219 | 0.0297 | 0.0029 | 0.1029 | **0.7238** | 0.0674 | 0.0176 | 0.0338 |
| 36-40 | 0.0030 | 0.0041 | 0.0041 | 0.0932 | 0.0771 | **0.7804** | 0.0126 | 0.0254 |

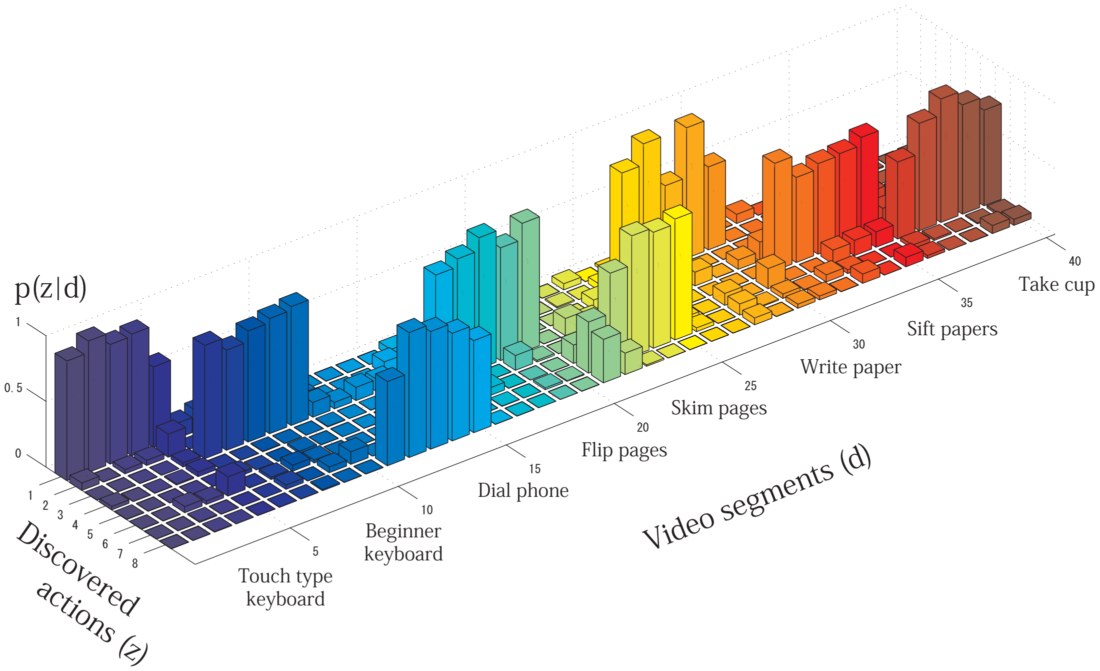


Figure 3.8. Baseline results using only **temporal** features for corpus $C_{OBJ}$. The horizontal axes gives the ground truth for each video $d$ and the discovered action category $z$. The vertical axis is the posterior probability $p(z|d)$.

### 3.4.2 Learning actions using object appearance

Now the proposed method using both motion and visual context is utilized to learn action categories from the same motion and object corpus $C_{OBJ}$. It is observed from the bar graph (Figure 3.10) of the posterior probability that all actions contained in the video corpus have been accurately discovered with high confidence. That is, the average PCC of the corpus $C_{OBJ}$ was increased to $99.05\%$. Leveraging the visual appearance of the action and related objects significantly increased the confidence of classification performance.

As a reference, the results (posterior probabilities) of action discovery using only spatial features is also given in Figure 3.9. It is observed that the action categories identified by both the visual context and motion helped to disambiguate each other. This resulted in an improvement in certainty regarding the classification of the segments. For example, when only the spatial features are used there is some uncertainty about the first two segments – a small distribution of the probability is shared between category 1 and category 3 of the $z$ axis. In other words, there is some confusion as to whether the first two segments contain a person touch typing on the keyboard or a beginner using the keyboard. Without the use of temporal features, the visual context of the keyboard and hands are very similar and therefore NMF is not able to differentiate the two actions completely. However, after the integration of temporal information via the bimodal model, the first two segments are classified with greater certainty. Note that the index of the discovered actions differs between the experiments. This has to do with the fact that the algorithm is unsupervised and is initialized randomly.

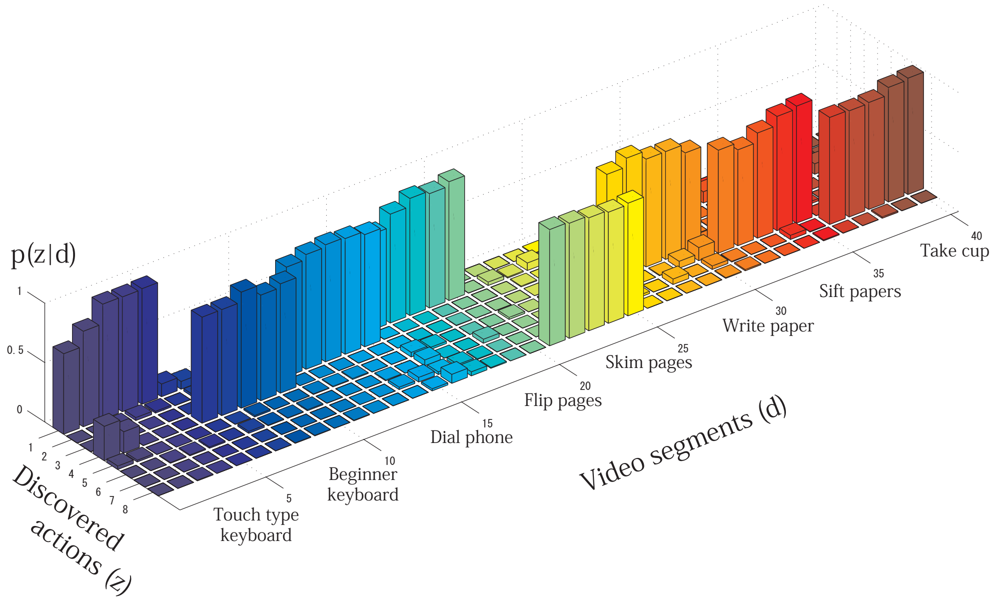


Figure 3.9. Using only **spatial** features for corpus $C_{OBJ}$. The horizontal axes gives the ground truth for each video $d$ and the discovered action category $z$. The vertical axis is the posterior probability $p(z|d)$.

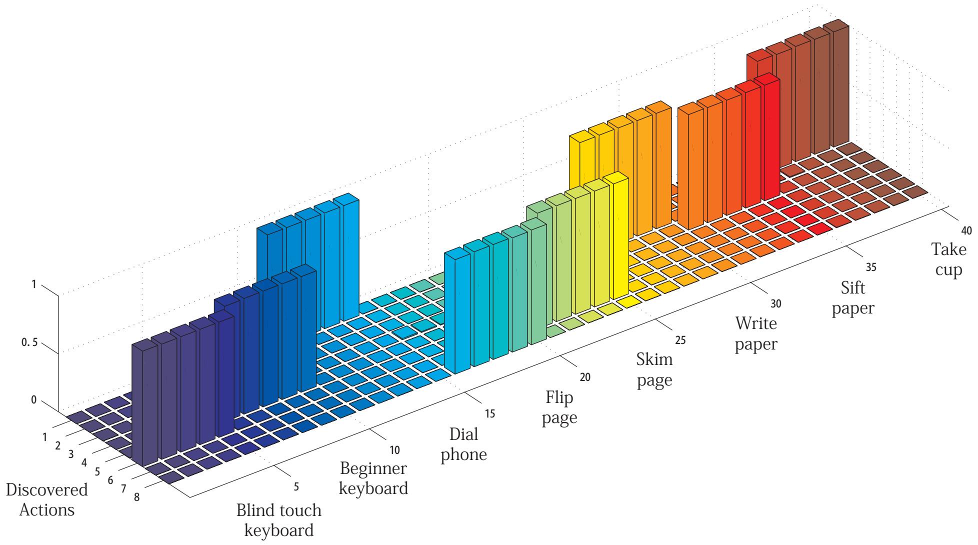


Figure 3.10. Posterior probabilities using the proposed **bi-modal** method with the corpus $C_{OBJ}$, which contained 8 different actions.

### 3.4.3 Learning actions using background appearance

The assumption of this approach is that visual context is relevant when it is constantly observed with a certain motion. Here the action and background corpus $C_{BG}$ is utilized to test whether the proposed method is able to distinguish between actions with very similar (same) motions, that can only be differentiated by their visual context. That is, given a video corpus containing three different motions performed in three different environments, can the proposed method discover nine unique actions from the database?

A bar chart of the posterior probability is given in Figure 3.14. Notice that each combination of visual context and motion have been correctly categorized with high probability (high confidence). The average PCC of the corpus was $92.29\%$ and the AUC was $1.000$. For this corpus, the visual features induced by the combination of motion and background enabled the system to differentiate the nine different action categories.

The independent results of the motion features and the visual features are given in Figure 3.11 and Figure 3.12. Notice that the PCC for action discovery with only temporal features was $44.96\%$ and the AUC was $0.878\%$ (Figure 3.13). Since there are only three differentiable motions in the database, the NMF decomposition of the temporal feature histograms is very difficult resulting in low performance.

In contrast, the PCC for action discovery using only spatial features was $91.35\%$ and the AUC was $1.0000$. In the special case of this corpus, it is observed that the information encoded by the visual context is sufficient to properly categorize the actions with high confidence. This experiment shows how the statics of static key frames can sometimes be sufficient to characterize certain types of motion.

As mentioned earlier, it is assumed that visual context is relevant when it is constantly observed with a certain motion. It is noted here that the proposed system will have problems when the number of irrelevant background types is less than the number of motion types in the database. That is, if a certain motion is frequently observed in front of the same unrelated object, the proposed method will include those visual features as part of the primitive action. In most cases, this should not be a problem given a sufficiently sized database generated over various backgrounds.

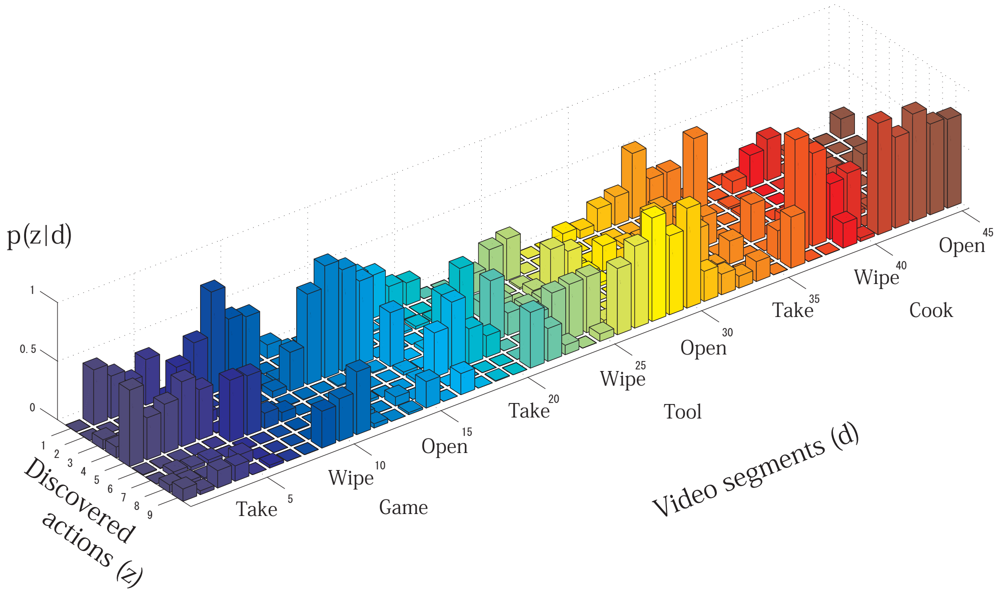


Figure 3.11. Results from corpus $C_{BG}$ using only **temporal** features.

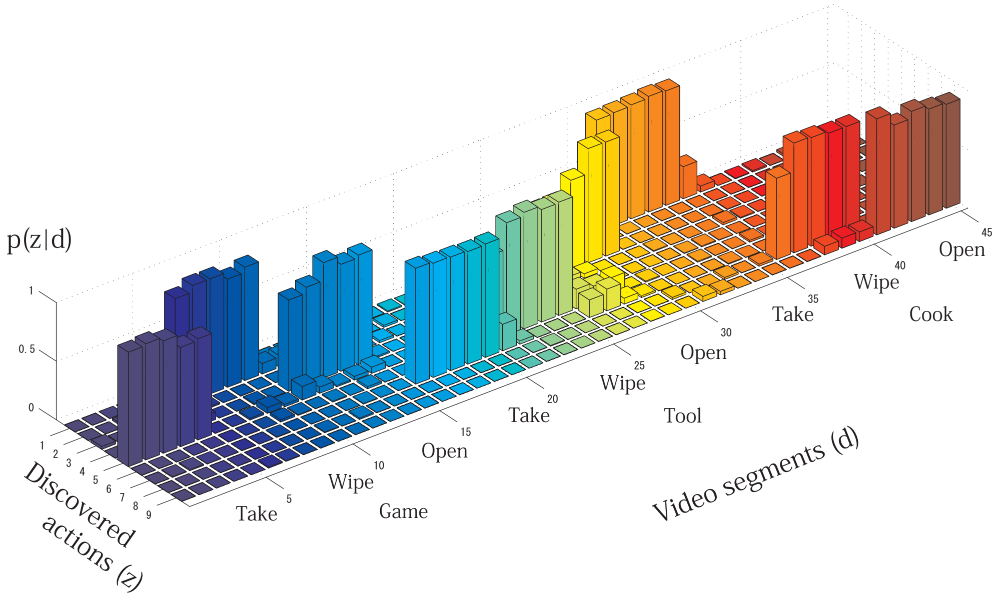


Figure 3.12. Results on the corpus $C_{BG}$ using only **spatial** features.

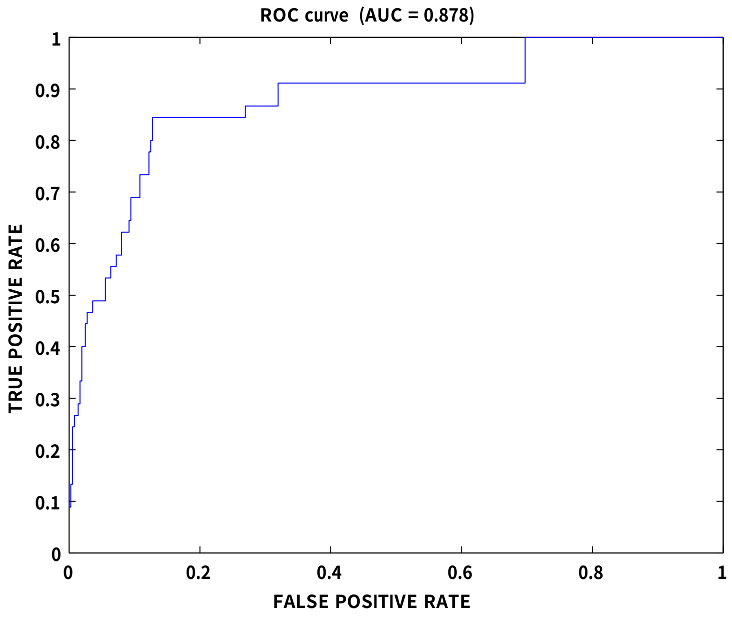


Figure 3.13. ROC and AUC for action discovery with only **temporal** features on corpus $C_{BG}$

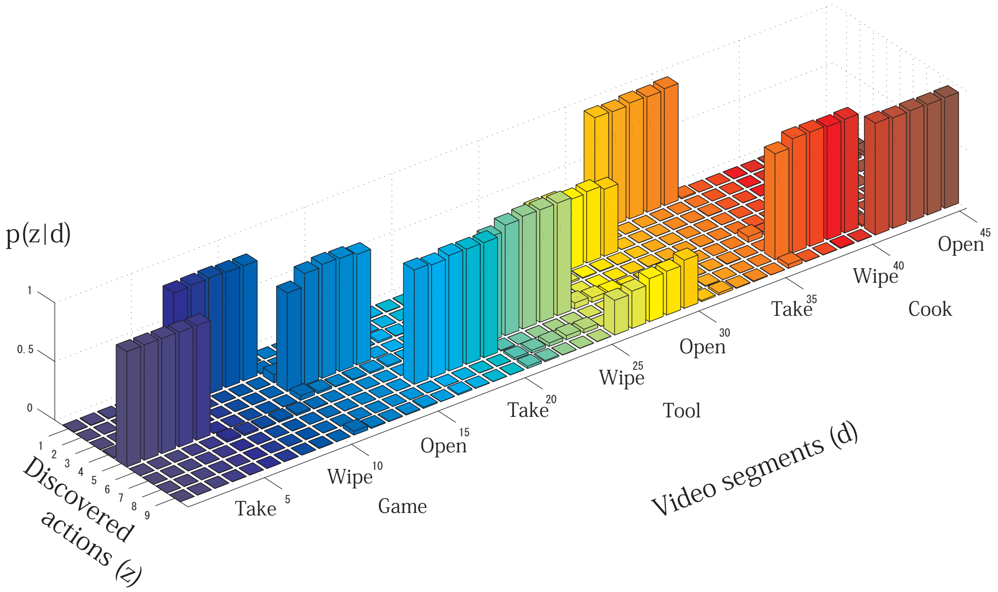


Figure 3.14. Results using the proposed method (**bimodal**) with the corpus $C_{BG}$ which includes 3 different motions and 3 different backgrounds. Ground truth labels are given along the axis.

### 3.4.4 Learning actions over a messy desk

In reality, primitive actions occur in various types of visual contexts and it is important to be able to leverage only the relevant visual features that should be associated with an action (Figure 3.1). In this experiment, the proposed method is applied to the motion with object and background corpus $C_{BGOB}$ and it is shown how the proposed method can leverage relevant visual features to discover actions categories, even with various cluttered backgrounds (visual noise).

Results show that the proposed approach is able to learn the actions of the corpus $C_{BGOB}$ with an average PCC of $93.62\%$ (Figure 3.15) and the AUC was $1.000$. A high AUC score was obtained despite the visual noise generated by the different background objects because the relevant visual features have a stronger signature (occur more often) in the histograms; a trait which is preserved during the NMF stage. The information gained from the relevant visual features is then used by the bi-modal model to effectively discover all of the eight latent primitive action categories.

Again, as a reference the PCC and AUC values achieved with only temporal feature and only spatial features are included. The PCC for action recognition with temporal features only was $77.32\%$ (Figure 3.18) and the AUC was $0.998$ (Figure 3.17). The PCC for action recognition with spatial features only was $82.68\%$ (Figure 3.16) and the AUC was $1.000$. It observed from the AUC values that the statistics of both the spatial features and the temporal features are sufficient to categorize the corpus. However, by integrating both modes, the bimodal model is able to categorize the corpus with higher valued probabilities.

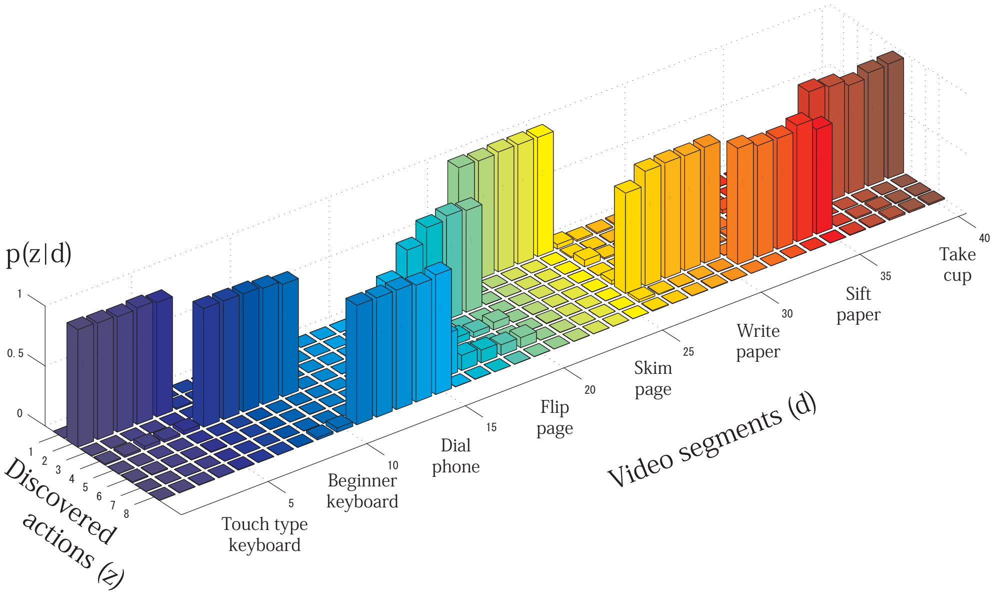


Figure 3.15. Results of the proposed method (**bimodal**) with corpus $C_{BGOB}$ which has 8 actions observed over various messy desktop environments.

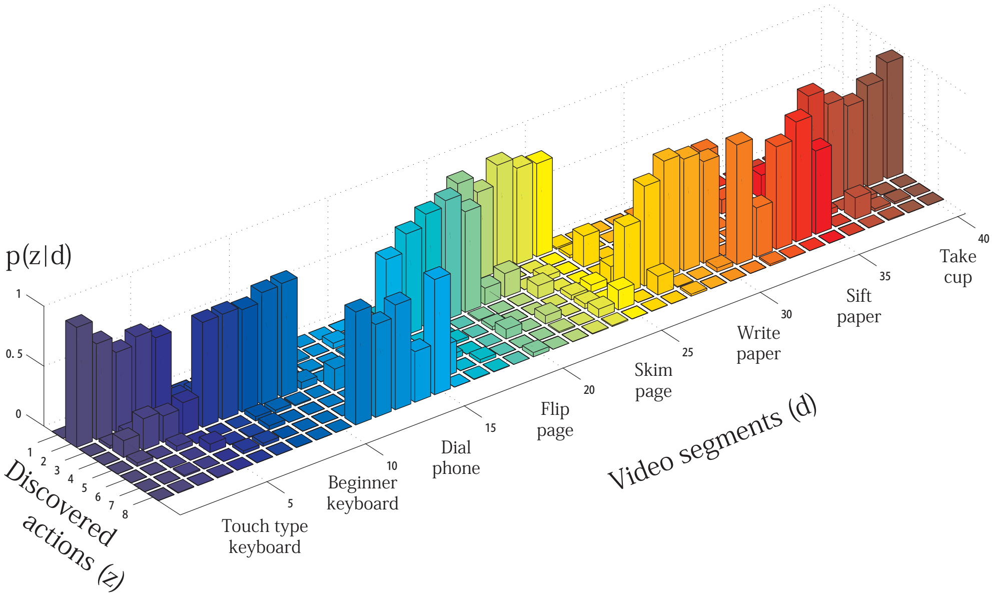


Figure 3.16. Results from the corpus $C_{BG}$ using only **spatial** features.

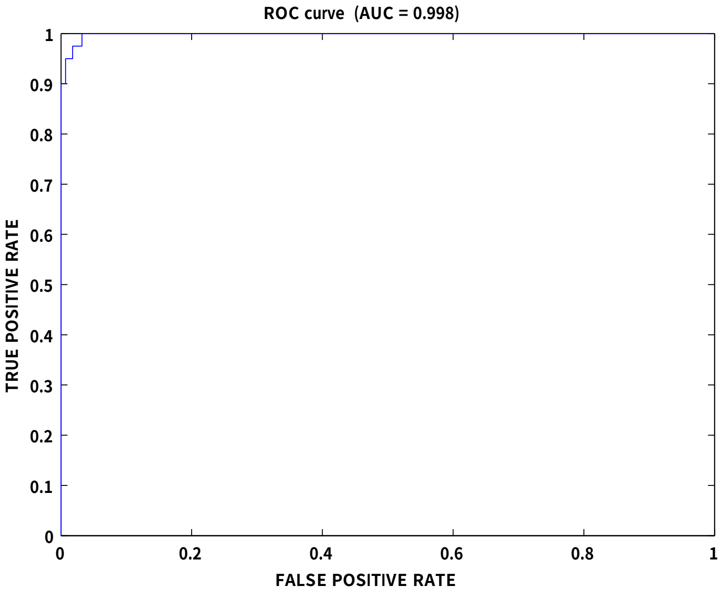


Figure 3.17. ROC and AUC for action discovery with only **temporal** features on corpus $C_{BGOB}$.

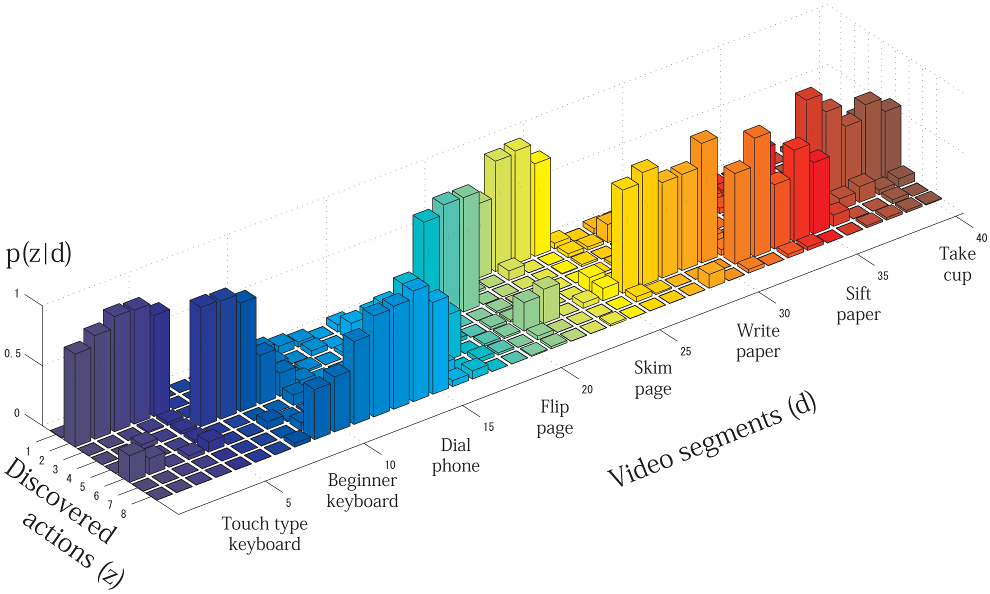


Figure 3.18. Results from the corpus $C_{BGOB}$ using only **temporal** features.

### 3.4.5 Learning actions from a corpus with extraneous motion

Up to this point, the previous experiments have worked with corpus elements (segments) that contain only the action categories to be discovered. It is conceivable that a video corpus will include segments that (1) do not contain target actions or (2) segments will contain only a portion of a target action or (3) segments will include non-target actions (extraneous motion) that might have an adverse effect on the discovery process. To examine the performance of the proposed method on a corpus that includes this type of noise, an experiment were executed on the extraneous motion corpus $C_{EXMO}$. The PCC for action recognition with both features was $61.69\%$ (Figure 3.20) and the AUC was $0.978$ (Figure 3.19).

Again as a reference, the results of using temporal features and spatial features independently is given. The PCC for action recognition with temporal features only was $64.62\%$ (Figure 3.22) and the AUC is $0.902$ (Figure 3.21). The PCC for action recognition with spacial features only was $93.16\%$(Figure 3.24) and the AUC is $0.996$ (Figure 3.22). Here it is observed that a bimodal model incurs a penalty when the statistics of one of the features is weak and not able to correctly categorize the corpus. In this case, the extraneous motions (temporal noise) in the database has adversely effected the statistics of the temporal features. However, with information gained from the visual context, the final categorization results in an average categorization performance between the two modes. In this case, better performance could be achieved by using only the visual features. However, this is dependent on the contexts of the video corpus and is not always the case. It has been observed from previous experiments that using both modes makes the system robust to a greater range of corpus content.

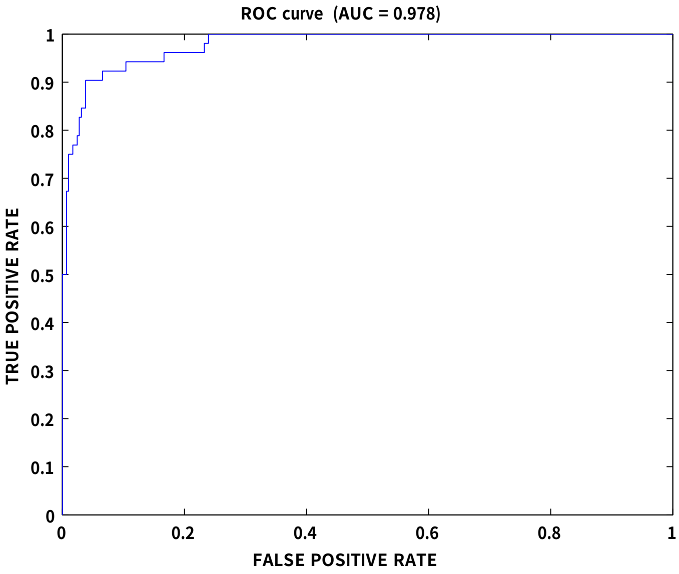


Figure 3.19. ROC from the extraneous motion corpus using the proposed **bimodal** model.

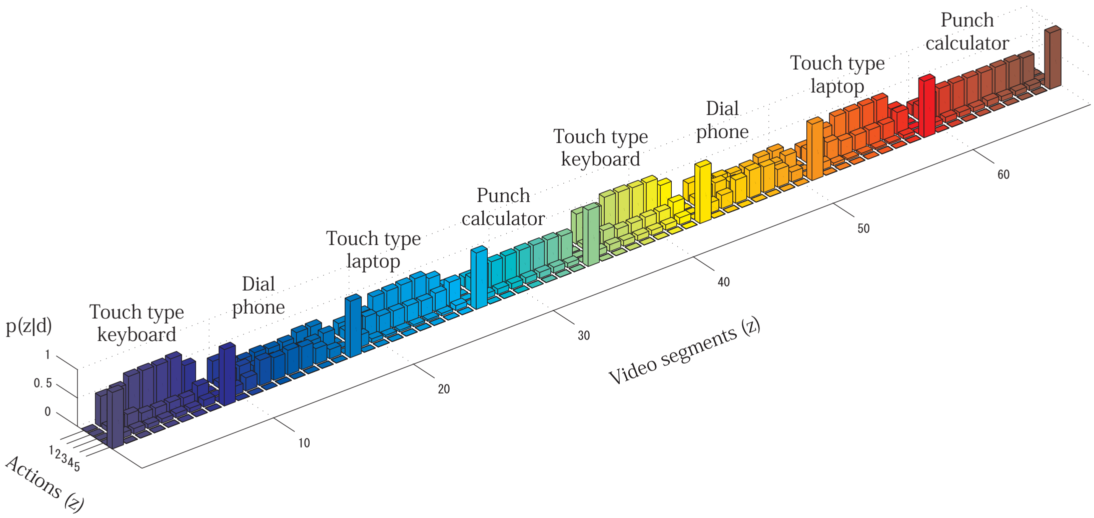


Figure 3.20. Results of the proposed **bimodal** method with corpus $C_{EXMO}$ with four actions including extraneous motion.

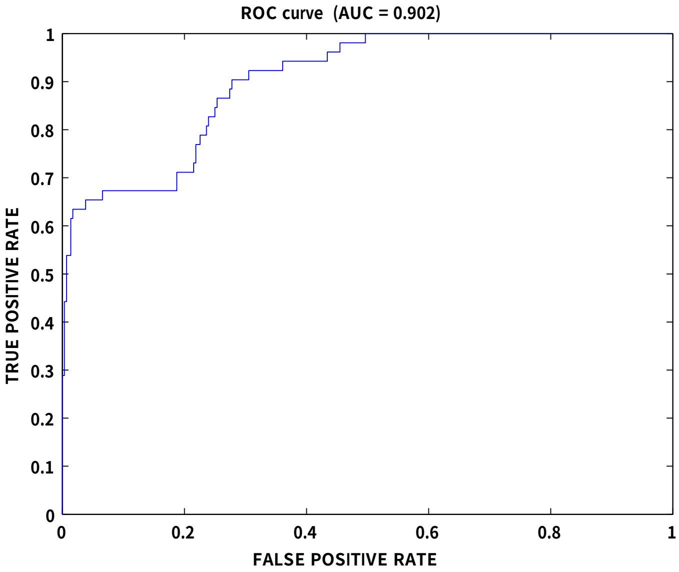


Figure 3.21. ROC of the corpus $C_{EXMO}$ using only **temporal** features.

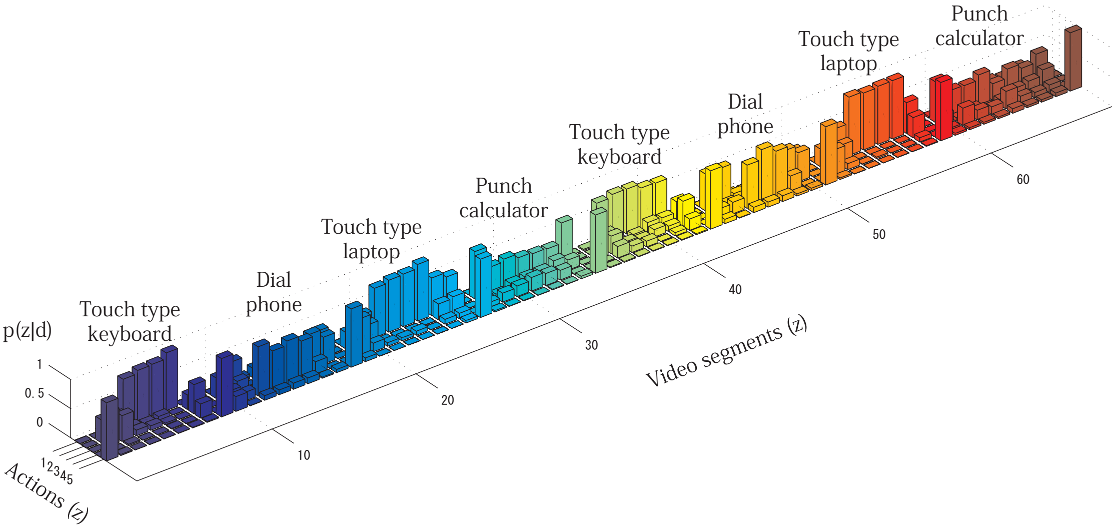


Figure 3.22. Results using only **temporal** features with corpus $C_{EXMO}$.

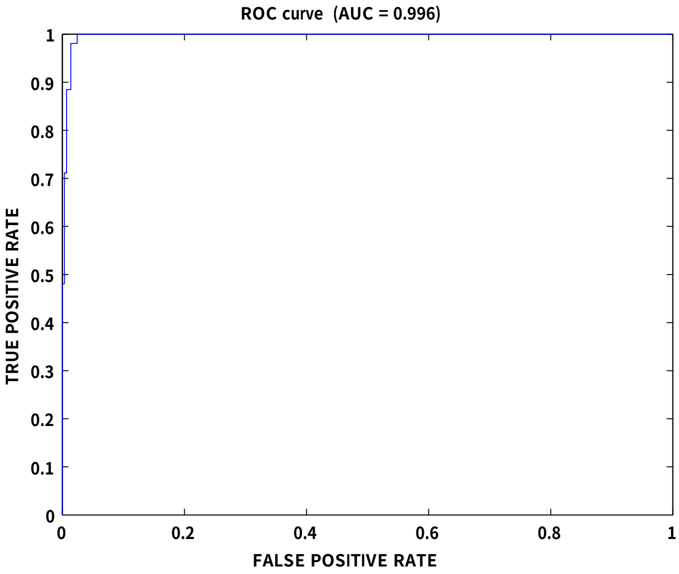


Figure 3.23. ROC curve of corpus $C_{EXMO}$ using only **spatial** features.

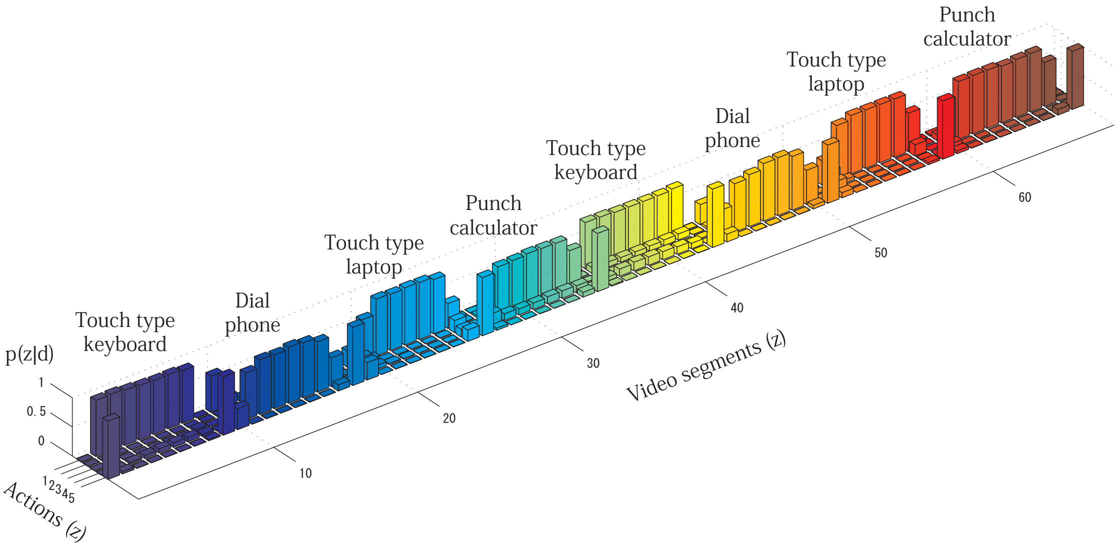


Figure 3.24. Results using only **spatial features** with corpus $C_{EXOB}$.

### 3.4.6 Comparing clustering methodologies

Many works using the bag-of-features representation of categories use offline algorithms to vector quantize the data by using such algorithms as the K-means clustering algorithm [CDW$^+$04], the EM algorithm with mixtures of Gaussians [Per08] or random sampling [DRCB05]. While offline approaches work well for text or images, the comparatively large number of features that can be extracted from video can pose some challenges for offline algorithms regarding computation time and memory usage.

This section makes a limited comparison between the standard offline K-means clustering algorithm and the proposed online nearest representative point clustering algorithm. It is shown that the overall performance of the NRPC approach is on par with K-means while at the same time offering faster processing and more efficient memory usage.

**K-means clustering**

The K-means algorithm [Mac67] is an iterative algorithm that clusters a set of data points by minimizing the distance (typically the Euclidean distance) between the points in a cluster. It utilizes termination criteria such as cluster centroid movement between iterations or a maximum number of iterations. The computation time is therefore dependent on the characteristics of the data points and the termination criteria. The termination criteria are usually set heuristically and the criteria have a direct impact on how long it will take the algorithm to converge.

The K-means algorithms also requires that all data records are stored in memory. While this is usually not a problem for small datasets, memory usage becomes an important issue when the number of data points becomes very large. For example, a computer with 4 GB of main memory can only store about 7 million SIFT features extracted (about 21 minutes of video if an average of 200 features are extracted from each frame). Since it is reasonable to assume that even a typical video database of home videos will contain several hours of video, the memory usage limitations of K-means is a very real issue.

**K-means performance on datasets**

A comparison between the performance of the NRPC algorithm presented in section 3.2.2 and the standard K-means clustering algorithm for the three datasets (section 3.3) is given in Table 3.2. It can be said that both NRPC and K-means obtain comparable results by correctly

Table 3.2. Comparison of NRPC and K-means on the same video corpus

| | NRPC | | Kmeans | |
|---|---|---|---|---|
| | PCC | AUC | PCC | AUC |
| Motion and object corpus $C_{OBJ}$ | 99.6 | 1.000 | 99.89 | 1.000 |
| Motion and background corpus $C_{BG}$ | 95.7 | 1.000 | 94.95 | 1.000 |
| Motion, object and background corpus $C_{BGOB}$ | 98.2 | 1.000 | 99.38 | 1.000 |
| Extraneous motion corpus $C_{EXMO}$ | 61.7 | 0.978 | 76.36 | 0.936 |

discovering all of the actions. It was observed that for simple datasets both clustering algorithms yield similar performance.

Table 3.3 contains a detailed comparison of the results for NRPC versus K-means clustering for different parameters.When the maximum number of iteration (Max It) and the stop criteria (Stop Crit) for K-means clustering are set to 10 and 1, respectively, it is observed that K-means clustering and NRPC have similar performance with respect to the AUC value and processing time. However, as stated earlier, speed and classification performance are data dependent and it would not be reasonable to make conclusions about the general case from this specific case. That being said, one possible reason for the success of NRPC is the fact that the features generated by video has the unique characteristic that the same spatial features are detected repeatedly for a series of frames that contain the same object. This means that small subsets of the data points are naturally grouped together in small clusters, which would explain why NRPC clusters the data with similar overall performance as K-means clustering.

The main difference that is observed is the memory usage. For example, the corpus $C_{BGOB}$ generates about 253,000 features, which requires about 130MB of memory to store to run K-means clustering. In contrast, NRPC processes the features online and only requires that the cluster centers be stored. For the corpus $C_{BGOB}$ NRPC requires only around 0.7 MB of space.

### Other considerations

Just as K-means clustering requires that the number of clusters is known, NRPC also requires that the cluster radius be known. In practice, a radius value between 0.01 and 0.02 was shown to be effective on preliminary tests with vectors (i.e. multi-dimensional data points) normalized to unit length. The data also shows that NRPC tends to produce more clusters in comparison to K-means. However, from the standpoint of memory usage the number of clusters is considerable smaller than the amount of space needed to store all of the data points.

Also, NRPC clustering can also be used as an initialization step for K-means clustering to leverage the strengths of both algorithms. Since K-means clustering can also be parallelized [OARS04], it would be possible to create a hybrid clustering technique that retains both the optimization and speed advantages of both algorithms.

## 3.5 Conclusion

A novel framework for discovering action categories by leveraging relevant visual context and motion features has been presented in this chapter. In the proposed framework, a fast two stage clustering algorithm was implemented via nearest representative point clustering and non-negative matrix factorization, to generate a term-by-document matrix as the input to the bimodal mixture model. The bi-modal mixture model used both visual features and temporal features to discover latent action categories. Through the experiments it was shown that the proposed approach is able to accurately classify actions by leveraging relevant visual appearance to disambiguate similar motions. It was also shown that the proposed method is robust against irrelevant visual features generated by the background while at the same time leveraging relevant visual features to accurately discover primitive action categories.

Table 3.3. Detailed comparison chart of performance based on clustering methodology. For NRPC the computation time for a single mode includes both feature extraction and clustering. For K-means clustering the time for clustering and extraction are given separately.

| | $\theta$ | Max Iter | Stop Crit | K | Temporal Features | Mem (MB) | Cluster (secs) | Extract (secs) | K | Spatial Features | Mem (MB) | Cluster (sec) | Extract (sec) | Total (sec) | AUC |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $C_{EXMO}$<br>K-means | - | 10 | 1.0 | 5,000 | 84,003 | 65.9 | 220 | 158 | 5,000 | 233,512 | 120 | 280 | 444 | 18 | 0.857 |
| | - | 10 | 1.0 | 2,000 | 84,003 | 65.9 | 52 | 157 | 2,000 | 233,512 | 120 | 95 | 403 | 12 | 0.876 |
| | - | 10 | 1.0 | 1,000 | 84,003 | 65.9 | 26 | 157 | 1,000 | 233,512 | 120 | 47 | 455 | 11 | 0.920 |
| | - | 10 | 1.0 | 500 | 84,003 | 65.9 | 13 | 157 | 500 | 233,512 | 120 | 24 | 455 | 11 | 0.936 |
| | - | 10 | 1.0 | 100 | 84,003 | 65.9 | 3 | 157 | 100 | 233,512 | 120 | 5 | 456 | 10 | 0.875 |
| | - | 100 | 0.1 | 500 | 84,003 | 65.9 | 19 | 157 | 500 | 233,512 | 120 | 36 | 464 | 11 | 0.945 |
| | - | 100 | 0.01 | 500 | 84,003 | 65.9 | 228 | 157 | 500 | 233,512 | 120 | 312 | 458 | 19 | 0.920 |
| $C_{EXMO}$<br>NRPC | 0.0050 | - | - | 27,460 | 84,003 | 21.5 | 447 | 0 | 7,495 | 235,216 | 3.84 | 570 | 0 | 17 | 0.989 |
| | 0.0075 | - | - | 12,644 | 84,003 | 9.9 | 285 | 0 | 4,283 | 235,216 | 2.19 | 536 | 0 | 14 | 0.938 |
| | 0.0100 | - | - | 5,307 | 84,003 | 4.2 | 208 | 0 | 2,399 | 235,216 | 1.23 | 500 | 0 | 12 | 0.993 |
| | 0.0150 | - | - | 1,719 | 84,003 | 1.3 | 179 | 0 | 698 | 235,216 | 0.39 | 448 | 0 | 10 | 0.983 |
| | 0.0200 | - | - | 780 | 84,003 | 0.6 | 170 | 0 | 241 | 235,216 | 0.12 | 460 | 0 | 11 | 0.906 |
| | 0.0300 | - | - | 254 | 84,003 | 0.2 | 166 | 0 | 48 | 235,216 | 0.03 | 417 | 0 | 10 | 0.763 |
| $C_{BGOB}$<br>K-means | - | 10 | 1.0 | 7,500 | 49,986 | 39.2 | 196 | 135 | 7,500 | 253,631 | 130 | 623 | 330 | 21 | 0.806 |
| | - | 10 | 1.0 | 5,000 | 49,986 | 39.2 | 133 | 135 | 5,000 | 253,631 | 130 | 280 | 316 | 14 | 0.985 |
| | - | 10 | 1.0 | 2,000 | 49,986 | 39.2 | 32 | 135 | 2,000 | 253,631 | 130 | 104 | 316 | 10 | 1.000 |
| | - | 10 | 1.0 | 1,000 | 49,986 | 39.2 | 16 | 136 | 1,000 | 253,631 | 130 | 52 | 319 | 9 | 1.000 |
| | - | 10 | 1.0 | 500 | 49,986 | 39.2 | 8 | 136 | 500 | 253,631 | 130 | 26 | 316 | 8 | 1.000 |
| | - | 10 | 1.0 | 100 | 49,986 | 39.2 | 1 | 136 | 100 | 253,631 | 130 | 5 | 316 | 8 | 1.000 |
| | - | 10 | 1.0 | 50 | 49,986 | 39.2 | 1 | 135 | 50 | 253,631 | 130 | 3 | 317 | 8 | 0.758 |
| $C_{BGOB}$<br>NRPC | 0.0050 | - | - | 15,282 | 49,986 | 12.0 | 227 | 0 | 6,947 | 253,631 | 5.4 | 434 | 0 | 11 | 0.860 |
| | 0.0100 | - | - | 2,658 | 49,986 | 2.1 | 152 | 0 | 2,794 | 253,631 | 2.2 | 377 | 0 | 9 | 0.846 |
| | 0.0150 | - | - | 838 | 49,986 | 0.7 | 141 | 0 | 903 | 253,631 | 0.7 | 349 | 0 | 8 | 1.000 |
| | 0.0200 | - | - | 388 | 49,986 | 0.3 | 139 | 0 | 311 | 253,631 | 0.2 | 339 | 0 | 8 | 0.979 |
| | 0.0300 | - | - | 137 | 49,986 | 0.1 | 137 | 0 | 66 | 253,631 | 0.1 | 332 | 0 | 8 | 0.751 |

# Chapter 4

# Learning the structure of activities

The previous chapter focused on the extraction of primitive actions from a video dataset by utilizing both motion features and visual context. As a result, the proposed method can be used to represent a video sequence as a temporal series of actions – an *action symbol string*. In this chapter, a new framework for discovering the basic temporal structure (grammar) within a action symbol string is proposed. To find the optimal grammar in a information-theoretic sense, the minimum description length principle is used to identify a set of primitive actions that defines an optimal stochastic context-free grammar.

## 4.1 Introduction

The stochastic context-free grammar (SCFG) is a model that has been widely utilized for natural language processing and in recent years, has also been shown to be effective in modeling human activities extracted from video [IB00, WSXZ01, ME02, MES03, OKA05, RA06]. The success of SCFGs in analyzing natural languages is largely due to its ability to represent the *hierarchical structure* found among words in a sentence. According to perceptual psychology [ZT01], this hierarchical structure is also characteristic of the primitive actions of a human activity[1] and like sentences, activities are perceived to have partonomic structure (a discrete temporal sequence of primitive actions). This similarity between strings of words and a series of actions gives one the rational basis for the use of an SCFG for activity analysis. Other non-hierarchical sequential state-based models (finite-state automata, hidden Markov models, *n*-

[1] As stated in chapter 2, the term *activity* is based on terminology introduced by Collins [CLK00] to refer to the high-level description of a temporal sequence of primitive actions.

grams, etc.) have also been successfully applied to human activity recognition but are limited by the fact that they do not explicitly describe the hierarchical structure of human activities.

One important task involved in using an SCFG for activity analysis is the task of *learning* the grammar. Most of the previous works however, have manually designed their own grammars and have avoided the issue of grammar learning. Ivanov [IB00] extracted primitive action words from a video sequence of a conductors arm using HMMs and was able to recognize the rhythmic meter using an SCFG. The grammar and it's probabilities however, were defined by Ivanov. Moore [ME02] used an SCFG to recognize people playing Black Jack and used the *a priori* information encoded in the grammar to deal with errors in the string of action words. Again, the grammar was defined by the author based on the basic rules of the game. Similarly, Minnen [MES03] leveraged the *a priori* knowledge of a predefined grammar to infer an action when the agent under analysis is occluded in the scene.

In contrast to works that used manually defined grammars, research dealing with the issue of automated learning has been minimal and assumes a pure data set for learning. Wang[WSXZ01] used an experimental scenario similar to Ivanov and implemented HMMs to produce primitive action symbols from a video segment of a conductors hand motions. The primitive actions produced by the HMMs were then fed into a pre-existing CFG learning algorithm COMPRESSIVE [NMW00] to learn the activity grammar. Due to the fact that COMPRESSIVE requires positive examples to generate the CFG, it can be shown that their system is very sensitive to noise in the input symbol string. That is, an unstable detector or an unrelated action would have an adverse affect on the learning process because this noise would be included into the learned grammar. While a noise-less input stream may be a reasonable assumption when learning a grammar from a string of words, it is a naive assumption when learning an activity grammar from a symbol string produced by stochastic detectors from a highly variable action sequence created by human actors.

In summary, most of the works using CFGs for activity analysis have used grammars manually designed by knowledge engineers while research focused on automated grammar learning has only used pre-exiting algorithms, assuming activities to be a noise-less stream of symbols. In contrast to previous works, this chapter proposes a new grammar learning method that deals with the issue of noise. The proposed method places an assumption of noise on different combinations of terminal symbols and tests that assumption using the minimum description length (MDL) principle. Then using the results of the MDL evaluation, the proposed method finds the best set of terminal symbols that yields the most compact and descriptive activity grammar.

## 4.2 Conceptual example

In order to understand the basic concepts underlying the proposed approach, this section gives a conceptual example to introduce the reasoning behind the proposed method. Given a symbol string $S$, the goal is to find the most *compact* yet *expressive* grammar that yields an optimal description of the symbol string. At first glance, no regularity is observed in the example string:

$$\texttt{S} \rightarrow \texttt{a x b y c a b x c y a b c x.}$$

Since it is assumed that noise exists in the string, it is possible that by removing different combinations of symbols, the underlying pattern may become more visible. Here for the sake of example, the convenient hypothesis is made that `y` is noise. Therefore, all of the `y` symbols are removed from the string[2]. This assumption shrinks the string into its new form:

$$\texttt{S} \rightarrow \texttt{a x b c a b x c a b c x.}$$

It is now observed that the substring `c a b` occurs twice in the string but no obvious *regularity* (some rule) that completely describes the symbol string can be observed. Therefore, another hypothesis is made that `x` is also a noise symbol, which results in the string:

$$\texttt{S} \rightarrow \texttt{a b c a b c a b c.}$$

Now it is clear that the substring `a b c` is repeated three times in the symbol string and so a new rule `A` is created and the symbol string $S$ is encoded with the new rule, yielding the compact description:

$$\texttt{S} \rightarrow \texttt{A A A}$$
$$\texttt{A} \rightarrow \texttt{a b c.}$$

Through this example it has been observed that by correctly assuming `x` and `y` to be noise, a compact grammar (`A` → `a b c`) and a deterministic description of the basic structure of the original symbol string was obtained as `S` → `A A A`. It will be shown later that this algorithm can be interpreted to be a type of hypothesis testing based on a description length criterion on the resulting grammar. The technical formulation of the concepts and methodology introduced here are given in the following sections.

[2] The `y` symbol is deleted for an illustrative purposes. Symbols are not actually deleted in the proposed method.

## 4.3 Preliminaries

### 4.3.1 The focus

It is necessary to first understand the focus of the proposed approach before proceeding to explain its details. First, the primary interest is high-level grammatical inference of human activities and not the methods for low-level primitive action extraction (as covered in chapter 3). The proposed method assumes a reliable low-level processing system (such as the one proposed in chapter 3) that returns a string of primitive action symbols. Second, this method works only with a strictly sequential string of primitive action symbols as the input (refer to section 2.2.4 for details). It is recognized that while most activities are sequential streams of primitive actions, intra-action relationships can also have other modes [All84], such as overlapping and concurrence. As such, other methods such as propagation networks [SHM$^+$04], Petri networks [GDDD04], deleted interpolation [KSS05] and other approaches using CFGs [RA06] have been proposed to address different temporal modes between primitive actions and activities for the *recognition* task. In contrast, in regards to the *learning* task, the claim of this chapter is that discovering the basic sequential structure between key actions is the first important step in establishing a strong *context* to discover other types of temporal modes. Here the issue of learning non-sequential temporal relationships is left for future work and this chapter will focus primarily on the discovery of temporally sequential action patterns. Therefore, given this focus, the discussion in this chapter is limited to the discovery of the grammar of a strictly sequential string of key action symbols.

### 4.3.2 Definition of noise

When considering the task of learning an activity from a string of action symbols, it is reasonable to expect different types of noise that might hide the basic structure of the activity that is to be learned. The first type of noise is inherent to human activities which is termed here as *inherent noise*. Inherent noise is caused by superfluous actions that do not play an important role in defining the activity to be learned. These secondary action symbols (noise symbols) tend to appear with irregular frequency and order, and fill in the gaps between the important action symbols. The second type of noise is *system noise* caused by the instability of the image processing system. System noise can be attributed to changes in appearance that cause the image processing system to insert, substitute or delete (miss) random symbols from the symbol string. Symbols that are inserted, substituted or deleted with a high frequency should not be used for learning because they introduce much randomness to the symbol string.

Since it is a very challenging task to address all the different modes of noise, several key assumptions are made to narrow the focus on a more manageable sub-problem, namely, *inherent insertion noise* in the training data. First, the assertion is made that a symbol is either a noise symbol or a non-noise symbol (a symbol cannot be noise and non-noise at the same time). Second, a non-noise symbol is defined to be a primary action symbol that defines or is directly related to the target activity. As for its properties, it shows regularity in its appearance and is observed with constant frequency and ordering. Noise symbols on the other hand are secondary action symbols that display random behavior with respect to frequency and ordering. The assumptions of this chapter are summarized as follows:

1. Noise symbols exist in the symbol string,
2. Non-noise symbols exist in the symbol string,
3. Noise and non-noise symbols are mutually exclusive,
4. Non-noise symbols occur with regularity.

While the primary assumption is that of inherent insertion noise, it is also shown in section 4.5.2 how the proposed method also shows robust performance when these assumptions are violated by using strings corrupted by both inherent insertion noise and system noise.

### 4.3.3 Context-free grammar

As mentioned before a context-free grammar (CFG) is used here to model human activity because of its ability to explicitly and compactly describe hierarchal structure. A CFG is defined by the 4-tuple $\mathbf{G} = \{\mathbf{T}, \mathbf{N}, S, \mathbf{R}\}$, where $\mathbf{T}$ is a finite set of terminal symbols, $\mathbf{N}$ is a finite set of non-terminal symbols, $S$ is the start symbol (a special non-terminal symbol) and $\mathbf{R}$ is the set of production rules. The production rules take the form $A \rightarrow \lambda^*$, which states that non-terminal symbol $A$ produces the string $\lambda^*$ of one or more symbols. When a probability $P(A \rightarrow \lambda^*)$ that satisfies the condition $\sum_i P(A \rightarrow \lambda_i^*) = 1$, is associated to each rule, the grammar becomes a stochastic content-free grammar (SCFG).

When a SCFG is used to model activity, each terminal symbol represents a primitive action and each non-terminal symbol represents an abstraction of a substring of terminal symbols. The start symbol $S$ represents a single activity, a complete symbol string produced by the grammar.

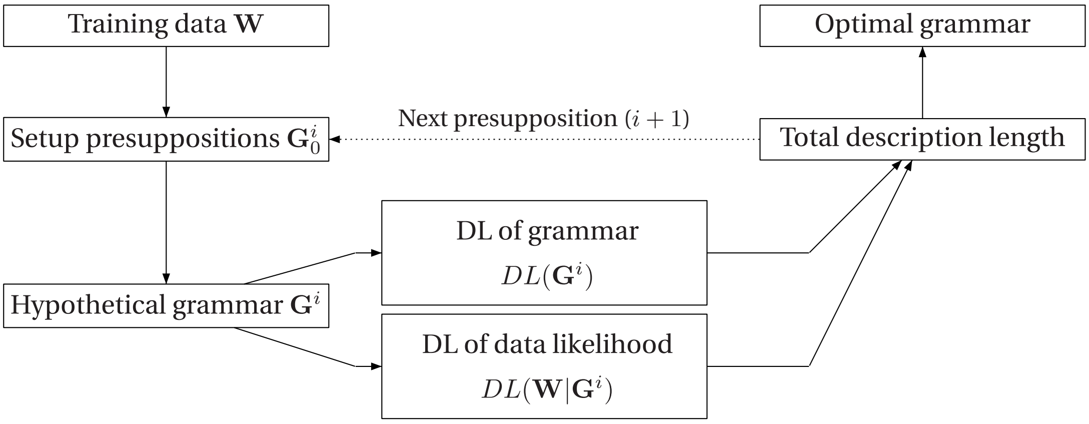

Training data $\mathbf{W}$
Optimal grammar
Next presupposition $(i+1)$
Setup presuppositions $\mathbf{G}_0^i$
Total description length
DL of grammar
$DL(\mathbf{G}^i)$
Hypothetical grammar $\mathbf{G}^i$
DL of data likelihood
$DL(\mathbf{W}|\mathbf{G}^i)$

Figure 4.1. Flowchart of the proposed MDL-based grammar learning method.

## 4.4 Proposed method

In this section, the key concepts introduced through the conceptual example in section 4.2 are formalized and it is shown how the MDL principle can be used to perform hypothesis testing to identify the correct noise symbols.

### 4.4.1 Setting up the presuppositions

To learn a grammar from the training data, it is required to first remove any noise that might be contained in the training data. Formally, given the training data $\mathbf{W} = \{W_1, \ldots, W_l\}$, a concatenation of $l$ activity sequences $W_i$, where each activity sequence $W_i = \{w_1, \ldots, w_p\}$ is a string of primitive action symbols $w_j \in \mathbf{T}$, it is the goal of this method to identify the symbols that are not useful (noise) for learning the grammar. However, since it is not know *a priori* which symbols are noise, it is proposed to set up various presuppositions (noise or not noise) against each unique primitive symbol and evaluate those presuppositions using an MDL criterion. Here it is explain how a single presupposition or hypothesis is set up.

A single hypothesis divides the set of primitive actions (terminal symbols) into two sets: the set of noise symbols $\mathbf{w}^f = \{w_1^f, \ldots, w_v^f\}$ and the set of non-noise symbols $\mathbf{w}^t = \{w_1^t, \ldots, w_u^t\}$. Next, an initial grammar is constructed to reflect the hypothesis. The initial grammar given in its general form is the set of production rules

$$\mathbf{R}_0 = \left\{ \begin{array}{c} S \rightarrow \mathbf{W}' \\ N_1 \rightarrow w_1^t \\ \vdots \\ N_u \rightarrow w_u^t \\ \eta \rightarrow \eta\ \eta \\ \eta \rightarrow w_1^f \\ \vdots \\ \eta \rightarrow w_v^f \end{array} \right\}. \tag{4.1}$$

The first rule of the form $S \rightarrow \mathbf{W}'$ is the start production rule. $S$ is a nonterminal symbol that represents all possible symbol strings produced by the grammar and in the initial stage $\mathbf{W}'$ is the concatenated training data encoded by the other production rules of the initial grammar. To attain the encoded input symbol string $\mathbf{W}'$, a plain input symbol string $\mathbf{W}$ is encoded to reflect the presuppositions made about each terminal symbol. This is done by replacing each terminal symbol $w_i$ with the appropriate nonterminal symbol using the preterminal production rules, which are defined next.

The set of production rules of the form $N_i \rightarrow w_i^t$ is created for each presupposed non-noise symbol, where $w_i^t$ is a non-noise terminal symbol and $N_i$ is a newly created nonterminal. These preterminal rules effectively preserve the unique identity of the symbol in the training data.

The set of generic preterminal production rules of the form $\eta \rightarrow w_j^f$ is created for each noise terminal symbol, where $w_j^f$ is a noise terminal symbol and the nonterminal $\eta$ is a generic nonterminal representing all noise symbols. The generic absorption rule $\eta \rightarrow \eta\ \eta$ is also created, which encodes a series of adjacent noise symbols. An example of setting up a hypothesis in the form of an initial grammar is given in Figure 4.2.

### 4.4.2 Learning the hypothesis grammar

Now that the presuppositions on the primitive action symbols have been encoded into the initial grammar, the next step is to learn the hypothesis grammar. This initial grammar is called the *hypothesis* grammar because it reflects a hypothesis (presupposition) about which symbols are noise and which symbols are not noise. In later sections it is shown how each hypothesis is tested by measuring the expressive power of each hypothesis grammar.

In this proposed method the heuristic CFG learning algorithm COMPRESSIVE is implemented to learn the grammar. When the original (hidden) grammar conforms to certain constraints and there is sufficient training data, the algorithm is able to learn a grammar that is

Input strings:

$W_1 = 1\, c\, a\, b\, a\, a\, b$

$W_2 = 2\, c\, a\, b\, a\, a\, b\, c$

$W_3 = 3\, a\, b\, a\, a\, b\, c$

Hypothesis:

$c$ is noise.

Initial grammar $\mathbf{R}_0$:

$S \rightarrow 1\, \eta\, A\, B\, A\, A\, B\, 2\, \eta\, A\, B\, A\, A\, B\, \eta\, 3\, A\, B\, A\, A\, B\, \eta$

$A \rightarrow a$

$B \rightarrow b$

$\eta \rightarrow c$

$\eta \rightarrow \eta\, \eta$

Figure 4.2. An example of setting up a hypothesis grammar.

of the same family of the original grammar. Four assumptions made regarding the original grammar are: (1) there are no cyclic (recursive) rules in the grammar, (2) there are no alternative expansions for non-terminals (only one expansion for a given non-terminal), (3) there are no abstractions (the number of symbols on the left-hand side of a rule is never 1) and (4) the grammar is optimal with respect to its description length.

When the original grammar does not conform to these assumptions, the grammar learned by the algorithm tends to be more complex (have more production rules) than the original grammar. However, since it is later shown that the primary concern is that of identifying the hypothesis that minimizes the overall description length, the relative difference in complexity between hypothesis grammars is more important than the absolute similarity (distance) to the original grammar.

COMPRESSIVE uses a function that quantifies the change in description length $\Delta DL$ to find the best *n-gram* in the grammar that minimizes (compresses) the overall size of the grammar. For a *n-gram* $\nu$ with length $n_\nu$ and occurrence $m_\nu$, the function is given as

$$\Delta DL = n_\nu \cdot m_\nu - (n_\nu + 1) - m_\nu. \tag{4.2}$$

In words, the change in description length is equivalent to the decrease caused by the removal

Input symbol string:

$S \to a\ b\ c\ d\ a\ b\ c\ d\ b\ c\ d\ a\ b\ a\ b$

| Pattern<br>$\nu$ | Occurrence Frequency<br>$n_\nu$ | Length<br>$m_\nu$ | Compression Factor<br>$\Delta DL$ |
|---|---|---|---|
| $b\ c\ d$ | **3** | **3** | **2** |
| $a\ b\ c\ d$ | 2 | 4 | 1 |
| $a\ b$ | 4 | 2 | 1 |
| $c\ d$ | 3 | 2 | 0 |

(1) Replace N-gram with maximum compression as new rule:

$A \to b\ c\ d$

(2) Encode input:

$S \to a\ A\ a\ A\ A\ a\ b\ a\ b$

(3) Repeat steps (1) and (2)

Figure 4.3. An example of COMPRESSIVE.

of $\nu$ ($m$ occurrences of length $n$), minus the increase of inserting a new rule $n + 1$, minus the increase of inserting of the new nonterminal symbol $m$ times. An example is given in Figure 4.3.

Once the best substring $\nu$ has been found and replaced by the new nonterminal, the algorithm repeats that process on the resulting grammar until there are no more *n-grams* can be found that decrease the size of the grammar. During the iterative process, the occurrence counts for the best *n-grams* are stored and are used later to calculate the rule probabilities.

Upon completion of COMPRESSIVE, the grammar is post-processed. Recall that the original segmented input symbol string $\mathbf{W}$ was encoded by the presuppositions to acquire $\mathbf{W}'$. Now after the completion of the COMPRESSIVE algorithm, the input string has been compressed to its new form $\mathbf{W}''$. In the post-processing step, the string $\mathbf{W}''$ is reverted back to its original $l$ activity sequences and sequences that have the same structure are grouped together. To do this, the $S$ rule, $S \to W_1'' \cdots W_l''$ is removed from the grammar. Next, each sequence separated and a new $S$ rule is placed back into the grammar for each unique sequence $S \to W_1'', \cdots, S \to W_h''$. Since unique sequences are only inserted once into the grammar $h \leq l$. The probability for each production rule is calculated with the following equation:

$$P(N \to \lambda_i^*) = \frac{c(N \to \lambda_i^*)}{\sum_j c(N \to \lambda_j^*)}, \tag{4.3}$$

such that $N$ is a nonterminal, $\lambda^*$ is the right-hand side of the rule and $c(\cdot)$ is a count function. Rules with zero probability are removed from the grammar.

This completes the step for learning the hypothesis grammar based on the initial presuppositions. The next section explains the framework used to evaluate the quality of the hypothesis grammar.

### 4.4.3 Testing using the MDL principle

The next goal is to find a presupposition on the primitive action symbols that yields both a *compact* yet *expressive* grammar that describes the input symbol string. Reworded in the framework of MDL, the goals it is find an optimal selection of non-noise symbols that will yield a grammar **G** that minimizes the sum of the description length of the grammar $DL(\mathbf{G})$ and the description length of the data encoded by the grammar $DL(\mathbf{W}|\mathbf{G})$ (data log-likelihood).

$$\begin{aligned} \hat{\mathbf{G}} &= \arg\min_G \{\ DL(\mathbf{G}) \ + \ DL(\mathbf{W}|\mathbf{G})\ \} && (4.4) \\ &= \arg\min_G \{-\log P(\mathbf{G}) - \log P(\mathbf{W}|\mathbf{G})\}. && (4.5) \end{aligned}$$

In this section, the encoding technique proposed by Stolcke [Sto94] is implemented to find the description length of the grammar and the inside (beta) probabilities introduced by Pynadath [PW98] are implemented to calculate the description length of the data likelihood.

#### Description length of the grammar

The first term of the MDL equation is the description length of the grammar $DL(\mathbf{G})$. $DL(\mathbf{G})$ is a measure of the compactness of the grammar and is an indicator of the *regularity* found in the training data.

Since the probability of the grammar can be interpreted as the joint probability of the *pa-*

*rameters* $\theta_G$ and *structure* $G_S$ of the grammar,

$$P(\mathbf{G}) = P(G_S, \theta_G) = P(\theta_G|G_S)P(G_S), \tag{4.6}$$

the description length of the grammar can be acquired by summing the description length of the grammar parameters $DL(\theta_G|G_S)$ and the description length of the grammar structure $DL(G_S)$. The description length of parameters $DL(\theta_G|G_S)$ is computed by using the parameter probability $P(\theta_G|G_S)$ and calculating the prior on the structure $DL(G_S)$ directly from the grammar using information theory.

First, the prior on the grammar parameters $P(\theta_G|G_S)$ is calculated as the product of Dirichlet distributions (equation 4.7), such that each Dirichlet distribution represents an uniformly distributed probability across all $q$ possible productions of a nonterminal symbol $N$.

$$P_N(\theta_G|G_S) = \frac{1}{B(\alpha_1, \ldots, \alpha_q)} \prod_{i=1}^{q} \theta_i^{\alpha_i - 1} \tag{4.7}$$

where the parameters for each nonterminal is represented by the multinomial distribution $\theta = (\theta_1, \ldots, \theta_q)$ and $B$ is a beta distribution. Since there is no prior knowledge about the distribution of the grammar parameters, the rule parameters $\theta_i$ and prior weights $\alpha_i$ are set to be uniformly distributed, similar to the original work [Sto94]. The description length of the parameters of the grammar is given by $-\log P(\theta_G|G_S)$.

Second, the structure probability $P(G_S)$ is calculated by directly computing the description length of the structure $DL(G_S)$. $DL(G_S)$ can be defined as the sum of two parts: (1) the description length of the production rule symbols and (2) the description length of number of symbols in the production rule. The description length of the number of symbols is computed from equation (4.8) on the assumption that the length of the production rule is drawn from a Poisson distribution ($\mu = 3$ is used for experiments) shifted by one since the smallest possible rule is of length two.

$$-\log P(r-1; \mu) = -\log \frac{e^{-\mu}\mu^{r-1}}{(r-1)!}. \tag{4.8}$$

Assuming all symbols have the same occurrence probability, $\log_2 |\Sigma|$ bits per symbol is needed, where $\Sigma$ is the set of all symbols. Therefore, the description length of $r$ symbols requires $r \log |\Sigma|$ bits to transmit. The total description length of the structure is given by:

$$DL(G_S) = \sum_{R \in \mathbf{R}} \left(-\log P(r_R - 1; \mu) + r_R \log |\Sigma|\right). \tag{4.9}$$

Further explanation and justification of the formulation of the description length of the grammar can be found in the original work [Sto94].

**Description length of the likelihood**

It is not enough to evaluate the description length of the grammar because a grammar chosen purely based on grammar size will favor a very small grammar which may not explain the data well. The second term in the MDL equation is the description length of the data likelihood $DL(\mathbf{W}|\mathbf{G})$. $DL(\mathbf{W}|\mathbf{G})$ works to balance the effect of the first term by quantifying the expressive power of the grammar.

First, the data likelihood is calculated and then converted into a description length using Shannon's coding theory (negative log of the probability). The data likelihood is calculated using a chart of $\beta$ probabilities created using the procedure outlined in the original work [PW98]. The chart defines a function $\beta(N, j, k)$, the probability that the non-terminal $N$ is the root node of a subtree, at abstraction level $k$, with a terminal substring of length $j$. Once a chart has been constructed for a sequence $W = \{w_1, \ldots, w_{j_{max}}\}$, the data likelihood can be computed as a sum of $\beta$ probabilities for all strings of length $j_{max}$ produced by the root node $S$. Due to the insertion of abstraction rules when constructing the initial grammar and the possible creation of abstraction rules at post-processing, the maximum abstraction level $k_{max}$ is two.

$$P(W_i|\mathbf{G}) = \sum_{k=1}^{k_{max}} \beta(S, j_{max}, k), \tag{4.10}$$

The total likelihood for all the sequences **W** is computed by equation (4.11) as a product of likelihoods for each sequence $W_i$. After the total likelihood has been computed, it is converted into a description length by taking the minus logarithm.

$$P(\mathbf{W}|\mathbf{G}) = \prod_{i=1}^{n} P(W_i|\mathbf{G}). \tag{4.11}$$

In summary, by calculating the description length of the grammar and the description length of the data likelihood, a framework for evaluating the quality of a presupposition made on the terminal symbols has been created. By identifying the hypothesis grammar that minimizes the total description length, the grammar that optimally describes the data is acquired.

### 4.4.4 The recovered grammar

The proposed method uses the MDL criterion to discover the most optimal grammar from a set of hypothesis grammars. This section gives a brief discussion on the nature of the recovered optimal grammar and also clarifies the focus of the proceeding quantitative analysis.

No claim is made that the recovered optimal grammar has a topology that is the same as the original grammar. Except in the special case where the grammar conforms to a set of assumptions made by the COMPRESSIVE algorithm (section 4.4.2), the underlying assumptions significantly inhibit the type of structures that can be learned.

This however is not a problem for the proposed framework since the aim is to identify a grammar that optimally characterizes the basic structure (rules) between the correct non-noise symbols. To this end, the propose method is primarily concerned with the relative differences between hypothesis grammars and not the difference from the original grammar. In fact, depending on the form of the original grammar, the basic structures that are learned might be less complex or more complex than the original grammar.

Next, the goal of the following quantitative analysis is to show that the proposed method can consistently assign an optimal score to the grammar that uses the correct non-noise symbols and learns the basic structure of the original grammar.

## 4.5 Experiments with synthetic data

This section explores the conditions under which the proposed method is valid through experiments with synthetic data generated by a known grammar. It is also shown through an experiment with real data that the proposed method is able to produce intuitive results that aligns well with a human understanding of the target activity.

The synthetic data for each experiment was created using a pre-defined stochastic context-free grammar written according to a set of conditions. A set of $d$ sample strings was generated by the artificial grammar and was used to analyze the proposed method. After the analysis, each hypothesis grammar was ranked according to its description length. Throughout this section, the grammar which uses the correct non-noise symbols is termed as the *true grammar* and use the rank of the grammar as a measure of the success of the proposed method. The desire is for the rank of the true grammar to always be first (i.e. the global solution of the MDL criterion). An example of a predefined grammar is given in Figure 4.4 and a ranked list of hypothesis grammars in given in Table 4.1.

| | | | |
|---|---|---|---|
| S | → | EN ACT EX | [1.0] |
| EN | → | A | [0.5] |
| EN | → | A INSERT | [0.5] |
| EX | → | B | [0.5] |
| EX | → | B INSERT | [0.5] |
| ACT | → | C | [0.5] |
| ACT | → | C INSERT | [0.5] |
| A | → | a | [1.0] |
| B | → | b | [1.0] |
| C | → | c | [1.0] |
| INSERT | → | nd | [0.333] |
| INSERT | → | ne | [0.333] |
| INSERT | → | INSERT INSERT | [0.334] |

Figure 4.4. An example of an one pattern synthetic grammar with three non-noise symbols and two noise symbols.

Table 4.1. Ranked list of hypothesis grammars - The true grammar marked in **bold** is given a sub-optimal rank due to a small sized training set.

| Rank | Symbols | | $DL(G)$ | $DL(W\|G)$ | Total |
|---|---|---|---|---|---|
| 1 | a b | 2 | 117.85 | 487.04 | 604.89 |
| 2 | a c | 2 | 126.81 | 489.01 | 615.82 |
| **3** | **a b c** | **3** | **348.69** | **327.84** | **676.52** |
| 4 | b c | 2 | 187.57 | 517.32 | 704.89 |
| 5 | a | 1 | 85.58 | 689.34 | 774.92 |
| 6 | c | 1 | 89.69 | 703.08 | 792.77 |
| 7 | b | 1 | 113.80 | 758.57 | 872.37 |
| 8 | a ne | 2 | 362.22 | 622.20 | 984.42 |
| 9 | a nd | 2 | 403.36 | 604.69 | 1008.05 |
| 10 | | 1 | 70.82 | 942.46 | 1013.28 |
| 11 | nd | 1 | 223.37 | 826.17 | 1049.53 |
| 12 | ne | 1 | 223.37 | 854.10 | 1077.46 |
| 13 | a c ne | 3 | 664.92 | 415.79 | 1080.71 |
| 14 | c nd | 2 | 540.10 | 566.35 | 1106.45 |
| 15 | c ne | 2 | 512.91 | 604.70 | 1117.61 |
| 16 | a c nd | 3 | 749.00 | 399.35 | 1148.35 |
| 17 | a b nd | 3 | 774.74 | 397.38 | 1172.12 |
| 18 | a b ne | 3 | 758.90 | 422.24 | 1181.14 |
| 19 | b nd | 2 | 608.16 | 636.22 | 1244.38 |
| 20 | b ne | 2 | 608.30 | 675.64 | 1283.94 |
| 21 | b c nd | 3 | 981.49 | 398.73 | 1380.22 |
| 22 | b c ne | 3 | 999.61 | 422.70 | 1422.31 |
| 23 | a b nd ne | 4 | 1268.58 | 260.39 | 1528.97 |
| 24 | a b c nd | 4 | 1300.13 | 257.39 | 1557.52 |
| 25 | a b c ne | 4 | 1300.13 | 257.39 | 1557.52 |
| 26 | a c nd ne | 4 | 1300.13 | 257.39 | 1557.52 |
| 27 | b c nd ne | 4 | 1300.13 | 257.39 | 1557.52 |
| 28 | a b c nd ne | 5 | 1300.13 | 257.39 | 1557.52 |
| 29 | nd ne | 2 | 885.68 | 706.60 | 1592.28 |
| 30 | a nd ne | 3 | 1145.23 | 489.99 | 1635.22 |
| 31 | b nd ne | 3 | 1151.62 | 487.99 | 1639.61 |
| 32 | c nd ne | 3 | 1280.75 | 377.39 | 1658.15 |

### 4.5.1 Inherent insertion noise

Three different grammar parameters were varied to examine the performance of the proposed method to different types of inherent noise. First, three types of artificial grammars with different numbers of patterns were defined to evaluate the response of the proposed method to grammars with increasing complexity. Type one grammars had only one basic pattern (one $S$ rule) while type two and type three produced two patterns (two $S$ rules) and three patterns (three $S$ rules), respectively. The basic patterns of type two and type three grammars were different permutations of the same non-noise symbols. An example of a type one grammar and a type two grammar are given in Figure 4.4 and Figure 4.6, respectively. Second, for each type of synthetic grammar, the number of terminal symbols were varied from 6 to 10. Several permutations between the number of noise and non-noise symbols were tested. An insertion noise rule was added for every non-noise production rule to simulate the random insertion of noise between non-noise symbols. Third, to evaluate the effect of the sample size on the results, several training sets consisting of $d = 50, 150, 300, 500, 1000$ randomly produced strings were analyzed for each artificial grammar. The parameters and results for each artificial grammar are given in Table 4.2.

The results show that the proposed method has identified the correct set of non-noise symbols when the sample size is sufficiently large (Table 4.2). Equivalently, the proposed method has been shown to produce sub-optimal results when the size of the training set was too small. The results also show that complex grammars require more training samples than do simple grammars. It was also observed that the rank of the true grammar converges faster to the top position for simpler grammars (Figure 4.5). Sub-optimal results were encountered when the sample size was not sufficient because the learned grammar was under-developed and the data likelihood was under-representative of the data. Specifically with respect to the learned grammar, the insufficient sample size means that the extent of the randomness of the real noise symbols is not fully observed and therefore not fully described by the learned grammar. As a result, grammars using noise symbols are under-developed and are not properly penalized with a long description length.

With respect to the description length of the data likelihood, an insufficient sample size means that the data is not representative of the true set of strings that could be produced by the hidden grammar. As a results, the description length of the data likelihood becomes a small value and is constrained to a narrow range of values. This means that the description length of the data likelihood plays a weaker role in determining the optimal grammar. When these two aspects are combined, a small sample size creates a strong bias toward simple grammars. In fact in the experiments with synthetic data, the true grammar was always outranked by smaller grammars when the sample size was insufficient (e.g. Table 4.1). Later a strategy for balancing the total description length is introduced in section 4.6.3.

Table 4.2. Results with synthetic data (inherent insertion noise).

| | | | $d = 50$ | $d = 150$ | $d = 300$ | $d = 500$ | $d = 1000$ |
|---|---|---|---|---|---|---|---|
| Type | Non-noise | Noise | Rank of the true grammar | | | | |
| 1 | 3 | 3 | 3 | 1 | 1 | - | - |
| 1 | 3 | 4 | 3 | 1 | 1 | - | - |
| 1 | 3 | 5 | 3 | 1 | 1 | - | - |
| 1 | 3 | 6 | 5 | 1 | 1 | - | - |
| 1 | 3 | 7 | 4 | 1 | 1 | - | - |
| 1 | 4 | 3 | 12 | 4 | 1 | 1 | 1 |
| 1 | 4 | 4 | 15 | 4 | 1 | 1 | 1 |
| 1 | 4 | 5 | 11 | 4 | 1 | 1 | 1 |
| 1 | 4 | 6 | 14 | 4 | 1 | 1 | 1 |
| 1 | 5 | 3 | 30 | 15 | 5 | 1 | 1 |
| 1 | 5 | 4 | 34 | 15 | 5 | 1 | 1 |
| 1 | 5 | 5 | 54 | 15 | 5 | 1 | 1 |
| 2 | 3 | 3 | 11 | 4 | 1 | 1 | - |
| 2 | 3 | 4 | 12 | 4 | 1 | 1 | - |
| 2 | 3 | 5 | 28 | 4 | 1 | 1 | - |
| 2 | 3 | 6 | 8 | 4 | 1 | 1 | - |
| 2 | 3 | 7 | 30 | 4 | 1 | 1 | - |
| 2 | 4 | 3 | 25 | 11 | 5 | 1 | 1 |
| 2 | 4 | 4 | 49 | 11 | 5 | 1 | 1 |
| 2 | 4 | 5 | 28 | 11 | 5 | 1 | 1 |
| 2 | 4 | 6 | 65 | 13 | 5 | 1 | 1 |
| 2 | 5 | 3 | 55 | 35 | 16 | 6 | 1 |
| 2 | 5 | 4 | 91 | 43 | 16 | 6 | 1 |
| 2 | 5 | 5 | 242 | 34 | 16 | 6 | 1 |
| 3 | 3 | 3 | 23 | 5 | 1 | 1 | - |
| 3 | 3 | 4 | 28 | 5 | 1 | 1 | - |
| 3 | 3 | 5 | 28 | 6 | 1 | 1 | - |
| 3 | 3 | 6 | 84 | 7 | 1 | 1 | - |
| 3 | 3 | 7 | 177 | 7 | 1 | 1 | - |
| 3 | 4 | 3 | 37 | 26 | 11 | 4 | 1 |
| 3 | 4 | 4 | 71 | 18 | 11 | 4 | 1 |
| 3 | 4 | 5 | 102 | 43 | 11 | 4 | 1 |
| 3 | 4 | 6 | 213 | 89 | 10 | 3 | 1 |
| 3 | 5 | 3 | 85 | 69 | 26 | 16 | 5 |
| 3 | 5 | 4 | 87 | 80 | 30 | 16 | 5 |
| 3 | 5 | 5 | 181 | 136 | 27 | 17 | 5 |

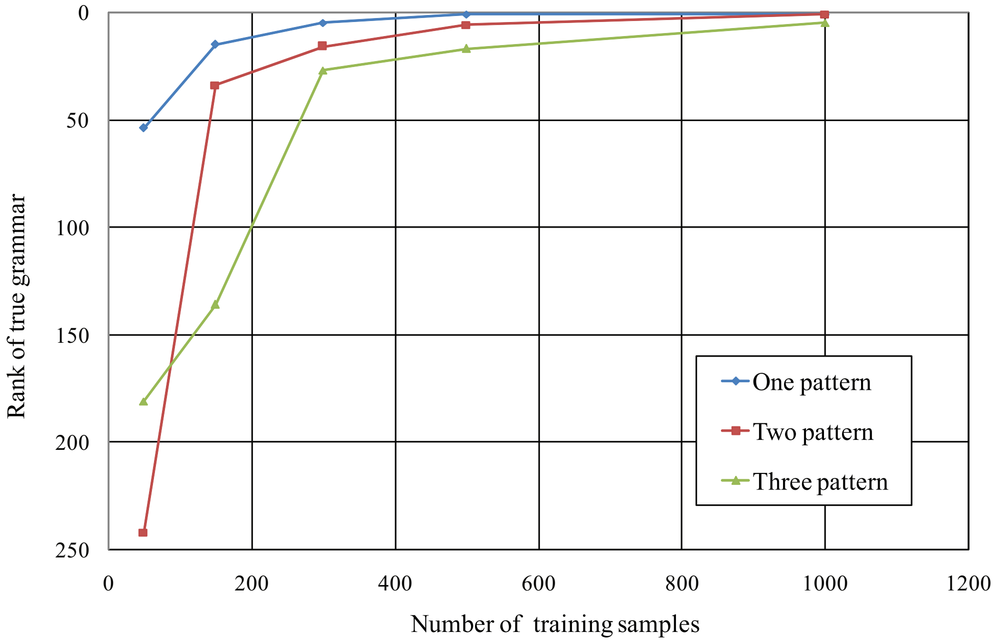


Figure 4.5. Rank of the true grammar converging to the top position for a grammar with five noise symbols and five noise symbols.

### 4.5.2 Synthetic system noise

Despite the fact that the method proposed thus far has been designed to address inherent insertion noise, it has been shown in preliminary experiments that the proposed method is also able to deal with system noise. More specifically, the results show that the proposed method is able to cope with random insertion, deletion and substitution errors. Insertion caused by system noise introduces the possibility of a non-noise symbol to appear randomly in the input sequence. The deletion of a non-noise symbol creates sequences with incomplete patterns. Substitution is a combination of a deletion and an insertion, where an important non-noise symbols is removed and replaced by either a noise symbols or another non-noise symbol.

One of the grammars used to produce the training samples is given in Figure 4.6. In addition to the insertion (INS) rules which represent inherent insertion noise, a substitution (SUB) rule was added to randomly insert a symbol in the place of a non-noise symbol. The parameters of the substitution rules have been distributed in such a way that non-noise symbols are inserted as noise 10% of the time. This is reasonable if it is assumed that key non-noise symbol detectors have high reliability. Using the artificial grammar, the training data was randomly generated for various values of $d$.

| | | | | | |
|---|---|---|---|---|---|
| S | → | EN | ACT | EX | [0.5] |
| S | → | ACT | EX | EN | [0.5] |
| EN | → | A | | | [0.45] |
| EN | → | A | INS | | [0.45] |
| EN | → | SUB | | | [0.10] |
| EX | → | B | | | [0.45] |
| EX | → | B | INS | | [0.45] |
| EX | → | SUB | | | [0.10] |
| ACT | → | C | | | [0.45] |
| ACT | → | C | INS | | [0.45] |
| ACT | → | SUB | | | [0.10] |
| A | → | a | | | [1.0] |
| B | → | b | | | [1.0] |
| C | → | c | | | [1.0] |
| INS | → | nd | | | [0.25] |
| INS | → | ne | | | [0.25] |
| INS | → | nf | | | [0.25] |
| INS | → | INS | INS | | [0.25] |
| SUB | → | nd | | | [0.30] |
| SUB | → | ne | | | [0.30] |
| SUB | → | nf | | | [0.30] |
| SUB | → | A | | | [0.0333] |
| SUB | → | B | | | [0.0333] |
| SUB | → | C | | | [0.0334] |

Figure 4.6. An example of a two pattern synthetic grammar with three non-noise symbols and system noise.

Table 4.3. Results with synthetic data (inherent insertion and system noise).

| | | | $d = 50$ | $d = 150$ | $d = 300$ | $d = 500$ | $d = 1000$ |
|---|---|---|---|---|---|---|---|
| Type | Non-noise | Noise | Rank of the true grammar | | | | |
| 1 | 3 | 3 | 12 | 3 | 1 | 1 | 1 |
| 2 | 3 | 3 | 15 | 7 | 4 | 2 | 1 |
| 3 | 3 | 3 | 23 | 17 | 7 | 4 | 1 |

Table 4.3 shows that the new modes of noise introduced by system noise increased the complexity of the task, which resulted in a need for more training samples to identify the true grammar. The proposed method was able to recover the correct non-noise symbols despite the increase of noise types because partial patterns could still be described by the CFG while incurring only a minimal increase in grammar size. As a result, the description length of the grammar and the data likelihood of the true grammar attained smaller values relative to those of other hypothesis grammars. These results show that as long as there is more order among the non-noise symbols compared to the noise symbols, an optimal solution can be identified. Consequently, if the structure between the non-noise symbols is corrupted to a degree, such that the randomness of the non-noise symbols becomes similar to the randomness of the noise symbols, the proposed method will only be able to identify a grammar using a subset of the correct set of non-noise symbols as the optimal solution.

### 4.5.3 Time complexity

Let $C$ be the maximum number of symbols in a single sequence and let $B$ be the number of training samples (sequences). The COMPRESSIVE algorithm has a theoretical time complexity of $O((BC)^2)$ because it makes multiple passes over the input string. However, in practice it is very fast compared to the calculation of the data likelihood when the speed-up techniques introduced in the original work [NMW00] are used. The linear time Sequitur algorithm[NMW97] could also be implemented for additional time savings.

The computation of the beta probabilities in the worst case is $O(PC^DK^D)$, where $P$ is the number of induced productions, $K$ is the maximum number of abstraction levels (for the proposed method $K = 2$) and $D$ is the maximum production length. The beta probabilities must be computed for each sequence, which means the time complexity for computing the data likelihood is $O(BPC^D2^D)$. Furthermore, since the proposed method evaluates every combination of terminal symbols, the total time complexity is $O(2^A(BPC^D2^D))$, where $A$ is the number of

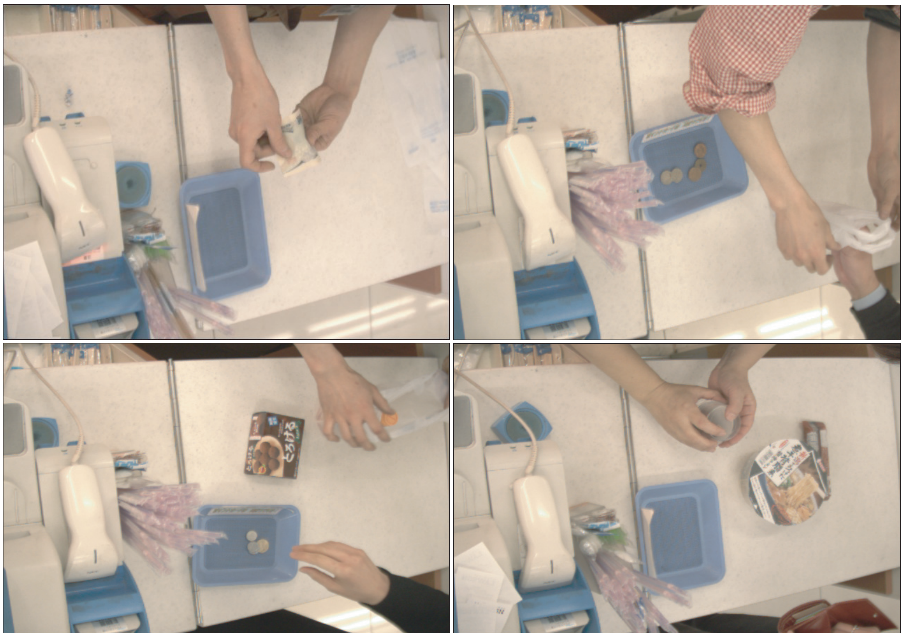


Figure 4.7. Overhead view of the CCD camera mounted above the counter.

terminal symbols. When the hidden grammar is complex, the calculation of the data likelihood dominates the computation time because the average number of symbols in a sequence $C$, the number of terminal symbols $A$ and the maximum production length $D$ become large. The Stolcke-Earley parser [Sto94] could be implemented as an alternative algorithm to speed up the calculation of the data likelihood.

## 4.6 Experiments with real data

A surveillance system in a local convenience store was setup to test the proposed method on real data. The system consisted of a single overhead CCD camera (Figure 4.7) that captured the hand movements of the employee and the customer. In the experiment a total of more than 9700 frames were recorded and processed offline according to the proposed method. Since the main goal was to learn the high-level grammar (not video segmentation) for a typical employee-customer transaction, the video was manually segmented for each new customer. While the issue of segmentation is not addressed in this paper, finding the beginning and ends of an activity will be an important task to address in future works when using a syntactic approach to learning.

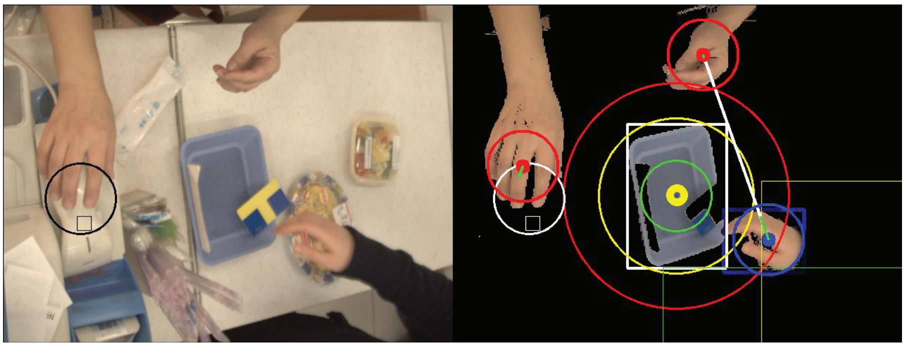


Figure 4.8. A frame from the image processing module showing the detection of hands and tray.

### 4.6.1 Extracting primitive action symbols

For this experiment primitive actions symbols were detected using simple image processing using application-specific domain knowledge for simplicity. However, a unsupervised technique such as the method proposed in chapter 3 could also be implemented. Skin color was detected in the HSV (hue, saturation, variance) space by merging a thresholded binary image from each channel. Similarly, the blue tray was detected using different thresholds in the HSV color space. The removal of the scanner and the receipt was detected by monitoring pixel changes over a small spatio-temporal window over the target region. Similarly, the addition and removal of money into the tray was detected by monitoring a spatio-temporal window over the center of the tray. An example of the results of the image processing module is shown in Figure 4.8. For this experiment a total of ten different types of primitive action symbols were extracted. An explanation of the terminals is given in Table 4.4. Again, a simple rule-based image processing system was implemented to extract the primitive action symbols in a top-down fashion. However, the proposed method will also work with any low-level image processing system that produces a string of primitive actions symbols.

A total of 369 symbols were automatically extracted from the convenience store surveillance video. The longest symbol sequence was eleven symbols long and the shortest sequence was three symbols long. Each sequence was concatenated into one long symbol string as the input to the propose algorithm. The size of the training data was $d = 55$ strings.

After acquiring the training data, each hypothesis was evaluated for every possible subset of primitive symbols as outlined in section 4.4.1. Since there were ten different terminals symbols, the system evaluated 1024 possible grammars. While the proposed method has the advantage of a complete search over the entire solution space, evaluating every possible com-

Table 4.4. Definition of the terminal symbols.

| NO. | TERMINAL SYMBOL | DESCRIPTION |
|---|---|---|
| 1 | CUS_AddedMoney | Money found in tray after customer comes in contact with the tray |
| 2 | CUS_MovedTray | Customer moves tray |
| 3 | CUS_RemovedMoney | Customer removes money from tray |
| 4 | EMP_HandReturns | Employee hand returns after long absence |
| 5 | EMP_Interaction | Employee interacts with customer |
| 6 | EMP_MovedTray | Employee moves the tray |
| 7 | EMP_RemovedMoney | Employee moves money from tray |
| 8 | EMP_ReturnedScanner | Employee returns scanner |
| 9 | EMP_TookReceipt | Employee takes the receipt from the register |
| 10 | EMP_TookScanner | Employee picks up scanner |

bination leads to a combinatorial explosion as the number of terminal symbols increase. While a brute force approach was adopted for this method, which resulted in the evaluation of every combination, results suggest that it may be possible to optimize the search by first evaluating grammars that use many non-noise symbols and limit subsequent evaluations to symbol subsets that are contained only in the top scoring set(s). This will be a topic for future work.

### 4.6.2 Initial results

The MDL identifies a single optimal grammar but from a practical perspective it is useful to present a list of the top hypothetical grammars. The top scoring hypothetical grammar for each class of grammars using the same number of non-noise symbols can be ranked as a list. While a certain user may be satisfied by a grammar that identifies two or three non-noise symbols, another user might desire a more descriptive grammar using five or more non-noise symbols despite the cost of a more complex grammar. Providing such a list would allow the user to choose the preferred grammar from a list of high scoring hypothetical grammars. A list of the top ranking grammars for each class of grammars using the same number of non-noise symbols $x$ is given in Table 4.5.

Table 4.5. Top hypothesis grammars - Optimal grammar marked in **bold**.

| $x$ | Non-noise Symbols | $DL(G)$ | $DL(W \mid G)$ | Total |
|---|---|---|---|---|
| 0 | | 140.11 | 1319.31 | 1459.42 |
| **1** | **EMP_TookScanner** | **221.41** | **1194.29** | **1415.70** |
| 2 | CUS_RemovedMoney<br>EMP_TookScanner | 245.34 | 1191.28 | 1436.62 |
| 3 | CUS_MovedTray<br>CUS_RemovedMoney<br>EMP_TookScanner | 294.19 | 1187.16 | 1481.35 |
| 4 | CUS_MovedTray<br>CUS_RemovedMoney<br>EMP_TookReceipt<br>EMP_TookScanner | 493.40 | 1054.20 | 1547.60 |
| 5 | CUS_MovedTray<br>CUS_RemovedMoney<br>EMP_MovedTray<br>EMP_TookReceipt<br>EMP_TookScanner | 658.55 | 1011.17 | 1669.72 |
| 6 | CUS_MovedTray<br>CUS_RemovedMoney<br>EMP_MovedTray<br>EMP_ReturnedScanner<br>EMP_TookReceipt<br>EMP_TookScanner | 1100.30 | 818.56 | 1918.86 |
| 7 | CUS_MovedTray<br>CUS_RemovedMoney<br>EMP_HandReturns<br>EMP_MovedTray<br>EMP_ReturnedScanner<br>EMP_TookReceipt<br>EMP_TookScanner | 1557.83 | 713.17 | 2271.00 |
| 8 | CUS_AddedMoney<br>CUS_MovedTray<br>CUS_RemovedMoney<br>EMP_HandReturns<br>EMP_MovedTray<br>EMP_RemovedMoney<br>EMP_ReturnedScanner<br>EMP_TookScanner | 2040.80 | 545.23 | 2586.03 |

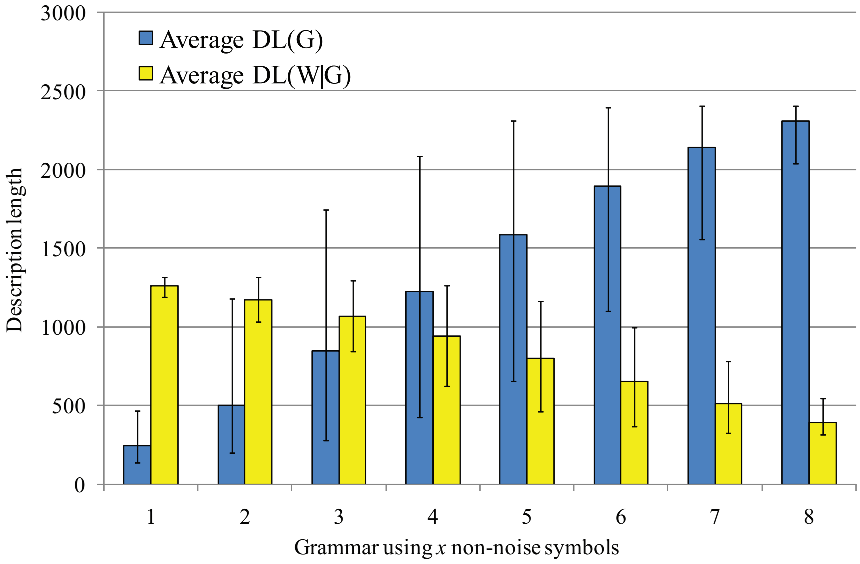


Figure 4.9. Imbalance between description lengths - Range of the grammar description length is consistently larger than the range of the likelihood description length (error bars shows range of description length).

It is expected that a minimum point will exist for a hypothetical grammar that uses actions such as *EMP_TookScanner* and *EMP_ReturnScanner* that are known to consistently occur during standard transaction sequences. However, it is also know from experiments with synthetic data that a sample size of 55 is likely to produce sub-optimal results when there are more than two true non-noise symbols. In fact, in these initial results a global minimum is found for a grammar that uses only one non-noise symbol *EMP_TookScanner*. As suspected, the proposed method has given more weight to the simplicity of the grammar and less weight to its descriptive ability. Furthermore, it is observed that high scores are given to grammars using symbols that occur less frequently in the data. For example, the top scoring grammar using $x = 2$ non-noise symbols includes the terminal *CUS_MovedTray*, an action that was only detected twice in the entire training set. Intuition requires that general rules should not be generated from symbols of rare occurrence.

### 4.6.3 Balancing description lengths

Figure 4.9 compares the range (difference between the minimum value and maximum value) of the description lengths of the grammar and the data likelihood produced by the real data. It is observed from this figure that the range of the description length of the grammar is consistently greater than the range of the description length of the data likelihood. This indicates that the size of the grammar always has a greater influence on the total description length.

As in this specific case, it may not always be possible to gather enough samples to apply an MDL criterion directly to the training data. In order to compensate for the imbalance between the description length of the grammar and the description length of the data likelihood, it is helpful to introduce a weighting scheme into the MDL criterion.

It is possible to heuristically balance the effect of the description length of the grammar and the description length of the data likelihood by introducing a factor $\gamma_x$ into the MDL equation, where $\gamma_x$ has been interpreted to be the *prior weight* of the grammar or the inverse of the *data multiplier*[Sto94], or the *representativeness* of the data[QR89].

$$\gamma_x DL(\mathbf{G}_x) + DL(\mathbf{W}|\mathbf{G}_x). \tag{4.12}$$

The value for $\gamma_x$ is defined as the ratio between the range of the description length of the likelihood and the description of the grammar, where $x$ is the number of non-noise symbols used in the grammar. This global prior weighting has the effect of minimizing the contribution of the description length of the grammar and boosts the contribution of the description length of the data likelihood, giving lower priority to grammars that use rare symbols.

$$\gamma_x = \frac{DL_{max}(\mathbf{W}|\mathbf{G}_x) - DL_{min}(\mathbf{W}|\mathbf{G}_x)}{DL_{max}(\mathbf{G}_x) - DL_{min}(\mathbf{G}_x)} \tag{4.13}$$

The top ranking grammar for each class, after compensating for the small size of the training data using the balanced total description length is given in Table 4.6. Figure 4.10 shows that the grammar with the smallest overall description length is the hypothesis grammar that uses the three symbols *EMP_ReturnedScanner*, *EMP_TookReceipt* and *EMP_TookScanner*. The grammar learned with these three symbols is given in Figure 4.12.

### 4.6.4 Recovered basic structure

The hierarchical structure (parse tree) learned for a common activity $H$ is given in Figure 4.11. The parse tree depicts the activity of an employee who first *begins* (node $E$) the transaction by taking the scanner to enter the bar codes of items for purchase into the register. Then, the employee *ends* (node $D$) the transaction, by returning the scanner to its holder and issuing the receipt.

Notice that the symbols identified as non-noise symbols are all predictable actions performed by the employee. Since the employee has been trained to follow a certain protocol, his

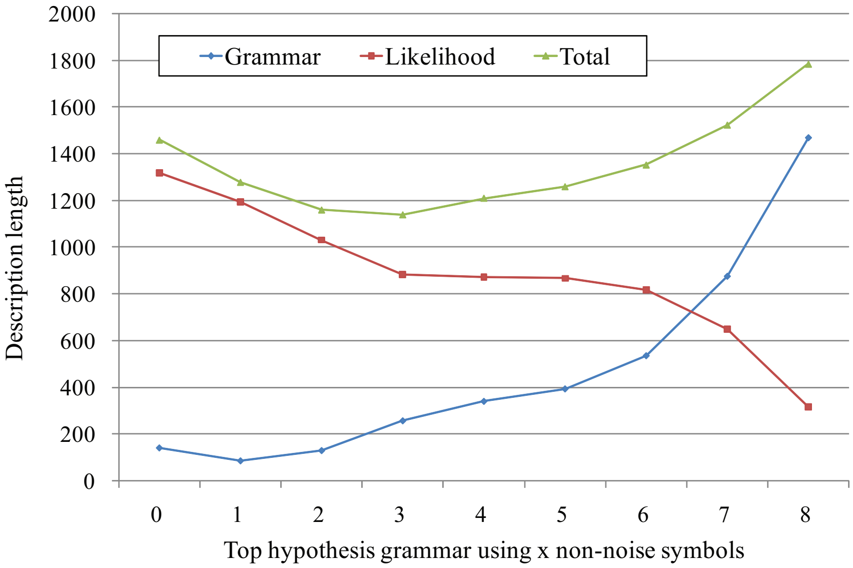


Figure 4.10. Description lengths for top hypothesis grammars - Global minimum at $x = 3$.

actions are predictable and ordered. In contrast, the actions of the customers show less regularity. Therefore, it makes sense that the MDL criterion identifies a grammar dependent only on the predicable actions of the employee as the optimal grammar.

Table 4.6. Top hypothesis grammars (Balanced) - Optimal grammar marked in **bold**.

| $x$ | Non-noise Symbols | $\gamma_x$ | $\gamma_x$DL(G)+DL(W\|G) |
|---|---|---|---|
| 1 | EMP_TookScanner | 0.383 | 1279.0016 |
| 2 | EMP_ReturnedScanner<br>EMP_TookScanner | 0.2897 | 1160.7076 |
| **3** | **EMP_ReturnedScanner**<br>**EMP_TookReceipt**<br>**EMP_TookScanner** | **0.3096** | **1140.0563** |
| 4 | CUS_MovedTray<br>EMP_ReturnedScanner<br>EMP_TookReceipt<br>EMP_TookScanner | 0.3847 | 1211.0414 |
| 5 | CUS_MovedTray<br>CUS_RemovedMoney<br>EMP_ReturnedScanner<br>EMP_TookReceipt<br>EMP_TookScanner | 0.4246 | 1260.2536 |
| 6 | CUS_MovedTray<br>CUS_RemovedMoney<br>EMP_MovedTray<br>EMP_ReturnedScanner<br>EMP_TookReceipt<br>EMP_TookScanner | 0.4859 | 1353.1436 |
| 7 | CUS_AddedMoney<br>CUS_MovedTray<br>CUS_RemovedMoney<br>EMP_MovedTray<br>EMP_RemovedMoney<br>EMP_ReturnedScanner<br>EMP_TookScanner | 0.5335 | 1523.8244 |
| 8 | CUS_MovedTray<br>EMP_HandReturns<br>EMP_Interaction<br>EMP_MovedTray<br>EMP_RemovedMoney<br>EMP_ReturnedScanner<br>EMP_TookReceipt<br>EMP_TookScanner | 0.6228 | 1784.4875 |

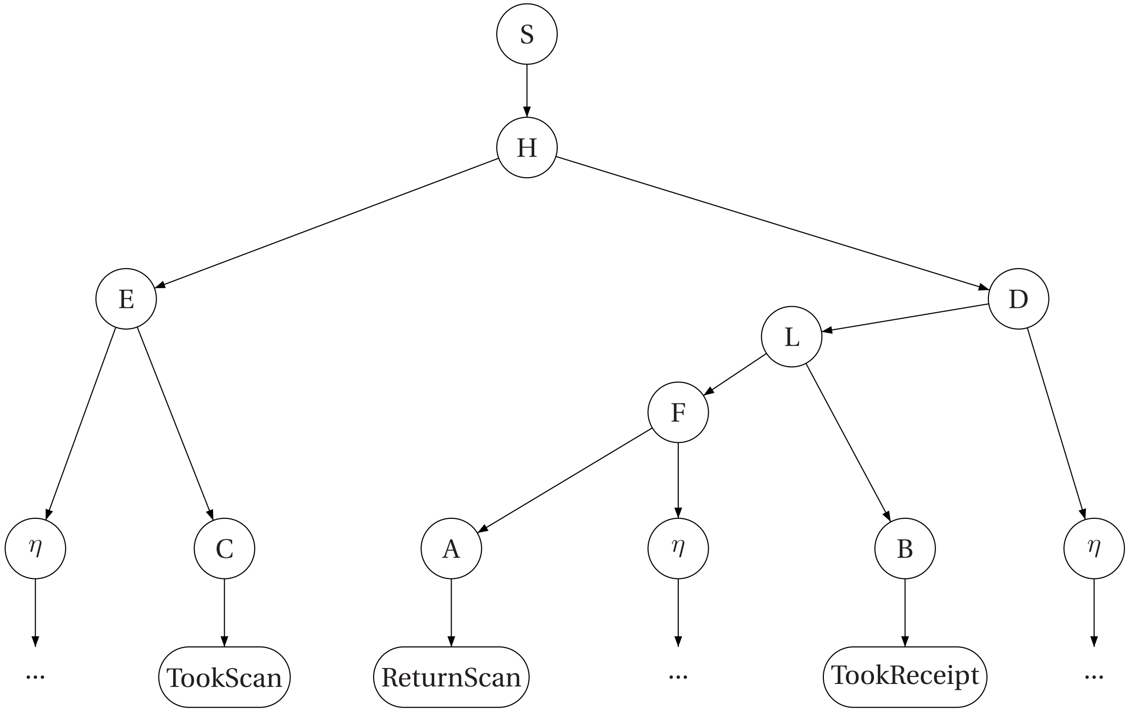


Figure 4.11. Parse tree of a common structure found in the training data.

| | | | |
|---|---|---|---|
| S → D | (0.02) | D → L $\eta$ | (1.000) |
| S → H | (0.16) | E → $\eta$ C | (1.000) |
| S → G | (0.18) | F → A $\eta$ | (1.000) |
| S → N $\eta$ | (0.04) | G → C D | (1.000) |
| S → J | (0.13) | H → E D | (1.000) |
| S → Q | (0.05) | I → $\eta$ B $\eta$ | (1.000) |
| S → $\eta$ | (0.02) | J → C F | (1.000) |
| S → N | (0.02) | K → $\eta$ D | (1.000) |
| S → R | (0.05) | L → F B | (1.000) |
| S → J B | (0.02) | M → C $\eta$ | (1.000) |
| S → M L | (0.04) | N → E A B | (1.000) |
| S → M A H | (0.02) | O → E $\eta$ | (1.000) |
| S → C K | (0.04) | P → E I | (1.000) |
| S → C A M F | (0.02) | Q → E K | (1.000) |
| S → O F | (0.02) | R → E L | (1.000) |
| S → M | (0.02) | | |
| S → O L | (0.02) | $\eta$ → $\eta$ $\eta$ | (0.309) |
| S → O | (0.02) | $\eta$ → CUS_AddMoney | (0.153) |
| S → P | (0.05) | $\eta$ → CUS_MovedTray | (0.006) |
| S → I | (0.04) | $\eta$ → CUS_RemMoney | (0.003) |
| S → K | (0.04) | $\eta$ → EMP_HandReturn | (0.080) |
| A → EMP_ReturnedScanner | (1.00) | $\eta$ → EMP_Interaction | (0.275) |
| B → EMP_TookReceipt | (1.00) | $\eta$ → EMP_MovedTray | (0.028) |
| C → EMP_TookScanner | (1.00) | $\eta$ → EMP_RemMoney | (0.147) |

Figure 4.12. Recovered optimal grammar using three non-noise symbols.

## 4.7 Conclusion

This chapter has introduced a new method for acquiring the basic structure of an activity from a noisy symbol string produced by video. The proposed method placed presuppositions on each combination of terminal symbols and tested that hypothesis using an MDL criterion. The MDL equation measured the balance between a compactness and expressiveness of a grammar to encode the data, and provided a means of quantifying the quality of each presupposition. Experiments with artificial data showed the proposed method is able to correctly identify an optimal grammar when the size of the training data was sufficient. Results also exemplified an inherent bias toward smaller grammars when the size of the training data was insufficient. Based on insights from experimental results, a heuristic method of balancing the MDL equation using a data multiplier $\gamma_x$ which minimized the bias toward smaller grammars was proposed. This new balanced equation resulted in the discovery of a compact grammar that captured the basic structure of activities found in the training data.

While creating a symbol string from video has allowed the proposed method to use preexisting syntactic analysis techniques to learn the optimal grammar, the method is far from utilizing the full range of information contained in video. For example, a more intuitive grammar could be attained by analyzing *temporal* information between two actions (e.g. one action always occurs 30 seconds after another) or by comparing the relative *location* (e.g. two actions occur in the same location) or by observing that two actions are always connected to a *common object*. Future work will use temporal, spatial and contextual information in the grammar learning process.

Furthermore, when one considers the applications of human activity learning techniques, most use cases will have some general *a priori* information about the activities to be learned. For example in the experiments, it is already known that an employee-customer interaction will begin with the placement of an item on the counter and end with a payment for the item. In future works, this type of rough *a priori* grammar will be used to guide the learning process, to discover more subtle and complex grammars found in human activities.

# Chapter 5

# Recognizing structured human activities

The two previous chapters have introduced a method for identifying the primitive actions in a video sequence and a method for learning the temporal structure between those primitive actions. In this chapter, a method for recognizing a string of primitive actions as an activity is introduced. The proposed method uses a weighted set of Bayesian networks, created from an underlying activity grammar, to detect activities occurring in the action symbol string.

## 5.1 Introduction

The automated real-time understanding of human activities from a video sequence is a topic of growing interest in recent times. In the field of video surveillance, detecting suspicious activity in real-time would mean stopping crimes while they are happening or even before they happen. In application to human-computer interfaces, computers could adjust according to the activity context of the user. An intelligent system that recognizes high-level human activities offers a wide range of applications to aid people in everyday activities.

To implement a system that recognizes human activities, the task can interpreted to be two-fold. The first task is to formulate a computational framework for characterizing human activity which is grounded in finding of experimental psychology. The second task is to create a computational technique for analyzing those activities.

First, the characteristics of human activities can be learned from perceptual psychology [ZT01]. According to recent findings, activities are hierarchical. That is, they are taxonomically organized, existing at various levels of abstraction. For example, walking and running are a *type of* moving. Activities are also *partonomical*, meaning that primitive actions are temporally ordered (sequential). For example, the activity of *leaving an object* in a room might consist of a sequence of primitive actions: (1) enter the room, (2) put down the object and (3) exit the room. Activities can also be temporally overlapped. For example, the transition of a person *walking through* a room might overlap with the activity of the person *departing* from the room. From the perspective of the system, it is difficult to identify the exact time at which the activity *walking through* has ceased and when the activity *departing* has started. Thus there is an inherent ambiguity at transitions between human activities which should be represented by a cognitive system.

To address the latter half of the problem, namely the computational recognition of human activities from a sequence of video images, an efficient algorithm for incorporating the partonomic characteristic of activity needs to be formalized. More specifically, the recognition system must encode hierarchical information, capture temporally constrained activities and accurately represent temporally overlapped activities.

The contribution of the proposed method described in this chapter lies in the novel application of deleted interpolation (DI) – a smoothing technique used in natural language processing – for recognizing temporally overlapped activities. This chapter addresses the issue of hierarchical structure by implementing a stochastic context-free grammar (SCFG). The SCFG is converted into a Bayesian network (BN) to create a hierarchical Bayesian network (HBN) which enables the system to execute more complex probability queries across the grammar. Then the HBN is applied to a string of observed primitive action symbols via DI to recognize various activities, especially those that are overlapped.

It is noted here that the issue of extracting symbols from a video sequence has already been described in chapter 3. Here it is assume that a set of reliable low-level observations (e.g. appearance and movement attributes) are available, an as such this chapter focuses on building up a scheme for activity recognition. Furthermore, the method of grammar creation has already been covered in chapter 4 and is therefore not the focus of this chapter. The activity grammar is assumed to be given.

The majority of models that have been proposed for activity analysis are models that represent an activity as a sequential transition between a set of finite states (i.e. NDA [WM98], FSA [AS01], HMM [YOI92], CHMM [ORP00], VLHMM [GJH01], LHMM [OHG02], DMLHMM [GX03], ODHMM [LC03], SHSMM [DBPV05]). However, due to the fact that most simple activities do not have complex hierarchical structure, these models have not explicitly incorporated the concept of hierarchy into the model topology.

On the other hand, there have been a few works that have proposed hierarchical models to recognize structured activities. Contributions from computer vision started with Brand [Bra96], when he utilized a deterministic action grammar to interpret a video sequence of a person opening a computer housing unit. Multiple parses over a stream of outputs from the low-level event detector were ranked and stored, giving priority to the highest ranking parse. Ivanov [IB00] first used a SCFG for action recognition using the Earley-Stolcke parser to analyze a video sequence of cars and pedestrians in a parking lot. Moore [ME02] also used a SCFG to recognize actions in a video sequence of people playing Blackjack. They extend the work of Ivanov by adding error correction, recovery schema and role analysis. Minnen [MES03] built on the modifications made by Moore by adding event parameters, state checks and internal states. They applied the SCFG to recognize and make predictions about actions seen in a video sequence of a person performing the Towers of Hanoi task. From a background in plan recognition, Bui [BVW01] used a hierarchy of abstract policies using Abstract Hidden Markov Models (AHMM) implementing a probabilistic state-dependent grammar to recognize activities. The system recognizes people going to the library and using the printer across multiple rooms. AHMMs closely resemble the Hierarchical Hidden Markov Models (HHMM) [FST98] but with an addition of an extra state node. Nguyen [NBVW03] used an abstract Hidden Memory Markov Model (AHMEM), a modified version of the AHMM, for the same scenario as Bui.

The aforementioned works used domains with high-level activities delineated by clear starting points and clear ending points, where the observed low-level action primitives are assumed to describe a series of temporally constrained activities (with the exception of Ivanov [IB00]). However, in this chapter the focus is placed on a subset of human activities that have the possibility of being temporally overlapped. It is shown that these types of activities can be recognized efficiently using the framework proposed in this chapter.

## 5.2 Modeling human action

Most human activities are ordered hierarchically much like sentences in a natural language. Thus an understanding of hierarchy about human activities should be leveraged to reason about those activities , just like one might guess at the meaning of a word from its context. It is asserted here that the SCFG and the BN lay the proper groundwork for hierarchical analysis of human activity recognition using a vision system.

The justification for using a SCFG to model human activity is based on the idea that it models hierarchical structure that closely resembles the inherent hierarchy in human activity. Just as series of words can be represented at a higher level of abstraction, a series of primitive actions can also be represented at a higher level of abstraction. By recognizing the close analogy between a string of words and a series of actions, it is reasoned that SCFGs are well suited for representing grammatical structure.

A SCFG is also able to describe an activity at any level in the hierarchy in the same way humans are known to perceive activities at different abstractions levels within a hierarchical structure. In contrast, standard sequential models like finite state machines, n-grams, Markov chains and hidden Markov models, do not explicitly model hierarchical structure.

Despite the expressive power of the SCFG, they were created to characterize formal language and thus in general, syntactic parsers are not well-suited for handling noisy data. Bayesian networks have the robustness needed to deal with faulty sensor data, especially when dealing with human actions. In contrast to standard parsing algorithms, the merit of using an BN is found in the wide range of queries that can be executed over the network [PW98]. In addition, BNs can deal with negative evidence, partial observations (likelihood evidence) and even missing evidence, making it a favorable framework for vision applications that deal with uncertain observations.

## 5.3 Recognition system overview

The proposed recognition system consists of three major parts (Figure 5.1). The first is the action grammar (a SCFG) that describes the hierarchical structure of all the activities to be recognized. Second is the hierarchical Bayesian network that is generated from the action grammar. Third is the final module that takes a stream of input symbols (level 1 action symbols) and uses deleted interpolation to determine the current probability distribution across each possible output symbol (level 2 action symbol).

The details of the proposed system are described here based on the use of the CAVIAR data set [CAV] to provide concrete explanation of each aspect of the algorithm. The CAVIAR data is a collection of video sequences of people in a lobby environment. The ground truth for each agent in each frame is labeled in XML with information about position, appearance, movement, situation, roles, and context. For practical reasons, the ground truth is used to produce a sequence of primitive action symbols as the low-level input into the proposed system for the experiments.

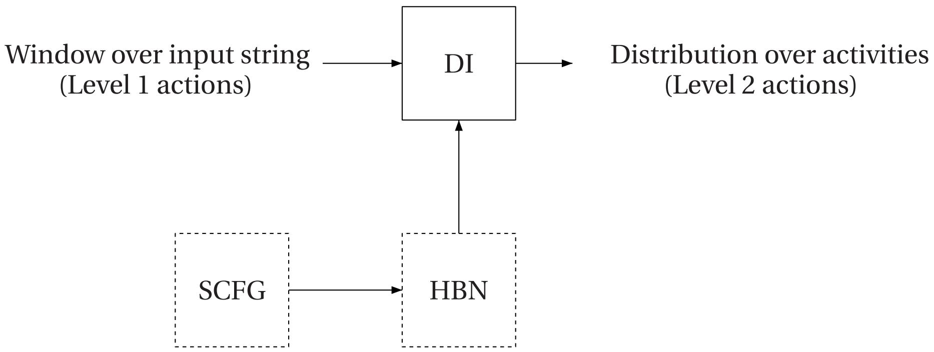


Figure 5.1. System flow chart. Dashed lines indicate off-line components and solid lines indicate online components. Level 1 actions symbols and the HBN are merged via the deleted interpolation step to produce level 2 actions.

### 5.3.1 Action grammar

The set of terminals (level 1 action symbols) is defined as **T**= {*en, ne, ex, mp, wa, in, br, pu, pd* } (definitions given in Table 5.2). The level 1 action symbols were generated directly from the CAVIAR XML ground truth data using logical relationships between the appearance, movement and position information for each frame (Table 5.1). The set of action symbols (called level 2 actions) $\mathbf{A} = \{BI, BR, TK, LB, PT, AR, DP\}$, along with a set of intermediate action symbols $\mathbf{I} = \{AI, MV, MT, MF\}$ were created manually to be the set of high-level actions to be used by the system (Table 5.3). Level 2 actions are a special subset of nonterminal symbols in the level 2 grammar because they are direct abstraction productions of $S$ (start symbol), i.e. they are directly caused by $S$. The set of nonterminals **N** is defined as $\mathbf{N} = \mathbf{I} \cup \mathbf{A}$. The set of production rules $\Sigma$ and their corresponding probabilities are given in Table 5.4. Again, it is noted here that since grammar creation is not the primary focus of this chapter, the grammar (including the rule probabilities) are manually defined. It is clear from chapter 4 that the grammar can also be learning from a sufficiently sized dataset.

Table 5.1. Grammar for producing level 1 symbols.

| Level 1 Actions | Appearance | Movement | Position |
|---|---|---|---|
| en | appear | - | - |
| ex | disappear | - | - |
| ne | visible | active/walking | near exit/entrance |
| br | visible | active/inactive | near a landmark |
| in | visible | inactive | - |
| mp | visible | active | - |
| wa | visible | walking | - |
| pu | referenced to object properties | | |
| pd | referenced to object properties | | |

Table 5.2. Definition of the level 1 actions (terminal symbols).

| Level 1 Actions | Meaning |
|---|---|
| en | enter : appears in the scene |
| ex | exit: disappears from the scene |
| ne | near exit/ entrance : moving near an exit / entrance |
| br | browse : standing near landmark |
| in | inactive: standing still |
| mp | move in place : standing but moving |
| wa | walk : moving within a certain velocity range |
| pd | put down : release object |
| pu | pick up : contact with object |

Table 5.3. Definition of the level 2 actions and intermediate actions (nonterminal symbols).

| Level 2 Actions | Meaning |
|---|---|
| AR | Arriving : Arriving into the scene |
| BI | Being Idle : Spending extra time in the scene |
| BR | Browsing : Showing interest in an object in the scene |
| TK | Taking away : Taking an object away |
| LB | Leaving behind : Leaving an object behind |
| PT | Passing Through : Passing through the scene |
| DP | Departing : Leaving the scene |

| Intermediate Actions | Meaning |
|---|---|
| AI | Action in Place: Taking action while in place |
| MV | Moving : Moving with a minimum velocity |
| MT | Move to : Moving in place after walking |
| MF | Move from : Walking after moving in place |

Table 5.4. Level 2 action grammar.

| | | | |
|---|---|---|---|
| S → BI | 0.20 | BR → br | 0.20 |
| S → BR | 0.10 | BR → MV br | 0.20 |
| S → TK | 0.05 | BR → br mp | 0.30 |
| S → LB | 0.05 | BR → MV br mp | 0.30 |
| S → PT | 0.30 | | |
| S → AR | 0.15 | LB → pd | 0.50 |
| S → DP | 0.15 | LB → MV pd | 0.20 |
| | | LB → pd mp | 0.05 |
| BI → AI | 0.10 | LB → pd wa | 0.05 |
| BI → MV AI | 0.10 | LB → pd mp wa | 0.10 |
| BI → AI MV | 0.10 | LB → mp pd mp | 0.10 |
| BI → mp AI MV | 0.10 | | |
| BI → mp | 0.20 | DP → ex | 0.40 |
| BI → MF mp | 0.10 | DP → wa ne ex | 0.30 |
| BI → MF | 0.10 | DP → ne ex | 0.20 |
| BI → MV ne MV | 0.10 | DP → wa ne | 0.10 |
| BI → AI wa ne | 0.10 | | |
| | | MV → MF | 0.20 |
| TK → pu | 0.50 | MV → MT | 0.20 |
| TK → MV pu | 0.20 | MV → wa | 0.30 |
| TK → pu mp | 0.20 | MV → mp | 0.30 |
| TK → pu wa | 0.10 | | |
| TK → MV pu MV | 0.10 | MF → mp wa | 1.00 |
| | | MT → wa mp | 1.00 |
| PT → en wa ex | 0.70 | | |
| PT → ne wa ne | 0.30 | AI → in | 0.60 |
| | | AI → br | 0.20 |
| AR → en | 0.50 | AI → pu | 0.10 |
| AR → en MV | 0.50 | AI → pd | 0.10 |

### 5.3.2 Hierarchical Bayesian network

A previously proposed method [PW98] is used to transform the action grammar (level 2 grammar) into a hierarchical Bayesian network (HBN). The term HBN is used here because information about hierarchy from the SCFG is embedded in the BN. However in a broader sense, all Bayesian networks (directed graphs) are hierarchical by definition.

As mentioned before, the SCFG is converted into a BN because it has the ability to deal with uncertainty. When the sensory input is uncertain, the BN can process a multinomial distribution across a discrete variable instead of a single value with a probability of one. In addition, the BN can also deal with missing evidence (a missed detection) by marginalizing over the values of the missed variable.

By converting the action grammar into a HBN, evidence nodes $\mathbf{E}$ contain terminal symbols, query nodes $\mathbf{Q}$ contain level 2 actions $\mathbf{A}$ and hidden nodes $\mathbf{H}$ contain production rules $\Sigma$ or intermediate action $\mathbf{I}$. Results of transforming the grammar in Table 5.4 into a HBN is depicted in Figure 5.2.

The probability density function (PDF) for level 2 actions[1] is denoted as $\mathbf{P}(\mathbf{A}|\mathbf{e})$ where $\mathbf{A} = \{A_1, A_2, \ldots, A_v\}$ is the set of all level 2 actions (states). The input vector $\mathbf{e} = [e_1, e_2, \ldots, e_l]$ is a string of evidence at the evidence nodes of the HBN where $l$ is the maximum length of the HBN. The probability of a specific level 2 action is defined as the sum of the probabilities from each of the query nodes,

$$\mathbf{P}(\mathbf{A}|\mathbf{e}) = \mathbf{P}(Q_1 = \mathbf{A}|\mathbf{e}) + \cdots + \mathbf{P}(Q_u = \mathbf{A}|\mathbf{e}). \tag{5.1}$$

When there are $v$ different level 2 actions, $\mathbf{P}(\mathbf{A}|\mathbf{e})$ represents a set of $v$ equations

$$\begin{aligned}
P(A_1|\mathbf{e}) &= P(Q_1 = A_1|\mathbf{e}) + \cdots + P(Q_u = A_1|\mathbf{e}), \\
P(A_2|\mathbf{e}) &= P(Q_1 = A_2|\mathbf{e}) + \cdots + P(Q_u = A_2|\mathbf{e}), \\
\ldots & \\
P(A_v|\mathbf{e}) &= P(Q_1 = A_v|\mathbf{e}) + \cdots + P(Q_u = A_v|\mathbf{e}).
\end{aligned} \tag{5.2}$$

The probabilities of the level 2 actions $\mathbf{A} = \{A_1, A_2, \ldots, A_v\}$ always sum to one when the evidence can be explained by the grammar because $\mathbf{A}$ is the set of all possible productions of $S$ (start symbol). Thus,

$$\sum_{i=1}^{v} P(A_i|\mathbf{e}) = 1. \tag{5.3}$$

[1] $\mathbf{P}$ will be used when dealing with probabilities of multi-valued discrete variables. It denotes a set of equations with one equation for each value of the variable.

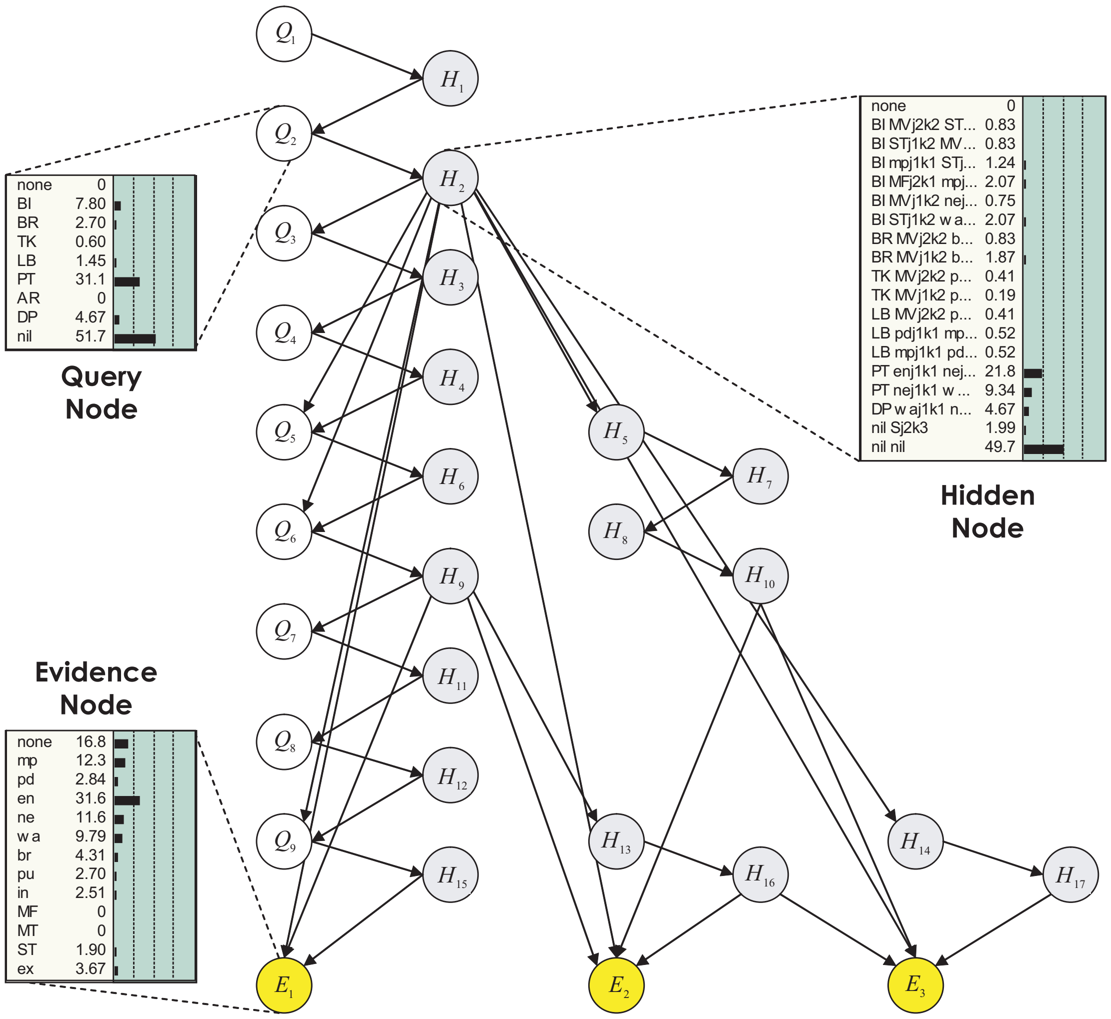


Figure 5.2. Hierarchical Bayesian Network (maximum length $l = 3$). The content of each node type is depicted by a bar chart.

**Computational cost**

The computational cost of calculating the beta probabilities is $O(Pn^m d^m)$ and the cost of building the Bayesian network is $O(Pn^{m+1}d^m T^m)$ (more details in the original paper [PW98]). $P$ is the number of rules induced by the grammar, $d$ is the maximum number of abstraction levels, $n$ is the maximum length of a sentential string, $m$ is the maximum production length and $T$ is the maximum number of entries of any conditional probability table in the network. Although the cost of building the network grows exponentially as the grammar grows in complexity the network only needs to be computed once offline. With respect to inference with the Bayesian network, exact inference becomes intractable as the network grows in size, which means that other well known approximation algorithms will need to be utilized for bigger networks.

### 5.3.3 Deleted interpolation

The concept of deleted interpolation (DI) involves combining two (or more) models of which one provides a more precise explanation of the observations but is not always reliable and the another which is more reliable but not as precise. A precise model requires that the input data be a close fit to the model and will reject anything that does not match. A reliable model exhibits greater tolerance in fitting the data and is more likely to find a match. Combining models allows one to fall back on the more reliable model when the more precise model fails to explain the observations. It is called *deleted* interpolation because the models which are being interpolated use a subset of the conditioning information of the most discriminating function [MS03].

In the proposed system it is assume that the analysis of a long sequence of evidence is more precise than that of a shorter length because a long sequence takes into consideration more information. However, when analysis over a long (more precise) input sequence fails one would like to fall back on analysis based on a shorter (more reliable) subsequence.

To implement this the current probability distribution S across level 2 actions, at each time instance, is calculated as a weighted sum of models,

$$\mathbf{S} = \sum_{i=1}^{l} \lambda_i \mathbf{P}(\mathbf{A}|\mathbf{O}_i), \tag{5.4}$$

where $\mathbf{O}_i$ is the string of full evidence when $i = 1$ and represents smaller subsets of the evidence sequence as the index $i$ increases. The weights are constrained by $\sum_{i=1}^{l} \lambda_i = 1$.

Representing the proposed system as a dynamic Bayesian network yields the network in Figure 5.3 where only the evidence is passed on across time slices. Memory nodes are added to

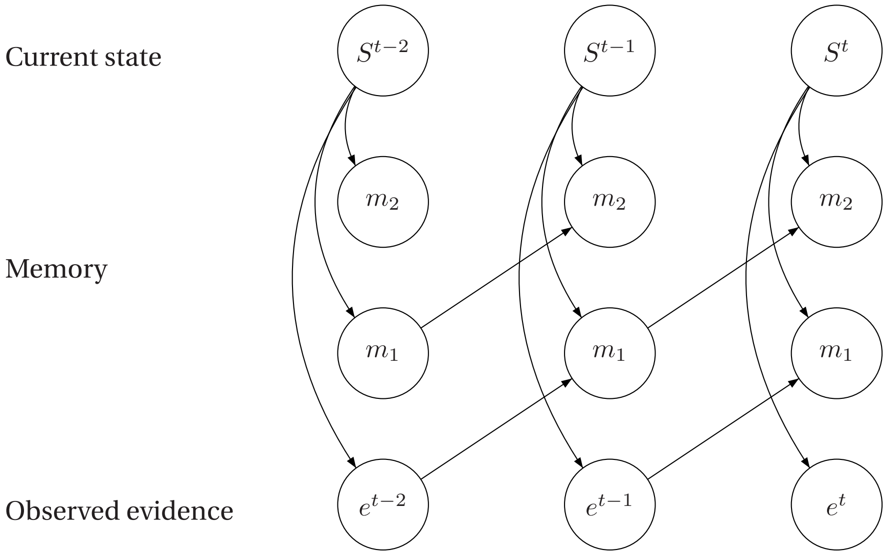


Figure 5.3. System depicted as a dynamic Bayesian network where the memory elements store past evidence.

the network to store past evidence and $l$ is the length of the analysis window. When $l = 3$, the current probability distribution of the level 2 actions over the temporal window is given by the equation

$$\mathbf{S} = \lambda_1 \mathbf{P}(\mathbf{A}|\mathbf{O}_1) + \lambda_2 \mathbf{P}(\mathbf{A}|\mathbf{O}_2) + \lambda_3 \mathbf{P}(\mathbf{A}|\mathbf{O}_3), \tag{5.5}$$

where[2]

$$\begin{aligned} \mathbf{O}_1 &= \{e_1^t, e_2^{t-1}, e_3^{t-2}\} && (5.6) \\ \mathbf{O}_2 &= \{e_1^t, e_2^{t-2}, e_3^{none}\} && (5.7) \\ \mathbf{O}_3 &= \{e_1^t, e_2^{none}, e_3^{none}\}. && (5.8) \end{aligned}$$

The first term $\mathbf{P}(\mathbf{A}|e_1^t, e_2^{t-1}, e_3^{t-2})$ is the activity probability distribution of the complete set of evidence and represents activities that have started at $t-2$. The second term $\mathbf{P}(\mathbf{A}|e_1^t, e_2^{t-2}, e_3^{none})$ is the activity probability distribution for a partial set of evidence and represents activities starting at $t-1$. Likewise, the last term $\mathbf{P}(\mathbf{A}|e_1^t, e_2^{none}, e_3^{none})$ is a probability distribution for activities starting at $t$. This is the mechanism that effectively allows the system to represent overlapped activities.

[2] $e^{none}$ is a terminal symbol that represents an end of the sequence.

Table 5.5. Logic equations for level 1 actions.

| Action symbol | Logic equation |
|---|---|
| enter | $\mathrm{en}_t = \mathrm{appear}_t$ |
| exit | $\mathrm{ex}_t = \mathrm{disappear}_t$ |
| near entrance | $\mathrm{ne}_t = \mathrm{visible}_t \wedge (\mathrm{active}_t \vee \mathrm{walking}_t) \wedge \mathrm{near\_doorway}_t$ |
| browse | $\mathrm{br}_t = \mathrm{visible}_t \wedge (\mathrm{active}_t \vee \mathrm{inactive}_t) \wedge \mathrm{near\_landmark}_t$ |
| inactive | $\mathrm{in}_t = \mathrm{visible}_t \wedge \mathrm{inactive}_t$ |
| move in place | $\mathrm{mp}_t = \mathrm{visible}_t \wedge \mathrm{active}$ |
| walk | $\mathrm{wa}_t = \mathrm{visible}_t \wedge \mathrm{walking}_t$ |
| pick up | $\mathrm{pu}_t = \mathrm{near\_object}(\mathrm{leaving\_object}_t \wedge \neg \mathrm{in}_t \wedge \mathrm{in}_{t-1})$ |
| put down | $\mathrm{pd}_t = \mathrm{near\_object}(\mathrm{leaving\_object}_t \wedge \mathrm{in}_t \wedge \neg \mathrm{in}_{t-1})$ |

## 5.4 Experiments

### 5.4.1 Extracting the action symbols

Since the ground truth for each agent in each frame is labeled in XML (information about position, appearance, movement, situation, roles, and context), the ground truth data is used directly as the low-level input into the system for practical reasons. Each video sequence was processed to create a sequence of level 1 action symbols by applying the logic equations given in Table 5.5 to the XML data. The extracted symbol stream for each sequence is given in Table 5.6. It is noted here that as presented in chapter 3, the actions symbols can also be produced by a probabilistic classifier.

Table 5.6. Action symbol sequences for each experimental video sequence

| | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Frame | 236 | 237 | 296 | 486 | 511 | | | | | | | | | | | | | | |
| Walk 1 | en | ne | wa | ne | ex | | | | | | | | | | | | | | |
| Frame | 212 | 213 | 257 | 319 | 378 | 459 | 460 | 470 | 474 | 506 | 699 | 772 | 1096 | 1097 | 1148 | 1352 | 1378 | | |
| Walk 2 | en | ne | wa | ne | ex | en | ne | en | ne | wa | ne | ex | en | ne | wa | ne | ex | | |
| Frame | 334 | 335 | 421 | 623 | 756 | 812 | 848 | 922 | 983 | | | | | | | | | | |
| Browse 1 | en | ne | wa | br | wa | ne | wa | br | mp | | | | | | | | | | |
| Frame | 182 | 183 | 279 | 599 | 620 | 643 | 691 | | | | | | | | | | | | |
| Browse 2 | en | ne | wa | br | wa | ne | ex | | | | | | | | | | | | |
| Frame | 238 | 239 | 273 | 465 | 466 | 535 | 639 | 747 | 748 | 830 | 901 | 902 | 1059 | 1114 | | | | | |
| Leave 1 | en | ne | wa | pd | wa | ne | ex | en | ne | wa | pu | wa | ne | ex | | | | | |
| Frame | 269 | 270 | 320 | 463 | 464 | 504 | 505 | 509 | 547 | 639 | 738 | 855 | 993 | 1015 | 1016 | 1027 | 1028 | 1106 | 1151 |
| Leave 2 | en | ne | wa | br | mp | pd | mp | wa | br | wa | br | wa | mp | pu | mp | br | wa | ne | ex |

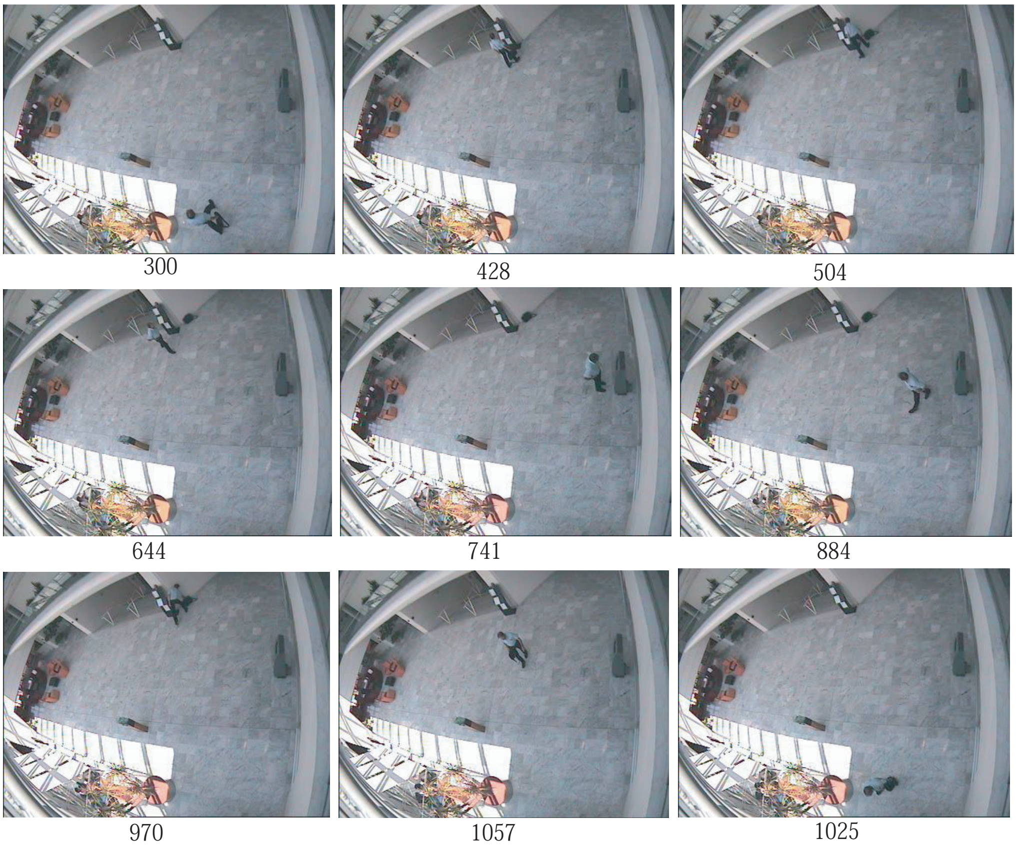


Figure 5.4. Key frames for the ”Leave Behind and Pick Up” (Leave1) sequence.

### 5.4.2 Results of activity recognition

The following experiments show that the proposed method is well-suited for recognizing sequential and overlapped single-agent activities. In the first two experiments it is shown that the use of DI improves performance as opposed to not using DI. In the latter two sections, the effect of the values chosen for grammar rule probabilities and the mixture weights are examined. It is shown that the parameters of the grammar and the parameters of the mixture weight have only a minimal impact on the results.

The video data used for this experiment was taken in a lobby environment (Figure 5.4) and the sequence of level 1 actions were generated using the labeled CAVIAR data. Analysis was run on six video sequences (Walk1, Walk2, Browse1, Browse2, Leave1 and Leave2) to test the performance of the system. The recognition results are depicted as stacked area graphs for each type of activity and are shown in Figures 5.5, 5.6 and 5.7. In each figure, the ground truth is given along with the results for each of the four different experimental setups.

Probabilistic inference with the BN was performed using an exact algorithm (belief propagation with the junction tree algorithm) with Netica [NET] for all of the experiments. However, as mentioned previously, as the size of the grammar (and the BN) increases, approximation algorithms such as loopy belief propagation [MWJ99] will need to be used to perform the inference task.

### 5.4.3 Ground truth

The ground truth was compiled from multiple users, as a normalized sum of the interpretations of the video data. Each labeler was given a definition (Table 5.7) for each level 2 action and directed to label every frame for each action independently. Users were given the option of labeling each frame with a *yes*, *maybe* or *no* (10 points for *yes*, 5 points for *maybe* and 0 points for *no*). No restrictions were placed on the number of times they could re-label or review the video sequences. They were not shown the string of primitive symbols extracted from the CAVIAR data.

Table 5.7. Definitions for ground truth labeling.

| | |
|---|---|
| Arriving | A period of time where the agent has just entered the scene. It must occur near a known exit or entrance. |
| Passing Through | Agent appears to be simply walking through the lobby. Pattern should look like: Enter + passing through + exit. Agent is not looking around. |
| Being Idle | The agent appears to be walking around aimlessly. Usually characterized by walking slowly and stopping in place. Sometimes includes browsing. |
| Browsing | Period of time where the agent is near a landmark (counter, magazine rack, information booth). The agent appears to be looking at a landmark. |
| Taking Away | Agent appears to be picking something up or preparing to pick something up. Includes movement just before and after picking up the object. |
| Leaving Behind | The agent appears to be putting something down or preparing to put something down. Includes movement just before and after putting down the object. |
| Departing | Period of time where it seems that the agent is about to exit the scene. Ceases once the agent exits the scene. |

Table 5.8. Definitions (a) Definition of the data types (b) Formulas for the different rates.

| | |
|---|---|
| A | Number of RELEVENT documents RETRIEVED |
| B | Number of RELEVENT documents NOT RETRIEVED |
| C | Number of IRRELEVENT documents RETRIEVED |
| D | Number of IRRELEVENT documents NOT RETRIEVED |

(a)

| | |
|---|---|
| Recall: A/(A+B) | Relevant data retrieved from relevant data |
| Precision: A/(A+C) | Relevant data retrieved from retrieved data |
| Miss: B/(A+B) | Relevant data missed (1-recall) |
| False: C/(C+D) | Irrelevant data retrieved from all irrelevant data |

(b)

### 5.4.4 Using deleted interpolation

The recall rate, precision rate, miss rate and false detection rates are given for each of the six video sequences in Table 5.10 when deleted interpolation was implemented using the grammar in Table 5.4. The definition of each rate is given in Table 5.8.

The precision rate was 88% after filtering out a common problem (explained later). *Arriving* and *Departing* had the highest precision rate (∼95%) because the activities were highly dependent on location (i.e. near a known exit or entrance) which made early detection relatively easy. In contrast, *Taking Away* had the lowest precision rate because the system was only able to detect the activity after the removed item was detected visually.

The frequent mis-detection of *Being Idle* as *Passing Through* had a negative effect on four of the six sequences, contributing to a 16% drop in the precision rate (Browse1, Browse2, Leave1 and Leave2). This drop in performance can be explained by the fact that the ground truth was collected under conditions that differ from the proposed system. Under normal conditions, a system cannot know if an agent will become idle or not and therefore can only label an initial detection of a mobile agent, as *Passing Through* the scene. In contrast, the ground truth was labeled with the foreknowledge of what the agent would do in the subsequent frames, giving the user the ability to mark an agent as being idle upon entry into the scene. Therefore, it is reasonable to remove the mis-detection of *Being Idle* as *Passing Through* to obtain a more realist precision rate.

The recall (capture) rate was 59% (equivalently, a miss rate of 41%) which indicates that the system was not able to detect the activity for the complete duration of the level 2 action as described by the ground truth data. The low recall rate is caused by similar reasons stated for the precision rate. The foreknowledge of the entire sequence gave the labeler the ability to recognize activities much earlier than the visual information permits. In contrast, the system changes its output only when a new terminal symbol (a significant visual change) is encountered.

The false alarm rate was 3% (not including the effects of *Passing Through*). The low false alarm rate is expected because the input symbols (level 1 actions) only change when there is a significant change in an agents visual characteristics.

An example of the detection of overlapping (concurrent) activities can be seen in the first transition from *Passing Through* to *Departing* in Figure 5.5(c). At about frame 315, both activities are detected and depicted as two stacked regions. Similar detections of overlapped activities are observed for *Browsing* and *Being Idle* in Figure 5.6(c).

In comparison to the ground truth, is was observed that the transitions between activities were abrupt for the experimental results. The sharp transitions can be attributed to the fact that the input into the system was a discrete sequence of primitive actions (level 1 actions), where each symbols was output only when a significant visual change was detected in appearance and movement (as defined by the CAVIAR data). In contrast, the ground truth was based on more detailed visual queues (e.g. body posture, head position) and foreknowledge of the entire sequence. The ground truth was also averaged over the labels of multiple users, which allowed the transition between activities to become smoother.

Table 5.9. Counts for the different rates.

| | Type | WALK1 | WALK2 | BROWSE1 | BROWSE2 | LEAVE1 | LEAVE2 | Total |
|---|---|---|---|---|---|---|---|---|
| Arriving | A | 52 | 144 | 87 | 90 | 116 | 38 | 527 |
| | B | 3 | 129 | 42 | 50 | 49 | 3 | 276 |
| | C | 8 | 0 | 0 | 7 | 2 | 13 | 30 |
| | D | 218 | 904 | 553 | 443 | 719 | 836 | 3673 |
| Passing Through | A | 215 | 569 | 0 | 0 | 0 | 0 | 784 |
| | B | 32 | 63 | 0 | 0 | 0 | 0 | 95 |
| | C | 0 | 48 | 202 | 320 | 263 | 143 | 976 |
| | D | 34 | 497 | 480 | 270 | 623 | 747 | 2651 |
| Being Idle | A | 0 | 0 | 360 | 63 | 229 | 645 | 1297 |
| | B | 0 | 0 | 211 | 346 | 208 | 158 | 923 |
| | C | 0 | 0 | 0 | 29 | 151 | 43 | 223 |
| | D | 281 | 1177 | 111 | 152 | 298 | 44 | 2063 |
| Browsing | A | 0 | 0 | 189 | 21 | 0 | 209 | 419 |
| | B | 0 | 0 | 65 | 112 | 0 | 155 | 332 |
| | C | 0 | 0 | 5 | 0 | 0 | 42 | 47 |
| | D | 281 | 1177 | 423 | 457 | 886 | 484 | 3708 |
| Taking Away | A | 0 | 0 | 0 | 0 | 82 | 12 | 94 |
| | B | 0 | 0 | 0 | 0 | 32 | 56 | 88 |
| | C | 0 | 0 | 0 | 0 | 76 | 0 | 76 |
| | D | 281 | 1177 | 682 | 590 | 696 | 822 | 4248 |
| Leaving Behind | A | 0 | 0 | 0 | 0 | 62 | 16 | 78 |
| | B | 0 | 0 | 0 | 0 | 27 | 47 | 74 |
| | C | 0 | 0 | 0 | 0 | 8 | 27 | 35 |
| | D | 281 | 1177 | 682 | 590 | 789 | 800 | 4319 |
| Departing | A | 26 | 158 | 20 | 48 | 151 | 45 | 448 |
| | B | 94 | 522 | 31 | 9 | 76 | 27 | 759 |
| | C | 0 | 0 | 16 | 0 | 1 | 0 | 17 |
| | D | 161 | 497 | 615 | 533 | 658 | 818 | 3282 |

Table 5.10. Rates for recall, precision, miss and false alarm.

| | Rate | WALK1 | WALK2 | BROWSE1 | BROWSE2 | LEAVE1 | LEAVE2 | Average |
|---|---|---|---|---|---|---|---|---|
| Arriving | Recall | 94.5% | 52.7% | 67.4% | 64.3% | 70.3% | 92.7% | 65.6% |
| | Precision | 86.7% | 100.0% | 100.0% | 92.8% | 98.3% | 74.5% | 94.6% |
| | Miss | 5.5% | 47.3% | 32.6% | 35.7% | 29.7% | 7.3% | 34.4% |
| | False | 3.5% | 0.0% | 0.0% | 1.6% | 0.3% | 1.5% | 0.8% |
| Passing Through | Recall | 87.0% | 90.0% | | | | | 89.2% |
| | Precision | 100.0% | 92.2% | | | | | 44.5% |
| | Miss | 13.0% | 10.0% | | | | | 10.8% |
| | False | 0.0% | 8.8% | 29.6% | 54.2% | 29.7% | 16.1% | 26.9% |
| Being Idle | Recall | | | 63.0% | 15.4% | 52.4% | 80.3% | 58.4% |
| | Precision | | | 100.0% | 68.5% | 60.3% | 93.8% | 85.3% |
| | Miss | | | 37.0% | 84.6% | 47.6% | 19.7% | 41.6% |
| | False | 0.0% | 0.0% | 0.0% | 16.0% | 33.6% | 49.4% | 9.8% |
| Browsing | Recall | | | 74.4% | 15.8% | 0.0% | 57.4% | 55.8% |
| | Precision | | | 97.4% | 100.0% | 0.0% | 83.3% | 89.9% |
| | Miss | | | 25.6% | 84.2% | 0.0% | 42.6% | 44.2% |
| | False | 0.0% | 0.0% | 1.2% | 0.0% | 0.0% | 8.0% | 1.3% |
| Taking Away | Recall | | | | | 71.9% | 17.6% | 51.6% |
| | Precision | | | | | 51.9% | 100.0% | 55.3% |
| | Miss | | | | | 28.1% | 82.4% | 48.4% |
| | False | 0.0% | 0.0% | 0.0% | 0.0% | 9.8% | 0.0% | 1.8% |
| Leaving Behind | Recall | | | | | 69.7% | 25.4% | 51.3% |
| | Precision | | | | | 88.6% | 37.2% | 69.0% |
| | Miss | | | | | 30.3% | 74.6% | 48.7% |
| | False | 0.0% | 0.0% | 0.0% | 0.0% | 1.0% | 3.3% | 0.8% |
| Departing | Recall | 21.7% | 23.2% | 39.2% | 84.2% | 66.5% | 62.5% | 37.1% |
| | Precision | 100.0% | 100.0% | 55.6% | 100.0% | 99.3% | 100.0% | 96.3% |
| | Miss | 78.3% | 76.8% | 60.8% | 15.8% | 33.5% | 37.5% | 62.9% |
| | False | 0.0% | 0.0% | 2.5% | 0.0% | 0.2% | 0.0% | 0.5% |

### 5.4.5 No deleted interpolation

To understand the advantage of using DI, an experiment was performed again on the same sequences but without the use of DI. The removal of DI is equivalent to the use of a single HBN shifted over time over a fixed temporal window to recognize activities. Since subsequences of the evidence are not used to interpolate the results, several level 2 actions based on smaller strings were not detected by the system.

The level 2 action that was effected the most was *Departing* because the sequence of primitive symbols $\{$*wa, ne, ex*$\}$ was never detected by the input sequences. Furthermore, since *Departing* relies heavily on the use of smaller substrings of one or two level 1 action symbols to detect, removing the DI framework significantly reduces the systems ability to recognize departures. In contrast, one instance of temporal concurrence was detected in Figure 5.7(b) between *Being Idle* and three other activities. This overlap was captured because in the grammar, a subset of the sequences of actions used by *Being Idle* was also used for the recognition of *Browsing, Leaving Behind* and *Taking Away*.

### 5.4.6 DI using uniformly distributed grammar parameters

The original grammar parameters (production rule probabilities) were set at the discretion of a knowledge engineer, giving greater weight to sequences that were more likely to occur. However, in this set of experiments, the probabilities were distributed uniformly among all possible productions for each nonterminal symbol. That is, the production probabilities $P(N \rightarrow \zeta_i)$ were uniformly distributed such that for the nonterminal $N$, $\sum_i P(N \rightarrow \zeta_i) = 1$ where $\zeta$ is a string of one or more symbols on the right-hand side of the production rule.

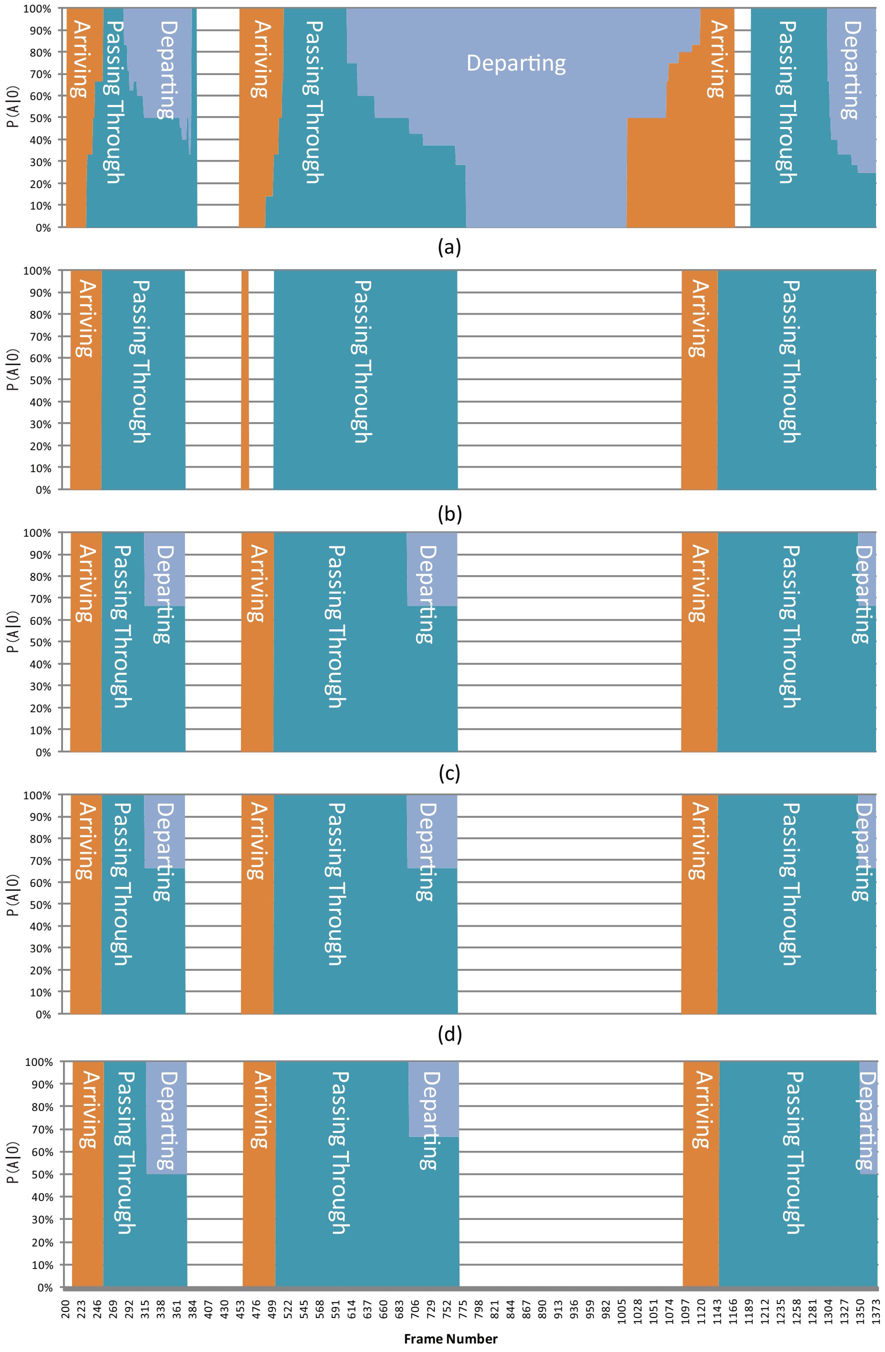
Since changing the probabilities of the rules changes only the proportions between overlapped activities and not the duration of activities themselves, the rates remain the same. It is interesting to observe that the proportion of the probabilities between activities remain virtually unchanged after rule probabilities have been changed (Figures 5.5(d), 5.6(d) and 5.7(d)). This is due to the fact that the structural analysis of a symbol sequence plays a larger role in determining the results compared to the role of the probabilities of the rules. Therefore, it is more important to include the correct rules in the grammar than to assign the optimal probabilities.

### 5.4.7 DI using uniformly distributed mixture weights

Previously, the mixture weights for deleted interpolation were set so that $\lambda_1 > \lambda_2 > \cdots > \lambda_l$, giving more weight to longer subsets of the data. For this experiment, the mixture weights

$\lambda_i$ were uniformly distributed, giving equal weight to each term in the interpolation equation. That is, $\sum_{i=1}^{l} \lambda_i = 1$ and $\lambda_i = 1/l$. A uniform weighting scheme can be interpreted as giving equal confidence to each of the $l$ terms in the DI equation.

Small changes in the proportions between overlapped probabilities were observed for the detection of *Departing* and *Passing through* (a higher probability for *Departing*), which was closer to the ground truth. In general however, the results remained similar to the results of using the original weighting scheme (Figures 5.5(e), 5.6(e) and 5.7(e)). As in the previous experiment, it was observed here that the structural constraints outweigh the values of the mixture weights such that the proportions between overlapped activities change only nominally when the mixture weights are varied. Again, since the mixture weights only effect the proportion between the probabilities of the actions and not their durations, the detection rates remain unchanged.



(a)

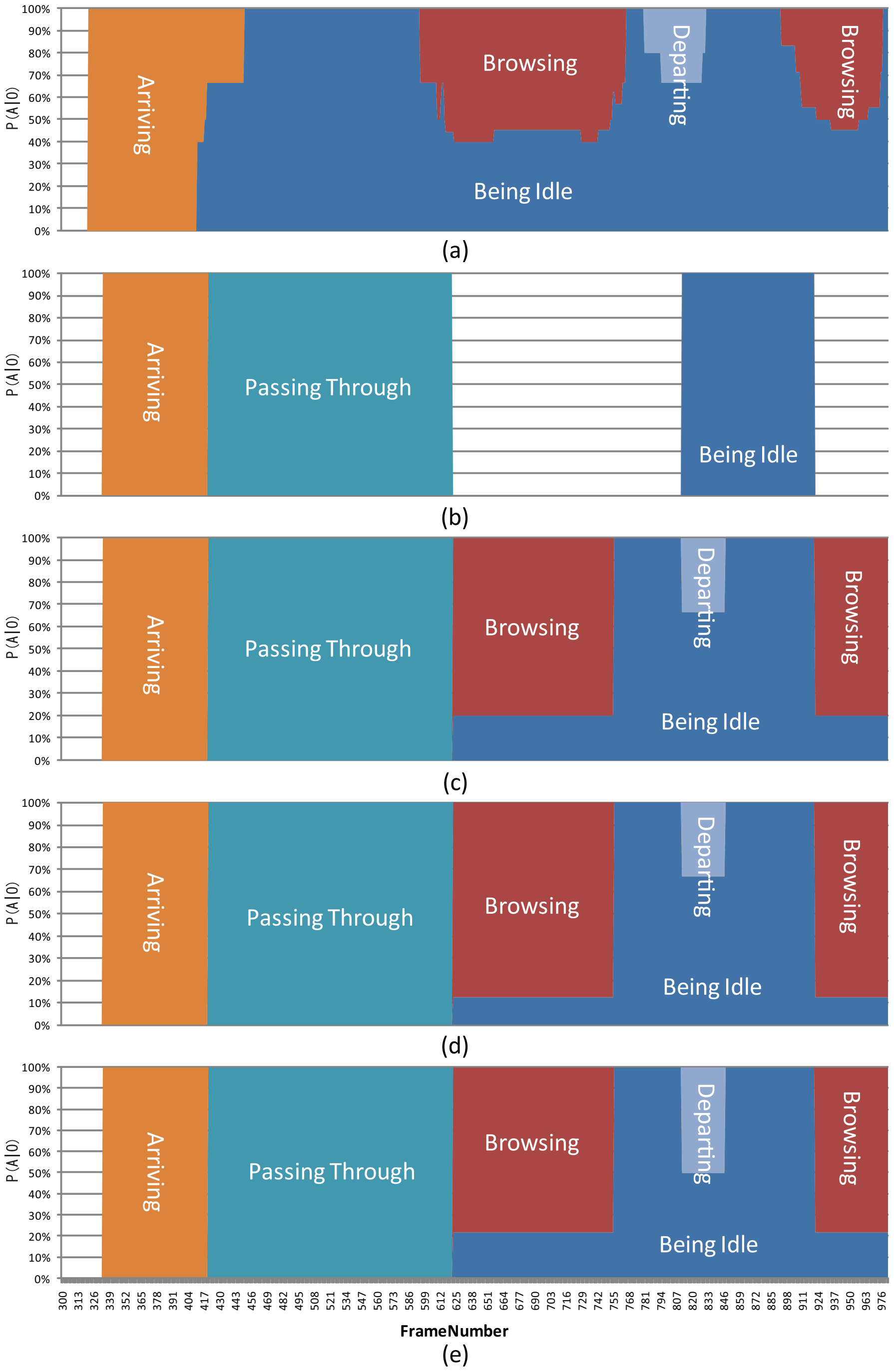
(b)

(c)

(d)

(e)

Figure 5.5. Walk sequence (a) ground truth (b) no DI (c) DI with user defined rule probabilities (d) DI with uniformly distributed rule probabilities (e) DI with uniform mixture weights



(a)

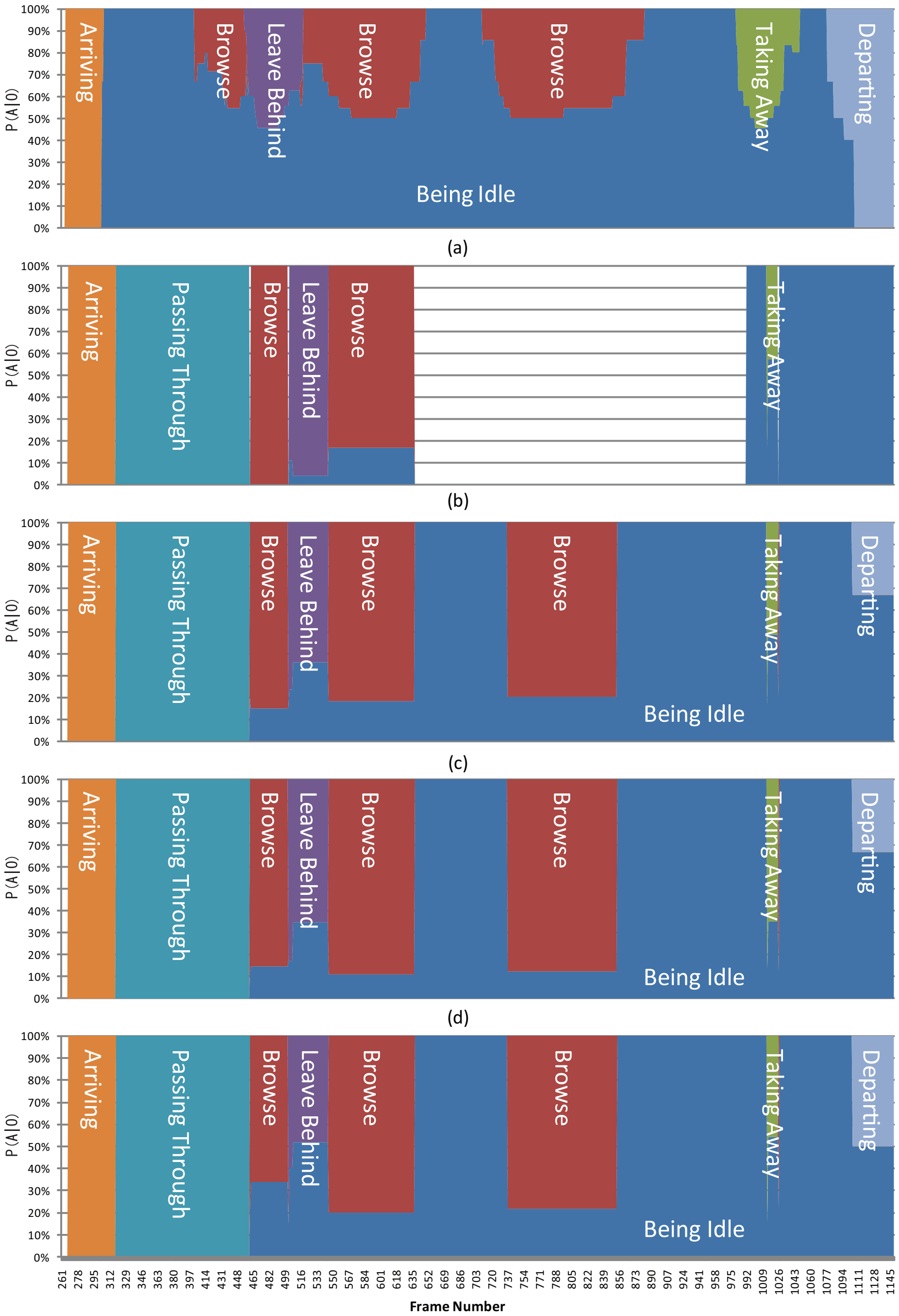
(b)

(c)

(d)

(e)

Figure 5.6. Browse sequence (a) ground truth (b) no DI (c) DI with user defined rule probabilities (d) DI with uniformly distributed rule probabilities (e) DI with uniform mixture weights



(a)

(b)

(c)

(d)

(e)

Figure 5.7. Leave bag sequence (a) ground truth (b) no DI (c) DI with user defined rule probabilities (d) DI with uniformly distributed rule probabilities (e) DI with uniform mixture weights

## 5.5 Conclusion

This chapter has addressed the issue of hierarchical representation of human activity by basing the proposed system on a SCFG. The SCFG was then converted to a HBN to allow the system to make complex probabilistic queries needed for uncertain inputs. As a preliminary test, the HBN was used to discover overlapped activities over a string of discrete primitive action symbols via DI. Through a set of preliminary experiments, it was shown that the proposed methodology is well-suited for detecting the overlap of simple single-agent activities.

Admittedly, manually defining the grammar may be problematic when a pre-defined grammar is not available or when the input string is noisy. However as covered in chapter 4, an unsupervised grammar learning technique can be implemented to extract a grammar from a noisy input sequence generated from a real video sequence [KSS07] when there is sufficient data.

Finally given the current capacity for computation within a reason amount of time, this proposed system is not feasible to real-time use for more complex (large grammars). That is, as the grammar grows in complexity, the complexity of the resulting Bayesian network also grows exponentially because it must explicitly model every possible string produced by the grammar. While a heavily syntactic approach is appropriate for strictly goal oriented behavior, a hybrid approach using both statistics and syntax may be more favorable for less organized behavior.

# Chapter 6

# Conclusion

## 6.1 Summary

This thesis has presented a bottom-up computational framework for modeling, learning and recognizing human activities. In the first section, it was shown that by describing primitive actions as a combination of both motion and visual context, the proposed algorithm was able to correctly categorize actions from a video database of actions. As a result, the segments of an action sequence were labeled according to the respective class yielding a string of action symbols. In the second section it was shown that by testing various hypothesis using an MDL criterion enabled the proposed system to discover the basic structure of an activity sequence from a symbol string of primitive actions corrupted by noise. As a result, an optimal SCFG expressing the grammar of the activities contained in the action string was acquired. In the third section it was shown that given a stochastic context-free grammar that describes human activity, the activities occurring within a stream of observations (a string of action symbols) can be detected, even when the activities are overlapped. Taken as a whole, the algorithms presented in this thesis describe a prototype system for learning and recognizing human activities from a video sequence.

## 6.2 Contributions

- Contributed a bimodal learning approach that used both **motion and visual context without the use of *a priori* scene knowledge**, whereas previous work used only motion or relied on *a priori* knowledge of the appearance of objects or actors.
- Contributed a **new unsupervised algorithm for learning syntactic structure from noisy data** (potentially all negative examples), whereas previous work on grammatical induction used a training set of positive examples.
- Contributed the first work that **robustly recognizes overlapped human activities** using a **syntactic** framework.

## 6.3 Discussion and future works

### 6.3.1 Learning primitive actions

It was shown that using both motion features and visual context (visual features) improves classification performance when the actions in the video corpus contains actions that are defined by both motion and visual context. On the other hand, in the case where one mode is not able to effectively categorize the corpus, it was also observed that the faulty mode degrades the overall performance. This phenomenon is based on the fact that the current model couple both modes and gives equal weight to both modes. Therefore the overall performance is effected when one mode fails to categorize an action. From the standpoint of a generative model it might be beneficial to model a weight parameter that determines the degree to which a given category effects the distribution over a mode. Intuitively, there are some actions like *walking* are distinguished mostly by motion features and depend very little on the visual context.

Considering the potential applications of the broader topic of latent category learning from a video corpus, the computational cost with respect to time and memory are very critical issues. The proposed method has taken only a preliminary step in considering an online algorithm for latent category learning but there are still many more variations to be explored. For example, the NMF (and likewise PLSA) algorithm has online counterparts [CSS$^+$07] that could be used to make the entire framework work online.

### 6.3.2 Learning the structure of activities

The proposed method for learning the structure of activities was presented as a bottom-up framework that assumes no available *a priori* knowledge about the grammar to be learned. However, in the case of activity learning for surveillance applications, it is highly likely that there will be some prior knowledge of the scenes to be encountered and activity patterns to be learned. The integration of top-down prior knowledge and bottom-up learning is a potentially fruitful direction for future research. In fact, hybrid approaches have been proposed for image segmentation to leverage both top-down and bottom-up information [BSU04].

### 6.3.3 Recognizing structured activities

The probabilistic syntactic model was shown to be successful at recognizing structured human activities. However, it is also true that many activities do not always conform to a strict syntactic rules. As such, syntactic approaches (grammars and state machines) to activity recognition are limited to a subset of ordered human activities. The use of petri nets or propagation networks have been proposed to address the loosely ordered nature of activities. While methods for learning such models is still a topic to be addressed, using more expressive models for recognition is also a topic that has yet to be fully explored in computer vision.

## 6.4 Final thoughts

In the final analysis, it still remains that mimicking the human process of learning and perceiving human actions is a very challenge task. This work has focused solely on a computational framework of learning and recognizing the *physical* phenomena of human activity using a bottom-up approach. Therefore, this work has yet to touch on the topic of actually closing the semantic gap that exists between physical motion and mental understanding. In other words, while the techniques introduced in this thesis can robustly identify the actions patterns found in noisy video data, there is no way to assign the proper *semantic* label without the interaction of human interpretation. This however should not be interpreted to be a short-coming of this work since even we as humans are not able to learn without someone teaching us the semantic expressions needed to describe an activity. The next step for human activity recognition is then, the integration of human interaction (top-down knowledge) as a means of semantic acquisition and bottom-up computation to model activities.

# Bibliography

[All84] James F. Allen. Towards a general theory of action and time. *Artificial Intelligence*, 23(2):123–154, 1984.

[AS01] Douglas Ayers and Mubarak Shah. Monitoring human behavior from video taken in an office environment. *Image Vision Comput.*, 19(12):833–846, 2001.

[BGS$^+$05] Moshe Blank, Lena Gorelick, Eli Shechtman, Michal Irani, and Ronen Basri. Actions as space-time shapes. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1395–1402, 2005.

[BI05] Oren Boiman and Michal Irani. Detecting irregularities in images and in video. In *Proceedings of the International Conference on Computer Vision*, pages I:462–469, 2005.

[BNJ03] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[Bou02] Jean Y. Bouguet. Pyramidal implementation of the Lucas Kanade feature tracker: Description of the algorithm, 2002.

[Bra96] Matthew Brand. Understanding manipulation in video. In *Proceedings of the 2nd International Conference on Automatic Face and Gesture Recognition*, page 94, 1996.

[BSU04] Eran Borenstein, Eitan Sharon, and Shimon Ullman. Combining top-down and bottom-up segmentation. In *Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition Workshop*, page 46, 2004.

[Bun02] Wray Buntine. Variational extensions to EM and multinomial PCA. In *Proceedings of the European Conference on Machine Learning*, pages 23–34, 2002.

[BVW01] Hung Hai Bui, Svetha Venkatesh, and Geoff A. W. West. Tracking and surveillance in wide-area spatial environments using the abstract hidden Markov model. *Inter-*

*national Journal of Pattern Recognition and Artificial Intelligence*, 15(1):177–195, 2001.

[CAV] EC funded CAVIAR project under the IST fifth framework programme (ist-2001-37540). Found at http://homepages.inf.ed.ac.uk/rbf/CAVIAR/.

[CDW$^+$04] Gabriela Csurka, Chris Dance, Jutta Willamowski, Lixin Fan, and Cedric Bray. Visual categorization with bags of keypoints. In *Proceedings of the ECCV International Workshop on Statistical Learning in Computer Vision*, 2004.

[CLK00] Robert Collins, Alan Lipton, and Takeo Kanade. Introduction to the special section on video surveillance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):745–746, 2000.

[CSS$^+$07] Bin Cao, Dou Shen, Jian-Tao Sun, Xuanhui Wang, Qiang Yang, and Zheng Chen. Detect and track latent factors with online nonnegative matrix factorization. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 2689–2694, 2007.

[CV96] Roberto Casati and Achille Varzi. *Events.* Dartmouth, 1996.

[CV02] Roberto Casati and Achille Varzi. The Stanford Encyclopedia of Philosophy: Events, Fall 2002. http://plato.stanford.edu/archives/fall2002/entries/events/.

[Dan63] Arthur Danto. What we can do. *The Journal of Philosophy*, 60(15):435–445, 1963.

[DBPV05] Thi V. Duong, Hung H. Bui, Dinh Q. Phung, and Svetha Venkatesh. Activity recognition and abnormality detection with the switching hidden semi-Markov model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 838–845, 2005.

[DHS00] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification.* Wiley-Interscience Publication, 2000.

[DRCB05] Piotr Dollár, Vincent Rabaud, Garrison Cottrell, and Serge Belongie. Behavior recognition via sparse spatio-temporal features. In *Proceedings of the IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pages 65–72, 2005.

[FA98] Andrew H. Fagg and Michael A. Arbib. Modeling parietal–premotor interactions in primate control of grasping. *Neural Networks*, 11(7-8):1277–1303, 1998.

[FST98] Shai Fine, Yoram Singer, and Naftali Tishby. The hierarchical hidden Markov model: Analysis and applications. *Machine Learning*, 32(1):41–62, 1998.

[FZMP05] Claudio Fanti, Lihi Zelnik-Manor, and Pietro Perona. Hybrid models for human motion recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, 2005.

[GD07] Abhinav Gupta and Larry S. Davis. Objects in action: An approach for combining action understanding and object perception. In *Proceedings of the IEEE Conference on Computer Vision*, pages 1–8, 2007.

[GDDD04] Nagia Ghanem, Daniel DeMenthon, David Doermann, and Larry Davis. Representation and recognition of events in surveillance video using petri nets. In *Second IEEE Workshop on Event Mining*, page 112, 2004.

[GG05] Eric Gaussier and Cyril Goutte. Relation between PLSA and NMF and implications. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 601–602, 2005.

[GJH01] Aphrodite Galata, Neil Johnson, and David Hogg. Learning variable-length Markov models of behavior. *Computer Vision and Image Understanding*, 81(3):398–413, 2001.

[GX03] Shaogang Gong and Tao Xiang. Recognition of group activities using dynamic probabilistic networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 742–749. IEEE Computer Society, 2003.

[HJB$^+$05] Raffay Hamid, Amos Y. Johnson, Samir Batta, Aaron F. Bobick, Charles L. Isbell, and Graham Coleman. Detection and explanation of anomalous activities: Representing activities as bags of event n-grams. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages I: 1031–1038, 2005.

[Hof99] Thomas Hofmann. Probabilistic latent semantic analysis. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 289–29, 1999.

[IB00] Yuri A. Ivanov and Aaron F. Bobick. Recognition of visual activities and interactions by stochastic parsing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):852–872, 2000.

[KSS05] Kris M. Kitani, Yoichi Sato, and Akihiro Sugimoto. Deleted interpolation using a hierarchical bayesian grammar network for recognizing human activity. In *Proceedings of the Second Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pages 239–246, 2005.

[KSS07] Kris M. Kitani, Yoichi Sato, and Akihiro Sugimoto. Recovering the basic structure of human activities from a video-based symbol string. In *Proceedings of the IEEE Workshop on Motion and Video Computing*, pages 9–9, 2007.

[Lap05] Ivan Laptev. On space-time interest points. *International Journal on Computer Vision*, 64(2):107–123, 2005.

[LC03] Xiaohui Liu and Chin-Seng Chua. Multi-agent activity recognition using observation decomposed hidden Markov model. In *Proceedings of the Third International Conference on Computer Vision Systems*, pages 247–256, 2003.

[Lin07] Chih-Jen Lin. Projected gradient methods for nonnegative matrix factorization. *Neural Computation*, 19(10):2756–2779, 2007.

[Low99] David G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision*, page II:1150, 1999.

[LS99] Daniel D. Lee and H. Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–791, 1999.

[Mac67] James B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297, 1967.

[MCUP02] Jiri Matas, Ondrej Chum, Martin Urban, and Tomás Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *Proceedings of the British Machine Vision Conference*, 2002.

[ME02] Darnell J. Moore and Irfan A. Essa. Recognizing multitasked activities from video using stochastic context-free grammar. In *Proceedings of the National Conference on Artificial Intelligence*, pages 770–776, 2002.

[MEH99] Darnell J. Moore, Irfan A. Essa, and Monson H. Hayes. Exploiting human actions and object context for recognition tasks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 80–86, 1999.

[MES03] David Minnen, Irfan A. Essa, and Thad Starner. Expectation grammars: Leveraging high-level expectations for activity recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages II: 626–632, 2003.

[MS03] Christopher D. Manning and Hinrich Schutze. *Foundations of Statistical Natural Language Processing*. The MIT Press, 2003.

[MWJ99] Kevin P. Murphy, Yair Weiss, and Michael I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Uncertainty in Artificial Intelligence*, pages 467–475, 1999.

[NBVW03] Nam T. Nguyen, Hung Hai Bui, Svetha Venkatesh, and Geoff A. W. West. Recognising and monitoring high-level behaviours in complex spatial environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages II: 620–625, 2003.

[NET] Netica. Found at http://www.norsys.com/.

[NFF07] Juan Carlos Niebles and Li Fei-Fei. A hierarchical model of shape and appearance for human action classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[NMTM99] Kamal Nigam, Andrew McCallum, Sebastian Thrun, and Tom Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 1999.

[NMW97] Craig G. Nevil-Manning and Ian H. Witten. Identifying hierarchical strcture in sequences: A linear-time algorithm. *Journal of Artificial Intelligence Research (JAIR)*, 7:67–82, 1997.

[NMW00] Craig G. Nevil-Manning and Ian H. Witten. Online and offline heuristics for inferring hierarchies of repetitions in sequences. In *Proceedings of IEEE*, number 11 in 88, pages 1745–1755, 2000.

[NWFF06] Juan Carlos Niebles, Hongcheng Wang, and Li Fei-Fei. Unsupervised learning of human action categories using spatial-temporal words. In *Proceedings of the British Machine Vision Conference*, pages III:1249–1258, 2006.

[OARS04] Fazilah Othman, Rosni Abdullah, Nur'Aini Abdul Rashid, and Rosalina Abdul Salam. Parallel K-means clustering algorithm on DNA dataset. In *Proceedings of the International Conference on Parallel and Distributed Computing*, pages 248–251, 2004.

[OHG02] Nuria Oliver, Eric Horvitz, and Ashutosh Garg. Layered representations for human activity recognition. In *Proceedings of the IEEE International Conference on Multimodal Interfaces*, pages 3–8. IEEE Computer Society, 2002.

[OKA05] Abhijit S. Ogale, Alap Karapurkar, and Yiannis Aloimonos. View-invariant modeling and recognition of human actions using grammars. In *Proceedings of the Workshop on Dynamical Vision*, 2005.

[OP07] Oleg Okun and Helen Priisalu. Fast nonnegative matrix factorization and its application for protein fold recognition. *EURASIP J. Appl. Signal Process.*, 2006(1):62–62, 2007.

[ORP00] Nuria M. Oliver, Barbara Rosario, and Alex Pentland. A bayesian computer vision system for modeling human interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):831–843, 2000.

[PB98] Claudio S. Pinhanez and Aaron F. Bobick. Human action detection using pnf propagation of temporal constraints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, page 898, 1998.

[Pea88] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988.

[Per08] Florent Perronnin. Universal and adapted vocabularies for generic visual categorization. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(7):1243–1256, 2008.

[PW98] David V. Pynadath and Michael P. Wellman. Generalized queries on probabilistic context-free grammars. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(1):65–77, 1998.

[QR89] J. Ross Quinlan and Ronald L. Rivest. Inferring decision trees using the minimum description length principle. *Information and Computation*, 80(3):227–248, 1989.

[RA06] Michael S. Ryoo and Jake K. Aggarwal. Recognition of composite human activities through context-free grammar based representation. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 1709–1718, 2006.

[SHM$^+$04] Yifan Shi, Yan Huang, David Minnen, Aaron F. Bobick, and Irfan A. Essa. Propagation networks for recognition of partially ordered sequential action. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 862–869, 2004.

[Sis00] Jeffrey Mark Siskind. Visual event classification via force dynamics. In *Proceedings of the National Conference on Artificial Intelligence and Conference on Innovative Applications of Artificial Intelligence*, pages 149–155, 2000.

[SLC04] Christian Schuldt, Ivan Laptev, and Barbara Caputo. Recognizing human actions: A local SVM approach. In *Proceedings of the International Conference on Pattern Recognition*, pages 32–36, 2004.

[SP95] Thad Starner and Alex Pentland. Real-time American sign language recognition from video using hidden Markov models. In *Proceedings of the International Symposium on Computer Vision*, page 265, 1995.

[ST94] Jianbo Shi and Carlo Tomasi. Good features to track. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1994.

[Sto94] Andreas Stolcke. *Bayesian Learning of Probabilistic Language Models*. PhD thesis, University of California at Berkeley, 1994.

[TJBB06] Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006.

[VG02] Alexei Vinokourov and Mark Girolami. A probabilistic framework for the hierarchic organisation and classification of document collections. *Journal of Intelligent Information Systems*, 18(2-3):153–172, 2002.

[WKC07] Shu-Fai Wong, Tae-Kyun Kim, and R. Cipolla. Learning motion categories using both semantic and structural information. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–6, 2007.

[WM98] Toshikazu Wada and Takashi Matsuyama. Appearance based behavior recognition by event driven selective attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 759–764, 1998.

[WMG07] Xiaogang Wang, Xiaoxu Ma, and Eric Grimson. Unsupervised activity perception by hierarchical Bayesian models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[WSXZ01] Tian-Shu Wang, Heung-Yeung Shum, Ying-Qing Xu, and Nan-Ning Zheng. Unsupervised analysis of human gestures. In *Proceedings of the IEEE Pacific Rim Conference on Multimedia*, pages 174–181, 2001.

[YOI92] Junji Yamato, Jun Ohya, and Kenichiro Ishii. Recognizing human action in time-sequential images using hidden Markov model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 379–385. IEEE Computer Society, 1992.

[ZT01] Jeffrey M. Zacks and Barbara Tversky. Event structure in perception and conception. *Psychological Bulletin*, 127:3–21, 2001.

# Publications

## International Journals

[1] Kris M. Kitani, Yoichi Sato, and Akihiro Sugimoto.
"Recovering the Basic Structure of Human Activities from Noisy Video-Based Symbol Strings." To appear in the *International Journal of Pattern Recognition and Artificial Intelligence*. 2008.

[2] Kris M. Kitani, Yoichi Sato and Akihiro Sugimoto.
"Recognizing Overlapped Human Activities from a Sequence of Primitive Actions via Deleted Interpolation."
To appear in the *International Journal of Pattern Recognition and Artificial Intelligence*. 2008.

## International Conferences and Workshops

[3] Kris M. Kitani, Takahiro Okabe, Yoichi Sato and Akihiro Sugimoto.
Discovering Primitive Action Categories by Leveraging Relevant Visual Context.
To appear in the proceeding of the *IEEE International Workshop on Visual Surveillance* (VS2008).
October 2008.

[4] Kris M. Kitani, Yoichi Sato and Akihiro Sugimoto.
"Recovering the Basic Structure of Human Activities from a Video-Based Symbol String."
Proceedings of the *IEEE Workshop on Motion and Video Computing* (WMVC 2007).
February 2007.

[5] Kris M. Kitani, Yoichi Sato and Akihiro Sugimoto. "An MDL Approach to Learning Activity Grammars." Proceedings of the *Korea-Japan Joint Workshop on Pattern Recognition* (KJPR 2006).
November 2006.

[6] Kris M. Kitani, Yoichi Sato and Akihiro Sugimoto.
"Deleted Interpolation using a Hierarchical Bayesian Grammar Network for Recognizing Human Activity."
Proceedings of the Second Joint *IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance* (VS-PETS05).
October 2005.

### Domestic Journals

[7] 木谷クリス真実，佐藤洋一，杉本晃宏.
ノイズを考慮した人物行動の文法の教師無し学習.
電子情報通信学会論文誌 D（画像の認識・理解特集号），
2008 年 8 月号.

### Domestic Conferences

[8] 木谷クリス真実, 岡部 孝弘, 佐藤洋一, 杉本晃宏.
視覚的文脈を考慮した人物動作カテゴリの教師無し学習.
画像の認識・理解シンポジウム（MIRU2008）
2008 年 7 月.

[9] 近藤雄飛，岡部孝弘，木谷クリス真実，佐藤洋一.
カテゴリの共起を考慮した物体認識.
画像の認識・理解シンポジウム（MIRU2008）
2008 年 7 月.

[10] 木谷クリス真実, 佐藤洋一, 杉本晃宏.
映像に基づく人物行動の文法学習.
画像の認識・理解シンポジウム（MIRU2007）
2007 年 7 月.

[11] 木谷クリス真実, 佐藤洋一, 杉本晃宏.
Deleted Interpolation Using a Hierarchical Bayesian Grammar Network for Recognizing Human Activity.
画像の認識・理解シンポジウム（MIRU2005）
2005 年 7 月.

### Domestic Presentations (No peer review)

[12] 近藤雄飛，岡部孝弘，木谷クリス真実，佐藤洋一.
カテゴリ共起を考慮した物体認識手法.

情報処理学会コンピュータビジョンとイメージメディア研究会（CVIM）,
2008年3月.

[13] 木谷クリス真実，佐藤洋一，杉本晃宏.
終端記号の部分集合と最小記述長原理を用いた文法学習.
電子情報通信学会技術研究報告. PRMU, パターン認識・メディア理解
vol. 106, no. 300, PRMU2006-94,
2006年10月.

[14] Kris M. Kitani, Takahiro Okabe, Yoichi Sato and Akihiro Sugimoto.
“Unsupervised Action Category Discovery Using Visual Context.”
MIRU International Workshop on Computer Vision (MIRU-IMCV2008).
July 2008.