Recognition of Complex Human Activities in Multimedia Streams using Machine Learning and Computer Vision

Thesis submitted to Cardiff University in candidature for the degree of Doctor of Philosophy.

Ioannis M. Kaloskampis

Institute of Medical Engineering and Medical Physics
Cardiff University
2013

DECLARATION

This work has not previously been accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.

Signed..... (candidate) Date .....

STATEMENT 1

This thesis is being submitted in partial fulfillment of the requirements for the degree of PhD.

Signed ..... (candidate) Date .....

STATEMENT 2

This thesis is the result of my own investigation, except where otherwise stated. Other sources are acknowledged by giving explicit reference.

Signed ..... (candidate) Date .....

STATEMENT 3

I hereby give consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organizations.

Signed ..... (candidate) Date .....

STATEMENT 4

I hereby give consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, after expiry of a bar on access approved by the Graduate Development Committee.

Signed..... (candidate) Date .....

ABSTRACT

Modelling human activities observed in multimedia streams as temporal sequences of their constituent actions has been the object of much research effort in recent years. However, most of this work concentrates on tasks where the action vocabulary is relatively small and/or each activity can be performed in a limited number of ways. In this Thesis, a novel and robust framework for modelling and analysing composite, prolonged activities arising in tasks which can be effectively executed in a variety of ways is proposed. Additionally, the proposed framework is designed to handle cognitive tasks, which cannot be captured using conventional types of sensors.

It is shown that the proposed methodology is able to efficiently analyse and recognise complex activities arising in such tasks and also detect potential errors in their execution. To achieve this, a novel activity classification method comprising a feature selection stage based on the novel Key Actions Discovery method and a classification stage based on the combination of Random Forests and Hierarchical Hidden Markov Models is introduced. Experimental results captured in several scenarios arising from real-life applications, including a novel application to a bridge design problem, show that the proposed framework offers higher classification accuracy compared to current activity identification schemes.

ACKNOWLEDGEMENTS

I would like to thank my supervisors Yulia Hicks and Dave Marshall for providing excellent guidance through the course of my PhD. Through the countless discussions we had, they encouraged and inspired me to develop new ideas. I thank them for all their contributions of time, ideas and funding to make my PhD experience productive, challenging and enjoyable.

I would also like to thank C. Katsaras, B. Mawson, Prof. J.C. Miles, Prof. J. Patrick and V. Smy for their help and advice on the development of the bridge design task.

Lastly, I would like to thank my family for their unconditional support.

LIST OF ACRONYMS

ASL	American Sign Language
AUC	Area Under Curve
CHMM	Coupled Hidden Markov Model
EM	Expectation-Maximisation
FP	False Positive
fps	Frames Per Second
FN	False Negative
GMM	Gaussian Mixture Model
GUI	Graphical User Interface
HHMM	Hierarchical Hidden Markov Model
HMM	Hidden Markov Model
HSV	Hue, Saturation, Value
IRR	Irrelevant/Relevant Ratio
KAD	Key Action Discovery
KBS	Knowledge Based System

LDA	Latent Dirichlet allocation
LSA	Latent Semantic Analysis
MPEG	Moving Picture Experts Group
NLP	Natural Language Processing
OOB	Out-Of-Bag
PCA	Principal Component Analysis
P-Nets	Propagation Nets
pdf	Probability Density Function
PHMM	Parallel Hidden Markov Model
pLSA	Probabilistic Latent Semantic Analysis
QSR	Qualitative Spatial Relations
RC	Regularity Count
RF	Random Forest
RF-ID	Radio-Frequency Identification
ROC	Receiver Operating Characteristics
SCFG	Stochastic Context-Free Grammar
SVM	Support Vector Machine
tfidf	Term FrequencyInverse Document Frequency
UAV	Unmanned Aerial Vehicle
VI	Variable Importance

VLMM Variable Length Markov Model

VSM Vector Space Model

LIST OF SYMBOLS

x	Scalar quantity
$\mathbf{x}$	Vector quantity
$\mathbf{X}$	Matrix quantity
$\bar{\mathbf{x}}$	Mean vector
$\hat{\mathbf{x}}$	Estimate of original quantity $\mathbf{x}$
$(.)^T$	Transpose operator
$(.)^H$	Hermitian transpose operator
$(.)^{-1}$	Matrix inverse
$(.)^*$	Complex conjugate operator
$\|.\|$	Matrix determinant
$\|\|.\|\|_F$	Frobenius Norm
1	Vector of Ones
$\mathbf{I}$	Identity matrix

List of Figures

List of Tables

CONTENTS

Chapter 1

INTRODUCTION

The objective of computer based activity recognition is to automatically recognise the actions and goals of one or more agents from a series of observations using computer systems [Patterson et al., 2003, Hodges and Pollack, 2007, Laerhoveni, 2012]. Activity recognition offers a wide variety of applications relevant to many study areas such as medicine, social sciences and informatics which explains the vast and growing body of research devoted to its investigation.

Activity recognition methods ease critical but at the same time tedious, time consuming tasks saving on time, cost and human effort. For example, an intelligent fall detection system for the elderly can monitor the patient's movements at all times and prompt the administrator when anomalous activity (*e.g.* fall) is detected. Human intervention is required only when alarm is signaled by the system. Such systems have the advantage that their accuracy cannot be affected by human factors such as fatigue, operating constantly at the same high productivity standards. They are also cost efficient since they reduce the need for private nursing. Additionally they offer considerable conveniences such as sending automated reports to relatives of the patients who might live far away or are travelling, putting their minds at ease.

At the same time, activity recognition is a technically challenging

research area. The goal is to build intelligent machines capable of reliably mimicking the human ability of recognising activities. Activity analysis consists of three stages, *sensing, learning* and *inferring* [Liao, 2006]. Thus, the machine must be equipped with instruments which make up for the human eyes, ears, hands, *etc.* which can be cameras or other sensors. Then it must be able to use the data acquired by the sensors to *learn* models describing the observed activities. Finally, in the *inference* stage, the machine should be able to use the learned models in order to recognise the observed activities. Technical difficulties are present in all three stages. For example, in the sensing stage environmental noise can deteriorate the quality of observations making the extracted streams very hard to analyse. Or an activity might be too complex to be sufficiently captured by existing state of the art modelling methods, making the learning and inference stages very challenging.

There are currently two main approaches in terms of *sensing*: *sensor based* activity recognition frameworks, which use wearable sensors, RFIDs, accelerometers or mobile phones to capture data from the real world for further processing and *vision based* systems which use video footage taken by various cameras. Sensor based methods have the disadvantage of being obstructive; they are not capable to monitor an agent who does not possess the necessary equipment. On the other hand, cameras are often limited to small environments. Therefore the choice of sensing equipment is clearly application specific. Since in both cases there is a significant amount of noise in the input stream, further data analysis (*i.e.* learning and inference stages) are typically carried out with the aid of statistical techniques.

A problem encountered often during data acquisition is whether the chosen data form can sufficiently represent the monitored activity. For example, video or sensor data cannot be used to monitor a discussion, since a large amount of important data can only be recorded in the audio stream. The gap between the real world activity and its representation in terms of captured data related to the activity is known as sensory gap [Schooler et al., 1993, Smeulders et al., 2000, Xie et al., 2008]. A common way to handle insufficient data representations is to use a combination of sensors to record an event. The disadvantage of this practice is the associated technical difficulties such as data synchronisation issues and increase of the amount of data storage space required for the recording of the event.

This Thesis investigates the analysis of complex human behaviour taking place in constrained, indoors, office-type environments. The methods and algorithms proposed in this work are designed to monitor procedures performed by professionals, carried out normally without the use of sensors or other obstructive equipment which could affect their performance. Therefore the activities performed by humans are captured using video cameras. However several of the activities investigated involve cognitive tasks [Clark et al., 2008] which cannot be captured in video stream or in any other type of stream resulting from recording using conventional types of sensors (*e.g.* microphones, RF-IDs). To solve this problem, a new method is introduced in this Thesis which involves recording human's interactions with specialised developed software. Thus the data content representing activities comprises two concurrent streams, each resulting from a different type of sensor. The first stream is video and the second is computer-based human

generated content.

1.1 Activity analysis in multimedia streams

The work presented in this Thesis investigates analysis of activities occurring in concurrent multimedia streams which result from video and computer-based human generated content. The focus of this Thesis is on the general problem of event analysis in multimedia streams. However, since the second type of data (computer-based human generated content) is unexplored in the field of activity analysis, in the presentation of related research emphasis is placed on work which uses mainly visual information.

Recognising human activities in multimedia streams is a challenging research topic. Its importance is underlined by the large number of application areas which require recognition of activities taking place in multimedia streams such as surveillance, entertainment, human-computer interaction and personal archiving. In simple terms, the problem can be defined as follows: "Given a multimedia stream illustrating one or more agents executing an activity, can a framework be developed which can automatically identify what activity is being performed?" [Turaga et al., 2008]. There is currently no universal solution to this problem [Amato and Di Lecce, 2011]. Many methods have been proposed, each exhibiting success in specific types of activities. Therefore the type of the studied activities determines the choice of the optimal analysis methodology.

Activities in multimedia streams are traditionally classified according to their duration into short actions or snapshots (*e.g.* grab, kick), simple but periodic actions (*e.g.* running), complex activities, consti-

tuting of simpler actions (*e.g.* playing tennis, preparing a meal), events, comprising activities (*e.g.* football match) and long-term events (building construction) [Niebles et al., 2010].

Activities longer than snapshots can be efficiently represented as sequences of their constituent actions [Hamid et al., 2007]. For short activities, it is observed that the temporal order of the actions constituting an activity becomes more immutable. For example, the actions comprising the complex activity of a *long jump* (running, jumping and falling) cannot be executed in a different order. On the contrary, when activity duration increases, leading to long-term events, it is evident that the activities become more variable. For instance, in the activity *prepare meal*, a human might decide, *e.g.* to cut bread first and then put vegetables in the casserole; another might execute these activities in the opposite order and a third might completely omit the first step or the second. This *variability* makes activities of this type more difficult to model, which explains why most of the activity analysis frameworks presented to date focus on monitoring relatively simple tasks.

1.2 Problem statement

This thesis attempts to bridge the gap between the methods of analysing simple everyday activities and long-term events. More specifically the focus of this work lies on events which comprise a large number of steps and these steps can be executed in a plethora of ways. Such activities occur in a large number of diverse contexts. A surgeon performing an operation, a patient performing the task of calibrating a blood glucose monitor or an engineer working at a study desk are some examples of such activities. Often more than one multimedia streams are required

in order to efficiently model these activities. The work presented in this Thesis focuses on building a framework capable of representing and learning such activities, which will be referred to as prolonged, composite activities. The purpose of the research presented here is to use the learned models to identify activities in new data streams. Furthermore, an important feature of the methodology developed in this work is the proposed system's ability to discriminate between correct and erroneous executions of the same task.

It has to be noted that there is a number of open challenges related to human activity recognition. For example, if the problem is viewed from the visual information perspective, robust object recognition/detection and tracking are tasks which are necessary for the feature extraction phase. The work presented in this Thesis does not attempt to tackle such problems; instead, it relies on current state-of-the-art solutions to handle these. Therefore this Thesis focuses on *representation and modelling of prolonged, composite human activities.*

1.3 Context and motivation

The work presented in this Thesis is a result of collaborative research between the schools of Engineering, Computer Science and Psychology of Cardiff University. The initial idea was to study the psychological implications of engineering design decision making. In the first phase of the project, case studies were designed which would enable the investigation of the engineer's thought process. It was decided that observational methods would be used to study the problem. This involved designing realistic engineering tasks, recording the behaviour of subjects while attempting to solve these tasks using various media

types and analysing their behaviour. The goal, from the psychological perspective, was to study the cognitive processes which lead to mistakes, impasse and insight. If the mechanics behind these processes are understood, psychological methods can be developed which could potentially help the engineer avoid mistakes and move rapidly from impasse to insight. From the engineering/computer science perspective, the goal was to develop methods, based on artificial intelligence principles, which would model the behaviour of the engineers taking part in the case studies for the purpose of detecting patterns of correct and erroneous executions of the given tasks.

As the project progressed, it became apparent that the artificial intelligence techniques developed were suitable for modelling and analysing, not only engineering tasks, but a variety of complex human activities; thus, emphasis was given in applying these techniques to several scenarios to show that they can generalise to solve different types of problems. The result of the research from the engineering/computer science perspective was an artificial intelligence framework capable of recognising complex human activities arising in multimedia streams using computer vision/machine learning techniques.

1.4 Definitions

For consistency, a list of key terms is presented here that will be used throughout this article. It is inspired by the terminology proposed in [Hamid et al., 2007]:

Key-object: An object present in a study scene, providing functionalities required for the execution of various interesting processes or operations in the scene. In the work presented in this Thesis, the set

of key-objects in a study scene is known *a priori*.

Action: An interaction amongst a subset of key-objects in the study scene that holds over a finite time period.

Action boundaries: The beginning and ending points of an action in time.

Action primitives: The basic elements used to denote an action. If there are concurrent actions in a task or a dataset, the action primitives are the action's boundaries. If not, the action primitives are the actions themselves.

Key action primitive: An action primitive which is important for the completion of a specific activity.

Common action primitive: An action primitive which is unimportant for the completion of a specific activity or a set of activities.

Activity: A finite sequence of actions. An activity's *start* and *end* points are signaled by special *landmark* actions. Since actions comprise action primitives, the definition of an activity as "a finite sequence of action primitives" is equivalent.

1.5 Project challenges

This Thesis analyses complex human behaviour arising in prolonged, composite tasks such as engineering design or calibrating a blood glucose monitor. Such tasks usually require a long time period to complete. The "long period" is a factor which increases the difficulty of analysing activities associated with these tasks and *distinguishes* the work presented in this Thesis from other approaches in analysis of complex human activities. Although the activities analysed here normally consist of a number of specific actions, these can be usually carried out

in a large number of ways and in different order. Therefore the "long period" factor implies that a framework designed to analyse prolonged, composite activities should be able to efficiently model complex temporal dependencies between the different steps comprising these tasks.

It is also possible that a human executing prolonged, composite tasks makes a mistake or is unsure about the correct step sequence. Thus, he may retrace his steps or make a few steps towards a potentially wrong direction before returning to the right track. This characteristic implies that there are many different ways to complete such tasks and therefore their structure is not known (or is difficult to be predicted) *a priori*. Additionally, it suggests that temporal dependencies between task steps have a non-local character. This means that although certain steps may have to be carried out in a specific order, they do not necessarily have to be executed one immediately after the other. Hence the temporal model should be able to handle these loose temporal dependencies.

Additionally, during the execution of complex activities certain actions can take place in parallel at the same time. For example in the glucometer calibration task a human might interfere with certain tools participating in the calibration procedure (*e.g.* test strip, blood vial) *while* operating the glucometer. Therefore a framework analysing prolonged, composite activities should be able to handle such *concurrent* actions.

Furthermore, the fact that prolonged, composite activities consist of a number of steps (or sub-activities) suggests that they are characterised by a natural hierarchical structure. Consequently a framework representing such tasks should be capable of capturing this hierarchy.

Additionally, current state of the art complex activity recognition algorithms are only suitable for a limited range of applications. One of the objectives of this Thesis is to present a methodology capable of solving a variety of complex tasks.

In this Thesis non-obstructive data acquisition methods are utilised in order to ensure that humans participating in the experiments of this study work under real life conditions. For this reason video camera is the main sensor used. To record data from cognitive tasks, which cannot be captured using conventional types of sensors, a non-obstructive method is required. Its development is one of the challenges of this Thesis.

State of the art activity recognition methods prove inadequate to efficiently analyse tasks arising in the problem space studied in this work as explained in the literature review (Chapter 2).

The hypothesis of this Thesis is the following:

Prolonged, composite human activities involving cognitive tasks can be sufficiently represented by data captured in a non-obstructive manner. Additionally, activities represented in this data can be modelled and identified using a methodology which combines feature selection, discriminative features and hierarchical statistical graphical models.

The following section discusses the contributions of this Thesis.

1.6 Contributions

Currently there is no model to represent prolonged activities of high complexity like the ones considered in this Thesis. Additionally, in such prolonged, composite activities not all actions are important for correct execution of an activity. A method is needed to identify such

actions automatically and to avoid including them in the models of activities of interest. Moreover, a method to unobtrusively extract cognitive activities is required.

In this Thesis a framework for analysing prolonged, composite human activities is developed, capable of overcoming the deficiencies of existing methods. Activities are represented using a model whose topology and parameters can be learned from data; it is capable of efficiently representing temporal relations between an activity's constituent actions and can handle noisy datasets. Furthermore, the method proposed in this Thesis is capable of capturing hierarchy of complex activities and is designed to work with actions that take place in parallel at the same time.

The contributions of this work are:

- A new feature extraction method which enables automatic construction of action sequences from data arising from multiple streams representing complex human activities is proposed. Contrary to existing methods in the area of complex activity analysis, this representation can model activities whose exact structure is not known *a priori* and can handle concurrent activities. This method first appeared in [Kaloskampis et al., 2011b] and is covered in Chapter 4 of this Thesis.

- A new method for recording cognitive activities, *i.e.* activities which aid in understanding cognitive thought process [Clark et al., 2008]. Central part of the proposed method is a Knowledge Based System (KBS) [Akerkar and Sajja, 2010]. This work was first presented in [Kaloskampis et al., 2011b] and is discussed in Chapter 4 of this Thesis.

- A new classification method, suitable for analysing prolonged, composite human activities, an area where currently existing methods prove inadequate, is proposed. It is based on the combination of Random Forests (RF) [Breiman, 2001] and Hierarchical Hidden Markov Models (HHMMs) [Fine et al., 1998]; combining these methods in the manner proposed in this Thesis allows the proposed algorithm to benefit from their strengths whilst avoiding their weaknesses. This work first appeared in [Kaloskampis et al., 2011c] and is covered in Chapter 5 of this Thesis.

- A method for identifying unimportant and important actions in action sequences arising from the execution of prolonged, composite human activities with the goal of improving classification accuracy, based on the Key Action Discovery concept. This work first appeared in [Kaloskampis et al., 2011a] and is discussed in Chapter 6 of this Thesis.

- An application of the proposed framework to the analysis of the conceptual stage of the bridge design task. This application was first described in [Kaloskampis et al., 2011b] and is covered in Chapter 7 of this Thesis.

The above methodologies are evaluated in scenarios resulting from real-life applications. Later in this work it is shown that the proposed framework can be successfully applied to detect mistakes in a bridge design scenario and the task of calibrating a blood glucose monitor. Additionally, the proposed method is applied to a dataset illustrating everyday human activities with the purpose of identifying these and it is observed that the proposed algorithm achieves state of the art

performance. Thus, the proposed methodology generalises well to solve a wide variety of problems.

The experimental results showed that proposed method compares favourably against state-of-the-art algorithms in the field of activity identification, such as the HHMMs (used for activity identification in [Nguyen et al., 2005]) and the Suffix Trees [Hamid et al., 2007], state-of-the-art classifiers such as RFs and Support Vector Machines (SVMs) [Cortes and Vapnik, 1995] and several classifier combinations (*e.g.* the combination of HHMMs and SVMs). Several of these comparisons were recommended by anonymous reviewers who refereed the publications resulting from this Thesis; all recommended comparisons were carried out.

A list of publications resulting from work presented in this Thesis is given below.

1. Kaloskampis, I., Hicks, Y., and Marshall, D. (2011). Analysing engineering tasks using a hybrid machine vision and knowledge based system application. In 12th IAPR International Conference on Machine Vision Applications (MVA), volume 1, pages 495-498, Nara, Japan.

2. Kaloskampis, I., Hicks, Y., and Marshall, D. (2011). Reinforcing conceptual engineering design with a hybrid computer vision, machine learning and knowledge based system framework. In 2011 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pages 3242-3249, Anchorage, AK, USA.

3. Kaloskampis, I., Hicks, Y., and Marshall, D. (2011). Automatic analysis of composite activities in video sequences using key ac-

tion discovery and hierarchical graphical models. In Proceedings of 2nd IEEE Workshop on Analysis and Retrieval of Tracked Events and Motion in Imagery Streams (IEEE ARTEMIS 2011), pages 890-897, Barcelona, Spain.

In the next section the outline of this thesis is presented.

1.7 Thesis outline

This Thesis is structured as follows: the main approaches relevant to the work presented in this Thesis are reviewed in Chapter 2. A brief overview of the proposed system is given in Chapter 3. Then the proposed framework's main components are explained, namely the Data acquisition unit in Chapter 4 and the Machine Learning component in Chapters 5, 6. Performance of the proposed system is assessed in Chapter 7, where testing methodology is first described and then the proposed system's performance is assessed on a real life complex civil engineering task. This thesis is concluded in Chapter 8, where plans for future work are also presented.

Chapter 2

RELATED WORK

In this Chapter the main research approaches relevant to the work presented in this Thesis are investigated. First the main aspects of activity analysis in multimedia streams are discussed (Section 2.1). This discussion aims at identifying open questions and problems in this research area which will be investigated in this Thesis. Then focus turns to investigation of techniques relevant to the design of algorithms capable of tackling these problems. In the field of activity analysis, algorithms typically comprise two phases, feature extraction and activity modelling. Thus previous work relevant to each of these phases will be critically reviewed in individual sections. Work related to the feature extraction phase is investigated in Section 2.2; the activity modelling phase is covered in Section 2.3.

2.1 Activity Analysis in Multimedia Streams

In this section three important aspects of activity analysis in multimedia streams are discussed. The first of these aspects is the data form in which the media are recorded. It is crucial for the performance of an activity analysis algorithm since it influences the fidelity with which a real life event is converted to a data form. Any detail of the event missing from the resulting data stream will be unavailable during

the analysis procedure and could possibly lead to less accurate results. Media form can also impact the algorithm's design since processing of complex streams may result in technical difficulties such as stream combining and synchronisation. The second aspect is the type of the activity analysis task which the developed algorithm will handle, as it influences the nature of the problem. The final aspect is the complexity of the activities to be modelled by the algorithm, which is related to their duration.

2.1.1 Media data forms

A factor that greatly influences the approach chosen to tackle an activity analysis problem is the form of the data which will be used for solving the problem. In [Xie et al., 2008] possible data forms are classified into four categories, which are data from a single stream captured in one continuous take, multiple concurrent streams, single stream captured in multiple takes and media collections.

The first category includes data from a single stream, captured in one continuous take. In this case data is captured using a single sensor (camera, microphone *etc.*). In practice this data form is the one encountered most often and can result from single camera surveillance which can be static [Lama et al., 2013, Bodor et al., 2003, Wren and Pentland, 1998, Beymer and Konolige, 1999] or moving, *e.g.* mounted on an unmanned aerial vehicle (UAV) [Manohar et al., 2006]. Since there is just a single stream data processing is simple as all problems related to syncrhronising data streams are avoided. However, there are often cases where a single stream cannot capture sufficient information for activity detection in the scene *e.g.* when a single view camera is

blocked by an obstacle. Therefore alternative media data forms are often considered.

Multiple concurrent streams is the second category of data forms used in the data acquisition phase of activity recognition. This form enriches available information as not only it can offer a new perspective (*e.g.* an additional viewpoint in visual surveillance) but also useful complementary information not detectable by a single data type. For example conversation recorded in an audio track can reveal information not visually observable. Thus combination of concurrent audio-visual data is frequently used in activity identification. In [Chen et al., 2004] multiple cameras and audio tracks are used for nursing home surveillance; in [McCowan et al., 2005] audio-visual information is exploited to analyse multimodal group actions in meetings. Other data type combinations can also be used for activity identification: in [Shi et al., 2004a] visual information is complemented by an RS232 stream from a medical instrument for identifying erroneous executions of a glucometer calibration task; in [Krahnstoever et al., 2005] video stream is combined with RFID to detect activities relevant to the retail domain such as a customer's interactions with products to prevent shoplifting. Using multiple concurrent streams can also enhance the information gathered from a scene by opening the way to certain technical possibilities. For example, employing two appropriately placed and calibrated cameras enables 3D scene reconstruction [Shcherbakov, 2009]. Thus depth information becomes available for scene objects, enhancing the accuracy of extracted information from video streams.

The third category concerns a single stream captured in multiple takes. One example of this media form is TV broadcast, where dif-

ferent video content (resulting from different shows scheduled in a TV program) appears sequentially in a unified stream. The difficulty of processing such streams results mainly from the boundaries of the takes, which cause discontinuities in time and space in the stream [Xie et al., 2008]. These discontinuities often cause problems to standard algorithms (*e.g.* tracking and object detection in computer vision) which rely on scene specific training data. Examples of research using this media type form for activity recognition are [Laptev et al., 2008], which introduces a dataset resulting from a collection of movies and the TRECVID project [Over, 2013] offering collections from news, documentary *etc.*

Media collections is the final media data form discussed in this section. This type concerns loosely related data collections resulting from various sources. An example of this data form is data resulting from forensics applications, which can contain diverse data types such as images, video/audio streams and documents [Krusse and Heiser, 2001]. The main challenge of activity identification in this category is the spatio-temporal correlation of the data. In [Naaman et al., 2004, Naaman et al., 2005] this problem is tackled by using geographical and time meta-data automatically assigned to imagery by the capturing device in order to generate label suggestions for unknown person identities in images.

Cognitive tasks

For certain types of human activities existing data acquisition methods are not capable of recording sufficient information to represent them. One example is activities which include cognitive tasks, *i.e.* actions

which aid in understanding cognitive thought process [Clark et al., 2008]. An example of a cognitive task could be selecting to follow a particular step of an industrial process and making a decision [Merriënboer, 1997]. Cognitive tasks include the perceptual, cognitive and motor demands of the studied activity [Remington et al., 2012]. Such tasks cannot be directly observed in video streams. Previous work in analysing cognitive tasks has used the think-aloud protocol [Lewis, 1982] which involves humans verbalising their thoughts while performing an activity. However, this method has three problems: firstly there is the possibility that thought verbalisation deteriorates human's performance [Lane and Schooler, 2004], an effect known as verbal overshadowing [Schooler et al., 1993]; secondly processing data acquired using this method is tedious and thirdly thought verbalisation during task execution is an unnatural process and though useful for understanding human cognition it cannot be employed in practical applications.

Current practice recommends, when dealing with activities which include cognitive tasks, the following procedure. First, a task analysis is carried out [Kirwan and Ainsworth, 1992] which reveals the primitive actions [Kieras, 1994] in the studied domain. These primitive actions are observable; for example, in the case of analysing the cognitive processes in the driving domain, task primitives could be actions related to steering wheel, brake and acceleration [Remington et al., 2012]. In the case of the ATM simulation task using computer software, task primitives are mouse clicks, mouse movements *etc.* [John et al., 2002]. Then, these primitive actions are linked to the covert cognitive actions which occur in the studied task. The cognitive actions cannot be observed and therefore have to be inferred from the theory of the studied

task. The main drawback of this approach is that there is no reliable method to determine the assignment of primitive actions to cognitive actions [Remington et al., 2012].

Therefore a new method is required for acquiring data describing cognitive tasks. Ideally, this method should avoid the problems associated with the think-aloud method and the ambiguities arising from the assignment of primitive actions to cognitive actions.

2.1.2 Type of activity analysis task

There are two main tasks of activity analysis frameworks in media data streams. The first task is detecting known activities and the second discovery of unknown activities. Both are presented below.

Detecting known activities

An important task is identification of an *a priori* known activity pattern in the data stream. For example, to detect a penalty kick in a video stream illustrating a football match. In this case the usual approach is to build a model of the known activity pattern using training data and apply this model to the video stream in order to detect instances of the modelled activity. This task can be applied in several problems such as stream annotation (place labels to the data stream), information retrieval (retrieve information relevant to a query from data) and verification (confirm a property in the data, *e.g.* whether a medical process was carried out correctly or erroneously). Several examples of known activity detection applications are described below.

Intelligent environments. Intelligent environments are spaces equipped with sensors to capture human activity. An artificial intelligence sys-

tem analyses the captured data and generates an appropriate response. This response varies with the application type. In [Yu et al., 2010] a fall detection system for the elderly in an intelligent room is presented. Human behaviour is analysed using the person's head velocity and shape as features; when a fall is detected the system notifies the nursing personnel. A smart clothes application based on activity recognition is presented in [Pentland, 1998]. Their implementation includes a camera mounted on a baseball cap. The camera is paired with an American Sign Language recognition facility based on hand gesture recognition which operates on a 40 word vocabulary with 97% accuracy.

Content based analysis and retrieval. The evolution in compression and streaming technologies have significantly increased our everyday exposure to media content. This development has initiated the need for a new category of applications which offer organisation of the digital media library. A fundamental problem is to detect interesting events in the media stream, such as a successful shot in a basketball game. In [Chang et al., 2001] a framework for filtering streaming videos illustrating sports activities is presented, capable of identifying canonical views (*e.g. serving* in tennis). Detecting interesting events in movies, such as *kissing, answering phone* and *hugging* is the objective in [Laptev et al., 2008]. An automatic football match analysis framework is introduced in [Xie et al., 2004] which can identify structural elements of the game such as *play* and *break*.

Surveillance - known activity detection. Surveillance is one of the traditional areas where human activity identification frameworks are applied. Advances in sensors technology have increased the quality of resulting media streams thus surveillance techniques are becoming

more accurate. Also, current security needs dictate the use of an increasing number of sensors, which translates into a growing need for automated surveillance. The problem of human activity recognition for surveillance is defined as follows: a human operator oversees activity captured by a set of appropriately placed sensors. The operator wishes to automatically identify in the video footage various events of interest. In this section detection of known events is covered and discovery of suspicious activities is reviewed where discovery of unknown activities is discussed (Section 2.1.2). In industrial applications, events of interest could be disruptions in a manufacturing work flow. A dataset for work flow recognition in industrial environments is presented in [Voulodimos et al., 2012] comprising video sequences taken from the production line of a major vehicle manufacturer. In medical applications interesting events could be the phases of the surgery and mistakes or omissions during a process. Operating room work flow is studied in [Padoy et al., 2009] where the task is to identify the stages of a surgery, such as *anaesthesy control* and *surgeon preparation*. The glucometer calibration process is investigated in [Shi et al., 2004a] where the goal is to verify correct task executions and detect mistakes in the procedure. The glucometer calibration dataset is publicly available and used in this Thesis for testing the proposed algorithms.

Discovering unknown activities

The second prominent task is to discover events in media streams without any *a priori* knowledge of their semantics. This case is similar to the clustering problem in machine learning [Duda et al., 2000] and is solved by forming clusters of natural groupings of the input data. The

natural groupings result using the regularity or self-similarity of event instances [Xie et al., 2008] *e.g.* a news broadcast takes place at approximately the same time every day.

Surveillance - unknown activity discovery. In security applications the task is to detect abnormal (anomalous) or suspicious behaviour. The problem is formulated as follows: given a media stream a surveillance system has to locate the position (in the case of a video stream the position is defined by the video frame(s) and the area within the video frames) that an unusual event takes place. The question that the machine has to answer is "what is abnormal behaviour?" If the answer to the question is that "abnormal behaviour is a type of behaviour which is not usually encountered in some context (where context could be, for example, airplane docking, surgery, *etc.*)", the problem is converted to statistically modelling the *usual* behaviour observed in this context. Then abnormal behaviour is defined as the examples which do not fit the model of usual behaviour. These examples are often referred to as *outliers*. The problem of airport surveillance is studied in [Vaswani et al., 2005], where trajectories of passengers are analysed to detect abnormal behaviour. A framework for bicycle theft detection is proposed in [Damen and Hogg, 2009] which is based on recognising linked events in video. Suspicious behaviour arising in a delivery vehicle loading dock is detected in [Hamid et al., 2005] with the aid of bags of event n-grams. The choice of abnormality detection method is application related. It depends on the availability of labelled data, the desired accuracy and how dissimilar are the outliers to the usual behaviour model.

In this work tasks of the first category are studied, where the analysed activities are known. More specifically, the problem of verification

is investigated *i.e.* models of known patterns are learned from labelled training data and are then applied to novel examples in order to determine whether they match the properties of the learned known patterns.

2.1.3 Activity duration

The duration of the studied activities is a factor which defines to a large extent the methodology which will be employed to study them. The main reason behind this is that duration is indicative of an activity's complexity, although there are exceptions to the rule *e.g.* a marathon run lasts longer than a football game but it is much simpler to model. In [Niebles et al., 2010] activities are classified according to their duration into (Figure 2.1):

- **Short actions** or **snapshots** (*e.g.* grab, kick). These usually last for short durations of time on the order of tens of seconds [Turaga et al., 2008]. Representative approaches for analysing short actions are featured in [Gupta et al., 2009], [Ikizler and Duygulu, 2009] and [Yao and Fei-Fei, 2010].

- **Simple but periodic actions** *e.g.* running, walking, swimming *etc.*. Methods for analysis of simple actions can be found in, *e.g.* [Efros et al., 2003] and [Laptev et al., 2008].

- **Simple activities** *e.g.* high jump, constituting of simpler actions and lasting for durations on the order of ten seconds. [Ikizler and Forsyth, 2007] and [Laxton et al., 2007] are two examples of methods proposed to analyse such activities.

- **Intermediate activities**, constituting of simpler actions but lasting for durations on the order of 10^3 seconds *e.g.* playing

(a) (b) (c)

(d) (e) (f)

Figure 2.1: Human activities can be classified according to their duration into **(a)**: snapshots (*e.g.* kicking), **(b)**: simple but periodic actions (*e.g.* walking), **(c)**: simple activities (*e.g.* high jump), **(d)**: intermediate activities (*e.g.* meal preparation), **(e)**: events and complex activities (*e.g.* rugby match), **(f)**:long-term events (*e.g.* building construction).

tennis, preparing a meal. Such activities were studied in [Sridhar et al., 2010] and [Nguyen et al., 2003].

- **Events and complex activities**, comprising activities *e.g.* football match. [Xie et al., 2002] and [Kuettel et al., 2010] are examples of event modelling.

- **Long-term events** *e.g.* building construction. Such events can last 10^{7-8} seconds and they have not yet been studied by any computer vision framework.

A special case of events are prolonged events, which typically comprise a large number of steps and these steps can be executed in a plethora of ways. This characteristic makes the structure of such events challenging to model. In simpler events, like a sports match certain constraints (*e.g.* laws of physics) simplify the prediction of the next step in an action sequence given the current step. For example, in a tennis game consider the state of the ball hitting the court. This state can only be followed by two possible states: either the ball is hit by a player or a point is scored. In the activity *high jump*, action *jump* is always followed by action *fall*. The temporal relationships between the actions comprising such simple activities have *local* character which means that the temporal dependencies between their actions are short-termed. Constraints apply in prolonged events, too; however these are much less restrictive than those encountered in normal events. For example, during meal preparation, the human opens the fridge. Eventually the fridge will be closed, which implies a temporal dependency between actions *open fridge* and *close fridge* but in the meantime a variety of random actions can take place. This temporal dependency is more relaxed compared to the table tennis and "jump-fall" examples. Prolonged events are time sequences whose temporal dependencies between their elements are long-termed and have *non-local* character. An aspect of such events is that their execution requires a certain level of expertise by the human who performs them. Sample applications for the proposed framework is a doctor performing a surgery, a patient calibrating a medical device or an engineer working on a design study.

A note on activity complexity

It was mentioned earlier that the duration of an activity is indicative of its complexity. This is an empirical approach to indirectly define complexity, which is accepted by the machine learning community (*e.g.* [Turaga et al., 2008, Niebles et al., 2010]) because of its simplicity. However, there is a huge body of literature on defining activity complexity, most of which stems from the domain of behaviour analysis. There are two definitions which are most widely used, as pointed out in [Gill and Hicks, 2006]. The first was proposed by Wood [Wood, 1986] and the second by Campbell [Campbell, 1988].

Wood [Wood, 1986] identifies three main sources of complexity, which are: the number of distinct steps required for the completion of the task, the form of the relationships between these steps and the evolution of the task's objectives during the execution of the task.

Campbell [Campbell, 1988] proposes four characteristics which contribute to complexity: the existence of multiple ways to complete the task, the existence of multiple desired outcomes, the potential conflicting interdependence among task objectives and the uncertain or probabilistic linkages between potential path activities.

Commenting on these two popular approaches, it is apparent that the sources of complexity considered in both of them imply an increase in information load, information diversity or information change. An activity becomes more complex as the number of the steps required for its completion increases and/or if its goal can be achieved using several different paths. From this perspective, the empirical taxonomy based on activity duration, which is used by the machine learning community, appears to be a reasonable heuristic: longer duration usually means an

increase in the number of the required steps and the possible paths to the task completion. Additionally, the evolution of the task's objectives is a characteristic of more time consuming, long-term activities.

A special reference is given now to the source of complexity described as "uncertainty" by Campbell. In [Campbell, 1988] the way that "uncertainty" affects complexity is explained: existence of probabilistic linkages increases information load and diversity, because candidate solution paths cannot be ruled out quickly; also, the number of paths to the desired outcome increases. This description is in line with the taxonomy based on activity duration: to model activities of equal or greater complexity than *simple activities*, a probabilistic framework is normally employed. In fact, Charniak and Goldman [Charniak and Goldman, 1993] argue that any complex activity analysis system which does not include any facility to handle uncertainty is inadequate.

It was mentioned earlier in this section that the activities studied in this thesis usually require a certain level of expertise by the human who performs them. It has to be noted that, as pointed out in [Kishore et al., 2004], when a certain degree of familiarity and experience with a task exists, such that the likelihood of successfully completing the task is high, the task's uncertainty (and therefore, its complexity) is low. The task then becomes similar to "routine tasks" which are regarded as non complex [Jehn et al., 1999, Schwarzwald et al., 2004]. However, the human's familiarity and expertise with the task do not necessarily make the task easier. Returning to Campbell's definition, the potential conflicting interdependence among task objectives is often present in tasks which require expertise; in such cases the task is complex, by Campbell's definition, as candidate solution paths cannot be ruled out

quickly and/or they need to be studied at a deeper level to be evaluated. An example is the game of chess: despite of the fact that there exist thousands of books about the theory of the game, the game is highly complex. Furthermore, there are cases in which, although the human who performs the task is proficient, the task is of high complexity because of the continuous flow of new information (*e.g.* air traffic controller simulation task) [Schwarzwald et al., 2004].

In this work the heuristic taxonomy of activities according to their duration, which is accepted by the machine learning community, is used. As mentioned earlier, it is only indicative of the complexity of an activity. There are cases where the taxonomy does not work (*e.g.* sleeping can last for a long time but it is easy to model). In [Sahaf et al., 2011] it is suggested that other factors should be taken into account along with activity duration, such as the human's participation during the activity and the presence of repetitive patterns. Regarding the concern that the human's expertise can transform a seemingly complex task into a routine task, care must be taken when a task is chosen so that there exists conflicting interdependence among task objectives and/or flow of new information during the execution of the task to ensure that the task is complex. Both of these requirements were taken into consideration in the task studied in chapter 7 of this thesis.

Types of activities studied in this work

Potential applications for the framework proposed in this thesis are, as mentioned earlier, a doctor performing a surgery, a patient calibrating a medical device or an engineer working on a design study. These tasks are, however, very different. The calibration of a medical device is a

purely procedural task. Given instructions, it is doubtful that the user will make any mistakes. However, as explained in the previous subsection, given that the task might comprise a large number of steps and last for a long period of time (as explained in the previous subsection, both of these factors increase uncertainty), the challenge may lie in modelling all the correct ways of executing the task. Also, a surgery can be seen as a procedural task, however, like in the case of the air traffic controller simulation task, the continuous flow of new information (*e.g.* complications during the procedure) increases the difficulty of modelling the activity. Regarding engineering tasks, it first has to be clarified that those studied in this thesis have a general (although usually "loose") structure which stems from the regulations and standards which have to be followed when these tasks are carried out. The complexity in these tasks stems from the potential conflicting interdependence among task objectives. This characteristic makes such tasks difficult to model as many iterations may be required until an acceptable solution is reached. Sports events and games is another type of applications in which the proposed framework can be applied. Such events also have some type of structure; in [Bettadapura et al., 2013], for example, football (soccer) is structured using transitions between different zones of the field.

Despite their differences, the activities studied in this thesis are similar in that they have some type of structure. Additionally, they include sources of uncertainty, such as a large number of steps, continuous flow of new information, conflicting interdependence among task objectives *etc.*. Frameworks designed to model simpler activities are not suitable to handle composite, long term activities as shown in the

results sections of this thesis.

2.1.4 Methodology of event analysis problems

The methodology applied to any event analysis problem typically comprises two stages: first, *interesting features* are extracted from the video footage. In the second stage these features are analysed at a high level and a complex human behaviour model is built. This model is then used to assign novel sequences to behaviour classes according to their context. Review of related work will focus on the investigation of methods previously applied in these fundamental stages. Thus, the choices made later on the selection of components which comprise the framework proposed in this Thesis will be justified.

Feature extraction techniques are covered in Section 2.2. The main approaches in the research area of complex activity identification are presented in Section 2.3.

2.2 Extracting features from video footage

As stated in [Wang et al., 2009] there are two approaches to feature extraction from video footage:

1. Objects of interest in a scene are detected (automatically or manually) and tracked; then their tracks exploited to understand activities (*e.g.* [Nguyen et al., 2005]).

2. Use of motion feature vectors instead of tracks (*e.g.* [Laptev et al., 2008, Niebles et al., 2010]).

The following Subsection 2.2.1 discusses literature related to the trajectory manipulation approach. Literature relevant to motion feature

vectors is covered in Subsection 2.2.2.

2.2.1 Key object detection and tracking

Methods of the first category can efficiently analyse relatively simple scenes *e.g.* where the video stream is captured by a single view camera and the set of key objects participating in the scene is known *a priori*. Such scenes include desktop activities (*e.g.* calibration of a medical device [Shi et al., 2004a]), everyday human activities (*e.g.* shopping [Xiang and Gong, 2006]), traffic control [Fernyhough et al., 2000, Wang et al., 2006], parking lot surveillance [Johnson and Hogg, 1996] and monitoring of elderly people in smart environments [Truyen et al., 2006]. In [Xiang and Gong, 2006] the disadvantages of such approaches are discussed. These disadvantages are given below in free interpretation.

1. Since the method relies on constantly tracking *key* objects in a scene, it is difficult to apply in video streams captured by low resolution CCTV surveillance cameras which might provide insufficient accuracy, especially in cluttered scenes.

2. In busy everyday scenes object tracking might be interrupted by occlusions resulting to potentially unusable tracks.

3. In certain cases, trajectories of moving objects cannot capture sufficient information to identify a human activity. For example, in a kitchen environment a person might walk towards the fridge, check if a certain product is present and then leave through the door or they might pick up an item from the fridge (*e.g.* a drink) and again leave the kitchen through the door. If the person's

movement trajectory is solely monitored, it is not possible to discriminate between the similar activities *check fridge* and *pick up an item from fridge*.

To these issues the fact that these methods heavily rely on the performance of the object tracker has to be added.

In view of these disadvantages, several researchers adopt an alternative methodology to exploit tracks resulting from the movement of key objects. Specifically, rather than quantitatively representing an object's trajectory (*i.e.* taking as features the trajectory's coordinates) features emerge by taking into account certain occurrences resulting from the object's interaction with other key objects in the scene. In [Hamid et al., 2009] these occurrences are an agent's interactions with kitchen facilities (such as stove, fridge, *etc.*). In this case, an extracted feature, $a(x)$ is in the form of $a(x) = \{agent\ interacts\ with\ facility\ x\}$ where x the code of a facility. Note that here the extracted features have qualitative rather than quantitative character. This idea is further investigated in [Sridhar et al., 2008] where features result from qualitative spatial relations between objects. Based on Allen's Interval Algebra [Allen, 1983] and Qualitative Primitives [Cohn et al., 2003] this work captures features resulting from the type of interaction between objects. Examples of features captured in this manner are $a_1 = \{object_x\ surrounds\ object_y\}$ and $a_2 = \{object_x\ touches\ object_y\}$. Another example of this methodology is [Nguyen et al., 2005] where the floor of a room is divided to a number of square segments. In this case, features result from the presence of an agent within the limits of a segment and are in the form $a(x) = \{agent\ in\ segment\ x\}$, where x is the code of a segment. Qualitative spatial relations have the advantage

that momentary loss of an object by the tracker does not necessarily render the track useless due to the qualitative rather than quantitative representation of the the trajectory. Therefore the requirements for perfect tracking and high quality data stream are relaxed. However, some problems still remain. Specifically, this method cannot discriminate between spatially similar actions such as writing and sketching which involve the same objects (hand, pencil and paper) in the same spatial setup (hand holding pencil over paper).

Therefore trajectory-based activity recognition might not be the optimal solution for some applications. Researchers also investigated techniques in which key object detection and tracking is not required. These approaches are reviewed in the next subsection.

2.2.2 Motion feature vectors

An early example of a motion feature vector approach is [Bobick and Davis, 2001] where temporal templates for simple human actions (*e.g. sit down, wave arms, crouch down*) are learned from labelled video sequences and are then used for identification in novel videos. During learning it is assumed that the object whose motion is used to construct a temporal template can be separated from the background. On the contrary this restricting assumption is not required in [Shechtman and Irani, 2005]. In this work event detection is achieved with the extension of the concept of 2D image correlation (*i.e.* 2D template matching) to 3D space-time video template correlation. With this method, behaviour patterns illustrating atomic actions (*e.g. walking, pool dive, ballet turn*) can be detected in long video sequences or video databases. The advantages of this approach is that no background-foreground segmentation

is required and that it can be used to detect actions which occur simultaneously. The disadvantage of this method is its sensitivity to large geometric deformations of the video template.

The popularisation of local feature representation for images (using algorithms such as SIFT [Lowe, 1999], SURF [Bay et al., 2008] and GLOH [Mikolajczyk and Schmid, 2005]) inspired the extension of this concept to space-time representations. In the same manner that local feature-based techniques represent an image as a vector of 2D interest points, space-time feature methods model a video as a vector of local 3D volume features in a space-time scale. One of the first attempts to model video sequences with the aid of such methods is [Chomat and Crowley, 1999] where local spatio-temporal features representing an activity are captured with the aid of motion energy models. In [Zelnik-Manor and Irani, 2001] *events* in a video sequence are modelled as local features captured at various temporal scales. Events that have similar local feature distributions at corresponding temporal scales are considered as similar. The advantage of the method is that no prior knowledge concerning the model events is required. An extension of the Harris and Föstner interest point operators [Harris and Stephens, 1988, Föstner and Gülch, 1987] is proposed in [Laptev and Lindeberg, 2003] to detect spatio-temporal interest points. The idea of local scale selection in the spatial and temporal domain [Lindeberg, 1998, Lindeberg, 1997] is used in this work to define the spatio-temporal extent of an event. Based on Lowe's remark that, although in certain cases feature sparsity might be desirable, excessive rarity of features might cause problems to a recognition framework [Lowe, 2004], work in [Dollar et al., 2005] proposes a local feature detection algorithm designed to produce more

features compared to [Laptev and Lindeberg, 2003]. Their goal is to capture subtle movements such as the jaw of a horse chewing on hay and the spinning wheel of a bicycle. Features are represented with cuboids containing spatio-temporally windowed pixel values and they apply their method to detect facial expressions, human behaviour and animal (mouse) activity (*e.g. drink, eat, sleep*).

Several researchers enrich the feature selection stage by investigating dependencies between extracted low level features. In [Niebles et al., 2006] features are extracted using the methodology of [Laptev and Lindeberg, 2003] and then the correlation between extracted features is investigated: a codebook is formed by clustering extracted local features with the k-means algorithm [Duda et al., 2000]. In this codebook, the center of each resulting cluster is defined as a codeword. Probabilistic Latent Semantic Analysis (pLSA) [Hofmann, 1999] is used to learn action models. The pLSA model cannot capture structural information (*i.e.* geometric relationship between extracted local features); thus in [Wong et al., 2007] the pLSA-Implicit State Model (pLSA-ISM) [Leibe et al., 2005] was employed to encode such information, offering improved performance over the pLSA algorithm. Performance of spatio-temporal features in realistic videos is tested in [Laptev et al., 2008] where a variation of [Laptev and Lindeberg, 2003] is used to learn and detect human actions in videos extracted from popular movies. In contrast to [Laptev and Lindeberg, 2003], a multi-scale approach is used and features are extracted at multiple levels of spatio-temporal scales. To tackle the problem of feature sparsity [Gilbert et al., 2009] use the 2D Harris corner detector for interest point localisation. Once extracted, local features are grouped hierarchically which speeds up

the classification process and improves accuracy. Based on the observation that most approaches using local spatio-temporal features disregard structural information (*i.e.* spatial and temporal relationships between extracted features), work in [Ryoo and Aggarwal, 2009] proposes the *spatio-temporal match* algorithm which compares temporal relationships (*e.g.* *before* and *during*) and spatial relationships (*e.g.* *near* and *far*) between extracted local features. This approach can successfully recognise two-person interactions (*e.g.* *shake hand, hug, punch*) as well as standard atomic actions.

A disadvantage of motion feature vector approaches is that are sometimes not able to handle complex temporal relations [Zhang et al., 2011].

2.3 Complex Activity Modelling

In the past, complex activity analysis systems have used pattern recognition techniques to analyse, *e.g.* kitchen activities [Sridhar et al., 2008], card games [Moore and Essa, 2002] and nursing activities [Inomata et al., 2009]. There are currently six approaches to complex activity modelling, specifically grammar-driven representations, vector space models, pattern recognition methods, local event statistic methods, flat statistical graphical models and hierarchical graphical models. These approaches are discussed in the following sections.

2.3.1 Grammar-driven representations

The first approach is based on grammar-driven representations where an activity is modelled as a string of symbols. In this string, each symbol represents an atomic action or primitive behaviour and the ac-

tivity's structure is captured in a set of production rules. An example of such methods are deterministic grammar models, which were used to analyse manipulations (*i.e.* meaningful sequences of body articulations such as *picking up*) in video [Brand, 1996]. Deterministic models have the limitation of relying on perfect low-level sensing [Ivanov and Bobick, 2000] and are therefore not suitable for noisy environments. To avoid this problem, a stochastic model specifically a Stochastic Context-Free Grammar (SCFG) was proposed in [Ivanov and Bobick, 2000] for parking lot surveillance. In SCGFs production rules are associated with probabilities and are therefore better suited for capturing complex tasks than deterministic grammars. Similar methods were employed for monitoring card games, [Moore and Essa, 2001] gymnastics and traffic events [Zhang et al., 2008] and simple human actions (such as *walk, turn, kneel*) [Ogale et al., 2007]. Attribute grammars [Knuth, 1968] is an extension of SCFGs which additionally associate production rules with conditions and can therefore describe features which finite symbols (produced by a purely syntactic grammar) cannot easily represent. Parking lot activity was analysed in [Joo and Chellappa, 2006] with the aid of attribute grammars. Propagation Nets (P-Nets) [Shi et al., 2004a] is a method which is related to grammar approaches. In this framework activity structure is explicitly pre-defined. Such methods cannot be applied when activity structure is not known *a priori*. Wyatt *et al.* [Wyatt et al., 2005] extended P-Nets so that structure is automatically learned from data, however their method is specific to the problem of identifying activities from text corpora such as the web and is therefore hard to generalise for complex activities [Shi et al., 2006]. In general, grammar representations are powerful activity mod-

elling tools but learning production rules is a difficult problem. Expert knowledge is required in order to construct the grammar, as pointed out in [Lavee et al., 2009] and therefore these methods are not suitable when activity structure is not known. For example, in [Cho et al., 2004, Cho et al., 2006, Guerra-Filho and Aloimonos, 2006, Kitani et al., 2007] model parameters are learned for known model topologies.

2.3.2 Vector Space Models

Vector Space Models (VSMs) [Salton et al., 1975, Dubin, 2004], used for street surveillance applications in [Stauffer and Grimson, 2000] is the second class of algorithms widely used to model complex activities. According to this method, an activity is represented as a vector of the frequencies of its constituent actions. This representation has its origins in the field of Natural Language Processing (NLP) (*e.g.* [Lebanon et al., 2007]) and Information Retrieval [Carrillo and Lpez-Lpez, 2010] where the method is often referred to as *Bag-of-words* [Salton and McGill, 1986]. Introduction of this theory in activity recognition signaled the arrival of important analysis techniques such as *tf-idf* weighting and feature selection [Chen et al., 2009] in the field. Temporal relations between actions are ignored in the VSM representation, which means that the model provides no information about the ordering of an activity's constituent actions. Extensions of VSMs using Latent Semantic Analysis (LSA) [Deerwester et al., 1990] and Probabilistic Latent Semantic Analysis (pLSA) [Hofmann, 1999] reduce dimensions of the vector space providing computationally more efficient representations. The main idea behind these methods is that a vector (representing an activity) can be viewed as a mixture of various topics (events). Latent

Dirichlet allocation (LDA) [Blei et al., 2003] is another extension of VSMs, similar to pLSA with the difference that the topic distribution is assumed to have a Dirichlet prior. These extensions, however, like the original VSMs do not encode information regarding the ordering of the vector's elements.

2.3.3 Pattern recognition methods

Application of pattern recognition methods is often encountered in activity analysis. There exist four categories of relevant algorithms which are presented in the following sections.

Parametric approaches

In parametric approaches it is assumed that the model of one or more activities can be satisfactorily approximated with a known distribution (*e.g.* Gaussian or Poisson [Duda et al., 2000]) or a mixture of known distributions (*e.g.* Gaussian Mixture Model [Bishop, 2007]). Then the problem is to define or learn the appropriate model parameters and classify examples according to their proximities to the distributions [Ben-Gal, 2005]. Such approaches often fail to model high dimensional datasets, especially when there is no information regarding the underlying data distribution [Papadimitriou et al., 2003].

Non-parametric approaches

Non-Parametric techniques make no assumptions regarding the underlying distributions of the data. There are mainly two types of non-parametric approaches. The methods of the first type estimate the probability density functions from data. An example of such approaches

is the Parzen-Rosenblatt window method [Rosenblatt, 1956, Parzen, 1962]. The methods of the second type directly estimate the *a posteriori* probabilities, (*i.e.* the probability of a sample to belong to a specific class). A method which belongs to this category is the k-nearest-neighbour algorithm [Duda et al., 2000]. The problem with such approaches is that a large number of samples is required in order to estimate the underlying distributions with accuracy. Therefore they are not suitable for small datasets.

Clustering techniques

Clustering techniques group the samples of a dataset in such a way that elements belonging to the same group (*cluster*) are similar between them and dissimilar to the elements belonging to other clusters. In such approaches, distance metrics are employed to classify novel examples. The main advantage of using clustering techniques is that they don't require training data, which comes handy in cases where labelled data are unavailable or the process of labelling the samples manually is tedious. Examples of clustering methods are k-means clustering [MacQueen, 1967,Steinhaus, 1956], hierarchical clustering [Sibson, 1973] and density-based clustering [Kriegel et al., 2011].

Discriminative feature approaches

These methods assume that the form of the discriminant function is known; parameters of the classifier are learned using training samples. The prominent algorithm of this category is the Support Vector Machines (SVMs) [Cortes and Vapnik, 1995]. SVMs are used often to learn activity models when features are extracted using the motion feature

vector methodology. For example, in [Schuldt et al., 2004] a framework for human action recognition (like *walk, jog, run*) is presented where action features are described using spatio-temporal interest points and action models are build using SVMs. Recently an alternative discriminative feature approach based on ensembles of cascade classifiers is gaining popularity, which is Random Forests (RFs) [Breiman, 2001]. This method has demonstrated better or at least comparable performance to other state-of-the-art methods in both classification [Breiman, 2001, Bosch et al., 2007], real-time keypoint recognition [Lepetit et al., 2005] and clustering applications [Moosmann et al., 2006]. Compared to their competitor, SVMs, RFs have the advantage of offering a variable importance index which reflects the "importance" of a variable based on the classification accuracy, taking into account interaction between variables [Breiman, 2001]. Also, their performance is not sensitive to the values of their parameters [Yeh et al., 2012]. Moreover, RFs extend "naturally" to multiple class problems unlike SVMs [Torralba et al., 2007, Criminisi et al., 2012].

Feature selection

Feature selection is an important aspect of pattern recognition methodology and it involves assessing the significance of extracted features in discriminating between different activities. Features found insignificant, or unimportant, or redundant are eliminated from the dataset. This results in reduction of the problem's dimensionality which translates to higher classification accuracy and computational efficiency. Feature selection is relatively unexplored in the context of activity analysis. In [Baos et al., 2010] a feature ranking technique based on

discrimination and robustness is proposed for the purpose of discriminating between activities such as *standing still*, *sitting and relaxing*, *running* and *walking*. In [Ribeiro and Santos-Victor, 2005] several feature selection algorithms are tested in order to reduce dimensionality in a dataset illustrating human activities *active, inactive, walking, running* and *fighting*. The best results by far were obtained by trying all possible combinations of dataset features rather than using established feature selection algorithms (the established Relief algorithm [Kira and Rendell, 1992] was among the methods tested). This illustrates the fact that feature selection in the activity analysis context is a complex process which is also application related. In [Hamid et al., 2005] a scheme of detecting *deficient* and *extraneous* actions in a dataset is proposed. This scheme aims at defining actions that could help discriminate between two or more activities.

Discussion on pattern recognition methods

Pattern recognition methods are powerful modelling tools which are often encountered in activity analysis algorithms. Out of the approaches mentioned above, parametric and non-parametric approaches are rarely employed in current practice: the former because activity complexity cannot be sufficiently approximated with simple distributions and the latter due to the large number of samples required to estimate the underlying distributions from data. Clustering techniques are usually employed in unsupervised activity analysis tasks, where the classes and their semantics are unknown. On the contrary, in the work presented in this Thesis, the number and type of classes is known. Discriminative feature approaches are techniques often encountered in modern activ-

ity analysis frameworks and will be thoroughly discussed throughout this work as they offer excellent classification accuracy when the number of classes is known. Their drawback is that they do not take into consideration temporal dependencies between features.

2.3.4 Local event statistic methods

Local event statistic methods, such as n-grams [Hamid et al., 2005] and Suffix trees [Hamid et al., 2007], which capture neighbouring temporal relations between an activity's constituent actions is the fourth approach. The algorithms of this category have been successfully applied to everyday activities and anomaly detection problems [Hamid et al., 2007]. The drawback of this approach is that in noisy datasets these neighbouring relations sometimes become less characteristic of the performed activity.

2.3.5 Flat statistical graphical models

The fifth approach is based on flat statistical graphical models, such as the HMM [Yamato et al., 1992, Cielniak et al., 2003]. These dynamic representations model activities as state chains of their constituent actions and encode temporal information in the form of transition probabilities between the elements of the chain. Early work with such models in the field of activity identification focused on recognition of American Sign Language [Costello, 2008] from sequences of hand gestures [Starner and Pentland, 1995] and hand's movement trajectories [Bobick and Wilson, 1997]. Due to scaling issues of the HMM in long sequences [Rabiner, 1989, Bui, 2004] researchers employed several variations of this approach to handle this problem. Methods em-

ployed include the Parallel Hidden Markov Model (PHMM) to model ASL [Vogler and Metaxas, 1999] and the Coupled HMM (CHMM) to model human activities [Oliver et al., 2000]. These methods improved the performance of the original HMM in handling prolonged sequences. A more complex model was introduced in [Natarajan and Nevatia, 2007] specifically the Coupled Hidden Semi Markov Model which outperformed CHMM in the ASL recognition task. One of the drawbacks of such extensions is that by complicating the model topology, standard, exact HMM learning and inference algorithms become inapplicable in these complex structures [Lavee et al., 2009]. Therefore, approximation algorithms are required to solve learning and inference problems when employing such methods. Another disadvantage of flat statistical models is that if the model's topology is fixed, representational capabilities are limited, which can potentially cause problems when modelling *variable* activities [Kaloskampis et al., 2011b]: a variable activity's constituent actions can be changed and actions can be added to the action sequence representing the activity or omitted from it without significantly altering the activity. Such behaviours cannot be efficiently captured using fixed topologies. This deficiency was tackled in [Galata et al., 2001] the Variable Length Markov Model (VLMM) [Guyon and Pereira, 1995] was used to model human exercise activities. The main problems of flat statistical graphical models is that they cannot represent efficiently the natural hierarchical structure of complex activities [Nguyen et al., 2005].

Flat statistical graphical models are related to CFGs as both models are grammar-based. Their difference is that CFGs represent activities by applying a set of production rules on a vocabulary of actions;

on the other hand, HMMs represent activities using a probabilistic model of discrete hidden states, where each state emits actions. In formal language terms, HHMs are essentially regular languages and CFGs context-free languages. Since, according to the Chomsky hierarchy [Chomsky, 1956], a normal language is a subset of a context-free language, CFGs have the advantage in terms of expressive power. However, this comes at the price of lacking efficient mechanisms which would enable learning of the structure of CFGs from data.

2.3.6 Hierarchical graphical models

The sixth class of complex activity models includes extensions of graphical models in a hierarchical manner, such as the Layered HMM [Oliver et al., 2002], the Abstract Hidden Markov Memory Model [Nguyen et al., 2003], the Abstract HMM [Osentoski et al., 2004] and the Hierarchical Hidden Markov Model (HHMM) [Nguyen et al., 2005]; these approaches demonstrate accuracy in modeling complex activities. The differences between LHMMs, HHMMs and AHMMs are the following. LHMMs are essentially cascades of HMMs, with each HMM (layer) operating at a different time scale. There is no dependency between the states of different layers. One problem with this method is that the model is not learned automatically; the number of layers and the time scale represented by each layer are handcrafted by taking into account intuition and domain knowledge provided by human experts. On the other hand, AHMMs and HHMMs are both organised in levels and not layers; dependencies are defined between their states at different levels. The difference between these two models lies in the types of the dependencies defined in each model, as their structures are different. Fig.

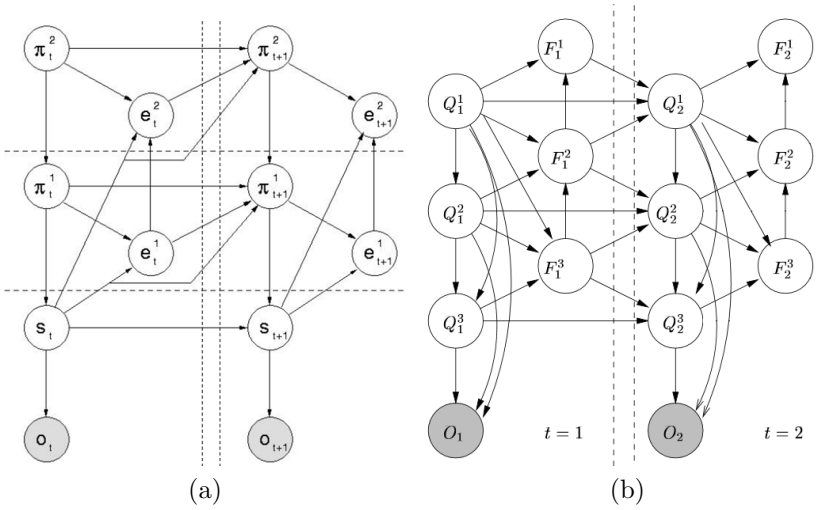


Figure 2.2: Structures of AHMM (modified from [Johns and Mahadevan, 2005]) and HHMM (modified from [Murphy and Paskin, 2001]) models in DBN form. **(a)**: AHMM **(b)**: HHMM.

2.2 shows the dynamic Bayesian network forms of the two models. As stated in [Nguyen et al., 2005] in the case of the AHMM the parameter learning process can become intractable when the number of levels increases. On the contrary, efficient parameter learning algorithms are available for the HHMM, *e.g.* [Nguyen et al., 2005, Murphy and Paskin, 2001]. HHMMs are capable of capturing hierarchy of everyday activities but have difficulties recognising more complex behaviour [Vishwakarma and Agrawal, 2012] and sometimes prove to be sensitive to noise.

2.3.7 Discussion

In Section 1.5 the challenges of analysing prolonged, composite tasks such the ones investigated in this Thesis were given. Current algorithms covered in this Chapter are now discussed in relation to these challenges. The methods listed above prove inadequate to efficiently analyse tasks arising in the problem domain investigated in this Thesis for the following reasons: activity structure is not known *a priori* in

the problem domain discussed in this work, therefore grammar driven representations are not applicable. Vector Space Models ignore temporal relations between actions and are therefore not suitable. Local event statistic methods focus on capturing neighbouring temporal relations between an activity's constituent actions. On the contrary, in the problem space discussed in this work, these neighbouring relations become less characteristic of the performed activity since in many cases "important" actions are preceded or/and succeeded by random actions, which can be thought of as "noise". Pattern recognition methods have the disadvantage that they do not take into consideration the ordering of features and therefore they cannot be readily applied to the problem investigated in this work. Flat statistical graphical models prove inadequate as they fail to capture the natural hierarchical structure of complex activities [Nguyen et al., 2005]. On the other hand, extensions of HMM in a hierarchical manner are capable of capturing hierarchy but prove to be sensitive to noise. Moreover, in all the above approaches (with the exception of [Shi et al., 2004a]) it is assumed that actions constituting activities take place in a sequential manner. Therefore these methods cannot readily handle parallel streams. In contrast, the method proposed in this Thesis is designed to handle *concurrent* actions, *i.e.* actions which take place in parallel at the same time.

In this Thesis, a novel methodology which deals with the shortcomings of the above presented approaches is proposed. In particular, the proposed method can model activities whose structure is not previously known. It is capable of efficiently representing the natural hierarchy of complex activities and encode the temporal relations between their constituent actions. Moreover, it is designed to operate in noisy

environments and can handle concurrent activities.

2.4 Summary

In this Chapter previous research which is relevant to the work presented in this Thesis was discussed. First the problem which is investigated in this Thesis was defined by analysing the main aspects of activity analysis in multimedia streams. Then techniques relevant to the design of an algorithm which will solve this problem were reviewed. Specifically, since activity analysis algorithms usually comprise two phases, feature extraction and activity modelling, the review covered the main trends and developments for these phases.

Chapter 3

SYSTEM OVERVIEW

3.1 Overview

New research presented in this Thesis spans several distinct areas, each contributing to a different part of the proposed framework. This chapter explains the design of the whole framework, capable of analysing prolonged, composite human activities arising in multimedia streams and the interaction between its different parts leaving the detailed descriptions of its individual parts to be presented later in appropriate chapters.

This chapter is structured as follows: the purpose of the proposed framework is first defined in Section 3.2 and a short general description is given in Section 3.3. A list of properties which should be accommodated by a complex behaviour analysis framework are then given (Section 3.4). Then the proposed framework's components are described in detail in Section 3.5. The Chapter is concluded in Section 3.6 in which the work presented here is summarised.

3.2 System scope

The objective of the proposed framework is to recognise prolonged, composite human activities in multimedia streams given a set of ob-

servations. These observations are data acquired from sources such as video footage, sensors, audio *etc.*. The system operates in a *supervised* manner: a part of the acquired data is annotated by experts. This data is used to build (train) models describing the observed activities. These models are then used to identify novel data, *i.e.* data not used in the training process.

3.3 General system description

The activity recognition process is now briefly described. The system first acquires data from multiple, parallel streams and converts them in the form of sequences of discrete actions. Sequences arising from parallel streams are merged in a unified stream which gives an account of what events took place during the observed period and in what order. A number of unified sequences are selected to serve as the training dataset. These are first assigned labels by human experts; the labels are high level descriptions of the activities observed in the sequences. The labelled sequences are then used to build activity models through a three-stage pipeline. The stages are: (1) removal of redundant sequence elements; (2) capture of discriminative importance of sequence elements; (3) encoding of temporal dependencies between sequence elements. Using the built activity models, activity recognition (for sequences not used for training) is achieved using a similar pipeline.

3.4 Desired framework properties

A set of desired properties is now listed which are desirable for a complex activity analysis model.

1. To be able to handle concurrent events. Human activities often naturally overlap one another. For example, in a kitchen environment one might prepare a side dish *while* the food is cooking. Yet, most current activity analysis frameworks assume that every activity must be finished before another is started [Hamid, 2008].

2. To be able to integrate and process data from multiple streams. In many cases, a single data stream is inadequate for capturing complex human behaviour. In many fields of observational research it is common to collect multiple data streams describing an activity, including digital video, system logs and sensor data [Fouse et al., 2011].

3. To be resilient to noise. "Noise" in human activities is defined in this work as random actions which take place during an activity without these actions being significant for the execution of the activity. For example, during the activity *washing clothes* a person can decide to take a break, performing the action *drink coffee*. This action is not relevant to the execution of *washing clothes* activity. In contrast, *e.g.*, one cannot wash clothes without switching on the washing machine. Therefore, the action *switch on washing machine* is important for the execution of the activity *wash clothes*. During the execution of complex, prolonged activities it is possible that the human sidetracks from executing actions relevant to these activities, performing several random, unpredictable actions. Therefore a complex activity analysis framework should be capable of dealing with such unpredictable, random actions.

4. To be able to handle datasets in which only a few labels are available. In practice it is difficult to obtain labelled data of complex activities such as engineering design processes since this data has to be annotated by experts. Therefore, there are many datasets X in which, if labelled data is denoted with X_l and unlabelled data with X_u then $|X_l| < |X_u|$ or $|X_l| \ll |X_u|$.

5. To be able to learn model topology from data. Manually defined model topology (*e.g.* [Shi et al., 2004b]) has the disadvantage of bounding a framework to a specific problem. Another disadvantage is that experts may be required to define the topology. This is usually a tedious process which requires extensive domain expertise [Shi et al., 2006] and may additionally result to a subjective model with poor generalisation potential.

6. To be able to learn model parameters from data. For the same reasons as in 5 it is desirable that model parameters are learned from data and are not manually defined.

7. To be able to efficiently represent activity hierarchy. Human activities have a natural hierarchy [Nguyen et al., 2005] as they typically comprise sequences of subtasks. For example, activity *high jump* consists of the subtasks or actions *running*, *jumping* and *falling*. As shown in [Nguyen et al., 2005], flat models such as the HMM and its extensions Coupled Hidden Markov Model [Brand et al., 1997] and Variable Length Hidden Markov Model [Galata et al., 2001] cannot model complex behaviour efficiently as they fail to capture the hierarchy naturally embedded in the behaviour.

8. To be able to provide on-line feedback in case of erroneous execu-

	Inomata et al. 2009	Nguyen et al. 2005	Shi et al. 2004	Hamid et al. 2009	Proposed
1. Concurrency	×	×	✓	×	✓
2. Multi streams	✓	×	✓	×	✓
3. Noise Resiliency	✓	✓	×	×	✓
4. Few training data	×	×	✓	×	✓
5. Auto topology	✓	×	×	✓	✓
6. Auto parameter	✓	✓	×	✓	✓
7. Hierarchy	×	✓	✓	✓	✓
8. On-line	✓	×	✓	×	✓

Table 3.1: Comparing characteristics of various current activity analysis frameworks.

tion of an activity or if any type of anomalous event is detected. This is very important for surveillance applications *e.g.* theft detection [Damen and Hogg, 2007] or nursing applications (*e.g.* fall detection for the elderly [Yu et al., 2010]) and industrial process monitoring applications [Voulodimos et al., 2012].

Table 3.1 shows how these properties are catered by four current activity analysis frameworks.

In this Thesis a system for analysing prolonged, composite human activities arising in multimedia streams is proposed which accommodates all properties listed above. The overview of the system is now presented.

3.5 System description

The proposed framework automatically analyses multiple data streams recorded during the execution of complex human activities. An overview of the overall system proposed in this Thesis is given in Fig. 3.1.

The proposed system comprises two main components, specifically, a *data acquisition and feature extraction* unit (Fig. 3.1a) and a *machine learning* unit (Fig. 3.1b and c).

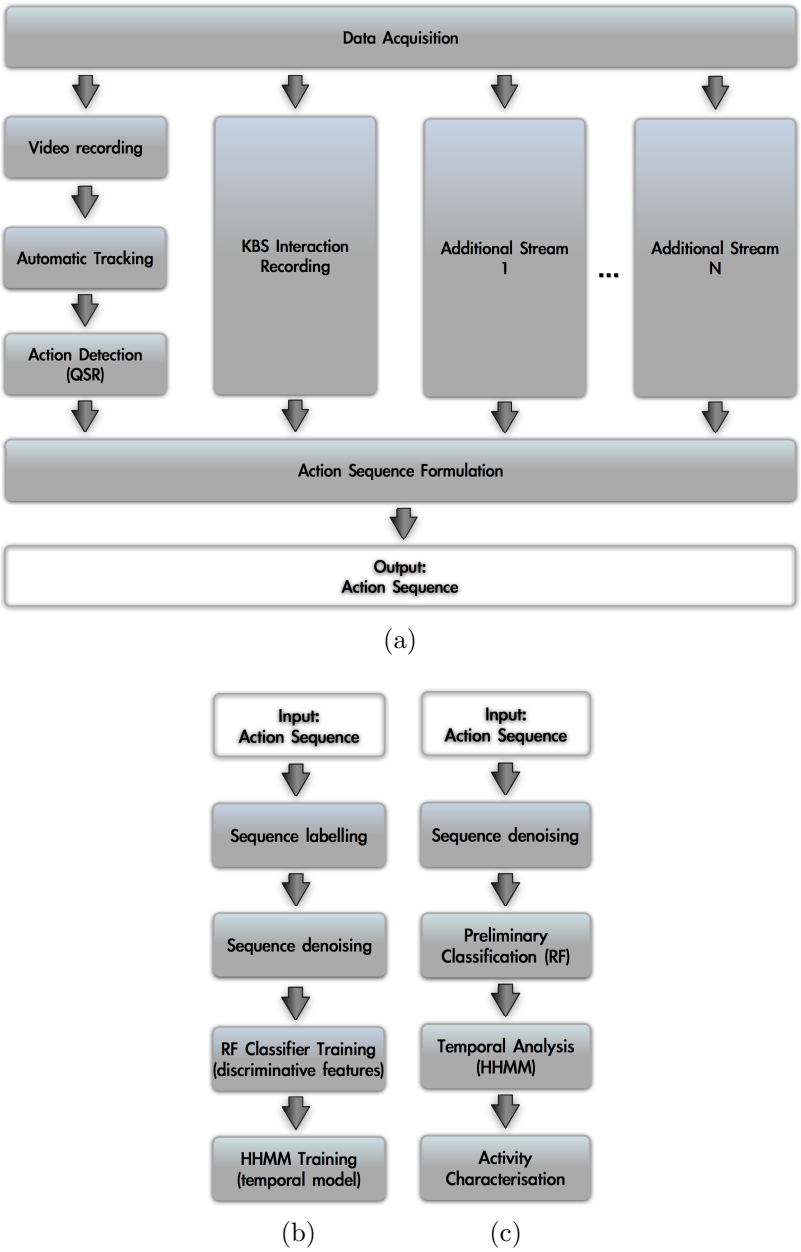


Figure 3.1: Overview of the proposed prolonged, composite activity analysis system. **(a):** Data acquisition unit. **(b):** Machine learning unit Phase I, training. **(c):** Machine learning unit Phase II, identification.

3.5.1 Data acquisition and feature extraction unit

The data acquisition unit extracts sequences of human actions from various data streams. These streams may result from video cameras, audio

recorders or any other type of sensors. In this work video streams are primarily used; additionally, in one of the studied applications a data stream which results from the user's interactions with a software application (KBS) is utilised. However, the system can handle streams resulting from any type of sensor, under the condition that the streams can be represented as time series of discrete elements (action primitives). If this condition is satisfied, the process of action sequence formulation, which prepares the data for further processing with the machine learning part of the system, is simply a matter of merging all data streams in a unified superstream by placing the elements of the data streams (action primitives) in chronological order.

The data acquisition unit consists of the following parts:

The *video recording* component, which uses a static video camera to record user's interactions with various scene objects (Fig. 7.1).

The *automatic tracker*, which picks out the movements of the moving objects in the scene. It is described in section 4.3.1.

The *action detection* unit, which converts the data generated by the tracking mechanism into *actions* by examining the *qualitative spatial relations* (QSR) [Sridhar et al., 2008] between the tracking windows of moving objects. This unit is described in section 4.3.2.

The *KBS interaction recording* unit which provides the user with on-demand expert knowledge and also records his queries and the exact time they occurred. The sequence formulation unit handles these queries in the same manner as the standard actions recorded by the QSR. As explained in Section 4.4, by monitoring the engineer's queries it can be deduced in which cognitive task he is involved, which would be impossible to determine by solely analysing the video stream. The

KBS interaction recording unit is described in section 4.4.

The *action sequence formulation* unit which generates a time sequence of actions that took place in the studied scene. This sequence, Q is of the form:

$$Q = \{p_a(t_{a,s}), p_b(t_{b,s}), p_b(t_{b,e}), p_a(t_{a,e}), \dots\} \tag{3.1}$$

where p_x an action (*e.g.* erasing, writing *etc.*) which starts at time $t_{x,s}$ and ends at time $t_{x,e}$. Sequence Q includes the complete timeline of the engineer's involvement with the engineering design task. This unit handles desired properties 1 and 2, which are handling concurrent events and data from multiple streams. It is described in section 4.5.

3.5.2 Machine learning unit

The action sequences extracted by the data acquisition unit are passed to the machine learning component. The latter is in essence a supervised classifier and operates in two phases, a *training* phase (Fig. 3.1b) and an *identification* phase (Fig. 3.1c).

Phase I: Training

The purpose of the training phase is to automatically train the classifier using data labelled by experts. The training phase accommodates desired properties 5 and 6 as model topology and parameters are learned from data. It comprises the following stages:

The *sequence labelling* stage is performed by experts, who label each action sequence by examining (a) the video footage corresponding to the action sequence (b) the elements of the sequence (c) the participant's study progress corresponding to the sequence. These la-

bels are high level descriptions of the activity observed in each action sequence, *e.g. soil condition examination executed correctly* or *base cost estimation executed erroneously*. Note that the correct solutions to the task are known to the experts. The proposed system can operate with a relatively small training dataset. Specifically, in the Gun Point dataset [Ratanamahatana and Keogh, 2004], discussed in chapter 5, 25% of the sequences are labelled. In the glucometer calibration task [Shi et al., 2004a], discussed in chapter 6, 14% of the total sequences are labelled. In the bridge design task, discussed in chapter 7, 43% of the total sequences are labelled. Thus, desired property 4 is satisfied as inequality X_u then $|X_l| < |X_u|$ stands (labelled data is denoted with X_l and unlabelled with X_u.)

The *sequence denoising* stage, in which redundant elements of the input sequences are removed. This process simplifies the dataset in a form which eases the classification task. This stage handles desired property 3, which is noise resilience and is described in sections 6.2 and 6.3.

The *RF classifier training* stage which builds a classification model on the basis of the constituent actions of the training sequences; the temporal order in which these actions are executed is not taken into account. It is performed with the aid of a *Random Forests* (RF) [Breiman, 2001] classifier. This stage handles desired property 4 as RFs are effcient in semi-supervised learning [Leistner et al., 2009]. It is described in section 5.3.1, Part I.

The *HHMM Training* stage which encodes information regarding the temporal order in which actions are executed within the sequences. This is achieved with an Hierarchical Hidden Markov Model (HHMM)

[Fine et al., 1998] whose topology represents the activities hierarchy and structure. This stage handles desired property 7 as HHMMs can efficiently represent activity hierarchy [Nguyen et al., 2005]. It is described in section 5.3.1, Part III.

Phase II: Identification

In the identification phase, novel unlabelled data (*i.e.* data not used during the training phase) are fed to the classifier which assigns them to classes. Each of these classes represent a complex activity, such as *transient loads evaluation executed correctly*. The identification phase consists of the following stages.

The *sequence denoising* stage which operates exactly as in the training phase and is described in sections 6.2 and 6.3.

The *preliminary classification* stage which assigns test sequences to activity classes using the trained RF classifier. The purpose of this stage is to statistically detect omissions of critical steps in the sequences. Sequences found to miss critical steps are not passed to the next step. Instead they are marked as erroneous, retaining the class label assigned to them by the RF classifier. This stage is described in section 5.3.2, Parts II and III.

The *temporal analysis* stage which assigns test sequences to activity classes using the trained HHMM model. This stage checks the temporal order in which actions are executed within the sequences marked as correct by the RF classifier and assigns them to activity classes. This stage is described in section 5.3.2, Part IV.

The *activity characterisation* stage is the decision making unit which makes a decision of the activity class label assigned on the basis of the

results obtained in the preliminary classification and temporal analysis stages. The form of activity characterisation is a single label providing a high level description of the activity performed in a test sequence (*e.g. glucometer calibration executed erroneously*, *base cost estimation executed correctly*). This stage handles desired property 8 as it provides on-line feedback in case of erroneous execution of an activity or if any type of anomalous event is detected. It is described in section 5.3.2, Part V.

Future extension: Unsupervised classification

Although a supervised classifier is currently used, in future work an unsupervised classifier will be considered. The classification phase will then work as follows. A number of sequences will be used for training. In the training stage, these sequences will be clustered using a suitable unsupervised clustering algorithm. Examples of such algorithms are k-means and Gaussian mixture models (GMMs) [Duda et al., 2000]. Ideally, each cluster will represent an activity. Then, in the identification phase, test sequences (not present in the training dataset) will be assigned to activities according to their proximities from the clusters representing activities. There are two problems that need to be solved which are discussed in the two paragraphs that follow.

First of all, it has to be ensured that each cluster, discovered in an unsupervised manner, actually corresponds to a single activity. In practice, a discovered cluster could represent different aspects of an activity (over-segmentation) or two similar activities (under-segmentation). In [Hamid et al., 2007] this problem was tackled by using labelled training sequences (where the labels corresponded to activities) and

following the three-step procedure of (1) over-segmenting the training dataset, (2) assigning each discovered cluster to the activity with the most samples in the cluster and (3) merging clusters assigned to the same activity. Of course, this method requires labelled training samples which implies a supervised classification procedure (although the method was named unsupervised). A fully unsupervised method would potentially consist of two steps, (1) over-segmentation of the training dataset (2) merging of the discovered clusters hierarchically, using, for example, the algorithm given in [Vasconcelos and Lippman, 1998].

The second problem is the choice of a distance metric which will determine the proximity of a test sequence to the discovered clusters representing activities.

3.6 Summary

In this Chapter a description of the proposed framework, capable of analysing prolonged, composite human activities arising in multimedia streams was given. The design of the whole framework was explained and the interaction between its different parts was analysed.

In the following chapters a detailed description of the system's individual components is presented. In Chapter 4 the data acquisition unit is described. Chapter 5 covers the classification component which comprises a discriminative features unit based on RFs and a temporal analysis unit which employs HHMMs. The system's denoising unit is described in Chapter 6.

Chapter 4

EXTRACTING ACTION SEQUENCES

4.1 Overview

In this chapter a methodology to extract features from multiple, parallel streams illustrating complex human activities which may involve cognitive actions is proposed. The goal of the method is to acquire a set of features from the input streams which can be used to recognise activities in the streams. The problem of activity recognition is studied later in this Thesis (Chapters 5 and 6).

For reasons explained in Section 2.1.1 current methods to record cognitive actions prove inadequate for practical applications. Thus, a novel method is proposed in this chapter which acquires data from cognitive tasks.

The contribution of this Chapter is therefore two-fold:

1. A new method to automatically extract action sequences from data streams representing complex human activities is presented. The proposed method has three important properties: firstly, it can handle multiple streams resulting from different acquisition sources; secondly, it is capable of modelling concurrent activi-

ties; thirdly it can model activities whose exact structure is not known a priori. No system simultaneously satisfying these three important properties has been previously presented.

2. A methodology for recording cognitive activities is introduced. In contrast to existing approaches, the proposed method operates in a non-obstructive manner and is suitable for practical applications.

The following section gives a general description of the proposed algorithm, explaining the methodological choices made for its development.

4.2 Framework design

It was stated that multiple concurrent streams are used as input. This decision is closely related to the proposed methodology of recording cognitive actions, which utilises at least two parallel streams: the first stream is video footage; the second stream is obtained by the human's interactions with a computer software. Note that in the analysis which follows, it is assumed that input data comprises two streams. However, the presented methodology can be extended to handle more than two streams. Additionally, in the experimental section it is shown that it can be efficiently applied to a single data stream.

The two extracted streams complement one another in the sense that each one stores information unavailable to the other. Semantically, the human activity represented in these streams results by merging them in a way that all recorded information appears in a unified stream which gives an account of what events took place during the observed

period and in what order. To ease the merging task, each stream was chosen to be represented as a finite sequence of discrete actions. Then building the unified sequence is simply a matter of placing the actions comprising each stream in absolute chronological order.

Converting the stream resulting from software to a sequence of discrete actions is achieved by associating each user's interactions with the software with a discrete action. For extracting simple actions from the video stream the choice of algorithm is application related, as it depends on the type of actions that have to detected. In the applications studied in this Thesis, often the framework has to discriminate between spatially similar but temporally different actions. This can only be achieved by analysing the trajectories of the objects involved in the similar actions.

Since the activities studied in this work are prolonged and involve multiple objects, trajectory recording and analysis are computationally expensive. To ease computational burden, qualitative spatial relations (QSR) [Sridhar et al., 2008] are employed. QSR is capable of monitoring interactions between objects. In this work, object interactions yielding spatially similar but temporally different actions are considered known. During the time intervals that QSR detects these interactions trajectories of involved objects are recorded. This is much more efficient than recording trajectories for the whole duration of the processed stream. Recorded trajectories are then analysed with the aid of continuous HMMs.

4.3 Extracting Actions from Video

The sequence classification algorithm proposed in this Thesis analyses time series of actions representing complex human activities. In this section a method to extract these actions from video of a human interacting with various objects on a table is presented. As stated in Wang *et al.* [Wang et al., 2009] there are two approaches in activity analysis: 1) objects detected (automatically or manually), tracked, then their tracks exploited to understand activities 2) use of motion feature vectors instead of tracks. For various practical reasons (*e.g.* tracking is still an unsolved problem unless constraints are applied in the study scene) most current action detection systems use the second method. However, these type of algorithms have the disadvantage of not being able to handle complex temporal relations [Zhang et al., 2011] and are not able to disambiguate between spatially similar but temporally different actions. The approach adopted in this Thesis falls in the first category as complex temporal relations and spatially similar actions are encountered in the examined problem.

4.3.1 Object tracking

In this Thesis it is assumed that all important objects are at least partially visible at all times. To monitor their movements, one video tracker on each object is placed manually at the first frame of each sequence. The work presented in this Thesis does not attempt to tackle the tracking problem; instead, it relies on current state-of-the-art solutions to handle it. Thus, an off-the-self approach is used which is sufficient for the applications studied in this Thesis. The chosen algorithm performs tracking using a colour histogram-based observation model

and a second order autoregressive dynamical model. This method can be found in [Pérez et al., 2002]. The main points of this algorithm are discussed in the following subsections.

Probabilistic Sequential Tracking with Monte Carlo Approximation

A state space model is first defined in which a Markovian prior on the hidden states is coupled with a conditionally independent observation process. At time t the hidden state is $\mathbf{x}_t$ and the observation $\mathbf{y}_t$. If the order of the dynamics is fixed to one then the sequence of filtering distributions $p(\mathbf{x}_t|\mathbf{y}_{0:t})$ to be tracked follows the equation:

$$p(\mathbf{x}_{t+1}|\mathbf{y}_{0:t+1}) \propto p(\mathbf{y}_{t+1}|\mathbf{x}_{t+1}) \int_{\mathbf{x}_t} p(\mathbf{x}_{t+1}|\mathbf{x}_t) p(\mathbf{x}_t|\mathbf{y}_{0:t}) d\mathbf{x}_t \qquad (4.1)$$

where $\mathbf{x}_{0:t} = (\mathbf{x}_0, \ldots, \mathbf{x}_t)$ and $\mathbf{y}_{0:t} = (\mathbf{y}_0, \ldots, \mathbf{y}_t)$.

Eqn. 4.1 cannot be handled analytically in visual tracking problems. Therefore in this work, following the recommendation of [Pérez et al., 2002] it is solved using a sequential Monte Carlo framework. More specifically the posterior $p(\mathbf{x}_t|\mathbf{y}_{0:t})$ is estimated by a finite set $\{\mathbf{x}_t^m\}_{m=1\ldots M}$ of M particles. A proposal transition kernel $f(\mathbf{x}_{t+1}; \mathbf{x}_t, \mathbf{y}t+1)$ is used to generate samples for $p(\mathbf{x}_{t+1}|\mathbf{y}_{0:t+1})$. If the set $\{\mathbf{x}_t^m\}_{m=1\ldots M}$ consists of representative samples from the filtering distribution at time t then the new particles $\tilde{\mathbf{x}}_{t+1}^m$ associated with their importance weights π_{t+1}^m are also representative samples of the new filtering distribution [Pérez et al., 2002]. The importance weights are given by the equation:

$$\pi_{t+1}^{m} \propto \frac{p(\mathbf{y}_{t+1}|\tilde{\mathbf{x}}_{t+1}^{m})p(\tilde{\mathbf{x}}_{t+1}^{m}|\mathbf{x}_{t}^{m})}{f(\tilde{\mathbf{x}}_{t+1}^{m};\mathbf{x}_{t}^{m},\mathbf{y}t+1)} \tag{4.2}$$

The set of representative samples for the distribution $p(\mathbf{x}_{t+1}|\mathbf{y}_{0:t+1})$ is denoted as $\{\mathbf{x}_{t+1}^{m}\}_{m=1\ldots M}$.

The output of the tracker at time t is given by the equation

$$\hat{\mathbf{x}}_t = \frac{1}{M}\sum_{m=1}^{M}\mathbf{x}_t^m \tag{4.3}$$

Dynamics Representation

A region of interest in a frame of a video sequence needs to be tracked. This region is represented as a 0-centered window, W which can be of any shape. Tracking is the process of determining the parameters of the transformation to be applied to W, for every frame of the video sequence. Following recommendations found in [Pérez et al., 2002, Bradski, 1998, Comaniciu et al., 2000], the parameters taken into account are the window's location in the image coordinate system $\mathbf{d} = (x, y)$ and the scale of the image, s. These parameters are the hidden variables of the dynamics representation. To estimate these parameters throughout the video sequence a second-order autoregressive dynamics model is used. The model's state at time (frame) t is defined as $\mathbf{x}_t = (\mathbf{d}_t, \mathbf{d}_{t-1}, s_t, s_{t-1})$. The dynamics model is given by the equation

$$\mathbf{x}_{t+1} = A\mathbf{x}_t + B\mathbf{x}_{t-1} + C\mathbf{u}_t \tag{4.4}$$

where $\mathbf{u}_t \sim \mathcal{N}(0, \Sigma)$.

Coefficients A, B and C as well as the covariance matrix, Σ can be learned from data illustrating correct tracks using the methodology

described in [Reynard et al., 1996]. In this work these parameters are empirically defined as described in Appendix 5.

Colour Representation

The colour model for a key object is obtained by applying histogramming techniques in the Hue, Saturation, Value (HSV) colour space [Forsyth and Ponce, 2011]. As in [Pérez et al., 2002] a HS histogram of $N_h N_s$ bins is formed and pixels with $saturation > 0.1$ and $value > 0.2$ are used. Acknowledging that the remaining pixels might contain useful information, an additional N_v bins are included which only contain Value, therefore the entire number of bins in the histogram is $N = N_h N_s + N_v$. The bin representing color vector $\mathbf{y}_t(\mathbf{u})$ in frame t and at pixel location $\mathbf{u}$ is denoted as $b_t(\mathbf{u})$, with $b_t(\mathbf{u}) \in \{1, \dots N\}$.

The system's state at frame t is given by the state vector $\mathbf{x}$. Colour information will be collected from region $R(\mathbf{x_t})$ with $R(\mathbf{x_t}) = \mathbf{d}_t + s_t W$. The color model of this region at frame t, $q_t(\mathbf{x})$ is a distribution which is given by a kernel density estimate as follows [Comaniciu et al., 2000]:

$$\mathbf{q}_t(\mathbf{x}) = \{q_t(n; \mathbf{x})\}_{n=1,\dots N} \tag{4.5}$$

$$q_t(n; \mathbf{x}) = K \sum_{\mathbf{u} \in R(\mathbf{x})} w(|\mathbf{u} - \mathbf{d}|)\delta[b_t(\mathbf{d}) - n] \tag{4.6}$$

In Eqn. 4.6, K is a normalisation constant ensuring $\sum_{n=1}^{N} q_t(n; \mathbf{x}) = 1$, δ is the Kronecker delta function [Spiegel and O'Donnell, 1997] and w is a weighting function. When a strictly colour-based tracking algorithm is used (*e.g.* mean shift [Comaniciu et al., 2000]) w is a smooth kernel, *e.g.* [Bradski, 1998, Comaniciu et al., 2000]. On the contrary, in parti-

cle filtering tracking methods this is not necessary since all candidate hypotheses associated with the particles have to be estimated [Pérez et al., 2002]. In such cases it is $w \equiv 1$. In general, the described color representation assigns a probability to each of the color histogram's N bins.

A colour representation $q_t(\mathbf{x})$, corresponding to a candidate state $\mathbf{x}$ will be compared to the reference colour representation $\mathbf{q}^*$ with:

$$\mathbf{q^*}_t(\mathbf{x}) = \{q^*(n)\}_{n=1,...N} \tag{4.7}$$

Similar to the case of $q_t(\mathbf{x})$, it is $\sum_{n=1}^{N} q^*(n) = 1$. The reference distribution can be defined manually or detected automatically given a certain colour profile. Colour models associated with candidate states are compared against the reference distribution using the distance D proposed in [Bradski, 1998, Comaniciu et al., 2000] with

$$D[\mathbf{q}^*, \mathbf{q_t}(\mathbf{x}_t)] = \left[1 - \sum_{n=1}^{N} \sqrt{q^*(n) q_t(n; \mathbf{x})}\right]^{\frac{1}{2}} \tag{4.8}$$

Distance D as given in Eqn. 4.8 is derived from the Bhattacharyya similarity coefficient. Following the recommendation of [Pérez et al., 2002] the probability of a candidate state is estimated as:

$$p(\mathbf{y}_t|\mathbf{x}_t) \propto exp{-\lambda D^2[\mathbf{q}^*, \mathbf{q_t}(\mathbf{x}_t)]} \tag{4.9}$$

Having applied the tracking algorithm, each object in the scene is now represented by a tracking window. In the next section it is described how interactions between tracking windows can be exploited to detect actions in video sequences.

4.3.2 Action detection in the video stream

In this section it is explained how actions are detected in the video footage.

QSR Framework

Actions from video footage are extracted by identifying patterns of *qualitative spatial relations* (QSR) [Sridhar et al., 2008] between the tracking windows of moving objects. For every pair of tracking windows, (W_j, W_k), a sequence of spatial relations is computed by considering the relative position of the windows. In [Sridhar et al., 2008], 3 types of spatial relations were used: two windows could be either spatially Disconnected (D), or connected through the surrounds (S) or Touches (T) relations. Here, this system is simplified by merging relations (S) and (T) and, therefore, 2 windows can be either Disconnected (D) or Interacting (I). A relation holds for a finite amount of time, from time point t_m to t_n. Therefore, a relation R_i between two tracking windows, W_j, and W_k can be represented with the 5-tuple:

$$R_i =< Q_i, W_j, W_k, t_m, t_n > \tag{4.10}$$

with Q_i the type of relation (*i.e.* (D) or (I)).

A timeline representation is formed which includes all spatial relations taking place in the examinant video sequence. In this representation, an action is defined as the temporal co-occurrence of one or more spatial relations that hold during a time interval. The mapping from spatial relation co-occurrence to actions is pre-defined, for example, the temporal co-occurrence of relations "hand (I) ruler" and "hand (I) map" yields action "measuring on map". Hence the set of possible

Class 1: Erasing	[(hand 1)T(eraser) || (hand 2)T(eraser)]&& (eraser)T(paper) [(hand 1)S(eraser) || (hand 2)S(eraser)] && (eraser)S(paper) [(hand 1)S(eraser) || (hand 2)S(eraser)] && (eraser)T(paper) [(hand 1)T(eraser) || (hand 2)T(eraser)] && (eraser)S(paper)
Class 2: Writing/ Sketching/ Waiting	[(hand 1)T(pencil) || (hand 2)T(pencil)] && (pencil)T(paper) [(hand 1)S(pencil) || (hand 2)S(pencil)] && (pencil)S(paper) [(hand 1)S(pencil) || (hand 2)S(pencil)] && (pencil)T(paper) [(hand 1)T(pencil) || (hand 2)T(pencil)] && (pencil)S(paper)
Class 3: Measuring on map	[(hand 1)T(ruler) || (hand 2)T(ruler)] && (ruler)T(map) [(hand 1)S(ruler) || (hand 2)S(ruler)] && (ruler)S(map) [(hand 1)S(ruler) || (hand 2)S(ruler)] && (ruler)T(map) [(hand 1)T(ruler) || (hand 2)T(ruler)] && (ruler)S(map) (hand 1)T(ruler) && (ruler)T(map) && (hand 2)T(ruler) (hand 1)T(ruler) && (ruler)T(map) && (hand 2)S(ruler) (hand 1)S(ruler) && (ruler)S(map) && (hand 2)T(ruler) (hand 1)S(ruler) && (ruler)S(map) && (hand 2)S(ruler) (hand 1)S(ruler) && (ruler)T(map) && (hand 2)T(ruler) (hand 1)S(ruler) && (ruler)T(map) && (hand 2)S(ruler) (hand 1)T(ruler) && (ruler)S(map) && (hand 2)T(ruler) (hand 1)T(ruler) && (ruler)S(map) && (hand 2)S(ruler)

Figure 4.1: Mapping from qualitative spatial relations to actions for three action classes. Subclasses of Class 2 can be distinguished by analysing the moving hand's trajectory. Index of symbols used: T : Touches, S : Surrounds, && : And, || Or.

actions detected from video is also predefined. An action p_z, starts at time point t_x when the spatial relations defining it start co-occurring and ends at time point t_y when the relations stop existing. The mapping from qualitative spatial relations to actions for three action classes is shown in Fig. 4.1.

Extended QSR

QSR cannot distinguish between spatially similar actions, such as *writing* and *sketching*: both of them are represented by the co-occurrence of spatial relations "hand (I) pencil" and "hand (I) paper". It is also possible that the participant simply holds a pencil over the paper (*waiting*).

QSR framework is extended here to disambiguate between these three actions by statistically analysing the motion trajectories of the objects involved in these. This analysis is performed with the aid of a continuous HMM. More specifically, a trajectory is considered to be a continuous quantity that is described as the position of the object

in time. A feature vector O_i is constructed for each trajectory i by using the object's relative coordinates (x_t, y_t) within a period of $t = 15$ successive frames. Therefore, the trajectory classification problem can be formulated as the assignment of an input vector, O_n, to a trajectory class c_m. This problem can be viewed as the maximisation of the quantity m^*, with:

$$m^* = \arg\max_m P(c_m|O_n). \tag{4.11}$$

To solve Equation 4.11, a HMM is defined where each state represents a trajectory class, c_m. The complete set of model parameters is given by the triplet:

$$\lambda = \{\pi, A, b_j\} \tag{4.12}$$

where π_j the initial probability, A the state transition probability matrix and b_j the probability density function (PDF) for state j, respectively. A Gaussian Mixture-based representation is used for the PDF, b_j. In the experiments, the HMM is trained with 750 sequences, each of length 15, with 250 sequences representing each class (*writing, sketching and waiting*).

Note that a related action detection framework appears in Yao and Fei-Fei [Yao and Fei-Fei, 2010]. However their method is not able to differentiate between spatially similar but temporally different actions such as writing and sketching and cannot be trivially extended to do so; the framework proposed here on the other hand is capable of performing this distinction.

4.4 Cognitive action detection

When performing a cognitive task (such as an engineering task), an engineer executes actions such as *"estimate seismic load"* and *"estimate soil condition"* which are related to the task and have to be detected by the system. These actions cannot be observed directly through the analysis of the video stream. To solve this problem, a novel action detection scheme is introduced in this section based on monitoring of the user's interactions with a computer software.

The action detection methodology presented in this section can be applied to tasks with the following three characteristics: 1) expert knowledge is required for the completion of the task which the user does not possess, 2) this knowledge can be acquired by the user by posing simple queries, 3) information which results from engineer's calculations can be directly associated with a query that a user posed.

For example, during the execution of a bridge design task the user requires information regarding the effect of wind load to the structure. He poses the query *wind load*. The software replies by displaying relevant information. After calculations the user reaches a result which reflects the effect of wind load. This result is directly associated with query *wind load*.

The computer software is in essence a Knowledge Based System (KBS); its structure is shown in Fig. 4.2. The software's Knowledge Base which stores all information required to solve the examined cognitive task is built by consulting relevant handbooks and regulations. The resulting database has been validated by experts. All information used for the purpose of the experiment is presented in Section 7.3.1. Prior to using the KBS the users are instructed to estimate any parameters

not included in the KBS on the basis of their knowledge.

The KBS is designed so that each piece of requested information when accessed is associated with a specific cognitive action of the executed task. Therefore, when the user queries the system, it can be deduced in which task he is involved. The user's query is automatically inputted as a spreadsheet entry in the KBS GUI (Fig. 4.3, D). The entry is timestamped by the system (Fig. 4.3, E). When the user obtains a *result* associated to the posed query, he inputs this *result* in a cell corresponding to this query (Fig. 4.3, F). The software automatically timestamps the result entry as well and associates it with the end of the corresponding cognitive process (Fig. 4.3, G). Each user's interaction with the computer software is denoted as $\omega(\phi, t)$ where ϕ is the type of the interaction (*e.g. wind load*) and t its timestamp. Thus, the KBS is used to generate a time sequence of the user's interactions with it by recording and timestamping the user's queries and result entries.

A cognitive action, p_ϕ is defined as the action which takes place during the time interval between a user's query $\omega_s(\phi_s, t_s)$ and the entry of the result associated to this query $\omega_e(\phi_e, t_e)$. A cognitive action is symbolised as $p_\phi = \{p_\phi(t_{\phi,s}), p_\phi(t_{\phi,e})\}$ with $t_{\phi,s} = t_s$ and $t_{\phi,e} = t_e$.

Compared to the existing methods of obtaining information regarding cognitive actions (discussed in section 2.1.1) the proposed method has the following advantages. First, it avoids the problem of overshadowing, associated with the think-aloud method. Second, there is no (or limited) ambiguity in the assignment of primitive actions to cognitive actions as the resulting cognitive action is directly associated with the user's input query. It has to be noted that the proposed method has also similarities with the previously proposed methods. First, task

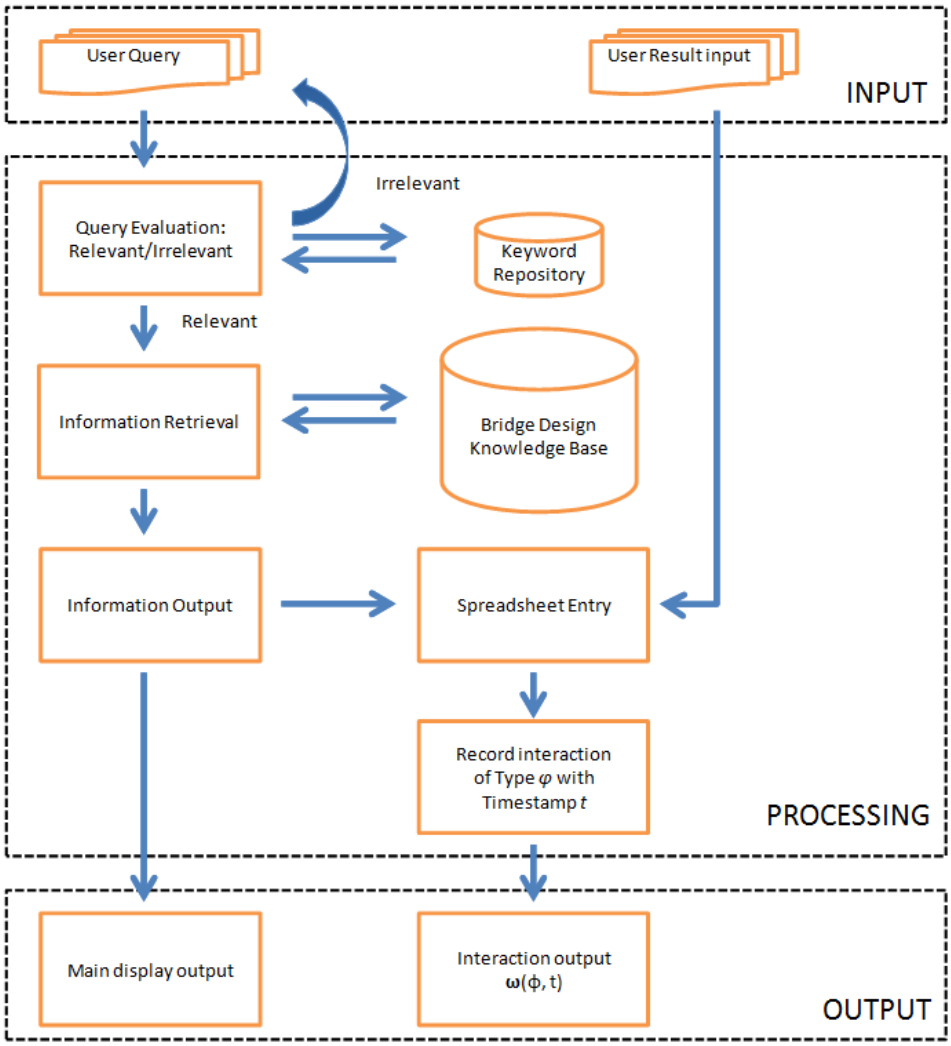


Figure 4.2: Structure of the KBS.

analysis is required to determine the cognitive actions of the studied domain. Second, it uses a key term recognition scheme in text, which is similar to the key term recognition scheme in speech utilised by the think aloud method. In other words, the think aloud method asks the participants to verbalise their thoughts; the proposed method asks the participants to convert their thoughts into text.

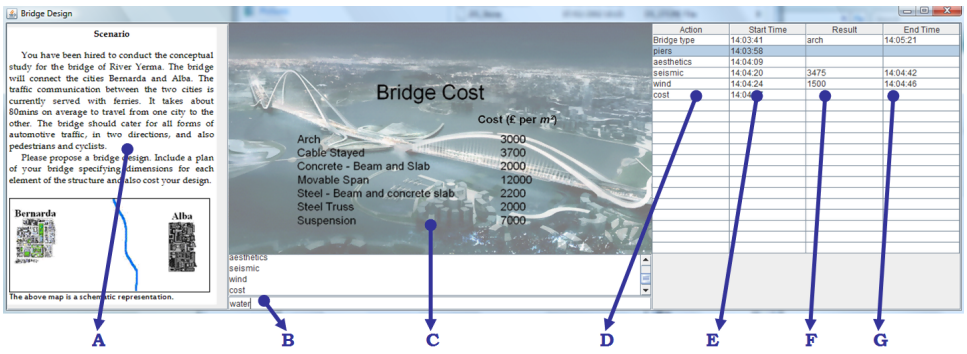


Figure 4.3: Overview of the KBS interface: (A) Task scenario, (B) input console, (C) returned expert knowledge, (D) entered queries and (E) their timestamps, (F) slot for user to input result and (G) result's timestamp.

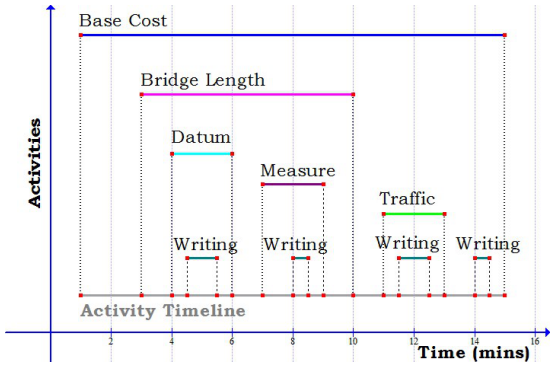


Figure 4.4: The activity timeline for a design task is formed from temporal occurrence of its constituent actions. Each action primitive (*e.g.* measure start, measure end, writing start, bridge length start *etc.*) is inputted as timepoint in the activity's (Base cost) timeline.

4.5 Action Sequence Formulation

Extraction of actions from input streams with the aid of QSR framework and KBS is a timeline of the user's actions during the studied task. Two data streams are obtained, one from the QSR action extraction unit and one from the KBS. The QSR stream is represented by the sequence Q_{QSR} with:

$$Q_{QSR} = \{p_a(t_{a,s}), p_b(t_{b,s}), p_b(t_{b,e}), p_a(t_{a,e}), \dots\} \tag{4.13}$$

with $p_x(t_{x,s}), p_x(t_{x,e})$ start and end of action p_x.

The KBS stream is represented by the sequence Q_{KBS} with:

$$Q_{KBS} = \{p_m(t_{m,s}), p_n(t_{n,s}), p_n(t_{n,e}), p_m(t_{m,e}), \dots\} \tag{4.14}$$

with $p_x(t_{x,s}), p_x(t_{x,e})$ start and end of action p_x.

Sequences Q_{QSR} and Q_{KBS} are then joined in one super sequence, Q_{total} with:

$$Q_{total} = \{Q_{QSR}, Q_{KBS}\} \tag{4.15}$$

$$\begin{aligned} Q_{total} = \{&p_a(t_{a,s}), p_b(t_{b,s}), p_b(t_{b,e}), p_a(t_{a,e}), \\ &p_m(t_{m,s}), p_n(t_{n,s}), p_n(t_{n,e}), p_m(t_{m,e}) \dots\} \end{aligned} \tag{4.16}$$

A notation which will help explaining sorting of the elements (action primitives) of Q_{total} more efficiently is now employed. Each element of Eqn. 4.16 $p_x(t_{x,\phi}), \phi \in \{s, e\}$ is written as $p_n(\psi_n, t_n)$ where $\psi_n = (x, \phi)$, $t_n = t_{x,\phi}$ and n the order of the element in sequence Q_{total}. If Q_{total} consists of N elements, then Eqn. 4.16 can be written as:

$$Q_{total} = \{p_n(\psi_n, t_n), p_{n+1}(\psi_{n+1}, t_{n+1}) \dots, p_N(\psi_N, t_N)\} \tag{4.17}$$

The final sequence is obtained by placing the elements of Q_{total} in chronological order starting with the action primitive which occurred first:

$$Q = \{p_m(\psi_m, t_m) \in Q_{total} : t_m < t_{m+1}\} \tag{4.18}$$

In Fig. 4.4 it is shown how a sequence Q is formulated: each action primitive (*e.g.* measure start, measure end, writing start, bridge length start *etc.*) is inputted as timepoint in the activity's timeline. Note that this representation can handle actions that take place in parallel (concurrent actions). The temporal relations between an activity's constituent action primitives are variable. This means that the order of action primitives forming an activity can be changed and action primitives can be added to the sequence or omitted from it without necessarily altering the correctness of the overall process.

4.6 The general case

The previous section described the sequence formulation process in the case that the system handles two streams, one resulting from video and the other from the user's interactions with the KBS. This section shows how this method can be applied to multiple streams, resulting from several different sources. It is assumed that there are N sources, then their corresponding time sequences of primitive actions, $Q_1, Q_2, ..., Q_N$ can be written as:

$$Q_1 = \{p_{11}(\psi_{11}, t_{11}), p_{12}(\psi_{12}, t_{12}), ..., p_{1k}(\psi_{1k}, t_{1k})\} \tag{4.19}$$

$$Q_2 = \{p_{21}(\psi_{21}, t_{21}), p_{22}(\psi_{22}, t_{22}), ..., p_{2k}(\psi_{2k}, t_{2k})\} \tag{4.20}$$

$$Q_N = \{p_{N1}(\psi_{N1}, t_{N1}), p_{N2}(\psi_{N2}, t_{N2}), ..., p_{Nk}(\psi_{Nk}, t_{Nk})\} \tag{4.21}$$

In the above equations it is assumed that the sequences $Q_1, Q_2, ..., Q_N$ contain equal number of elements, k, to simplify the notation. Obviously, in practice these sequences usually include different numbers of elements. A super-sequence is then formed, Q_{total} such that:

$$Q_{total} = \{Q_1, Q_2, ..., Q_N\} \tag{4.22}$$

$$\begin{aligned} Q_{total} = \{&p_{11}(\psi_{11}, t_{11}), p_{12}(\psi_{12}, t_{12}), ..., p_{1k}(\psi_{1k}, t_{1k}), \\ &p_{21}(\psi_{21}, t_{21}), p_{22}(\psi_{22}, t_{22}), ..., p_{2k}(\psi_{2k}, t_{2k}), ..., \\ &p_{N1}(\psi_{N1}, t_{N1}), p_{N2}(\psi_{N2}, t_{N2}), ..., p_{Nk}(\psi_{Nk}, t_{Nk})\} \end{aligned} \tag{4.23}$$

The elements of Q_{total} are then placed in chronological order, starting with the element which occurred first:

$$Q = \{p_m(\psi_m, t_m) \in Q_{total} : t_m < t_{m+1}\} \tag{4.24}$$

The final sequence, Q is the unified representation of the time sequences $Q_1, Q_2, ..., Q_N$, resulting from N different data streams.

4.7 Experimental results

In this section qualitative results are presented for the proposed sequence extraction and representation method in two datasets. The first dataset arises from execution of a bridge design task. The second

illustrates execution of the glucometer calibration task.

4.7.1 Bridge design task

This dataset is described in detail in Chapter 7 where the activity recognition results are presented. Here a brief overview is given.

Actions of engineers working on a bridge design task are recorded using a static video camera and the software described in Section 4.4. Regarding the analysis of video footage, key objects are manually selected and a video tracker, operating as described in Section 4.3.1 is placed manually on each of them. Key objects are all objects present in the studied scene, *i.e.* the participant's hands, a pencil, an eraser, a ruler, a map and a paper. Each key object in the scene is now represented by a tracking window. Actions are then detected by recording the spatial interactions between the tracking windows using the QSR framework as described in Section 4.3.2. Mapping between object interactions and actions is achieved using the manually defined scheme of Fig. 4.1. For convenience each interaction observed in the experiment is represented by a short code; all assigned codes are shown in Table 4.1. The resulting two streams are merged into a unified stream following the methodology proposed in Section 4.5.

The proposed algorithm results in formulation of Gantt charts. Charts resulting from a selection of four clips of the dataset are shown in Fig. 4.5 where the extracted action sequences appear below the Gannt charts. Note that in the charts, time interval between two consecutive action boundaries is set to five time units for representational purposes. Actions extracted from the video stream appear in red colour and cognitive actions, detected by the software, in blue colour. It is clear that the

No.	Action boundary code	Description	Source stream
1	p	measuring start	video
2	v	measuring end	video
3	b	sketching start	video
4	u	sketching end	video
5	x	writing start	video
6	w	writing end	video
7	y	erasing start	video
8	z	erasing end	video
9	s	transient loads start	KBS
10	t	transient loads end	KBS
11	d	river traffic start	KBS
12	e	river traffic end	KBS
13	f	wind load start	KBS
14	g	wind load end	KBS
15	h	seismic load start	KBS
16	i	seismic load end	KBS
17	j	base cost start	KBS
18	k	base cost end	KBS
19	l	bridge length start	KBS
20	m	bridge length end	KBS
21	n	traffic requirements start	KBS
22	o	traffic requirements end	KBS
23	a (1)	soil evaluation in zone 1 start	KBS
24	c (1)	soil evaluation in zone 1 end	KBS
25	a (2)	soil evaluation in zone 2 start	KBS
26	c (2)	soil evaluation in zone 2 end	KBS
27	a (3)	soil evaluation in zone 3 start	KBS
28	c (3)	soil evaluation in zone 3 end	KBS
29	a (4)	soil evaluation in zone 4 start	KBS
30	c (4)	soil evaluation in zone 4 end	KBS
31	q	reference datum start	KBS
32	r	reference datum end	KBS
33	E	excavations start	KBS
34	K	excavations end	KBS
35	P	intermediate piers start	KBS
36	R	intermediate piers end	KBS
37	F	foundations start	KBS
38	G	foundations end	KBS
39	A	aesthetics start	KBS
40	Z	aesthetics end	KBS

Table 4.1: Vocabulary of observed action boundaries in the bridge design task with their corresponding codes.

proposed representation is capable of efficiently extracting actions from multiple parallel streams. The charts show that many of the recorded actions overlap. By modelling each action using its boundaries (*i.e.* its start and end point), the method is able to represent such concurrent actions.

4.7.2 Glucometer calibration

The proposed action sequence extraction framework is also applied to the videos of the publicly available dataset presented in [Shi et al., 2004a]. The videos illustrate executions of the task of calibrating a blood glucose monitor, which is a common task for elderly people who

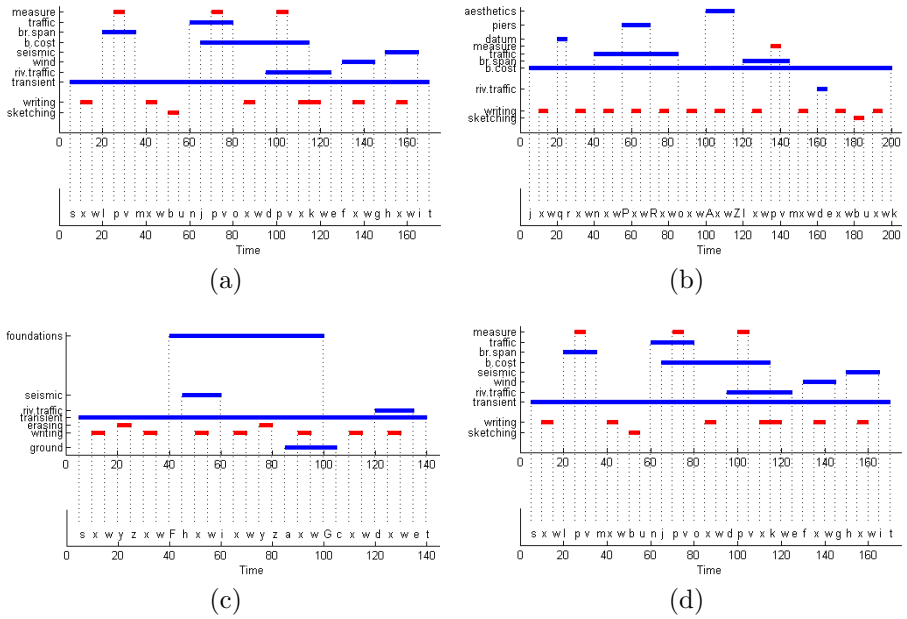


Figure 4.5: Gantt charts resulting from four clips of the bridge design task. Actions extracted from the video stream appear in red colour and cognitive actions, detected by the software, in blue colour. Extracted action sequences are shown below the charts.

develop late stage diabetes. In similar fashion to the bridge task, in-depth analysis of this dataset and results in activity recognition are given later in this Thesis, in Section 6.7. Here analysis is restricted to qualitative assessment of the action sequence representation resulting from the application of the proposed framework to this dataset.

In this case there is only one stream, resulting from the video footage. However, action concurrency poses a significant challenge. Key objects are manually defined and tracked using the tracking technique described in Section 4.3.1. Rather than manually specifying certain actions of interest, all possible object interactions are recorded. Each interaction observed in the experiment is represented by a code (Table 4.2).

Formulated Gannt charts for a selection of four clips from the dataset

No.	Action primitive code	Description	Source stream
1	a	hand interacts with glucometer, start	video
2	b	hand interacts with glucometer, end	video
3	c	hand interacts with liquid, start	video
4	d	hand interacts with liquid, end	video
5	e	hand interacts with test strip, start	video
6	f	hand interacts with test strip, end	video
7	g	glucometer interacts with test strip, start	video
8	h	glucometer interacts with test strip, end	video
9	i	test strip interacts with liquid, start	video
10	j	test strip interacts with liquid, end	video
11	k	shake liquid, start	video
12	l	shake liquid, end	video
13	m	shake test strip, start	video
14	n	shake test strip, end	video
15	o	liquid interacts with glucometer, start	video
16	p	liquid interacts with glucometer, end	video

Table 4.2: Vocabulary of observed action primitives in the glucometer task with their corresponding codes.

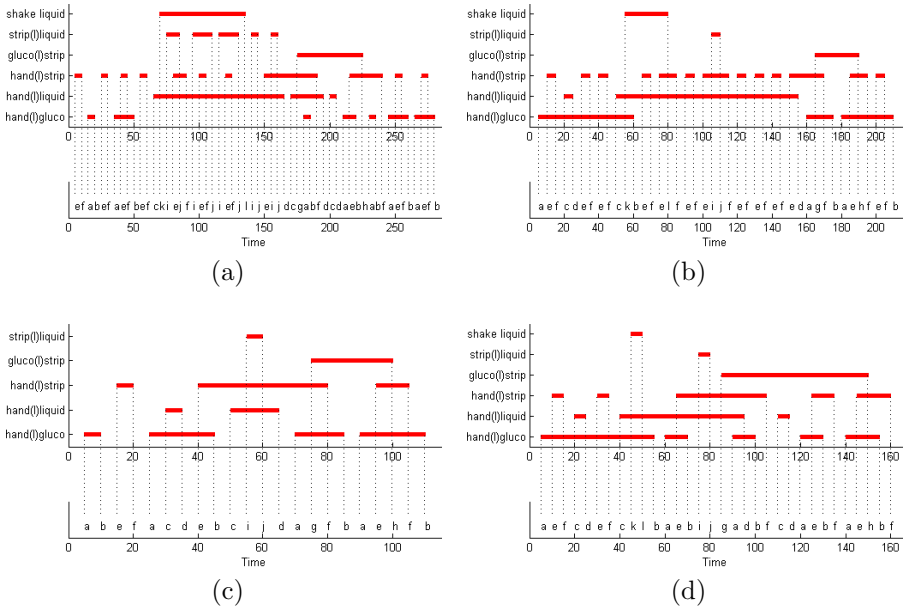


Figure 4.6: Gantt charts resulting from four clips of the glucometer task. The extracted sequences of action primitives are shown below the charts.

are shown in Fig. 4.6. Resulting action sequences appear below the charts. As shown in the charts, many of the recorded actions overlap. The proposed framework models all actions using their start and end points to overcome this problem.

Note that so far it was shown that the proposed framework is capable of obtaining qualitatively logical results as the information

present in the data streams is smoothly converted into action sequences. The resulting sequences are afterwards passed to the machine learning unit which identifies the activities represented in them. The relevant methodology is discussed in Chapters 5 and 6. The efficiency of the proposed algorithm is therefore assessed in the following chapters. High accuracy in activity identification will prove that the proposed action extraction and representation method is suitable for prolonged, composite human activities.

Similarly, the effectiveness of the proposed cognitive action detection method will be further investigated in Chapter 7 where the bridge design dataset is analysed from the aspect of activity identification.

4.8 Summary

In this Chapter a methodology to extract features from multiple, parallel streams illustrating complex human activities which may involve cognitive actions was presented. The extracted features are in form of sequences of simple actions. Two streams are used, video footage and user's interactions with a computer software. The methodology proposed to use human's interactions with the computer to record cognitive tasks is novel. In contrast to existing approaches, the proposed method operates in a non-obstructive manner and is suitable for practical applications. The idea of complementing video features with a stream acquired by the human's interactions with a computer for activity analysis is novel as well. The proposed method can handle multiple streams resulting from different acquisition sources, it is capable of modelling concurrent activities and can model activities whose exact structure is not known a priori. No system simultaneously satisfying

these three important properties has been previously presented.

Chapter 5

ACTION SEQUENCE RECOGNITION

In this Thesis activities which fall in the category of events in terms of duration are examined. The focus of this work lies specifically on activities which comprise a large number of steps and these steps can be executed in a plethora of ways. This characteristic makes the structure of such activities challenging to model. In simpler events, like a sports match certain constraints (*e.g.* laws of physics) simplify the prediction of the next step in an action sequence given the current step. For example, in a tennis game consider the state of the ball hitting the court. This state can only be followed by two possible states: either the ball is hit by a player or a point is scored. Although constraints apply in the events studied in this Thesis, these are much less restrictive than those encountered in normal activities. Therefore activities studied in this Thesis are time sequences whose temporal relationships between their elements have *non-local* character. An aspect of such activities is that their execution requires a certain level of expertise by the human who performs them. They can be observed in a wide variety of tasks such as surgery, calibration of a medical device or an engineering design study. These activities are referred to as prolonged, composite activities

in this work.

The contribution of this chapter is a classification algorithm designed to efficiently recognise composite, prolonged activities. The proposed method comprises two individual components: a preliminary classification unit based on RFs and an HHMM, whose topology implements the activities' hierarchy and structure. Previously proposed algorithms are not capable of recognising composite, prolonged activities with high accuracy.

The decision to propose this combined classification model was based on the observations that:

1. Discriminative feature approaches (such as RFs and SVMs) perform well in noisy datasets but are not readily capable of handling temporal relations arising from sequential data.

2. Markov type approaches represent temporal relationships in time sequences efficiently, however their accuracy deteriorates in noisy sequences.

In this work, it is shown how discriminative feature and Markov type methods can be combined in a model which yields "the best of two worlds" for datasets illustrating complex human behaviour. There exist no classifiers that posses both discriminative and temporal encoding properties; modelling prolonged, composite activities requires both.

Regarding the choice of individual methods, RFs is used in this work as the discriminative feature algorithm as they have demonstrated better or at least comparable performance to other state-of-the-art methods in classification [Breiman, 2001, Bosch et al., 2007], real-time keypoint recognition [Lepetit et al., 2005] and clustering applications

[Moosmann et al., 2006]. Compared to their main competitor, SVMs, RFs have the advantage of offering a variable importance index which reflects the importance of a variable based on the classification accuracy, taking into account interaction between variables [Breiman, 2001]. Also, their performance is not sensitive to the values of their parameters [Yeh et al., 2012]. Moreover, RFs extend naturally to multiple class problems unlike SVMs [Torralba et al., 2007, Criminisi et al., 2012]. In this work a multiclass problem is solved; also measuring feature importance is desirable so that a reduced set of features is passed to the temporal part of the proposed algorithm. Therefore RFs are used. Concerning the choice of the temporal modelling algorithm, a hierarchical graphical model is used since flat statistical graphical models (like HMMs) cannot represent efficiently the natural hierarchical structure of complex activities [Nguyen et al., 2005]. The HHMM was finally chosen over other algorithms of its category because it offers efficient parameter learning algorithms, which are actually generalisations of the standard parameter learning algorithms for HMMs [Fine et al., 1998].

The proposed algorithm receives as input action sequences illustrating prolonged, composite human activities extracted using the methodology described in Chapter 4. It then analyses sequences in a supervised manner: using a subset of labelled extracted sequences it automatically trains a combined RF and HHMM classifier; this classifier is then used to analyse novel sequences, *i.e.* sequences not used in training. The analysis results in assigning each of the novel sequences a label describing the type of activity represented in the sequence as predicted by the combined RF and HHMM classifier.

Note that the action sequences contain action boundaries (*i.e.* start

and end points of actions), in cases where concurrent actions exist in a dataset. If no action concurrency is observed (which can be determined by forming Gannt charts, such as those in Fig. 4.5) the action sequences contain simply actions. Therefore the action primitives are action boundaries when action concurrency is observed; in the opposite case, the action primitives are the actions themselves.

As results show (Section 5.4), the proposed combined model achieves higher classification accuracy than other classification frameworks both in synthetic and real data.

In the next section, a brief review of RFs is given.

5.1 Random forests

A RF is an ensemble classifier that consists of many decision trees and outputs the class that is the statistical mode of the classes output by individual trees. The method combines "bagging" concept [Breiman, 2001] and random selection of features [Ho, 2002] in order to construct a collection of decision trees with controlled variation.

For a RF consisting of N independent decision trees, the n^{th} tree of the ensemble is denoted as $f_n(x, \theta_n) : \mathcal{X} \rightarrow \mathcal{Y}$ mapping each element of the sample space $\mathcal{X}$ to a label in the label space, $\mathcal{Y}$. θ_n is a random vector containing the stochastic elements of the tree (*e.g.* the randomly subsampled training set or selected random tests at its decision nodes). The entire forest is denoted as $\mathcal{F} = \{f_1, ..., f_N\}$. The estimated probability for predicting class c for a sample is given by equation

$$p(c|x) = \frac{1}{N} \sum_{n=1}^{N} p_n(c|x), \tag{5.1}$$

where $p_n(c|x)$ is the estimated density of class labels of the leaf of the n^{th} tree where x falls with $p_n(c|x) = p_n(c|L)$. The class probability at leaf L, $p_n(c|L)$ can be directly estimated from *Eqn. 5.14*. Note that *Eqn. 5.1* is obtained in a data-driven fashion as illustrated later by its derivation in Section 5.3.1; it does not require knowledge of the forms of underlying probability distributions. Therefore its classification accuracy (as reflected in the results) depends on the representativeness of the data used for training, which is the filtered training dataset $U_{Tr,F}$ consisting of N labelled action sequences. The multi-class decision function of the forest is defined as

$$C(x) = \arg\max_{c \in \mathcal{Y}} p(c|x). \tag{5.2}$$

Breiman [Breiman, 2001] defined the classification margin of a labeled sample (x, y) as

$$m_l(x, y) = p(y|x) - \max_{\substack{k \in \mathcal{Y} \\ k \neq y}} p(k|x). \tag{5.3}$$

For a correct classification $m_l(x, y) > 0$ should hold. The generalization error is given by

$$GE = E_{(X,Y)}\ (m_l(x, y) < 0\ ), \tag{5.4}$$

where the expectation is measured over the entire distribution of (x, y). Breiman [Breiman, 2001] has shown that this error has an upper bound in the form of

$$GE \leq \bar{p}\frac{1 - s^2}{s^2}, \tag{5.5}$$

where $\bar{p}$ is the mean correlation between pairs of trees in the forest and

s is the strength of the ensemble (*i.e.*, the expected value of the margin over the entire distribution).

In the next section an overview of the Hierarchical Hidden Markov Model is given.

5.2 The Hierarchical Hidden Markov Model

Algorithms such as RFs do not encode the temporal relationships between the elements of a time sequence. In the problem domain investigated in this Thesis, these relationships are important, since the order in which the actions constituting an activity are executed is vital to the characterisation of a sequence. To encode temporal information, in this work the elements of sequences are represented as states of an activity chain, where transitions from one state to another are allowed or not subject to the set of rules that govern the event. Such representations have been employed frequently in the past. They usually take the form of the HMM and some of its variations. However, as pointed out in [Nguyen et al., 2005], these flat models cannot sufficiently represent complex activities, as they fail to model the hierarchic structure that describes them. More recent work has adopted extensions of the HMM in a hierarchical manner, such as the HHMM [Fine et al., 1998], which is used here.

Each state of the HHMM (Fig. 5.1) can either emit observations ("production states") or strings of observations ("abstract states"). Each abstract state is a sub-HHMM that can be called recursively and integrates *end states*, which signal when the control is returned to the parent HHMM. In this work discrete HHMMs are used, which are formally defined by a 3-tuple $< \zeta, \Sigma, \vartheta >$: the topological structure,

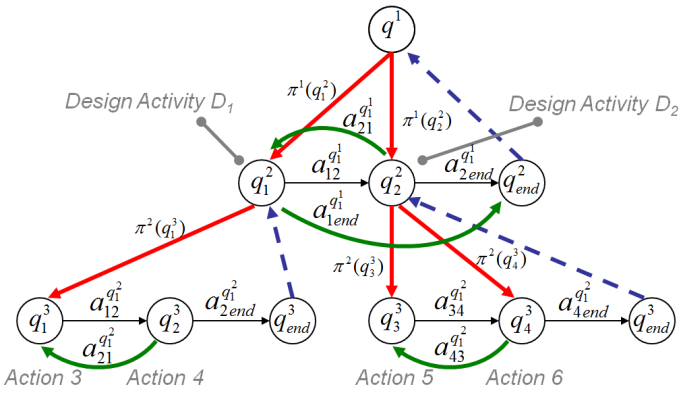


Figure 5.1: An example three-level HHMM which represents two sample design activities, D_1 and D_2. Actions below a production state represent the symbol that is emitted. Each state is symbolised as q_i^d, $d \in \{1, \ldots, D\}$ with i the state index and d the hierarchy index. Transition probabilities are denoted with $a_{ij}^{q^d}$ and the initial state probabilities with $\pi^{q^d}(q_i^{d+1})$.

ζ, defines the number of levels, the state space at each level and the parent-children relationship between levels. The observation alphabet, Σ, is the set of the symbols emitted by the model's states and the set of parameters, ϑ, includes the matrix of transition probabilities between nodes, the initial probability distribution between the children of each node and the observation probability distribution.

Denoted as Σ^* is the set of all possible activities formed by the actions in Σ. An activity can be represented as a sequence $\bar{O} = \{o_1 o_2 \ldots o_T\} = o_{1:T}$ where T the length of the sequence. Each state of the HHMM is symbolised as q_i^d, $d \in \{1, \ldots, D\}$ with i the state index and d the hierarchy index (for the root it is $d = 1$, for production states $d = D$). The number of substates of an internal state q_i^d is denoted as $|q_i^d|$. The state index will be omitted in cases where it is clear from the context and thus a state at level d will be denoted as q^d. Each level, excluding the root level has an ending state, q_{end}^d.

The state transition probabilities between the internal states at level $d+1$ are given by the matrix $A^{q^d} = (a_{ij}^{q^d})$ with $a_{ij}^{q^d} = P(q_j^{d+1}|q_i^{d+1})$ the probability of transitioning from state i to j within level $d+1$. The initial distribution over the substates of q^d is given by the vector $\Pi^{q^d} = \{\pi^{q^d}(q_i^{d+1})\} = \{P(q_i^{d+1}|q^d)\}$. Note that $P(q_i^{d+1}|q^d)$ is the probability that parent state q^d will initially activate substate q_i^{d+1}. The production states, q^D emit actions as specified by their output probability vector $B^{q^D} = \{b^{q^D}(k)\}$. In this case, $b^{q^D}(k) = P(\sigma_k|q^D)$ is the probability that q^D will produce action primitive $\sigma_k \in \Sigma$. The set of parameters for the entire HHMM can be symbolised as $\lambda = \{\lambda^{q^d}\}_{d\in\{1,\ldots,D\}} = \{\{A^{q^d}\}_{d\in\{1,\ldots,D-1\}}, \{\Pi^{q^d}\}_{d\in\{1,\ldots,D-1\}}, \{B^{q^D}\}\}$.

Algorithm 5.2.1: Learn HHMM Structure

```
// Learn HHMM structure from training sequences
V set of all possible actions in the task
l =‖ V ‖
U_{c,i}, i ∈ {1 ... N_c} a training sequence
illustrating activity class c ∈ {1 ... C}
O root node of the HHMM
for u ← 1 to C
do { for v ← 1 to N_c
     do { b^(Uu,v) binary vector of length l
          for j ← 1 to l
          do { if action primitive V_j is present in U_{u,v}
               then { b_j^(Uu,v) = 1
               else b_j^(Uu,v) = 0
     V_u = { V_k : ( Σ_{n=1}^{N_C} b_k^(Uu,n) ≥ 1 ) }
     Form a chain D_u with the elements of V_u as nodes
     Add an end state to the chain D_u
     Add a node I_u to the root node O
     Attach chain D_u to node I_u
Chain formed by nodes I_u, u ∈ {1 ... N_C} is model's
first level; add an end state to this chain.
```

The topology of the HHMM implements the rules that govern the activities taking place within the studied task. Previous approaches us-

ing HHMM for activity recognition (*e.g.* [Nguyen et al., 2005]) have used a manually specified model topology. Here, the topology of the model is learned automatically from annotated training sequences using a data-driven, heuristic approach proposed in this Thesis. The algorithm takes as input a set of labelled training sequences $U_{c,i}, i \in \{1 \dots N_c\}$, with N_c the number of sequences for each class $c \in \{1 \dots C\}$. It works as follows. The vocabulary of the task, V, which contains all possible actions in the task, is first extracted from the labelled training sequences and its length $l = ||V||$ is calculated. All sequences $U_{c,i}$ of the same class c are processed as follows. For each sequence a binary vector $b_{c,i}$ of length l is first formed. Each element, $b_j^{(Uc,i)}$ of the binary vector, corresponds to an action primitive of the task vocabulary so that $b_j^{(Uc,i)} = 1$ if action primitive V_j is present in $U_{c,i}$, otherwise $b_j^{(Uc,i)} = 0$. Then a set V_c is formed, representing the vocabulary of the class, so that $V_c = \left\{ V_k : \left(\sum_{n=1}^{N_C} b_k^{(Uc,n)} \geq 1 \right) \right\}$. This set includes all actions which appear in at least one sequence of the class c in the training dataset. From V_c a Markov chain, D_c is formed using the elements of V_c as nodes and an end node is added to the chain. A node, I_c, representing class c is added to the root O of the HHMM network and the chain D_c is attached to I_c. This procedure is repeated for all classes present in the dataset. Thus, the simplest possible 3-level HHMM is formed, where at the third level, each class is represented as a flat HMM. Pseudocode for this method is given in Algorithm 5.2.1. This structure can be optimised, given sufficient data, using Markov Chain Monte Carlo techniques [Xie, 2005]. The structural optimisation is beyond the scope of this Thesis; it will be covered in future work.

The parameters of the model are then estimated by discovering the

most probable set of parameters, λ^* with $\lambda^* = \arg\max_{\lambda} P(\{\bar{O}_t\}|\lambda)$. This is achieved with the aid of the generalised Baum-Welch algorithm [Fine et al., 1998]. The only information that the algorithm requires is the training dataset $U_{Tr,F}$ of N labelled action sequences. More details about the generalised Baum-Welch algorithm are given in Appendix 1.

5.3 The combined RF+HHMM activity analysis method

In this section a methodology is presented to analyse activities using a classifier which combines the discriminative capabilities of RFs and the efficiency of HHMMs to efficiently encode complex temporal relations between elements of an action sequence. Classification is performed in a supervised manner: a model is first learned automatically using a set of labelled sequences; this model is then used to analyse novel input sequences (*i.e.* sequences not present in the training dataset).

5.3.1 Model training

Consider a set U_{Tr} of N labelled action sequences, $U_{Tr} = \{U_{Tr,i} = (S_i, c_i)\}$ where S_i is an action sequence and c_i is the sequence's class label. The set U_{Tr} will be used to train the model. The classifier comprises two parts, a RF classifier and an HHMM classifier. The two parts are linked as follows. First, the RF classifier is trained using the labelled action sequences, U_{Tr}. During the classification process, the RF classifier assesses the significance of actions using a variable importance facility which is integrated in the RF algorithm. Using this facility, the action sequences U_{Tr} are simplified by removing actions with low importance score. Thus, a simplified (filtered) training dataset is obtained, $U_{Tr,F}$. This filtered dataset is used to train the HHMM

classifier.

Elimination of unimportant actions from the dataset reduces problem dimensionality, resulting in sequences with fewer features. Consequently, input sequences also become shorter. This allows the HHMM to operate more robustly since algorithms of its category may run into numerical underflow problem models as the length of the observation sequence increases [Bui et al., 2004].

Part I: RF Classifier Training (discriminative action primitive classification)

For the RF classifier, each tree is grown as follows: a bootstrapped [Efron and Tibshirani, 1994] sample of the training dataset is taken for the tree which is denoted as U_{Bt}. For each non-leaf node of the tree a split function has to be defined

$$f_{\Phi}(U_{Bt,i}) \in \{0, 1\} \tag{5.6}$$

which provides the optimal separation of the training sequences. The function evaluates one or more features of sequence $U_{Bt,i}$ and decides about whether it will be sent to the left ($f_{\Phi}(U_{Bt,i}) = 0$) or right child ($f_{\Phi}(U_{Bt,i}) = 1$) of the node. With Φ the set of parameters of the split function are denoted. These parameters are optimised during the training process, which is summarised in the following steps:

1. The algorithm starts at the root node of the tree with the training set $U_{Bt} = A_{node}$.

2. A random set of parameters, $\Phi = \{\phi_k\}$ is generated.

3. The training set A_{node} is divided into two subsets, A_L and A_R $\forall \phi \in \Phi$ as follows:

$$A_L(\phi) = \{U_{Bt,i} \in A_{node} | f_\phi(U_{T,i}) = 0\} \tag{5.7}$$

$$A_R(\phi) = \{U_{Bt,i} \in A_{node} | f_\phi(U_{T,i}) = 1\} \tag{5.8}$$

4. The split parameters ϕ^* are selected so that they optimise a gain function g with:

$$\phi^* = \arg\max_{\phi \in \Phi} g(\phi, A_{node}) \tag{5.9}$$

where the gain function g is given in the equation:

$$g(\phi, A_{node}) = H(A_{node}) - \sum_{M \in \{L,R\}} \frac{|A_M(\phi)|}{|A_{node}|} H(A_M(\phi)). \tag{5.10}$$

The function H measures the gain of the classification accuracy of the children nodes in comparison to the current node. The following entropy-based classification function H is given in [Gall et al., 2012]:

$$H(A_{node}) = -\sum_c p(c|A_{node}) log(p(c|A_{node})) \tag{5.11}$$

where the class probability, $p(c|A_{node})$, can be calculated from the equation

$$p(c|A_{node}) = \frac{|A_c^{node}| \cdot r_c}{\sum_c (|A_c^{node}| \cdot r_c)} \tag{5.12}$$

with A_c^{node} the set of sequences with class label c reaching the studied node after training and

$$r_c = \frac{|A|}{|A_c|} \tag{5.13}$$

where A_c the set of sequences with class label c within the whole training set, A.

5. If the stopping criteria are not satisfied, the tree continues to grow using the subsets A_L and A_R . Else a leaf node is created which stores the statistics of the training data A_{node}. Therefore the class probability $p(c|L)$ at leaf L can be estimated with the equation

$$p(c|L) = \frac{|A_c^L| \cdot r_c}{\sum_c (|A_c^L| \cdot r_c)} \tag{5.14}$$

Part II: Selecting *good* features

One of the advantages of using RF is that they integrate a variable importance facility which assesses the significance of each feature participating in the classification process [Breiman, 2001, Genuer et al., 2010]. The algorithm operates as follows:

1. For every tree in the forest, the classification of the out-of-bag (OOB) samples of the training set is predicted and the misclassification rate is estimated. The OOB_t samples for a tree t are defined as the training samples not used during the construction of t. The misclassification rate is defined as the tree's out-of-bag error.

2. Values of every variable in the tree are permuted and compute

the out-of-bag error is estimated. By comparing this error to the misclassification rate of the tree an indication of the variable's importance is obtained. The increase of misclassification rate is defined as the variable's importance measure for the tree.

3. Out-of-bag errors and importance measures from all trees in the forest are then aggregated to obtain the overall out-of-bag error rate and variable importance measures.

Part III: HHMM Training (temporal model)

After selecting the important features using the Variable Importance assessment method, non-important features are removed from the original training sequences. Therefore, a filtered training dataset $U_{Tr,F}$ of N labelled action sequences is obtained. The filtered dataset is used to train the HHMM. Using Algorithm 5.2.1 the HHMM's structure is obtained and model's parameters are learned using the methodology found the methodology in [Fine et al., 1998] which is given in Appendix 1.

5.3.2 Activity identification

Consider the task of classifying a test dataset, U_{Te} of M labelled action sequences not included in the training dataset $U_{Tr,F}$. The process is as follows:

Part I: Sequence denoising

Non-important elements (*i.e.* actions with low importance score as given by the variable importance facility of RFs in section 5.3.1, Part II) detected by RF during the training process are removed from the

test dataset and thus the filtered test dataset, $U_{Te,F}$ is obtained. Elimination of unimportant actions from the dataset reduces problem dimensionality, resulting in sequences with fewer features. Therefore, the sequence denoising stage improves the computational efficiency of the proposed algorithm.

Part II: RF Classification

The filtered test dataset $U_{Te,F}$ is passed to the RF classifier. Classification process results in mapping the elements of $U_{Te,F}$ to classes corresponding to correctly executed activities $U_{Te,F,Corr}$ and erroneously executed activities $U_{Te,F,Err}$. The RF classification step classifies input sequences using their discriminative features. A high-level, qualitative explanation of this step is that the algorithm, by taking into consideration the discriminative features of the activities, which were learned in the training stage, makes an initial decision regarding the type of activity performed in each input sequence. This step is necessary, since the temporal model used later in the algorithm does not have strong feature discriminative properties, since it lacks a facility which can perform classification by taking into account the absence or presence of features in a sequence and their frequencies. On the other hand, discriminative feature classification, as performed in this stage, does not take into account the ordering of the features. Therefore it is complemented with a temporal model, as described in Part IV. The next part describes how the discriminative and the temporal models are linked.

Part III: Linking RF with HHMM

This part describes how the proposed algorithm links the discriminative model (RF) with the temporal model (HHMM). Two subsets of the filtered test dataset, $U_{Te,F}$ are passed to the HHMM: the first contains all sequences marked as correct by the RF classifier; the second comprises sequences which, although were marked as erroneous by the RF, have close proximity to the correct executions of an activity according to a similarity criterion proposed here. The idea behind the similarity criterion used here is that, if a sequence is classified as "erroneous" by the RF classifier, but contains all important elements of an activity, as detected by the variable importance facility of RFs, it was potentially misclassified by the RF classifier; therefore the responsibility for the final classification decision is passed to the temporal model. This part aims at reducing the classification error of the RF classification stage; it comprises six steps.

At a high level, the process can be described as follows. In the first two steps, a model is built for each correctly executed activity, which is in the form of a binary vector. Each element of the binary vector corresponds to an important element (the important elements for the dataset are those detected in section 5.3.1, Part II). If an important element exists in all training sequences of a correctly executed activity, its corresponding element in the binary vector representing this activity is equal to one; else it is equal to zero. In the third step, each sequence classified as erroneous is converted to a binary vector so that, if it contains an important element, the element in its binary vector corresponding to this important element is equal to one; else it is equal to zero. Step four estimates the similarity of each sequence classified

as erroneous with each model of correctly executed activities. Step five selects the sequences, which, although classified as erroneous, are given a "second chance" and are therefore passed to the temporal model. Note that the sequences classified as "correct" by the RF classifier are unaffected by the process so far. In step six, sequences classified as "correct" by the RF classifier and sequences which, as described above, will be given a "second chance" are passed to the temporal model.

The whole process is described in detail below.

1. A binary vector, $b^{(Imp)}$ is formed which represents the important variables of the dataset. If I_{mp} is the set of important elements, the length l of $b^{(Imp)}$ is $l =\parallel I_{mp} \parallel$. Furthermore, it is $b_j^{(Imp)} = 1$, $j \in \{1 \dots l\}$.

2. For all correctly executed activities of the filtered training dataset, $U_{Tr,F,Corr}$ binary vectors $b^{(U)}$ are formed of length l such that, if an important variable, j is present in a sequence then $b_j^{(U)} = 1$ else $b_j^{(U)} = 0$. The *important variable vector* for a correctly executed activity, B is defined as:

$$V_B = \left\{ j : \left(\sum_{i=1}^{N} b^{(U)} = N \right) \forall j \in V_B \right\} \tag{5.15}$$

The *binary importance vector* of B is defined as a binary vector of length l with $b_j^{(B)} = 1$, if $j \in V_B$, otherwise $b_j^{(B)} = 0$. This vector contains all actions present in all the correct executions of the activity B. This vector is used later in the algorithm as a means of measuring the similarity of a test sequence to the model of correct executions of an activity.

3. For all sequences classified as erroneous, $U_{Te,F,Err}$ binary vectors $b^{(U)}$ are formed of length l such that, if an important variable, j is present in a sequence then $b_j^{(U)} = 1$ else $b_j^{(U)} = 0$.

4. Similarity score $Sim(B, b^{(U)})$ is estimated as follows:

$$Sim(B, b^{(U)}) = \sum_{j=1}^{l} b_j^{(U)} \cdot b_j^{(B)}. \tag{5.16}$$

5. A set of sequences of erroneously executed activities which satisfy the condition $Sim(B, b^{(U)}) = \| V_B \|$ is formed (where $|| \cdot ||$ stands for the length of the vector) and is denoted with $U_{Te,F,Pas}$. Sequences in this set, as stated by the condition, contain all actions encountered in all correct executions of an activity in the training dataset. These sequences are given a "second chance" since they include all steps present in all correct executions of an activity.

6. Sequences of sets $U_{Te,F,Corr}$ and $U_{Te,F,Pas}$ are passed to the HHMM.

Part IV: HHMM Classification

Inference for the sequences of the sets $U_{Te,F,Corr}$ and $U_{Te,F,Pas}$ is performed with the generalised Viterbi algorithm [Fine et al., 1998]. More details about this algorithm are given in Appendix 2. The generalised Viterbi requires only the sets $U_{Te,F,Corr}$ and $U_{Te,F,Pas}$ and assigns a class to each sample of the sets.

Part V: Activity Characterisation

The class assigned to each of these sequences by the HHMM is the output of the combined RF+HHMM algorithm. Sequences of set $U_{Te,F,Err} \setminus$

$U_{Te,F,Pas}$ (here the symbol '\' denotes deduction) retain the class label assigned to them by the RF classifier.

In the next Section classification results for the proposed algorithm, RF+HHMM are presented in two datasets. Further experiments on real life data are presented in Chapters 6 and 7.

5.4 Results: activity identification and error detection

In this section the classification accuracy of the proposed algorithm RF+HHMM is assessed in two datasets.

An everyday human activity classification problem is investigated in Section 5.4.1 where the task is to discriminate between activities relevant to meal preparation. The purpose of this experiment is to demonstrate the proposed algorithm's ability to classify relatively simple action sequences arising from everyday activities with high classification accuracy. This dataset consists of time sequences comprising discrete data. Complexity of this dataset is then increased by synthetically adding erroneous variations of the existing everyday activities. This addition aims at demonstrating that the proposed algorithm is capable of detecting mistakes. Finally, the dataset is modified using synthetically generated noise which perplexes the identification task. This experiment aims at demonstrating the proposed algorithm's resilience to noise.

In Section 5.4.2 a threat detection problem from a publicly available dataset is considered. Specifically, the task is to discriminate between the similar actions *draw gun* and *raise hand pointing forward*. In this case time sequences comprise continuous data. The purpose of this experiment is to demonstrate the proposed algorithm's ability to work

with continuous data, although it was not specifically designed to do so.

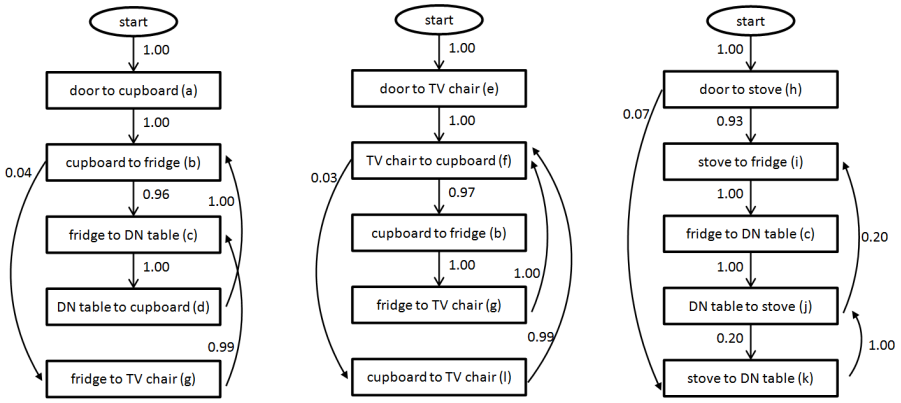
In both scenarios the proposed algorithm is compared against several current activity identification methods.

5.4.1 Everyday activity problem

In this section, experimental results of the method described in this chapter are presented on a dataset derived from real world data. More specifically, the activity representation model described in [Nguyen and Venkatesh, 2005] is replicated and used to generate action sequences for three activity classes which are *have coffee*, *have snack* and *have meal*. The structures of these activities, as given in [Nguyen and Venkatesh, 2005] are shown in Fig. 5.2. Examples of action sequences expected from these structures are: *start, door to cupboard, cupboard to fridge, fridge to dining table, dining table to cupboard, cupboard to fridge, fridge to TV chair* (have coffee); *start, door to TV chair, TV chair to cupboard, cupboard to fridge, fridge to TV chair, TV chair to cupboard, cupboard to TV chair* (have snack); *start, door to stove, stove to fridge, fridge to dining table, dining table to stove, stove to dining table* (have meal).

Activity Classification and Error Detection

One of the most important aspects in this Thesis is the detection of errors during execution of complex activities. However the activity model of [Nguyen and Venkatesh, 2005] only generates action sequences illustrating correctly performed activities. To produce sequences of erroneously executed activities, the activity generators of [Nguyen and



(a) (b) (c)

Figure 5.2: The structures of everyday activities studied in [Nguyen and Venkatesh, 2005]. **(a):** Have coffee, **(b):** have snack, **(c):** have meal.

Venkatesh, 2005] are modified and the output sequences of the modified generators are regarded as erroneous. The modification of the activity generators is done by hand; random noise is added later to the dataset to alleviate potential bias which could be introduced by the hand-coding. Six activity classes are now present, the original three (*have coffee*, *have snack* and *have meal*) and their erroneously executed counterparts. The erroneous sequences introduced by the modified generators illustrate two types of erroneous activity executions:

Type I: An action primitive is missing from the sequence. Erroneous executions of activities *have coffee* and *have meal* fall into this category.

Type II: No action primitive is missing from the sequence but action primitives are performed in a different order than the normal behaviour. Erroneous executions of activity *have meal* fall into this category.

For each of the activity classes 25 action sequences are obtained, 15 of which are used for training and 10 for testing (thus keeping the train-

ing/testing samples ratio identical to [Nguyen and Venkatesh, 2005]). The dataset comprises 90 training and 60 testing action sequences. The vocabulary of the dataset consists of 12 primitive actions (the same as in [Nguyen and Venkatesh, 2005]).

Four algorithms are tested against the proposed algorithm, RF + HHMM, specifically HHMM (used for activity recognition in [Nguyen and Venkatesh, 2005]), Suffix Trees (used for activity recognition in [Nguyen and Venkatesh, 2005]), RFs and SVMs. A three-level HHMM was learned using algorithm 5.2.1 from the data; it is shown in Fig. 5.3. In Fig. 5.4 (a) classification results are shown in the form of ROC curves. It is observed that three methods, RF+HHMM, HHMM and Suffix Trees achieve maximum accuracy (100% classification accuracy with area under curve (AUC) = 1). RFs and SVMs on the other hand misclassify several sequences of Type II. This result was expected since these algorithms do not encode temporal relations between an activity's constituent actions which is the challenge posed by sequences of Type II.

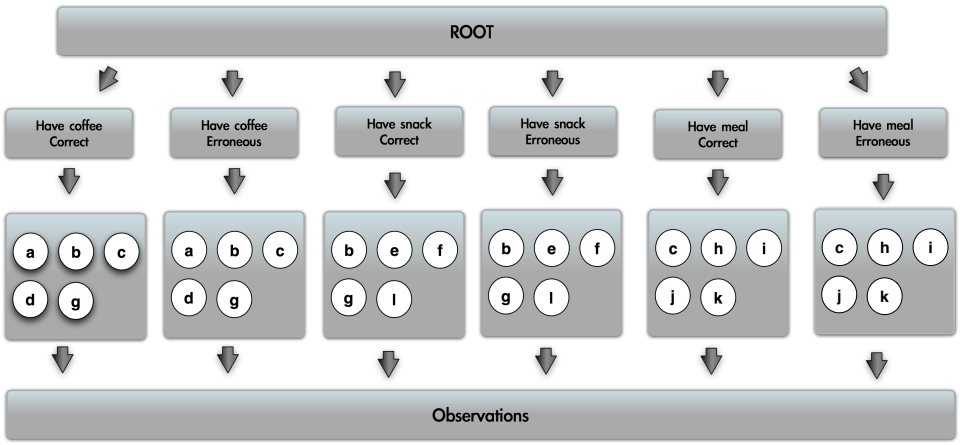
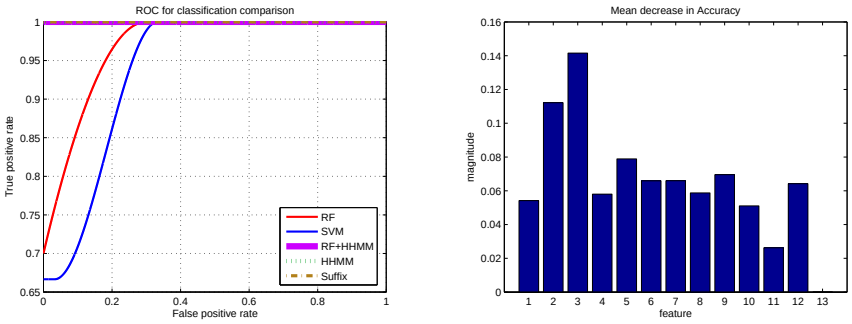


Figure 5.3: The learned three-level HHMM representing activities performed in the everyday activity problem. Each action primitive is represented with a letter using the mapping of Fig. 5.2.



(a) (b)

Figure 5.4: **(a):** ROC curves illustrating classification accuracy **(b):** Variable importance.

In Fig. 5.4 (b) the importance of actions present in the dataset is shown as determined by the RF variable importance module (Section 5.3.1). At this stage no action primitive in the dataset can be characterised as *unimportant*.

Further commenting on the results, the classification accuracy of HHMMs reported in [Nguyen and Venkatesh, 2005] (100%) is validated by the experiment presented in this section. However, since three different algorithms achieved maximum accuracy a concern is raised regarding the difficulty of the classification task in this dataset. Therefore the results presented in this section serve as a *sanity check* for the proposed method. In the next section difficulty of the classification task is increased with the addition of noise to the original dataset.

Adding noise to the original dataset

The purpose of the algorithm presented in this Chapter is to classify complex human behaviour. When the sequences representing activities only contain actions directly related to the activity performed, encoding temporal relations between actions and determining discriminative features is an easy task for modern classification algorithms as illus-

trated in Section 5.4.1. However in action sequences actions which are *irrelevant* to the performed activity are sometimes encountered. For example, a human might pick up the phone to answer a phone call during the execution of the *washing clothes* activity. In this case the action *pick up phone* is irrelevant to the *washing clothes* activity. In this section it is investigated how such actions can influence the performance of current activity classification algorithms.

The effect of *random, irrelevant* actions is simulated here with the addition of noise to the original dataset. Towards this effort 14 new features (actions) are inputted in the original dataset in places determined by the van der Corput low discrepancy sequence [van der Corput, 1935a, van der Corput, 1935b]. Appendix 6 explains how this sequence is generated. The amount of noise added to the dataset is determined as follows: the irrelevant/relevant actions ratio (IRR) is estimated for the dataset presented in chapter 7 which comprises action sequences recorded during the execution of a complex engineering task. This dataset is used as a reference to estimate the amount of added noise because it illustrates a typical prolonged, composite human activity and therefore contains an amount of noise which is representative for activities of this category. If for the dataset in chapter 7 it is $IRR = a$, noise is added to the dataset described in Section 5.4.1 so that its IRR equals a. The resulting noisy dataset will be referred to as *basis noise dataset*.

The same algorithms are tested in this new, noisy dataset and the results are shown in Fig. 5.5 (a). In Fig. 5.5 (b) the importance of actions present in the dataset is shown as determined by the RF variable importance module (Section 5.3.1). For the temporal encoding part of

the proposed algorithm RF+HHMM actions with variable importance (VI) less than 0.01 are discarded. It is observed that classification accuracy for HHMM and Suffix Trees significantly deteriorates under the effect of noise. On the other hand, the proposed method RF+HHMM is not affected by noise since the variable importance module used successfully discards most irrelevant features. Further experiments with added noise are carried out. Specifically, 20% - 80% noise is added to the *basis noise dataset* and the results are shown in Fig. 5.5 (a, c, e) and Fig. 5.6 (a, c). All results from experiments where noise is added to the *basis noise dataset* are aggregated in Fig. 5.7 where the effect of added noise to the area under the ROC curves for the tested algorithms is illustrated. Suffix Trees and SVMs are the algorithms most affected by noise whereas HHMM and RF are more resilient. The proposed RF+HHMM algorithm is not affected by noise in this experiment. Note that the feature selection facility used is part of the RF algorithm employed in the first classification step of the proposed RF+HHMM method. Therefore no external feature selection algorithm is used.

Performance Interpretation

The result tables show that RFs and SVMs have problems with distinguishing between activities of Type II (no action primitive missing from sequence but action primitives are performed in a different order than the normal behaviour). Since these algorithms do not take into account the temporal order between sequence elements, they cannot differentiate efficiently between these two activity classes.

Regarding the performance of HHMM and Suffix Tree algorithms,

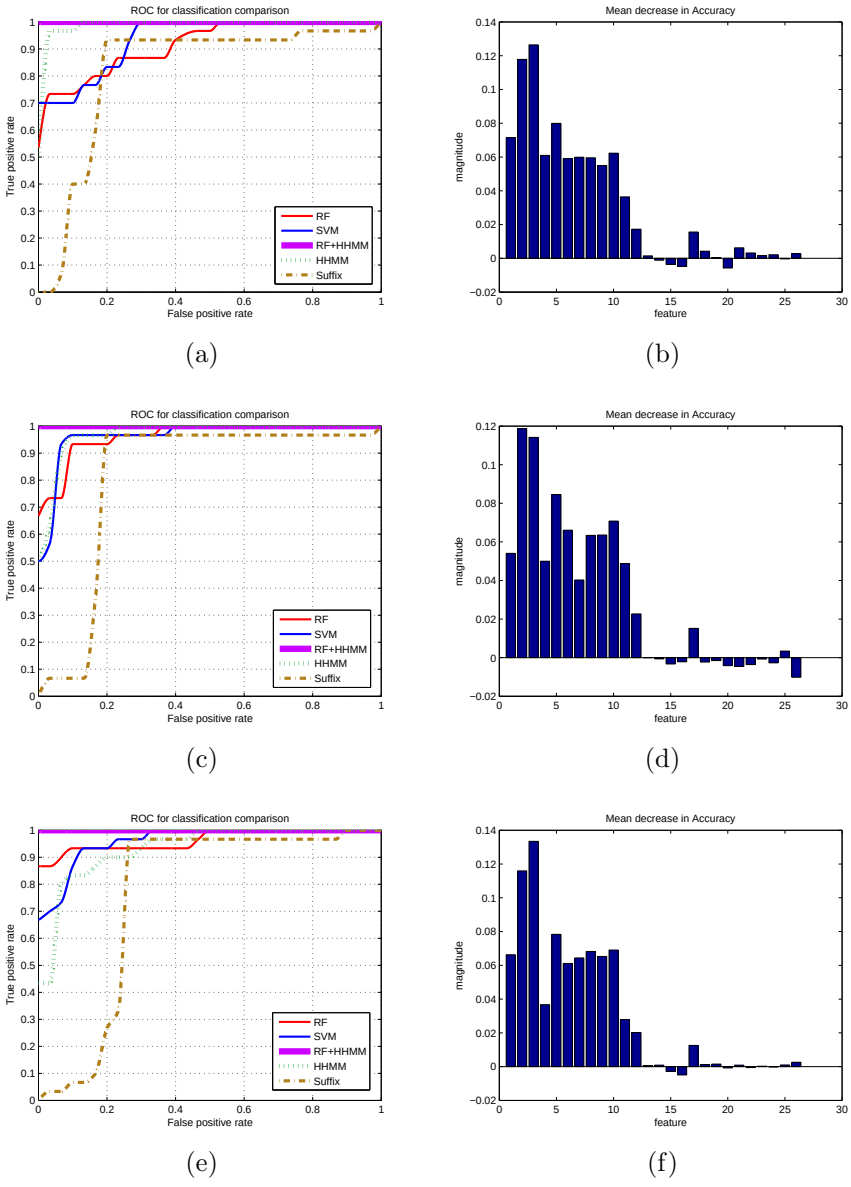


Figure 5.5: ROC curves illustrating classification accuracy under the effect of noise **(a):** Base dataset, **(c):** Added 20% noise, **(e):** Added 40% noise. Feature importance for each case shown in **(b)**, **(d)**, **(f)**.

performance decrease is directly related to noise increase. When noise is added, their accuracy deteriorates.

On the contrary, it is observed that the algorithm proposed in this Chapter is capable of encoding complex temporal relations between an activity's constituent actions and is additionally resilient to noise. Note

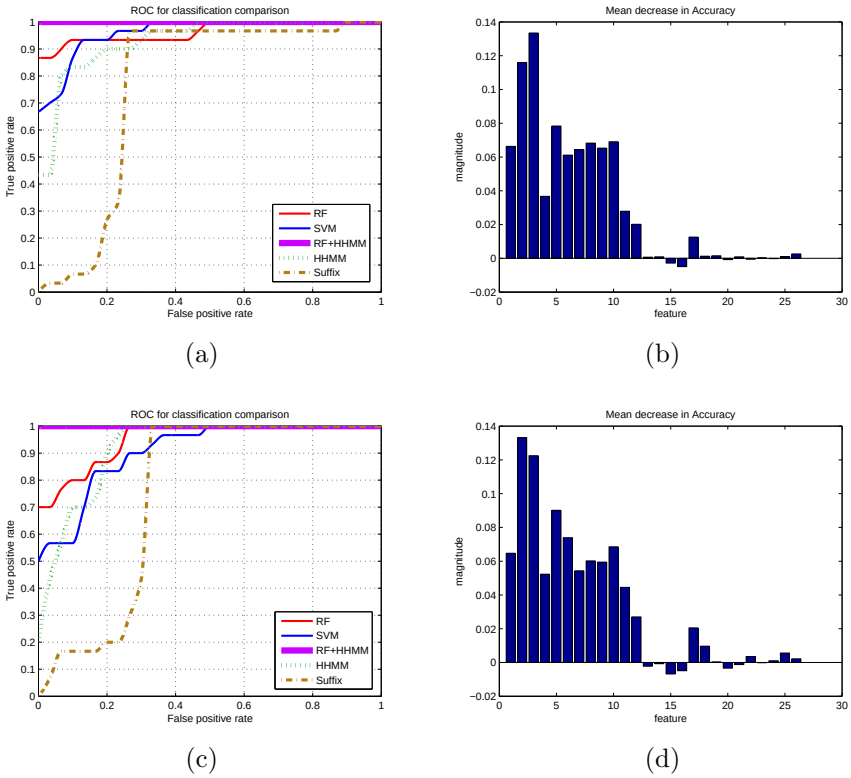


Figure 5.6: ROC curves illustrating classification accuracy under the effect of noise **(a):** Added 60% noise, **(c):** Added 80% noise. Feature importance for each case shown in **(b)**, **(d)**.

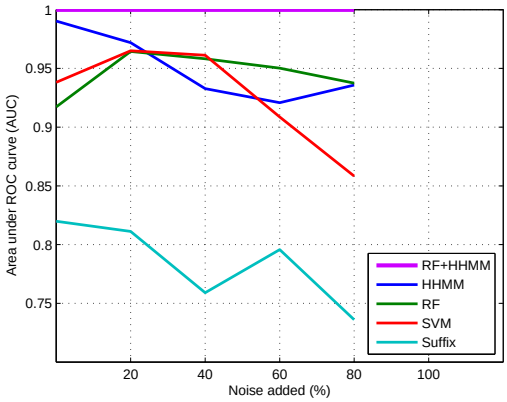


Figure 5.7: Area under curve measurements for noise added to the *basis noise dataset*.

that the feature selection facility used is part of the RF unit employed in the first classification step of the proposed RF+HHMM method. Therefore no external feature selection algorithm is used.

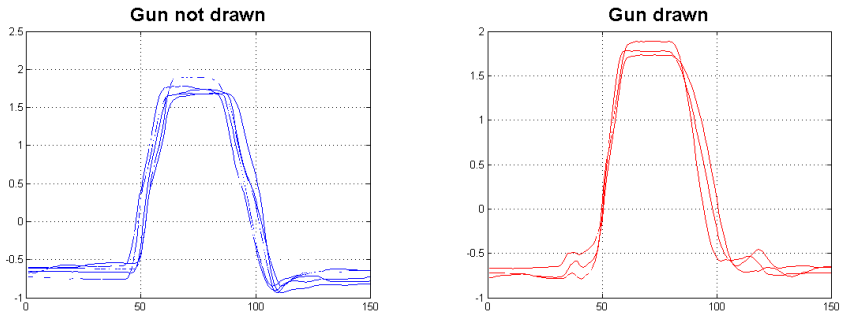
5.4.2 Continuous data problem

In this section, experimental results of the method described in this Chapter are presented on the publicly available Gun-Point dataset [Ratanamahatana and Keogh, 2004]. The purpose of this experiment is to demonstrate the proposed algorithm's ability to work with continuous data, although it was not specifically designed to do so. The dataset has two classes, each containing 100 sequences. 50 of these sequences are used for training and 150 for testing; the partitioning recommended by the authors is used in the experiments. All instances are of the same length (150 data points). The two classes are:

Gun Draw: The actors start with their hands by their sides. They then draw a gun from a hip-mounted holster, point it to a target for approximately one second, then return the gun to the holster, and their hands to their sides.

Point: The actors start with their hands by their sides. They then point with their index fingers to a target for approximately one second, then return their hands to their sides.

Data in this experiment was captured by placing a video tracker on the centroid of the right hand in both the horizontal and vertical axes. However, the dataset contains the track of the hand in the vertical axis only. The resulting data is in continuous form. Sample trajectories from the Dataset are shown in Fig. 5.8. Since the proposed algorithm RF+HHMM works with discrete data, continuous features are discretised into 40 bins using equal width discretisation.



(a) (b)

Figure 5.8: Sample trajectories from the Gun-Point dataset**(a):** Class *Point*, **(b):** Class *Gun Draw*.

Activity Classification and Error Detection

The proposed RF+HHMM algorithm operates in datasets where correct and erroneous executions of an activity are present. Therefore in the Gun-Point dataset, one of the two classes has to be considered as *correct* and the other as *erroneous*. To determine this, the training dataset is split into two parts and classification is performed using one part as a training subset and the other as testing subset. A bootstrapped sample of the training dataset consisting of 35 sequences is the training subset; the remaining 15 sequences are used for testing. Two experiments are carried out: in the first *Gun Draw* class is selected as *correct* and in the second class *Point* is selected as *correct*. Performance of RF+HHMM is measured in both experiments. Since the algorithm performs better in the second experiment, class *Point* is selected as correct.

The four algorithms tested against the proposed algorithm RF + HHMM in Section 5.4.1, specifically HHMM, Suffix Trees, RFs and SVMs are tested in this dataset too. In Fig. 5.9 and Table 5.1 classification results are shown in the form of ROC curves and correct classification percentages respectively. It is observed that the proposed method RF + HHMM yields higher classification accuracy compared

to the other methods tested. Note that the feature selection facility of the proposed RF+HHMM algorithm was not used in this experiment.

It is noteworthy that the performance of SVMs and RFs is considerably inferior compared to that of the rest of the algorithms. This is due to the fact that SVMs and RFs ignore the temporal relationships between the elements of the sequences. Since the Gun Point dataset consists of time series, temporal relationships are a prominent characteristic of the dataset's sequences which explains the low classification accuracy outputted by these two methods.

It is also noteworthy that both discriminative feature methods exhibit high accuracy in the Point class and low accuracy in the Gun Draw class. This is attributed to the fact that only a small percentage of the total number of sequences in the dataset is used for training (25%) is used for training; in most applications, this percentage is usually above 50%. Thus, both of the discriminative models built are not descriptive enough to discriminate between the two classes: in the case of SVMs, all testing samples are assigned to a single class; in the case of RFs, 79% of the testing samples were attributed to a single class. Consequently, this leads to high detection rate for one class and low for the other.

	Gun Point Classes		
Methods	Gun Draw	Point	Total
RF+HHMM	0.96	0.78	0.86
HHMM	0.91	0.76	0.83
RF	0.42	1.00	0.71
SVM	0.00	1.00	0.49
Suffix	0.93	0.65	0.79

Table 5.1: Classification accuracy in the Gun Point dataset.

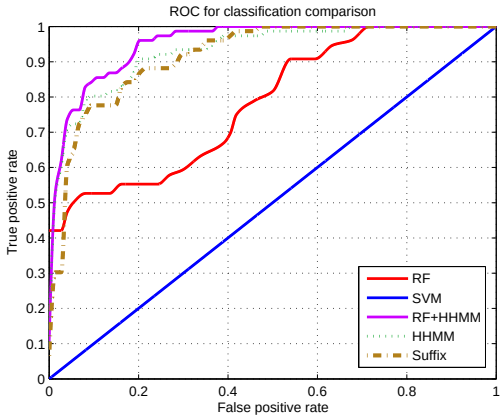


Figure 5.9: Classification accuracy in the Gun Point dataset.

5.5 Summary

In this chapter the novel RF + HHMM classification method was presented, which was used here to analyse complex human activities. In contrast to existing approaches the proposed method possesses both discriminative and temporal encoding properties; these properties are essential for modelling prolonged, composite activities. The proposed technique operates in a supervised manner: a combined classifier based on RFs and HHMMs is first built using training data; novel data (*i.e.* action sequences not used during training) is then fed to the model which maps input action sequences into classes corresponding to types of correctly executed activities and classes corresponding to erroneous activity executions. Results in two datasets derived from real data show that the proposed method is capable of encoding complex temporal relations between an activity's constituent actions and is additionally resilient to noise.

Chapter 6

KEY ACTION DISCOVERY SYSTEM

The novel Key Action Discovery (KAD) system is presented in this chapter, designed to discover important, *key* action primitives in action sequences illustrating prolonged, composite human activities with the goal of improving classification accuracy in activity recognition applications. The proposed method comprises two phases:

1. Exclusion of sequence elements (action primitives) from the dataset which perplex the activity classification task. This phase is performed in an *unsupervised* manner using a data-driven approach inspired from Natural Language Processing statistics and is discussed in Section 6.2.

2. Detection of important, *key* sequence elements (key action primitives). This phase treats action sequences as bags-of-words [Salton and McGill, 1986] and utilises a training dataset consisting of labelled action sequences. The labels are high level descriptions of the activity illustrated in each sequence (an example label is (*washing clothes - correct execution*). This phase is analysed in Section 6.3.

The contribution presented in this Chapter is the KAD algorithm, a method for identifying unimportant and important action primitives in action sequences arising from the execution of prolonged, composite human activities. Furthermore, it is described how this method can be combined with the RF + HHMM algorithm proposed in Chapter 5 and several other widely used classifiers. It is experimentally shown that these combinations offer improved accuracy in classification. Previously proposed feature selection methods cannot efficiently detect important and redundant action primitives in action sequences illustrating prolonged, composite human activities.

The next section presents the definitions of certain action primitive types which will be used in this Chapter.

6.1 Action primitive types definitions

In this Thesis an activity is defined as a sequence of action primitives. Some of these action primitives are critical for the execution of a certain activity *e.g.*, one cannot wash clothes without switching on the washing machine. Such important action primitives are defined as *key* action primitives for a specific activity. On the other hand, in action sequences, action primitives which are *irrelevant* to the performed activity are sometimes found. For example, a human might pick up the phone to answer a phone call during the execution of the *washing clothes* activity. In this case the action primitive *pick up phone* is irrelevant to the *washing clothes* activity. In complex activities a third category of action primitives is often encountered, in specific those action primitives which although are *relevant* to the executed activity they do not alter its semantic meaning. For example, during the activity *take literature*

exam a candidate will certainly perform the action primitive *write* and might also need to *erase*. However, presence of action primitives *writing* and/or *erasing* in a sequence does not alter its semantic meaning: one cannot determine if the candidate's answers were correct or erroneous by the presence or absence of these two action primitives. Furthermore, since *writing* and *erasing* can appear in a large variety of contexts, they cannot help in differentiating between activities such as *take literature exam*, *take law exam* or even *write letter*. On the contrary, they enlarge the computational burden required to analyse complex behaviour since their presence results in longer sequences and further complicates the temporal relations between an activity's constituent action primitives. Such action primitives are defined as *common* action primitives in this Thesis. Successful detection of common action primitives and their elimination from the dataset reduces problem dimensionality as it simplifies input sequences allowing hence the dataset contains fewer features. Dimensionality reduction also means that the dataset sequences are shortened. This allows the HHMM which is used in the classification stage to operate more robustly since algorithms of its category may run into numerical underflow problem models as the length of the observation sequence increases [Bui et al., 2004].

The idea behind the proposed KAD method is to first detect and remove *common action primitives* from the dataset and then use context statistics to discover *key* action primitives.

6.2 Detecting *common* action primitives

The vocabulary relevant to a specific task (*e.g.* bridge design or meal preparation), V_T, is defined as the set which includes all possible action

primitives in the context of this task. The next step is to discover the set of *key* action primitives for each individual activity in the task. *Key* action primitives are important for the execution of certain activities while *common* action primitives such as *erasing* do not alter the semantic meaning of any performed activity and can therefore be ignored during the activity identification process.

The concept of *common* actions is new in the area of activity recognition. Related schemes are present in literature but serve different purposes: *e.g.* in [Hamid et al., 2005] a scheme of *deficient* and *extraneous* action primitives was proposed. However, this classification aims at defining action primitives that could help discriminate between two or more activities. In the work described in this Thesis, such methods are not readily applicable since at least two different activities (*e.g.* correct and erroneous execution of a task) could comprise the same action primitives but differ in the order that these same action primitives are executed. Rather than classifying activities, identification of *common* action primitives aims at denoising the dataset to allow a later classification stage to operate efficiently in noisy environments.

Eliminating *common* action primitives from an action sequence is closely related to the discovery of "stop words" (articles, prepositions *etc*) in the field of document analysis [Salton, 1971, Salton and Lesk, 1968]. Similar concepts have been applied for genome comparison and clustering in Bioinformatics [Miller and Attwood, 2001]. Conventionally, a list of "stop words" is prepared manually. However, this process is subjective and may not be optimal for every application domain.

In this Thesis a data-driven algorithm (Alg. 6.4.1) based on sliding context windows (Fig. 6.1) [Chotimongkol, 2008] is proposed to elim-

inate "common action primitives". The algorithm is now described in detail. First, all context sliding windows in the dataset are detected. For an action sequence S, with $S = \{w_1, w_2, w_3, ..., w_\nu\}$ the context sliding windows are given by the set of bi-grams $\{w_1w_2, w_2w_3, ..., w_{\nu-1}w_\nu\}$. Let us denote the total number of context sliding windows in the dataset with z and the frequency of each action primitive in the dataset with f_w. For each action primitive w present in the dataset, its regularity count, $RC(w)$ is defined as the number of context sliding windows in which w participates (Fig. 6.1). The action primitive's regularity weight, $RW(w)$ is estimated as $RW(w) = \frac{z-RW(w)}{z}$. By multiplying the regularity weight with the frequency of the action in the dataset, the regularity (Reg) of the action primitive is obtained. The feature vector of the action primitive, M_w consists of the tuple $< Reg, RC(w) >$, *i.e.* its regularity and regularity count.

The proposed algorithm is related to [Chotimongkol, 2008] with the important difference that in [Chotimongkol, 2008], a threshold is applied in the final classification stage which is usually learned from annotated data. In the research area covered in this Thesis, such annotated data is not available. Therefore *common action primitives* are discovered in an unsupervised manner using k-means (Alg. 6.4.1, last line) [Duda et al., 2000].

In this algorithm, the rows of matrix $\mathbf{M_n}$ which correspond to common action primitives are removed and the resulting matrix, $\mathbf{M_{res}}$ is retained. Having found the set of *common* action primitives, V_U, it is subtracted from V_T, which gives the set of task *key* action primitives, $V_{T,C}$:

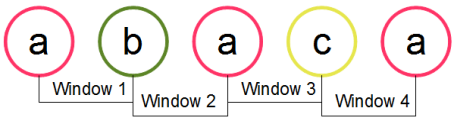


Figure 6.1: Using context sliding windows to calculate regularity count $RC(w)$ for each action primitive w in the sequence $\{a, b, a, c, a\}$. It is $RC(a) = 4$, $RC(b) = 2$, $RC(c) = 2$.

$$V_{T,C} = V_T \setminus V_U. \tag{6.1}$$

6.3 Discovering *key* action primitives

To discover the set of *key* action primitives for an individual activity, $V_{A,C}$, the following method is proposed: For every sequence describing A in the training set, $S_i, i \in \{1 \dots N\}$, a subsequence of length l, is formed, with $l = \| V_{T,C} \|$. A binary vector $w^{(Si)}$ of length l represents the presence or absence of each of the possible action primitives in S_i such that, if an action primitive j is present in S_i, $w_j^{(Si)} = 1$, otherwise $w_j^{(Si)} = 0$. Then, the set of key action primitives for A is defined as:

$$V_{A,C} = \left\{ j : \left(\sum_{i=1}^{N} w_j^{(Si)} = N \right) \forall j \in V_{A,C} \right\} \tag{6.2}$$

which means that a *key* action primitive for an activity A is a task *key* action primitive which is, in addition, present in all training sequences describing A. The characteristic vector of A is defined as a binary vector of length $l = \|V_{T,C}\|$ with $w_j^{(A)} = 1$, if $j \in V_{A,C}$, otherwise $w_j^{(A)} = 0$.

6.4 Sequence classification

In order to classify a test sequence X, its similarity to each activity A can be computed in three steps:

1. Obtain set $X' = X \cap V_{A,C}$.

2. Convert X' to a binary vector $w^{(X')}$ as above.

3. Compute similarity score, $Sim(A, X)$ as follows:

$$Sim(A, X) = \sum_{j=1}^{l} w_j^{(X')} \cdot w_j^{(A)}. \tag{6.3}$$

If $Sim(A, X) = \|V_{A,C}\|$ for an activity class, then input vector X is assigned to this class. If this condition applies to more than one class, X is temporarily assigned to all classes satisfying the criterion. Disambiguation is achieved with the aid of the activity structure investigation method discussed in Section 6.5. If $Sim(A_k, X) < ||V_{A,C}||$ for all classes, X is characterised as an erroneous activity.

For an erroneous activity it is also important to find the type of the activity which was erroneously performed. To find this information, the proximity of an erroneous activity to each class of correctly executed activities is evaluated by computing the similarity between their characteristic vectors. Input vector X is considered as an erroneous execution of the activity class A_k for which the similarity score $Sim(A_k, X)$ is maximised and therefore:

$$k^* = \arg\max_{k \in \{1 \ldots c\}} Sim(A_k, X). \tag{6.4}$$

where c is the number of distinct correctly executed activity types.

Algorithm 6.4.1: DETECT COMMON ACTIONS

// Identify common actions in dataset D

z total number of context windows in D

f_w frequency of action primitive w in D

$RC(w)$ regularity count of w in D (Fig. 6.1)

$n = \|V_T\|$

for $w \leftarrow 1$ **to** n

do
- Estimate $RC(w)$ and f_w
- $RW(w) = \frac{z - RC(w)}{z}$
- $Reg = f_w \cdot RW(w)$
- $M_w =< Reg, RC(w) >$

kmeans($\mathbf{M_n}$, *2*) //Separate $\mathbf{M_n}$ in 2 clusters (i)

6.5 Encoding temporal information

It is important to know that having finished an activity the participant completed certain stages which are key for the activity's correct execution. This step is accomplished with the use of KAD system, as described earlier. However, the temporal order in which these stages are completed is equally important; imagine the activity *"wash clothes"* executed in the order {*put clothes in washing machine, take clothes out of the washing machine, start washing machine, stop washing machine*}. Although all necessary steps are there, the activity is abnormal. The temporal order between stages (or actions) defines the activitys structure which is modelled with the use of HHMMs.

When modelling an activity with the aid of HHMMs, two problems need to be solved:

1. Given a set of sequences illustrating an activity, the parameters of the HHMM have to be learned from data. The sequences used to learn the parameters of the HHMM are defined as *training* sequences.

2. Given a set of sequences illustrating activities, the HHMM has to be utilised for the purpose of inferring to which activity class each of these sequences belongs.

The first problem is solved using the Expectation-Maximisation (EM) algorithm [Duda et al., 2000]. The solution to this problem is presented in Appendix 1. To solve the second problem (inference) the generalised Viterbi algorithm is employed [Fine et al., 1998]; this method is discussed in Appendix 2.

6.6 DeRFHHMM: Combining KAD with RF+HHMM

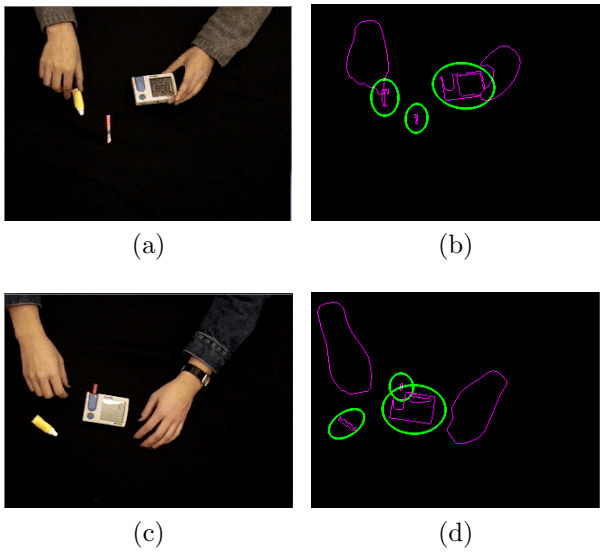
The classification method proposed Section 6.4 can be substituted with the RF+HHMM algorithm proposed in Chapter 5. This is useful when detected Key Actions cannot sufficiently discriminate between similar activities. In this case RF+HHMM algorithm is directly applied to the simplified dataset, $V_{T,C}$ obtained by Eqn. 6.1. The resulting algorithm, combining common actions denoising and RF+HHMM classification is called DeRFHHMM.

The common actions denoising unit can be modified to include variable importance measure obtained by the RF algorithm as explained in Section 5.3.1. Specifically, after obtaining the matrix of importance

measures generated by the RF, its maximum value, VI_{max}, is estimated. To detect unimportant actions the k-means clustering technique on matrix $\mathbf{M_{res}}$ retained from Section 6.2 is applied as follows (in similar fashion to Alg. 6.4.1) : kmeans($(\frac{1-VI_{max}}{VI_{max}} M_{res})^{\circ 2}, 2$), where the symbol $^{\circ}$ is used to denote element-wise matrix operation.

6.7 Experimental results

As stated in Section 4.3, approaches in which objects are first detected and tracked and then their tracks are used to model activities are investigated in this Thesis. In this research area, very few public data sets exist [Zhang et al., 2011]. One of these is found in [Shi et al., 2004a]; it analyses the problem of evaluating correct execution of the task of calibrating a blood glucose monitor, which is a common task for elderly people who develop late stage diabetes. This task is relevant to the work described in this Thesis for the following reasons: (1) the glucose monitor calibration task is a real life, complex task which comprises a large number of individual steps (2) these steps can be carried out in a large number of ways (3) these steps can be typically executed concurrently. For more details regarding the glucose monitor calibration task please see [Shi et al., 2004a]. The video sequences used as dataset for the evaluation of the task are publicly available from the authors's website [Shi et al., 2004b]. The dataset consists of 41 video sequences depicting this activity carried out by 3 participants. In the experiments described below a similar testing methodology with the one proposed in [Shi et al., 2004a] was used: in specific six sequences were used for training of the model and the rest for testing. Sample frames from this dataset are shown in Fig. 6.2a and 6.2c.



(a) (b)

(c) (d)

Figure 6.2: **(a), (b)**: Sample frames extracted from the dataset presented in [Shi et al., 2004a]. **(c), (d)**: Object tracking results obtained by the key object tracking algorithm.

In [Shi et al., 2004a] a framework based on Propagation Networks (P-Nets) was introduced to solve this task. This method has the following conceptual similarities with the method proposed in this Thesis: (1) an activity is, as in the methodology proposed in this Thesis modeled as a sequence of actions (2) the activity is represented as a discrete state model. However, there are several key differences between the two approaches, namely (1) the state model proposed in [Shi et al., 2004a] is manually designed which means that it is specific to the glucose monitor task. For a different task a different state topology has to be defined; in contrast, the state model of the method proposed in this Thesis is learned from training data (2) the transition probabilities between two states of the model are provided by experts in [Shi et al., 2004a], however in the method proposed in this Thesis these are learned from training sequences (3) the method proposed in this Thesis includes a sequence classification step based on ensemble classifier techniques;

for this reason, the model automatically built using the methodology discussed in this Thesis has to be trained with a dataset which includes sequences illustrating both correctly and erroneously executed activities; on the contrary, the topology of the model proposed in [Shi et al., 2004a] is manually defined using only sequences illustrating correctly executed activities. Despite these differences the comparison is informative and useful as the activity detection rate reported in [Shi et al., 2004a] provides a measure for the success rate which is expected for a system designed to analyse the glucose monitor task.

Objects of interest in the scene are the glucometer, the participant's hands, the test strip and the liquid bottle containing the blood sample. Each of these key objects was tracked using methodology described in Section 4.3.1. Results of application of the proposed system's tracking component to the video sequences of this dataset are shown in 6.2b and 6.2d. Using the QSR action detection framework (Section 4.3.2), action primitives which correspond to interactions of key objects in the scene were extracted.

6.7.1 Performance without denoising

First the proposed method RF+HHMM is applied (Chapter 5) without the denoising stage (Chapter 6). The proposed method's performance in the glucose monitor calibration task is shown in Table 6.1 and is compared to the performance of the P-Nets as reported in [Shi et al., 2004a] (Table 6.2). The method proposed in Chapter 5 is able to classify sequences marked as "correct" in the ground truth data with accuracy of 90.5%. However, it is able to detect sequences marked as "missing one" with 100% accuracy and sequences marked as "missing six" with

90% accuracy. P-Nets achieves 100% accuracy at identifying "correct" executions but when it comes at detecting erroneous sequences accuracy at detecting sequences of type "missing one" is 80% and accuracy at detecting sequences of type "missing six" is 50% of these sequences. The accuracy of 100% achieved by P-Nets in identification of "correct" executions is attributed to the fact that this model is only trained with sequences illustrating correctly executed activities, whereas the method proposed in Chapter 5 is trained with sequences illustrating both correctly and erroneously executed activities. More specifically P-Nets use six sequences to build the "correct" model while the proposed method RF+HHMM uses only two sequences. The approach proposed in Chapter 5 is able to correctly classify the sequences of the testing dataset in 93% of cases; performance of P-Nets is 83%. Note that KAD method only works with the denoising step therefore it is not tested at this stage.

6.7.2 Performance with denoising

The proposed method DeRFHHMM is applied which comprises a denoising stage using the common actions framework (Section 6.1) and a classification stage (Chapter 5). Results of the denoising phase are shown in Fig. 6.3. Actions enclosed in cluster 1 are regarded as *common* and are therefore removed from the dataset. These actions are: $\{a, b, e, f\}$ using the codes of Table 4.2. Classification is afterwards performed as in Section 6.7.1 using RF+HHMM algorithm (Chapter 5) The proposed method's performance in the glucose monitor calibration task is shown in Table 6.3. The method proposed in this Thesis is able to classify sequences marked as "correct" in the ground truth data

Table 6.1: Performance of the proposed method RF+HHMM, glucose monitor calibration task.

Sequence Category	Total	Correct	Almost right	Negative
Correct	19/21	90.5%	9.5%	0%
Missing one	10/10	0%	100%	0%
Missing six	9/10	0%	10%	90%

Table 6.2: Propagation Nets [Shi et al., 2004a] performance, glucose monitor calibration task (reported in [Shi et al., 2004a]).

Sequence Category	Total	Correct	Almost right	Negative
Correct	21/21	100%	0%	0%
Missing one	8/10	20%	80%	0%
Missing six	5/10	0%	50%	50%

Table 6.3: Performance of the proposed DeRFHHMM method, glucose monitor calibration task.

Sequence Category	Total	Correct	Almost right	Negative
Correct	20/21	95%	5%	0%
Missing one	10/10	0%	100%	0%
Missing six	10/10	0%	0%	100%

Table 6.4: Performance of the proposed KAD+HHMM method, glucose monitor calibration task.

Sequence Category	Total	Correct	Almost right	Negative
Correct	20/21	95%	5%	0%
Missing one	9/10	0%	90%	10%
Missing six	10/10	0%	0%	100%

with accuracy of 95%. It is also able to detect sequences marked as "missing one" with 100% accuracy and sequences marked as "missing six" with 100% accuracy. Compared to the results without the denoising stage the proposed method illustrates significant improvement in classification accuracy yielding an overall performance of 98% in this dataset.

The proposed method KAD+HHMM is also tested. Key actions detected are $\{c, d, g, h, i, j, k, l\}$. The algorithm achieves overall classification accuracy of 95%. In Table 6.4 its results are presented analytically.

6.7.3 Investigating alternative denoising methods

In this subsection alternative denoising techniques are considered for the glucometer calibration task. These techniques are then compared against the denoising algorithm proposed in this Chapter. The comparison methodology is the following: each denoising technique is applied to four classifiers which are RFs, HHMM and the proposed KAD+HHMM and RF+HHMM. Note that Suffix Trees algorithm was also tested but did not present competitive classification accuracy therefore its results are not shown here. The denoising algorithms which are tested are: RF variable importance [Breiman, 2001], SVM variable importance [Maldonado and Weber, 2009], Brute Force feature selection [Ribeiro and Santos-Victor, 2005] and the Common actions method proposed in this

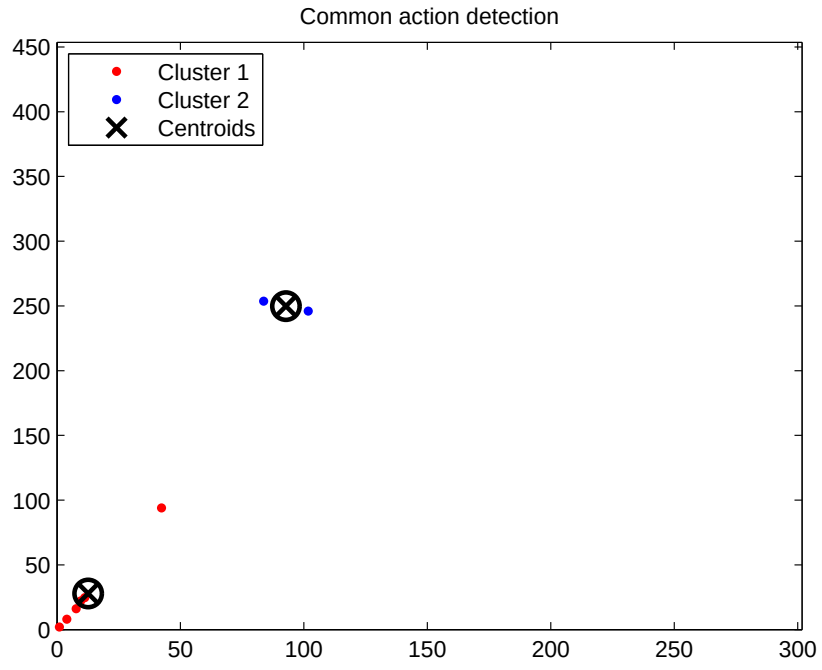


Figure 6.3: Common action primitives detection in the glucometer dataset. Elements of cluster 2 are regarded as *common* and are therefore removed from the dataset.

Chapter. Implementation details for each method are given below.

RF variable importance. The algorithm outputs the mean decrease in Gini index [Gini, 1912] for all variables. Gini index is a standard measure of variable importance [United Nations, 2010, Hillebrand, 2011]. Variable importance is analogous to decrease in Gini index, *i.e.* more important variables display higher values of decrease in Gini index. The test was repeated 10 times and the average decrease in Gini index for all variables was estimated. Results are shown in Fig. 6.4. Actions $\{g, h, m, n\}$ with Gini index $= 0$, are considered redundant and are removed from the dataset.

SVM variable importance. The algorithm randomly partitions the training dataset into two subsets. The first is used to train an SVM classifier. Important features are those which maximise classification accuracy of the trained SVM classifier on the second subset. The training dataset of the glucometer dataset is very small (six samples) and the method does not produce logical results in a single run. Therefore the algorithm is executed multiple times (n) and the number of occasions each action primitive a is selected as important, $n_{a,i}$ divided by n is calculated. Then a manually defined threshold is applied over the resulting $n_{a,i}$ values to select redundant actions. The algorithm is first executed

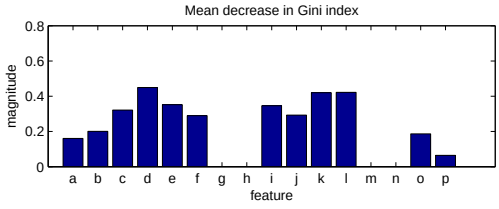
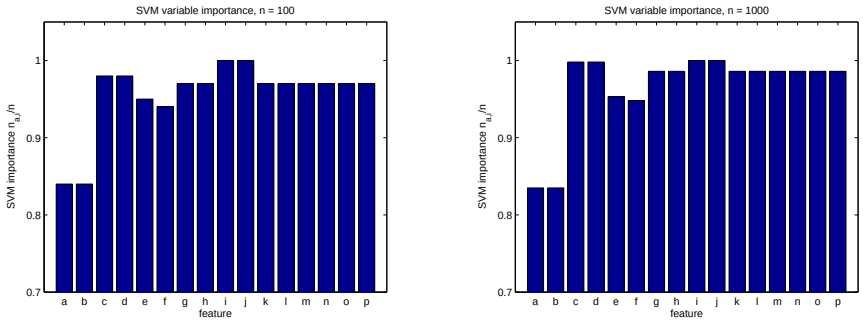


Figure 6.4: Decrease in Gini index as outputted by the RF variable importance algorithm.



(a) (b)

Figure 6.5: Variable importance $n_{a,i}$ as outputted by the SVM variable importance algorithm.

$n = 100$ times and results are shown in Fig. 6.5a. A logical threshold for this experiment is $n_{a,i}/n = 0.90$ and for this value actions $\{a, b\}$ with $n_{a,i}/n = 0.84$ are considered redundant and are removed from the dataset. Then the algorithm is run $n = 1000$ times and results are shown in Fig. 6.5b. A logical threshold for this experiment is again $n_{a,i}/n = 0.90$ and for this value actions $\{a, b\}$ with $n_{a,i}/n = 0.84$ are considered redundant and are removed from the dataset.

Brute Force feature selection. This method was found to give the best results in [Ribeiro and Santos-Victor, 2005]. In [Ribeiro and Santos-Victor, 2005] important feature selection is based on classification results of the original dataset partitioning in training and testing data. However, selecting features and testing this selection on the same data can lead to overfitting. A modified version of this method is therefore applied here. The training dataset is partitioned into two subsets. The first is used to train a set of classifiers. Then the algorithm exhaustively searches amongst all possible feature combinations and selects those maximising a certain metric. This metric is based on classification accuracy of the trained classifiers on the second subset. Important

features are those present in most of the selected combinations and redundant those present in the least of them. Exhaustive search is limited to feature combinations which include at least 8 features. HHMM and RF classifiers are used, the classification accuracy of which is symbolised as acc_{HHMM} and acc_{RF} respectively. The chosen metric m_{brute} is the average performance of the classifiers, $m_{brute} = \frac{acc_{HHMM}+acc_{RF}}{2}$. If k feature combinations yield maximum metric value $m_{brute,max}$ and a feature, a is present in k_a of these combinations, its importance for a specific training dataset partitioning can be measured as $imp_{brute,a} = k_a/k$. The algorithm is run n times, each for a random partitioning of the training dataset and the overall importance for a feature is calculated as $imp_{brute,a}(n) = \frac{1}{n}\sum_{j=1}^{n}\frac{k_{a,j}}{k_j}$. In this work the algorithm is run $n = 10$ times and results are shown in Fig. 6.6. A logical threshold for this experiment is $imp_{brute} = 0.50$ and for this value actions $\{a, b, c, d\}$ are considered redundant and are removed from the dataset. It is observed that action primitive $\{n\}$ is close to the threshold and measurements are repeated with this action primitive added to the redundant action primitives set.

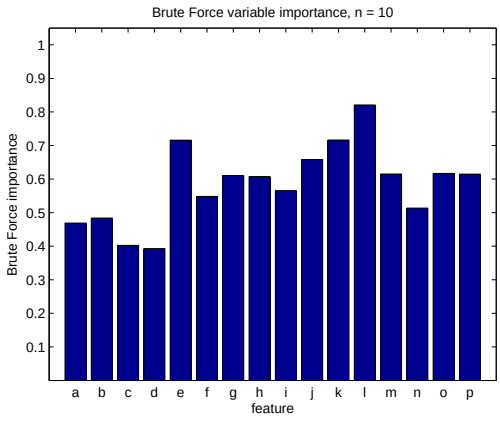


Figure 6.6: Brute Force variable importance.

Performance comparison. Feature selection results for each tested method are applied to the glucometer dataset and classification performance of four classifiers (RFs, HHMM and the proposed KAD+HHMM and RF+HHMM) is measured on the filtered versions of the dataset. Classification results are shown in Fig. 6.7a. The proposed dataset denoising algorithm based on the common actions concept offers higher classification accuracy for all tested classifiers. It is also noteworthy that all tested classifiers clearly benefit from the proposed denoising method as their classification accuracy increases when the common action primitive scheme is applied before classification. Performance evaluation of the tested algorithms in terms of speed is shown in Fig. 6.7b. Matlab implementation of all tested algorithms was used on an Intel Core i7-870 quad core CPU with 8 GB of RAM. The proposed method is the second fastest in this experiment; on the other hand Brute Force algorithm is by far the slowest.

6.7.4 Modifying the glucometer dataset

Results presented in Subsection 6.7.3 show that the proposed denoising method, based on the common actions concept, when combined with several current classifiers results in increase of classification accuracy in the glucometer dataset. In Fig. 6.7a it is also shown that the RF classifier offers the highest classification accuracy. This result is attributed to the fact that discriminative features (*i.e.* the type of actions executed) are more important for classification than temporal dependencies between features (*i.e.* the ordering of actions) in the glucometer dataset. The power of the proposed algorithms RF+HHMM and KAD+HHMM, however becomes apparent in datasets where dis-

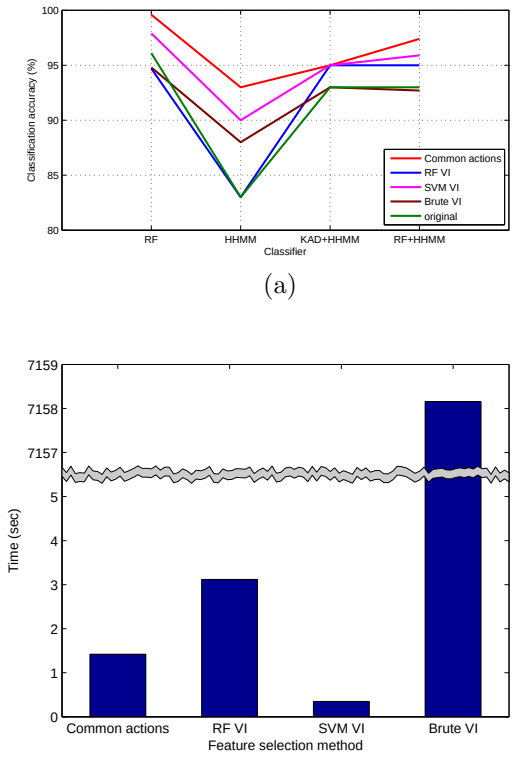


Figure 6.7: Feature selection methods comparison.

criminative features and temporal dependencies are equally important. To prove this point, the glucometer dataset is modified as follows: actions $\{k, l\}$ are attached to the end of six sequences of the class "missing one" and the modified sequences are added to the dataset as a new class. Two of the new sequences are used for training of the new class. The new sequences illustrate erroneous executions since, although they include all necessary steps for the glucometer calibration procedure, ordering of actions is erroneous as $\{k, l\}$ should be carried out before actions $\{i, j, g, h\}$. All classifiers tested in Subsection 6.7.3 are evaluated in the modified dataset. The proposed common action primitive denoising scheme is applied to all algorithms before classification. Results are shown in Fig. 6.8. The proposed methods RF+HHMM and

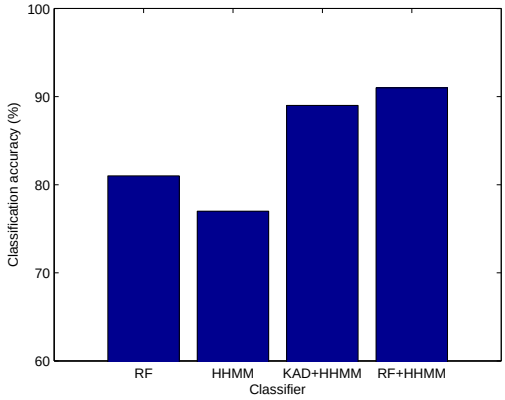
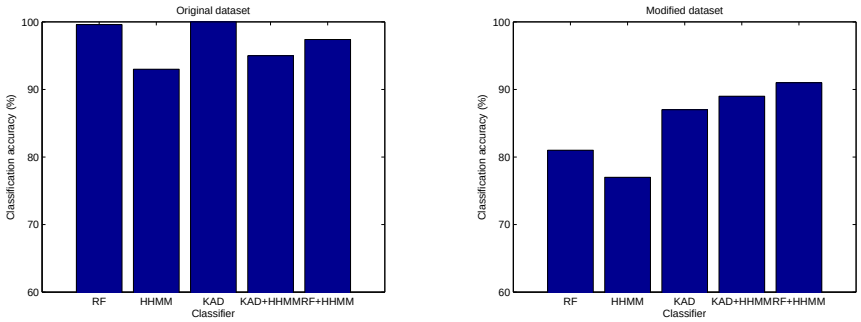


Figure 6.8: Classifier comparison, in the modified glucometer dataset.

KAD+HHMM clearly outperform the rest of the classifiers in classification accuracy with performances of 91% and 89% respectively.

The finding that discriminative features are more important for classification than temporal dependencies between features in the glucometer task motivates the application of KAD method without the temporal analysis step both to the original and the modified dataset. Results are shown in Fig. 6.9 where all algorithms were applied in combination with the common actions denoising step. It is observed that in the original dataset (Fig. 6.9a) KAD method achieves classification accuracy of 100%. On the contrary, in the modified dataset (Fig. 6.9a), when temporal dependencies come into play performance of KAD method drops to 87% as it has no means of encoding the ordering of actions.

Note that activity identification was achieved in the examined dataset with high precision. This proves that the action primitive extraction and representation method proposed in Chapter 4 is suitable for prolonged, composite human activities.



(a) (b)

Figure 6.9: Comparative performance of KAD in the original and the modified dataset.

6.8 Summary

In this chapter the novel KAD method was presented, which identifies *key* actions in datasets illustrating complex human behaviour. The technique simplifies input data and therefore eases further classification process. Furthermore, it was shown how the main concepts of the proposed methodology can be used to form an activity classification scheme. Experiments showed that the proposed algorithm, combined with several classifiers offers improved accuracy in classification. Previously proposed feature selection methods cannot efficiently detect important and redundant actions in action sequences illustrating prolonged, composite human activities.

Chapter 7

ENGINEERING APPLICATION

7.1 Overview

In this chapter a novel application of activity recognition is presented, in which the proposed system is tested in a real-life bridge design scenario, developed with the help of three bridge design experts. The bridge design task was chosen because of its relative complexity and due to the fact that many conventional Decision Support Systems focus on this problem, *e.g.* [Moore, 1991, Choi and Choi, 1993, Philbey et al., 1993, Moore et al., 1997, Hong et al., 2002, Malekly et al., 2010]. In particular the conceptual stage of bridge design is investigated.

The conceptual stage of design engineering is the stage where the basic solution path is laid down [Pahl et al., 2007]. It is traditionally performed on pen and paper. However, in recent years several computer based applications have been proposed for the purpose of supporting this preliminary design stage. In initial computer based approaches (*e.g.* [Moore, 1991]), the decision making process was controlled by an expert system, which asked questions until it had sufficient information and then provided a solution. It is apparent that such an approach limits engineer's creativity. More recent systems attempted to enhance the user's role by allowing him to make decisions regarding the choice of

components and the style of the structure [Moore et al., 1997, Mashood et al., 2007, Nimtawat and Nanakorn, 2009] or even allowing modifications in the final design [Sisk et al., 2003]. However, in these systems the engineer's contribution is still constrained to a supervisory role.

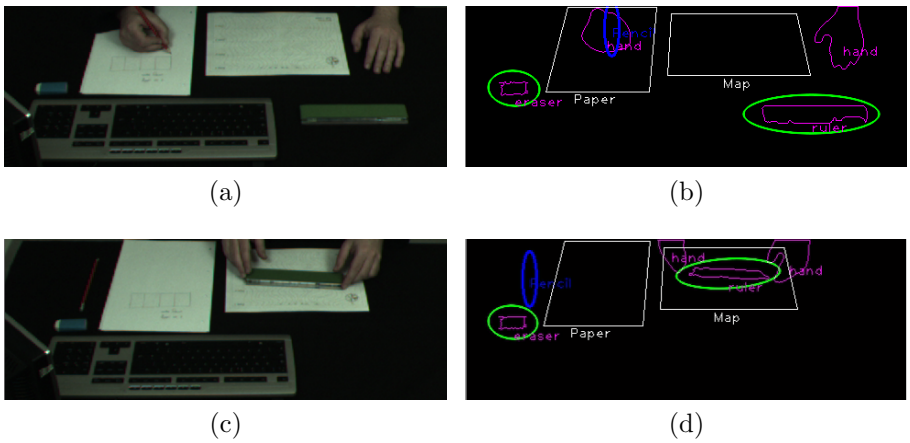
The proposed system monitors an engineer's behaviour using a static video camera as he works on a design task at a table (Fig. 7.1). Technical information is provided to the engineer on demand with the aid of a Knowledge Based System (KBS); their interactions with the KBS are also recorded. The system automatically extracts the actions performed by the engineer from the video footage and their interactions with the KBS during the conceptual design phase in a stream which is called the *task's timeline*. This timeline is then analysed using the framework presented in Chapter 3. After the completion of a design phase, the system either verifies its correct execution or points out potential mistakes. Thus the proposed system, in contrast to previously existing decision supports systems, gives feedback to the engineer without providing ready-made solutions to the task.

7.2 System Development

The procedure followed in order to formulate a framework that would enable the monitoring of the behaviour of engineers when working on a design task is described in this section.

7.2.1 System requirements

One of the most challenging aspects of the work presented in this Thesis is to design a platform which will enable us to study and analyse the behaviour of engineers when working on a design task. Two important



(a) (b)

(c) (d)

Figure 7.1: The experimental environment: the engineer interacts with various objects at a study desk. **(a), (c)**: Sample frames extracted from the bridge task dataset. **(b), (d)**: Object tracking results obtained by the key object tracking algorithm (Section 4.3.1) and application of the extended qualitative spatial representation framework (Section 4.3.2). **(b)**: Action writing, sketching or waiting. Spatial relationship: {(hand 1) Surrounds (pencil) And (pencil) Surrounds (paper)} **(d)**: Action measuring on map. Spatial relationship: {(hand 1) Touches (ruler) And (ruler) Surrounds (map) && (hand 2) Touches (ruler)}.

factors which have to be accommodated are:

1. **Solution variety.** There are multiple correct ways of solving a design task. To be able to draw conclusions regarding the efficiency of a solution, a set of parameters which are essential for its completion need to be identified *a priori*. These parameters will serve as a list of *checkpoints*. A solution will be then evaluated with respect to how many of these *checkpoints* were completed and (in certain occasions) the temporal order of their completion.

2. **Participant's expertise.** Engineer's experience in relevant tasks might hinder the attempt to analyse his behaviour during the design procedure. Given an easy design task, an experienced engineer might be able to provide an immediate solution, without the

need to complete any *checkpoints*.

In the next section, it is described how the case study was developed in order to handle these issues.

7.2.2 Facing the challenges

To satisfy *solution variety* factor, the experiment was developed in close collaboration with bridge design experts. Widely used bridge design handbooks [Troitsky, 2000, Menn, 1990] and regulations [Fahoum, 2010] were taken into consideration. Finally, discovered *checkpoints* were cross-validated with findings from relevant research [Moore, 1991, Moore and Miles, 1991]. To ensure that the task is not trivial and therefore the participant will eventually be obliged to complete the *checkpoints* in order to provide a scientifically sound solution (*participant's expertise* factor), the experiment was designed so that:

1. The design scenario given to the participant contains only *abstract* information concerning the task. Thus, the engineer has to ask for further information to solve the problem. This information is carefully correlated to the checkpoints so that it can be deduced in which part the engineer is working by his queries. More details about this method are given in Section 4.4.

2. There are multiple correct solutions to the problem; by careful setting of the task's parameters, it is ensured that none of these is obviously "optimum". Several alternatives have to be thoroughly considered before the final decision is made. A circumstantial decision is only possible after consideration of all problem parameters; this necessitates the completion of designated *checkpoints*.

7.3 The bridge design task

This section presents an extensive evaluation of the proposed system. Here an overview of the experiments is given: after presenting the dataset used in the experiments (Section 7.3), testing methodology is described (Section 7.3.3) and an overview of the evaluation methodology is given (Section 7.3.4). Then the experimental results are presented where it is shown that the proposed sequence analysis algorithm performs better than other existing approaches on the bridge task dataset (Section 7.3.5). Section 7.3.6 investigates the cases where the proposed algorithm failed to classify correctly. It is shown that the proposed methodology offers high classification accuracy on the studied dataset and it is therefore suitable for detecting mistakes in tasks illustrating complex human behaviour.

7.3.1 Technical specifications

In this section technical details about the bridge design task are given. The scenario presented to each participant is discussed in Section 7.3.1 and the parameters of the task are analysed in Section 7.3.1.

Bridge Design Scenario

The topographical map of Fig. 7.2 is presented to the engineer. The map is divided into four zones, each with different characteristics (*e.g.* soil condition, wind load *etc.*). According to the scenario, a river flows somewhere in the map. The exact location of the river and its characteristics (*e.g.* depth, width) are not given but can be estimated by querying the KBS and performing calculations using the information given by the software. Two cities are located in the map on opposite

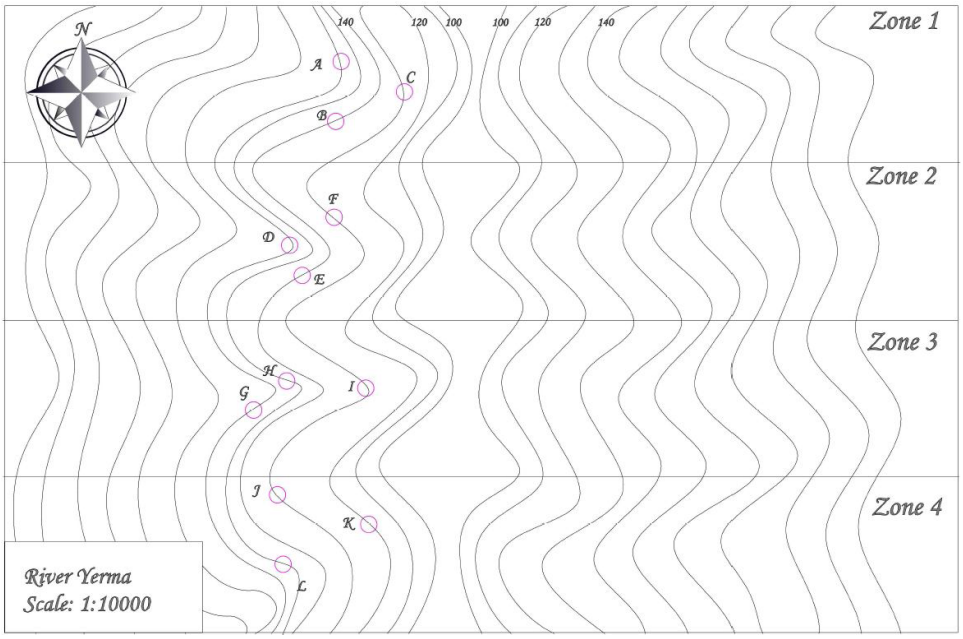


Figure 7.2: The topographical map accompanying the Bridge Design Task.

sides of the river. Their locations are not given. The cities are currently connected by ferries and it takes about 80 minutes on average to travel from one city to the other. The engineer is asked to design a bridge which connects the two river banks and cost it. All technical aspects of the bridge have to be taken into consideration. The bridge should cater all forms of automotive traffic in two directions and also pedestrians and cyclists. The final design should comprise the following information: location of the bridge (preferred zone and coordinates), bridge type (*e.g.* suspension, arch, cable *etc.*), total cost, total travel time between the cities after the construction and a sketch of the bridge. The goal is to design the best possible solution with respect to cost and travel time. Justification for individual choices should also be given.

Bridge Design task parameters

The basic scenario does not offer sufficient data for the solution of the task. All additional necessary information is provided by the KBS in the form of answers to the engineer's queries. Appropriately placed queries reveal information about the parameters of the Bridge Design problem which were defined after thorough research in the bridge design literature, with the main sources being [Troitsky, 2000, Menn, 1990, Fahoum, 2010, Moore, 1991, Moore and Miles, 1991]. Three bridge design experts verified the parameters of the task and the associated values. It is not claimed that the experiment presented here captures the full complexity of the bridge design task however it features its main design principles and is suitable for educational purposes.

The parameters taken into consideration in the Bridge Design Task are the following:

Bridge width, which depends on the traffic lanes carried.

Bridge length, which depends on the river's width.

Traffic requirements, with the minimum being a dual carriageway with two lanes in each direction. Concerning the traffic lane width, width of motor vehicle lane should be between 3.0-4.0m, cycle lane if incorporated should be between 1.5-2.0m, pedestrian pavement if incorporated should be at least 1.5m and pavements with bike lanes have a minimum width 2.5m.

Travel time between cities, after the bridge is constructed which will vary depending on where in the zones the bridge is located. This is because of the available current road infrastructure. For Zone 1 it is 50 minutes, 30 for Zone 2, 20 for Zone 3 and 60 for Zone 4.

Bridge types with the following being available for the experiment:

concrete beam and slab, steel beam and concrete slab, steel truss, cable stayed, arch, suspension, movable span.

Bridge spans, which depend on the chosen bridge type. Minimum and maximum spans for the bridge types of the experiment are shown in Table 7.1.

Bridge base cost, which depends on the chosen bridge type and the size of the bridge. Costs for the bridge types of the experiment are given in Table 7.1. All calculations regarding additional costs should use this value as basis.

Intermediate piers. If maximum span is exceeded, intermediate piers will be required to ensure the bridge is safe. It is assumed that an intermediate pier will increase the overall cost of the bridge by 5%. This increase does not include the additional cost of the foundation that may be required.

Foundations. If the ground/terrain at the location of bridge construction requires deep foundations or piling, costs can be expected to increase by 10% of the total cost of the bridge for each 10m of depth for each foundation. An additional 7% of the total cost of the bridge is required if the structure is founded on land and 15% if founded in the river.

Excavations. If the ground/terrain at the location of bridge construction requires bulk excavation, costs can be expected to increase by

	Bridge Specifications	
Bridge Type	Cost ($£/m^2$)	Min-max span (m)
Concrete beam and slab	2000	20-40
Steel beam, concrete slab	2200	30-90
Steel Truss	2000	90-360
Cable Stayed	3700	180-360
Arch	3000	10-120
Suspension	7000	180-1500
Movable Span	12000	60-120

Table 7.1: Specifications for bridge types available in the experiment.

5% for each 10m of depth for earth and 10% for rock.

Wind load. Wind speed can reach up to 160km/hr in Zone 4. Max wind speed in Zones 1 -3 lower, max 120km per hr. If the bridge is built in Zone 4, costs can be expected to increase by 5%.

Seismic activity. Previous seismic activity has been greater in Zones 3 and 4. If the bridge is built in these zones, costs can be expected to increase by 5%.

Safety for river traffic. Where possible, river traffic should be unaffected by the presence of the bridge.

Water height above reference datum. Minimum +118, maximum +120.

Ground type which depends on zone. It can be evaluated using given boreholes to solve a three-point geology problem. Boreholes are given for only one side of the river for each zone but it can be assumed that the soil structure is the same on both sides of the river for the same zone. Borehole specifications are as follows (note that the letter preceding the borehole measurements refers to its corresponding point in map of Fig. 7.2):

Zone 1: A: Surface - 50m: Sand, 50 - 80: Clay, 80 - : Rock. B: Surface - 30m: Sand, 30 - 60: Clay, 60 - : Rock. C: Surface - 20m: Sand, 20 - 50:Clay, 50 - :Rock.

Zone 2: D: Surface - 60m: Clay, 60 - : Rock. E: Surface - 40m: Clay, 40 - :Rock. F: Surface - 30m: Clay, 30 - : Rock.

Zone 3: G: Surface - 70m: Clay, 70 - : Rock. H: Surface - 60m: Clay, 60 - :Rock. I: Surface - 40m: Clay, 40 - : Rock.

Zone 4: J: Rock. K: Rock. L: Rock.

Aesthetics, which depends on the chosen bridge type and the cho-

sen location. It is assumed that in Zones 2 and 3 the bridge should be aesthetically pleasing.

7.3.2 Results

Six civil engineering professionals and 14 civil engineering students participated in the study, resulting in a total of 30 hours of video footage. From this video footage 54 sequences were extracted (each of length 5-15 minutes) to serve as the training set. In each sequence, participants execute one of three complex tasks: *evaluate soil condition*, *estimate transient loads* and *evaluate bridge cost*. The test data is a different set of 72 sequences obtained in a similar way. Table 7.5 shows the distribution of the sequences of the dataset in activity classes as provided by expert's labelling.

The data acquisition and feature extraction component extracted sequences of actions from the video stream and the user's interactions with the KBS as explained in Section 4.3. The task vocabulary, V_T, which consists of 40 action primitives is shown in Table 7.3. In this Table it is also specified whether each action primitive is detected by the video analysis unit or the KBS. For action primitives related to soil evaluation suffixes indicating specific zones are dropped, thus action primitives 23-30 in Table 7.3 are merged into two: *soil evaluation start* and *soil evaluation end*. A subset of the sequences used for testing is given in Table 7.4. All sequences used in the experiments (both for training and testing) are given in Appendix 3.

Ground truth (*i.e.* identification of activities performed in sequences and evaluation if these activities were executed in a correct or erroneous manner) was provided by bridge design experts by manual labelling of

Table 7.2: Bridge Design Task dataset, labelling details

Class No.	Code	Description
Class 1	A1.1	Soil Condition examination executed correctly
Class 2	A1.2	Soil Condition examination executed erroneously
Class 3	A2.1	Transient Loads evaluation executed correctly
Class 4	A2.2	Transient Loads evaluation executed erroneously
Class 5	A3.1	Base Cost estimation executed correctly
Class 6	A3.2	Base Cost estimation executed erroneously

the extracted video sequences and examination of each participant's study output. The labelling process took place as follows: (1) an expert examined an input sequence and evaluated the participant's study output corresponding to this sequence, (2) based on this information, they assigned the input sequene a label which indicated (a) type of the activity observed in the sequence (b) whether the activity was executed in a correct or erroneous manner. Activity classes for the bridge design task are shown in Table 7.2.

7.3.3 Model specifications

The proposed sequence classification algorithm was applied to the extracted sequences, as specified in Section 5. Denoising was first applied (Section 6.2) which characterised four of the actions in the dataset vocabulary, specifically *writing start, writing end, erasing start, erasing end* as common, forming the set V_{com}. These are excluded from the set of key actions, $V_{T,C}$.

The combined RF+HHMM was then built as described in Section 5.3.1 using the 54 training sequences. For the RF, 100 trees were used and the number of split variables, m was set to $m = 6$. Data was classified into six classes (two for each activity: one including correct executions of the activity and one erroneous executions). Concerning the temporal analysis stage, a three-level HMMM was automatically

No.	Action primitive code	Description	Source stream
1	p	measuring start	video
2	v	measuring end	video
3	b	sketching start	video
4	u	sketching end	video
5	x	writing start	video
6	w	writing end	video
7	y	erasing start	video
8	z	erasing end	video
9	s	transient loads start	KBS
10	t	transient loads end	KBS
11	d	river traffic start	KBS
12	e	river traffic end	KBS
13	f	wind load start	KBS
14	g	wind load end	KBS
15	h	seismic load start	KBS
16	i	seismic load end	KBS
17	j	base cost start	KBS
18	k	base cost end	KBS
19	l	bridge length start	KBS
20	m	bridge length end	KBS
21	n	traffic requirements start	KBS
22	o	traffic requirements end	KBS
23	a (1)	soil evaluation in zone 1 start	KBS
24	c (1)	soil evaluation in zone 1 end	KBS
25	a (2)	soil evaluation in zone 2 start	KBS
26	c (2)	soil evaluation in zone 2 end	KBS
27	a (3)	soil evaluation in zone 3 start	KBS
28	c (3)	soil evaluation in zone 3 end	KBS
29	a (4)	soil evaluation in zone 4 start	KBS
30	c (4)	soil evaluation in zone 4 end	KBS
31	q	reference datum start	KBS
32	r	reference datum end	KBS
33	E	excavations start	KBS
34	K	excavations end	KBS
35	P	intermediate piers start	KBS
36	R	intermediate piers end	KBS
37	F	foundations start	KBS
38	G	foundations end	KBS
39	A	aesthetics start	KBS
40	Z	aesthetics end	KBS

Table 7.3: Vocabulary of observed actions in the bridge design task with their corresponding codes.

No.	Class	Sequence
1	1	ahixwqrxwdebulxwFxwpvGdmyzxwc
2	3	sxwyzxwFhxwixwyzaxwGcxwdxwet
3	5	jdxwyzepvxwbuxwqrExwKxwpvAxwZlmyzxwk
4	2	axwbuxwyzjxwnoyzxwc
5	4	sfxwgxwdhiexwt
6	6	jdxwexwyzqxwFGryzlxwpvxwmk
7	1	axwbudexwbujyzxwc
8	1	abuxwqxwburyzxwc
9	2	axwhxwyzxwixwyzxjwyzxwc
10	2	axwdxwnxwyzpvqxwc
11	3	sxwlpvmxwbunjpvoxwdpvxkwefxwghxwit
12	4	sdbupvxwnoxwlmyzxwqrnxwexwt
13	3	sxwhxwixwdehxwiyzxwfgxwyzxwt
14	4	sxwyzxwdbuxwpvxwlmyznoyzxwqrnxwqret
15	5	jxwqrxwnxwPxwRxwoxwAxwZlxwpvmxwdexwbuxwk
16	5	jqrxwpvlmAxwZyzhxwFxwGxwnxwok
17	6	jxwdxweyzxwqxwryzlExwyzKxwpvxwmxwk
18	6	jyzxwnoxwlxwmyzqxwPxwRxwyzrxwpvk

Table 7.4: Example sequences used for testing.

created using algorithm 5.2.1. The learned model is shown in Fig. 7.3 where the actions are represented using the code system of Table 7.3. Parameters of the learned HHMM are given in Appendix 4.

7.3.4 Performance evaluation

Using the learned model of Section 7.3.3 the 72 test sequences are classified following the proposed algorithm as described in Section 5.3.2.

Evaluation of the framework's efficiency in behaviour recognition is based on system's ability to identify correctly performed activities and detect mistakes in the test sequences.

The performance standards on which the proposed method is evaluated are the following:

1. Comparison of the proposed method's (DeRFHHMM) classification accuracy against the classification accuracy of its individual components (RF and HHMM). The purpose of this test is to show that the proposed method outperforms its individual components in classification accuracy hence combining these two algorithms is meaningful. The comparison is achieved using Receiver Operator Charac-

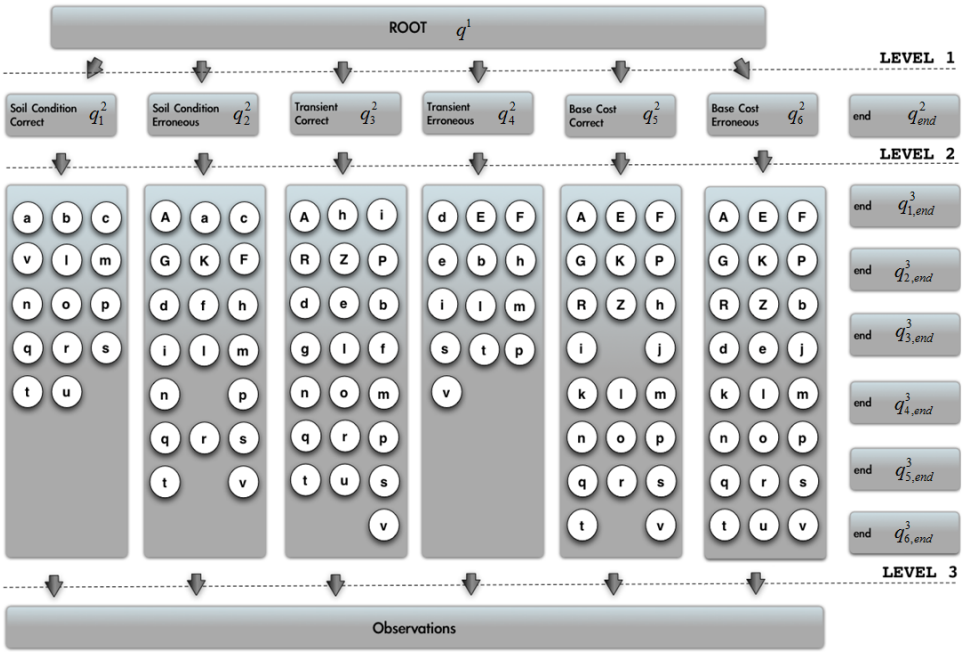


Figure 7.3: The learned HHMM representing activities performed in the bridge design task.

teristic curves [Provost and Fawcett, 2001]. This method is standard for evaluating the robustness of classification performance. This test is repeated with added noise to the original dataset to illustrate the proposed method's resilience to noise.

2. Comparison of the proposed method's (DeRFHHMM) classification accuracy against the classification accuracy of several existing methods. The purpose of this test is to show that the proposed method outperforms current state-of-the-art in classification accuracy in the experiment described in Section 7.3.1. The comparison is achieved using standard classification accuracy estimation for multi-class problems. Therefore classification accuracy, $accuracy(k)$, for a class k from Table 7.2 is defined as:

$$accuracy(k) = \frac{\text{number of correctly identified samples of } k}{\text{total number of samples of } k} \qquad (7.1)$$

Accuracy is estimated for all tested methods, for all activities of Table 7.2.

3. Discriminatory power of the proposed method. The purpose of this test is show that the proposed method mislabels similar classes less frequently than other tested methods. This is achieved using confusion matrices [Kohavi and Provost, 1998].

4. Classification accuracy in relation to activity complexity. The purpose of this test is to illustrate how classification accuracy of the proposed method is affected by increase of activity complexity. If for an activity there are N correct and M erroneous executions present in the test dataset, a sequence illustrating correct execution of the activity

is denoted with Q_i and erroneous with Q_j and activity's complexity is defined as:

$$complexity = \frac{\sum_{i=1}^{N} |Q_i| + \sum_{j=1}^{M} |Q_j|}{N + M} \tag{7.2}$$

where $|Q|$ is the cardinality of an action sequence, Q. In this case, the total classification accuracy for each tested method is considered, defined as:

$$total\ accuracy = \frac{\text{total number of correctly identified samples}}{\text{total number of samples}} \tag{7.3}$$

7.3.5 Comparative performance

For the behaviour analysis component of the proposed system a combination of a RF, a hierarchical graphical model and a denoising unit is proposed. The superiority of this ensemble model is now demonstrated over its individual components on the bridge task dataset (RF classifier and HHMM). An illustrative way to achieve this is to employ ROC curves to measure the efficiency of each algorithm in distinguishing "correct" from "erroneous" behaviours. The ROC curves are obtained as follows: each tested method outputs, for each sample of the test

Table 7.5: Bridge Design Task dataset, distribution of dataset sequences to activities. "ID": correctly performed activities. "ERR": erroneously performed activities.

Activities	ID	ERR	Total
Soil Condition	20	16	36
Transient Loads	25	21	46
Base Cost	23	21	44
Overall	68	58	126

dataset, a tuple $< I, p >$ where I is the classification result which corresponds to an activity class of Table 7.2 and p the classifier's score in the interval $[0, 1]$ for the result which is a measure of the classifier's confidence for the result. The list of tuples for each method are then used to form its ROC curve using the algorithm from [Provost and Fawcett, 2001]. An interpretation of the ROC curve visualisation is given in Fig. 7.4c. Curves corresponding to higher classification accuracies bulge further outward to the upper-left, nearing the point of perfection at (0,1). Note that in Fig. 7.4c curves corresponding to low and higher accuracy are indicative. ROC curves visualising classification accuracy for the proposed method (DeRFHHMM), RF and HHMM are presented in Fig. 7.4a, where it is shown that the proposed method achieves higher classification accuracy.

The response of the three tested algorithms to the addition of independent noise to the original dataset is also evaluated. For the reasons explained in Section 6.1 it is expected that addition of independent noise will increase the difficulty of the classification task. Independent noise is added by inputting additional "common" actions in the testing dataset. The proposed algorithm has already detected four of those, forming the set V_{com} as described in Section 7.3.3. This set is expanded as follows. Following the recommendation of [Kaloskampis et al., 2011b] a set comprising actions of $V_{T,C}$ which are present in all correct executions of at least one activity in the training dataset is formed. By subtracting these actions from $V_{T,C}$ set $V_{T,R}$ is obtained. The set of actions V_{noise} which will be used as noise in the experiment is $V_{noise} = V_{T,R} \cup V_{com}$. It is found that $V_{noise} = \{x, w, y, z, E, K, P, R, F, G, A, Z\}$ where the codes of Table 7.3 are used to represent the actions. Input

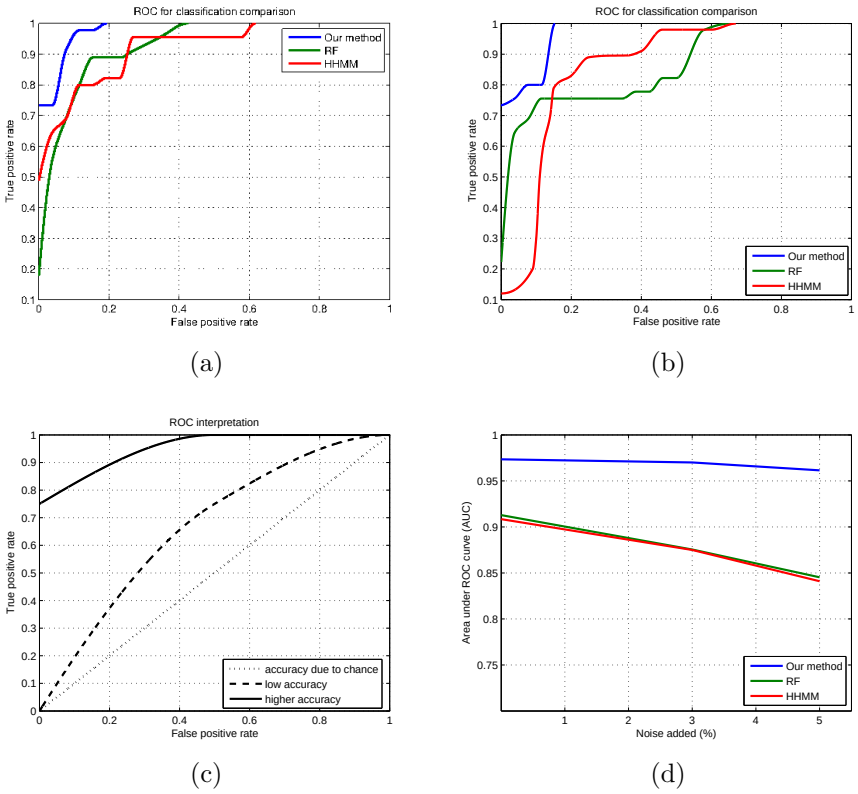


Figure 7.4: ROC curves for different approaches in classification of performed activities in *correct* or *erroneous* of the experiments. **(a):** Original dataset, **(b):** Added noise, **(c):** Performance reference, **(d):** Area under curve (AUC).

points for the actions within the dataset are determined using the van der Corput low discrepancy sequence [van der Corput, 1935a, van der Corput, 1935b]. A randomly selected action primitive from set V_{noise} is inputted at each input point. ROC curves for the added noise experiment are shown in Fig. 7.4b, where it is clear that the proposed method is able to maintain its edge compared to RF and HHMM algorithms under the effect of independent noise.

In a ROC diagram, a measure of classification accuracy is the area under a curve (AUC). An AUC of 0.5 reflects random forecasts and $AUC = 1$ implies perfect forecasts. An AUC for a ROC curve $y(x)$ can be directly computed with the formula $AUC = \int_0^1 y(x)\,\mathrm{d}x$. In Fig. 7.4d

the change of AUC size under the effect of independent noise is shown for the three tested algorithms. It is observed that classification accuracy for both RF and HHMM algorithms significantly deteriorates due to added independent noise; on the other hand, the proposed method's performance exhibits a very mild decrease rate which keeps accuracy at an almost steady level.

The experiments therefore show that the proposed method outperforms its constituent components in classification accuracy in the bridge task dataset.

To further demonstrate the efficiency of the proposed DeRFHHMM algorithm in sequence analysis, the system's performance is compared against several alternative methods. The examined bridge task involves complex temporal relations, therefore approaches in which objects are first detected and tracked and then their tracks are used to model activities are investigated. Input sequences are analysed with flat HMM (a popular choice in activity analysis, *e.g.* [Cielniak et al., 2003]), simple RF [Leistner et al., 2009] and HHMM (used for activity identification in [Nguyen et al., 2005]) algorithms. A comparison against an unsupervised method involving Suffix Trees which appears in Hamid *et al.* [Hamid et al., 2009] is also carried out. Furthermore the proposed method is compared against two algorithm combinations: sequence analysis is performed by combining RF with the flat HMM and finally combine HHMM with Support Vector Machines (SVMs) [Cortes and Vapnik, 1995]. SVMs were chosen as an alternative non-parametric classification algorithm due to its popularity in current activity recognition frameworks, *e.g.* [Niebles et al., 2010, Duchenne et al., 2009]. Finally, comparisons against algorithms RF+HHMM [Kaloskampis et al.,

2011c] and KAD+HHMM [Kaloskampis et al., 2011b] are carried out which were discussed earlier in this Thesis. The results are presented in Fig. 7.5, where it is shown that the proposed approach exhibits higher or equal accuracy in activity identification than the other tested methods. Performance analysis is based on each algorithm's ability to identify correctly performed activities (column "ID") and detect mistakes in the test sequences (column "ERR").

An important finding of the experiments is that methods combining a discriminative feature classifier and a Markov model (*i.e.* DeRFHHMM, SVM+HHMM, RF+HMM) perform better than single step classification methods (*i.e.* RF, HMM, HHMM and Suffix Trees).

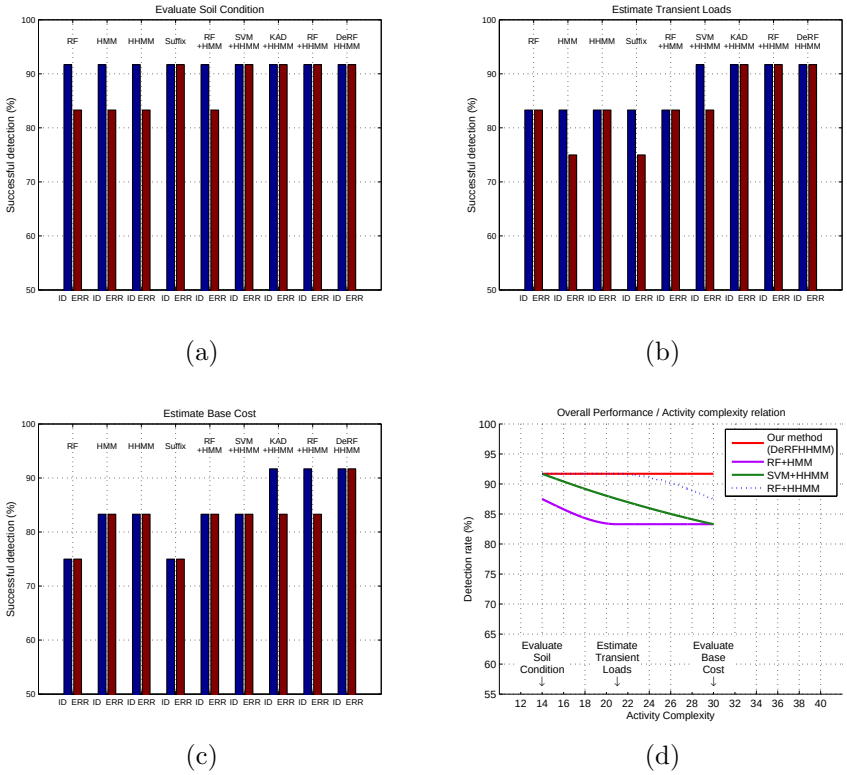


Figure 7.5: **(a), (b, (c):** Comparative performance of the proposed system for three activities. **(d):** System performance in analysis of complex behaviours.

To further investigate this finding, confusion matrices of RF, HHMM, Suffix trees and the proposed method (DeRFHHMM) are plotted (Fig. 7.6). Denoted as A1.1, A2.1, A3.1 in Fig. 7.6 are the activities *estimate soil condition, evaluate transient loads* and *estimate base cost* respectively executed correctly. Denoted as A1.2, A2.2, A3.2 are their corresponding erroneously executed counterparts. It is observed that it is easy for single step approaches to confuse correct with erroneous activity executions. This is attributed to the fact that non-parametric approaches (*e.g.* RF) are not able to handle temporal relations; on the other hand, models following Markovian properties (*e.g.* HHMM) are not able to represent temporal relations accurately in noisy environments. Suffix Trees encode *neighbouring* temporal relations between an activity's constituent actions; in the examined problem space, these neighbouring relations become less characteristic of the performed activity since in many cases "important" actions are preceded and/or succeeded by random actions, which can be thought of as *noise*.

Furthermore, the relationship of system's performance with the complexity of each performed activity is investigated as defined in Section 7.3.4. In Fig. 7.5d overall performance (*i.e.* combined performance in identification and error detection) of composite algorithms RF+HMM, SVM+HHMM, RF+HHMM and the proposed DeRFHHMM is plotted with respect to activity complexity. It is observed that algorithms RF+HMM, SVM+HHMM, RF + HHMM perform equally well in the task *evaluate soil condition*, which has an average sequence length of 14 actions. For more complex tasks, performance slightly deteriorates for these models. On the contrary, it is clear that the proposed DeRFHHMM algorithm maintains its accuracy level as complexity in-

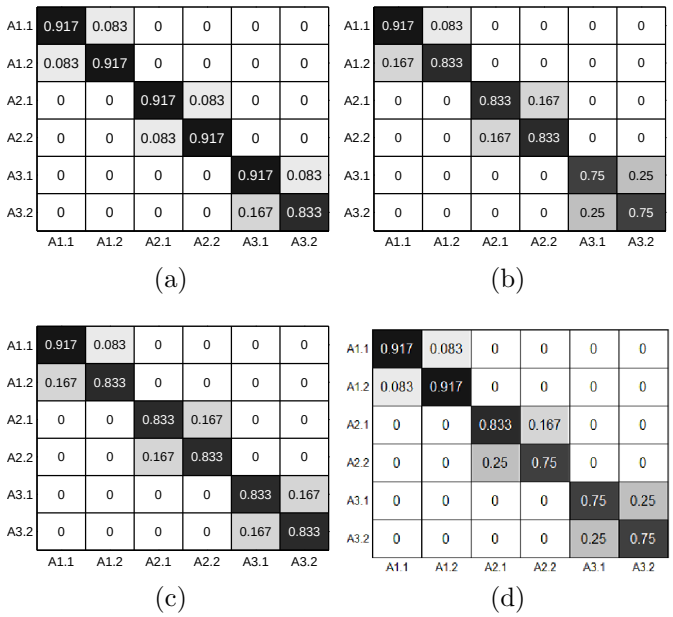


Figure 7.6: Confusion matrices for different approaches in identification of the six activity classes of the experiments. **(a):** The proposed method, **(b):** RF, **(c):** HHMM, **(d):** Suffix Trees.

creases.

7.3.6 Interpretation of misclassifications

Out of the total 72 test sequences, six were misclassified by the proposed system. Out of these, three were false positives (FPs) and three false negatives (FNs). All FPs occurred due to calculation errors, *i.e.* although participants followed all necessary *checkpoints* in the correct order, failure of combining input data lead to erroneous results. Regarding the occurrence of FNs, there were two causes:

1. In one of the FNs, occurrence of *non critical* (see Section4.5) actions in an input sequence was observed, *i.e.* the participants requested information not related to the performed activity during a design procedure.

2. In the two remaining FNs, participants did not request *critical*

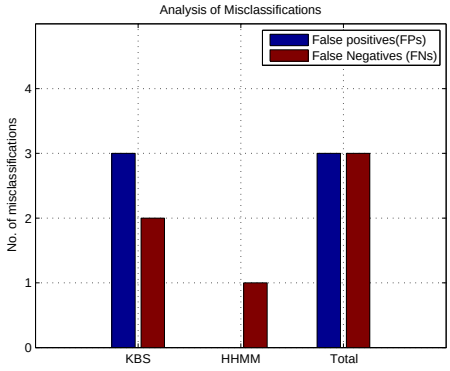


Figure 7.7: Analysis of system's misclassifications.

details concerning a performed activity but reached acceptable results by assuming reasonable values for the missing data. This is common practice for the more experienced engineers in the conceptual stage of the design process.

In Fig. 7.7, misclassifications are attributed to individual system components.

7.3.7 Importance of actions extracted from video

This section reviews the importance of actions recorded in video in the identification of activities in the bridge design task. As already mentioned in Section 7.3.3, two of those, specifically *writing start, writing end, erasing start, erasing end* were characterised by the proposed system as *common* (unimportant) and were removed from the dataset. However this was not the case with actions relevant to *sketching* (*sketching start*, *sketching stop*) and *measuring* (*measuring start*, *measuring stop*).

For the activity *soil condition* it was discovered that all correct executions included action *sketching*. This is logical from a technical point

of view as during the execution of this activity the engineer has to solve a geological 3-point problem [pit, 1985]. Since this problem is solved by determining the positions of soil layers in the three dimensional space, sketching is an essential part of the solution. It has to be noted that sketching is not an indication that the engineer attempted the task; this indication is the primitive action *soil evaluation in zone x start*. Sketching is an optional primitive action which, in the context of the soil evaluation process, was carried out in most cases only by participants who understood that a 3-point problem had to be solved and knew the procedure. Therefore detection of action *sketching* is important for discriminating between correct and erroneous executions of this activity. Of course, even if an engineer sketches they might still produce an incorrect result possibly by errors in their calculations. In the experiments however no such incidents were encountered. In conclusion, omission of sketching is a good reason for the proposed system to notify the user that they have possibly omitted a step during the execution of the task and therefore detection of this action by the system is important. Detecting errors in calculations is an interesting problem and will be analysed in future work.

It was also discovered that for the activity *base cost*, engineers who did not perform measuring after querying *reference datum* executed this activity erroneously. This is logical since, although ground morphology as given in the geological map (Fig. 7.2) implies that the river is located somewhere in the middle of the map (at the lower point, which is between the lowest contours of 100m), its width is not known. Only after the engineer queries *reference datum* it is discovered that water reaches the altitude of 118m. This means that the river flows between

the two contours representing the altitude of 120m hence the width of the river becomes known for every map location. Therefore, omission of *measuring* sometime after the *reference datum* query is a good reason for the system to notify the user about the potential omission of the step. Thus detection of this action by the system is important.

7.4 Action sequence extraction assessment

Activity identification is achieved in the bridge design dataset with high precision. This proves that the action extraction and representation method proposed in Chapter 4 is suitable for prolonged, composite human activities. Furthermore, cognitive tasks can be successfully extracted using a KBS as proposed in Section 4.4.

7.5 Suitability of the bridge design task to test the performance of the proposed method

The engineering design task was found suitable to test the performance of the proposed method for the following reasons. First of all, it involves cognitive actions. Second, it comprises two parallel data streams. Certainly, most of the information is included in the KBS stream and the video plays a secondary role. Yet, without the information obtained from video it is not possible to discriminate between correct and erroneous executions of some performed activities as the action primitives found in the video have discriminative characteristics. Therefore, the two streams complement each other. Third, the resulting classification task is difficult; in fact, as experimental results show, many state-of-the-art algorithms perform poorly - especially after the addition of in-

dependent noise to the dataset. The main difficulty of the classification task is that it requires a method which has both discriminative feature and temporal analysis properties. One of the main contributions of this work is the proposed classification algorithm; this problem shows that this algorithm compares favourably to state-of-the-art methods in a challenging task.

However, acknowledging the fact that a more difficult task is required to test the efficiency of the video analysis component of the proposed system, it was tested additionally in a publicly available dataset where video is the prominent stream, as described in chapter 6.

Finally, there is a logical concern regarding whether certain action primitives, such as erasing, should be included in the task analysis since they seem to be irrelevant to the objectives of the task. Initially several researchers working on the development of the task had the idea that erasing could be used as a discriminative feature *i.e.* there was a possibility that erroneous executions could include erasing in higher frequencies compared to correct executions. This hypothesis was not proven correct, as results showed, for the activities analysed in the studied experiment. However, the results might be different for other types of engineering activities. Therefore, that erasing is a *common* action was a hypothesis that had to be proven experimentally. Since it was validated, the action primitive *erasing* can be ignored when the proposed system is applied in practice.

7.6 Summary

In this chapter a novel application of activity recognition was presented, in which the proposed system was tested in a real-life bridge design

scenario, developed with the help of three bridge design experts. Engineer's behaviour was monitored using a static video camera as they worked on a design task at a table. The system automatically extracted the actions performed by the engineer following the methodology presented in Chapter 4 and analysed them using the techniques described in Chapters 5 and 6. The results showed that the proposed method offers high accuracy in recognising activities and detecting mistakes in the Bridge Design Task. Furthermore it was shown that the proposed algorithm outperforms several current activity recognition algorithms in classification accuracy in the application examined in this Chapter. In this application the proposed framework can be viewed as a decision support system as it gives feedback to the engineer after the completion of each design stage. In contrast to previously existing decision supports systems, the proposed method aids the engineer in the design process without providing ready-made solutions.

Chapter 8

CONCLUSIONS AND FUTURE WORK

The work presented in this Thesis investigates analysis of activities occurring in concurrent multimedia streams which result from video and computer-based human generated content. The general problem of event analysis in multimedia streams is investigated with the main focus lying on events which comprise a large number of steps and these steps can be executed in a plethora of ways. The studied events may include cognitive tasks. Human activities arising in such events are referred to as prolonged, composite activities.

Currently there is no model to represent prolonged activities of high complexity like the ones considered in this Thesis. Additionally, in such prolonged, composite activities not all actions are important for correct execution of an activity. A method is needed to identify such actions automatically and to avoid including them in the models of activities of interest. What's more, there exists no method to unobtrusively extract cognitive activities.

Therefore in this Thesis a framework for analysing prolonged, composite human activities is developed, capable of overcoming the deficiencies of existing methods. Activities are represented using a model

whose topology and parameters can be learned from data; it is capable of efficiently representing temporal relations between an activity's constituent actions and can handle noisy datasets. Furthermore, the method proposed in this Thesis is capable of capturing hierarchy of complex activities and is designed to work with actions that take place in parallel at the same time.

The novelties presented in this Thesis are the following:

- A new feature extraction methodology which enables automatic construction of action sequences from data arising from multiple streams representing complex human activities is proposed. Contrary to existing methods in the area of complex activity analysis, this representation can model activities whose exact structure is not known *a priori* and can handle concurrent activities.

- A new methodology for recording cognitive activities, *i.e.* activities which aid in understanding cognitive thought process. Central part of the proposed method is a KBS.

- A new classification algorithm, suitable for analysing prolonged, composite human activities, an area where currently existing methods prove inadequate, is proposed. It is based on the combination of RFs and HHMMs; combining these methods in the manner proposed in this Thesis allows the proposed algorithm to benefit from their strengths whilst avoiding their weaknesses.

- A method for identifying unimportant and important actions in action sequences arising from the execution of prolonged, composite human activities with the goal of improving classification accuracy, based on the Key Action Discovery concept.

- An application of the proposed framework to the analysis of the conceptual stage of the bridge design task.

The proposed methodology offers higher accuracy in activity recognition and error detection than other leading methods. This is assessed using extensive experiments. Moreover, the algorithm's performance is assessed in several datasets with results showing that the proposed method generalises well to solve a variety of different problems.

The next section gives a brief overview of the impact of the work presented in this Thesis in the studied research area and its importance is discussed.

8.1 Impact of the proposed methodology

Analysis of prolonged, composite activities is a step further in identifying complex behaviour as it is an unexplored area. The techniques described in this Thesis can potentially offer a variety of applications, such as monitoring and identification of mistakes in composite tasks, improved automated surveillance and reinforced learning. The proposed methodology paves the way for numerous important applications in the fields of training for industrial engineering, education and cognitive psychology: automation in error detection as provided by the proposed system saves on time and human effort. Moreover, since the proposed methodology provides a means of analysing complex human behaviour, it can be used for understanding complex cognitive processes in psychology studies.

Current state of the art complex activity identification algorithms are only suitable for a limited range of applications. On the contrary, the methodology presented in this Thesis is designed to solve a variety

of complex tasks. It is shown that the proposed framework can be successfully applied to detect mistakes in a bridge design scenario and the task of calibrating a blood glucose monitor. Additionally, the proposed method is employed in a dataset illustrating everyday human activities and it is observed that the proposed algorithm achieves state of the art performance. Thus, the proposed methodology generalises well to solve a wide variety of problems.

An important aspect of the work presented in this Thesis is the ability of a system to identify mistakes in the execution of prolonged, composite activities. Such mistakes are typically difficult to detect since a correct and an erroneous execution of the same task are strongly correlated and therefore hard to distinguish. Being able to identify mistakes during the execution of prolonged, composite activities is very important: a mistake during a surgery can cost the patient's life. Mistakes in engineering studies can also prove costly: engineering design consists of a number of consecutive stages and the outcomes from each stage feed into the next one; thus, any undetected errors made at an earlier stage can feed into the next stages propagating through the entire process. Since mistakes in construction engineering can cost human lives [Frejus, 2009] or have severe environmental and/or financial consequences [Pellegrini, 2008], detecting mistakes as early as possible is of priority.

The work presented in this Thesis offers several important applications which are listed below.

Automated surveillance is a field which can benefit from the techniques presented here. In particular, the proposed method makes it possible to identify complex activities in noisy environments in multi-

media streams. This is useful for monitoring complex long-term manufacturing procedures in industrial environments.

Detecting mistakes in engineering tasks is a tedious, time consuming task which in addition requires the presence of experienced reviewing engineers. Thus, automation in activity verification and error detection as provided by the proposed system saves on time and human effort.

The benefit of the proposed system as an educational tool emanates once more from its ability to detect mistakes; recurring patterns of mistakes occurring in groups of people trained in the same institute might indicate a potential deficiency in its training programme.

An important aspect of this work is to investigate the cognitive processes underlying an engineer's decisions. In the bridge design task a KBS is used to obtain data from cognitive tasks. This method can be used for understanding engineering cognition using problem solving analysis methodologies.

8.2 Limitations

This section discusses the limitations of the methods proposed in this work. There exist two limitations.

The first limitation is that time sequences of discrete elements (which represent primitive actions) are required for the method to work. Yet, in theory all continuous time series can be converted to discrete by discretising the data into bins. In fact, section 5.4.2 demonstrates how the proposed method can handle a continuous data problem. However the discretisation process results in loss of accuracy which means that a method designed to work with continuous data would normally yield better performance.

The second limitation of the proposed framework concerns the method used to extract cognitive actions. Specifically, one of the requirements of the method presented in this thesis is that the participant types text using a suitable device (desktop computer, laptop, tablet, *etc.*). Therefore for applications where the participant is in motion (*e.g.* driving a vehicle) the proposed method might not be appropriate.

8.3 Future work

In future work, applicability of the proposed system to other areas involving complex, flexible tasks will be explored. An interesting area is surgery monitoring, where verification of procedures followed by doctors and detection of potential mistakes in them is an important problem.

Another research path which can be followed is to use the activity recognition methods proposed in this Thesis to improve performance of algorithms tackling technical problems such as object tracking in video or action detection. For example, the position of an occluded object, lost by the video tracker could be inferred by the activity performed. Also, given a detected activity an algorithm could disambiguate between two actions yielding equal probability of occurrence with greater confidence.

The algorithms developed in this work are supervised. Although they operate with high classification accuracy in the chosen applications, they require human labelled training data. In certain cases acquisition of training data is a time consuming, tedious process which may require employment of experts. It has to be noted that experts are hard to find and are sometimes unable to dedicate the amount of time required for data annotation. Therefore, an extension of the proposed

algorithms so that they operate in an unsupervised manner is a logical research direction.

8.4 Conclusion

The work presented in this Thesis investigates analysis of activities occurring in concurrent multimedia streams. The studied events may include cognitive tasks. The proposed algorithms are experimentally validated in several different datasets. It is thus proven that they can recognise prolonged, composite activities arising in these streams with high classification accuracy, even at the presence of noise. Hence the proposed methodology can be regarded as a small but significant step towards understanding prolonged, composite human activities.

Appendix 1

ESTIMATING PARAMETERS OF AN HHMM

The set of parameters for the entire HHMM is symbolised as $\lambda = \{\lambda^{q^d}\}_{d\in\{1,\ldots,D\}} = \{\{A^{q^d}\}_{d\in\{1,\ldots,D-1\}}, \{\Pi^{q^d}\}_{d\in\{1,\ldots,D-1\}}, \{B^{q^D}\}\}$. Parameters are learned given the structure of the HHMM, an initial set of parameters, λ and a sequence from the training set, $\bar{O} = \{o_1 o_2 \ldots o_T\} = o_{1:T}$ where T the length of the sequence, using the generalised Baum-Welch algorithm [Fine et al., 1998]. Directly following [Fine et al., 1998] it is shown how the variables used in the expectation step of this algorithm are calculated. Auxiliary parameters α, β, ξ are first defined as follows:

$$\alpha(t, t+k, q_i^d, q^{d-1}) = P(o_t \ldots o_{t+k}, q_i^d \text{ finished at } t+k | q^{d-1} \text{ started at } t) \tag{1.1}$$

where $\alpha(t, t+k, q_i^d, q^{d-1})$ the probability that the sequence $o_t \ldots o_{t+k}$ was generated by state q^{d-1} and that q_i^d was the last state activated by q^{d-1}. Also, $t \in [1, \ldots, T]$ and $t+k \in [1, \ldots, T]$.

$$\beta(t, t+k, q_i^d, q^{d-1}) = P(o_t \dots o_{t+k} | q_i^d \text{ started at } t, q^{d-1} \text{ finished at } t+k) \tag{1.2}$$

where $\beta(t, t+k, q_i^d, q^{d-1})$ is the probability that the sequence $o_t \dots o_{t+k}$ was generated by state q^{d-1} which finished at time $t+k$ having activated q^{d-1} at time t.

The probability of a horizontal transition from q_i^d to q_j^d, which are substates of q^{d-1} at t after emission of o_t and before emission of o_{t+1} is symbolised as $\xi(t, q_i^d, q_j^d, q^{d-1})$ and is given by the equation:

$$\xi(t, q_i^d, q_j^d, q^{d-1}) = P(o_1 \dots o_t, q_i^d \rightarrow q_j^d, o_1 \dots o_T | \lambda) \tag{1.3}$$

Two more auxiliary variables, γ_{in} and γ_{out} are based on ξ. Specifically, γ_{in} is the probability of a horizontal transition to q_i^d before emission of o_t and γ_{out} the probability of a horizontal transition from q_i^d to any of the states of level d after emission of o_t. These variables can be estimated by *Eqs.* 1.4 and 1.5:

$$\gamma_{in}(t, q_i^d, q^{d-1}) = \sum_{k=1}^{|q^{d-1}|} \xi(t-1, q_k^d, q_i^d, q^{d-1}) \tag{1.4}$$

$$\gamma_{out}(t, q_i^d, q^{d-1}) = \sum_{k=1}^{|q^{d-1}|} \xi(t, q_i^d, q_k^d, q^{d-1}) \tag{1.5}$$

The path variable χ is defined as the probability that q^{d-1} was entered at t before emission of o_t and activated q_i^D and is given by the equation:

$$\chi(t, q_i^d, q^{d-1}) = P(o_1 \dots o_{t-1}, \begin{matrix} q^{d-1} \\ \downarrow \\ q_i^d \end{matrix}, o_t \dots o_T|\lambda) \tag{1.6}$$

With the aid of the above defined variables the following expectations can be estimated:

First, the expected number of horizontal transitions from q_i^d to q_j^d. These two states are both substates of q^{d-1}.

$$\sum_{t=1}^{T-1} \xi(t, q_i^d, q_j^d, q^{d-1}) \tag{1.7}$$

$$\sum_{t=2}^{T} \gamma_{in}(t, q_i^d, q^{d-1}) = \sum_{k=1}^{|q^{d-1}|} \sum_{t=2}^{T} \xi(t-1, q_k^d, q_i^d, q^{d-1}), \tag{1.8}$$

which gives the expected number of horizontal transitions to state q_i^d from all substates of level d.

$$\sum_{t=1}^{T-1} \gamma_{out}(t, q_i^d, q^{d-1}) = \sum_{k=1}^{|q^{d-1}|} \sum_{t=2}^{T} \xi(t, q_i^d, q_k^d, q^{d-1}), \tag{1.9}$$

which gives the expected number of horizontal transitions from state q_i^d to any substate of level d.

$$\sum_{t=1}^{T} \chi(t, q_i^d, q^{d-1}), \tag{1.10}$$

which gives the expected number of vertical transitions from state q^{d-1} to q_i^d.

$$\sum_{i=1}^{|q^{d-1}|} \sum_{t=1}^{T} \chi(t, q_i^d, q^{d-1}), \tag{1.11}$$

which gives the expected number of vertical transitions from state q^{d-1}

any of its substates at level d.

$$\sum_{t=1}^{T} \chi(t, q_i^D, q^{D-1}) + \sum_{t=2}^{T} \gamma_{in}(t, q_i^D, q^{D-1}) = \sum_{t=1}^{T-1} \gamma_{out}(t, q_i^D, q^{D-1}), \quad (1.12)$$

which gives the expected number of vertical transitions from state q^{D-1} to production state q_i^D.

Having estimated the above expectations, the new set of parameters is given by the following equations:

$$\hat{\pi}^{q^1}(q_i^2) = \chi(t, q_i^2, q^1) \quad (1.13)$$

$$\hat{\pi}^{q^{d-1}}(q_i^d) = \frac{\sum_{t=1}^{T} \chi(t, q_i^d, q^{d-1})}{\sum_{i=1}^{|q^{d-1}|} \sum_{t=1}^{T} \chi(t, q_i^d, q^{d-1})}, \ (2 < d < D) \quad (1.14)$$

$$\hat{a}_{ij}^{q^{d-1}} = \frac{\sum_{t=1}^{T} \xi(t, q_i^d, q_j^d, q^{d-1})}{\sum_{t=1}^{T} \gamma_{out}(t, q_i^d, q^{d-1})} \quad (1.15)$$

$$\hat{b}_{q_i^D}^{q^{d-1}}(\upsilon_k) = \frac{\sum_{o_t=\upsilon_k} \chi(t, q_i^D, q^{D-1}) + \sum_{t>1, o_t=\upsilon_k} \gamma_{in}(t, q_i^D, q^{D-1})}{\sum_{t=1}^{T} \chi(t, q_i^D, q^{D-1}) + \sum_{t=2}^{T} \gamma_{in}(t, q_i^D, q^{D-1})} \quad (1.16)$$

Appendix 2

INFERENCE IN HHMM

The most probable state sequence can be estimated using the generalised Viterbi algorithm. This algorithm is presented in this section, directly following [Fine et al., 1998].

Three variables are retained for each pair of states (q^{d-1}, q_i^d):

1. $\delta(t, t+k, q_i^d, q^{d-1})$ which represents the likelihood of the most probable state sequence producing observation sequence $\{o_t, \ldots, o_{t+k}\}$ under the assumption that it was produced by a recursive activation starting at t from q^{d-1} and ending at q_i^d and returned to q^{d-1} at $t+k$.

2. $\psi(t, t+k, q_i^d, q^{d-1})$ which represents the most probable state activated by q^{d-1} before q_i^d. In the case that such a state does not exist it is $\psi(t, t+k, q_i^d, q^{d-1}) := 0$.

3. $\tau(t, t+k, q_i^d, q^{d-1})$ which is defined as the time at which state q_i^d was most probable to be activated by parent state q^{d-1}. In case the whole subsequence was produced by q_i^d it is $\tau(t, t+k, q_i^d, q^{d-1}) := t$.

For notation simplification, the functional $\mathcal{MAX}$ is defined as follows:

$$\mathcal{MAX}_{l \in S}\{f(l)\} := \left(\max_{l \in S} \{f(l)\}, \arg\max_{l \in S} \{f(l)\} \right) \tag{2.1}$$

Starting from the production states the algorithm calculates δ, ψ and τ bottom-up.

For the production states it is:

1. Initialisation:

$$\delta(t, t, q_i^D, q^{D-1}) = \pi^{q^{D-1}}(q_i^D) b^{q_i^D}(ot) \tag{2.2}$$

$$\psi(t, t, q_i^D, q^{D-1}) = 0 \tag{2.3}$$

$$\tau(t, t, q_i^D, q^{D-1}) = t \tag{2.4}$$

2. Recursion:

$$\begin{aligned} &(\delta(t, t+k, q_i^D, q^{D-1}), \psi(t, t+k, q_i^D, q^{D-1})) = \\ &\mathcal{MAX}_{1 \le j \le |q^{D-1}|} \left\{ \delta(t, t+k-1, q_j^D, q^{D-1}) \alpha_{ji}^{q_i^D} b^{q_i^D}(o_{t+k}) \right\} \end{aligned} \tag{2.5}$$

$$\tau(t, t+k, q_i^D, q^{D-1}) = t + k \tag{2.6}$$

For the internal states it is:

1. Initialisation:

$$\delta(t, t, q_i^d, q^{d-1}) = \max_{1 \le j \le |q_i^d|} \left\{ \pi^{q^{d-1}}(q_i^d) \delta(t, t, q_r^{d+1}, q_i^d) \alpha_{r,end}^{q_i^d} \right\} \tag{2.7}$$

$$\psi(t, t, q_r^{d+1}, q_i^d) = 0 \tag{2.8}$$

$$\tau(t, t, q_r^{d+1}, q_i^d) = t \tag{2.9}$$

2. Recursion:

   (a) For $t' = t+1, \ldots, t+k$ set:

$$R = \max_{1 \le r \le |q_i^d|} \left\{ \delta(t', t+k, q_r^{d+1}, q_i^d) \alpha_{r,end}^{q_i^d} \right\} \tag{2.10}$$

$$(\Delta(t'), \Psi(t')) = \mathcal{MAX}_{1 \le j \le |q^{D-1}|} \left\{ \delta(t, t'-1, q_j^d, q^{d-1}) \alpha_{ji}^{q^{d-1}} R \right\} \tag{2.11}$$

   (b) For t set:

$$\Delta(t) = \pi^{q^{d-1}}(q_i^d) \max_{1 \le r \le |q_i^d|} \left\{ \delta(t', t+k, q_r^{d+1}, q_i^d) \alpha_{r,end}^{q_i^d} \right\} \tag{2.12}$$

$$\Psi(t) = 0 \tag{2.13}$$

   (c) The most probable switching time can be found as follows:

$$\left(\delta(t, t+k, q_i^d, q^{d-1}), \tau(t, t+k, q_i^d, q^{d-1}) \right) = \mathcal{MAX}_{t \le t' \le t+k} \Delta(t') \tag{2.14}$$

$$\psi(t, t+k, q_i^d, q^{d-1}) = \Psi(\tau(t, t+k, q_i^d, q^{d-1})) \tag{2.15}$$

The probability of the most probable state sequence can be calculated from the equation:

$$(P^*, q_{last}^2) = \mathcal{MAX}_{q_i^2} \left\{\delta(1, T, q_i^2, q^1)\right\} \tag{2.16}$$

To find the most probable state sequence the lists ψ and τ starting from $\delta(1, T, q_{last}^2, q^1)$ and $\psi(1, T, q_{last}^2, q^1)$ have to be scanned.

Appendix 3

BRIDGE TASK DATASET

All sequences used in the experiments are given in this section. Sequences used for training are shown in Table 3.1 and for testing in Table 3.2.

No.	Class	Sequence	Training dataset
1	1	axwbumnoc	
2	1	abuxwqxwryzxwc	
3	1	axwbulxwmyzxwc	
4	1	axwbusnopvtc	
5	1	axwbuyzxwc	
6	2	axwyzsxwtc	
7	2	axwlxwqrxwmyzxwc	
8	2	axwFxwyzpvxwGfxwhixwc	
9	2	axwqdxwFxwGyzAxwyzKpvxwrc	
10	1	axwbuqxwrc	
11	1	axwbunopvyzxwc	
12	2	axwhxwiyznxwc	
13	3	sxwdxwefxwghxwit	
14	3	sdexwfgyzhxwit	
15	3	sxwAxwZfxwgdxwbueyzxwhit	
16	3	sxwfxwgqxwderhxwit	
17	3	sxwfxwgxwdeyzxwhxwit	
18	3	sfgxwdehnopvxwit	
19	3	sfxwgdxweyzhxwit	
20	3	sxwPxwRxwhxwixwfgdxwet	
21	3	sxwhxwixwdehxwiyzxwfgxwyzxwt	
22	3	sxwhixwdxwexwyzxwfxwgt	
23	3	shxwideyzxwfgxwt	
24	3	sxwhxwixwdelxwmyzxwfxwgt	
25	4	shxwpvixwdyzlmexwt	
26	4	sxwyzxwdxwexwlbuhixwmt	
27	4	sfxwgpvxwdexwlmt	
28	4	sxwyzbufgxwyzhixwt	
29	4	sdxwehpvxwlmixwt	
30	4	sfxwgxwhpvxwlmit	
31	4	sxwhixwyzpvfgxwt	
32	4	sfxwgxwpvdeyzxwlmxwt	
33	4	sxwhxwidyzxwet	
34	6	jxwlxwmxwnyzxwoxwpvxwk	
35	5	jnxwoxwqrxwpvExwKxwlyzpvmAxwZxwk	
36	5	jqxwrpvxwlmAZhxwFGixwPxwRnxwoxwk	
37	5	jxwqrxwpvxwnAZxwyzxFxwyzGwoxwPxwRxwlyzxwmxwk	
38	6	jlxwmxwqrxwExwKnoyzxwFxwGpvxwPxwRxwk	
39	6	jxwlmxwnxwopvEKxwAxwZxwyzxwk	
40	6	jxwyzxwlmxwqrnxwopvxwyzxwk	
41	5	jnxwoxwFxwGqyzPxwRrpvxwPxwyzxwRxwlpvmxwk	
42	5	jxwqrxwpvlmEKxwnAZoxwk	
43	5	jqrxwpvxwFxwGxwPxwRxwnAxwZstxwoxwlmk	
44	6	jlxwmnxwFGoyzAZxwPxwRxwpvxwyzxwk	
45	5	jxwnxwoAZxwqrpvxwFGlmxwk	
46	5	jqrxwpvlmyzExwKxwnAxwZxwok	
47	6	jxwpvnoxwyzxwlmFxwGxwk	
48	6	jxwlxwmnxwopvAZyzxwk	
49	6	jlmxwnxwbuFGodepvxwk	
50	6	jxwlxwmxwnxwbuAxwZoyzxwpvxwk	
51	5	jxwnoxwqrxwpvAZlmyzxwFGxwk	
52	5	jqrxwpvxwlxwEKmxwFGnyzxwPxwRoxwyzxwk	
53	6	jnxwyzxwolmAxwZxwpvbuxwqxwFxwGrxwk	
54	6	jlxwmxwAqxwZstPbuRrnoxwpvxwyzk	

Table 3.1: Training dataset.

No.	Class	Sequence	Testing dataset
1	1	axwbumnopvxwc	
2	1	abuxwqxwburyzxwc	
3	1	axwbulxwhmyzxwc	
4	1	axwbudexwbujyzxwc	
5	1	axwbuAKlxwmyzxwpvxwc	
6	1	ahixwqrxwdebulxwFxwpvGdmyzxwc	
7	1	axwbulxwmyzxwc	
8	1	axwbumnopvxwc	
9	1	axwbuyzxwpvxwyzxwc	
10	2	axwfxwlxwsxwc	
11	2	axwlmyzExwKxwc	
12	2	apvxwhixwdestc	
13	2	axwfgxwdestc	
14	2	axwlmyzxwc	
15	2	axwbuxwyzjxwnoyzxwc	
16	1	abustxwc	
17	1	axwbuqxwrc	
18	1	axwbunopvyzxwc	
19	2	axwdxwnxwyzpvqxwc	
20	2	axwyzfxwgpvxwc	
21	2	axwhxwyzxwixwyzxjwyzxwc	
22	2	axwyznxwyzxwc	
23	2	axwyzxwpvxwc	
24	2	axwyzxwc	
25	3	sxwlpvmxwbunjpvoxwdpvxkwefxwghxwit	
26	3	sxwlpvmfxwgxwnjohxwidxkwexwt	
27	3	sfgxwdehnopvxwit	
28	3	sfxwgdxweyzhxwit	
29	3	shxwixwfgdxwet	
30	3	sxwyzxwFhxwixwyzaxwGcxwdxwet	
31	3	shxwifgnoxwpvdexwt	
32	3	sxwhxwixwfxwgxwdxwet	
33	3	sxwhxwifnopvxwgyzxwdyzxwet	
34	3	syzxwhxwidxwyzxwefxwgxwtxwlbuhi	
35	3	sxwhxwixwdehxwiyzxwfgxwyzxwt	
36	3	sxwhxwidexwfyzxwgt	
37	4	sdbupvxwnoxwlmyzxwqrnxwexwt	
38	4	sdxwexwlhixwmt	
39	4	sxwhixwyzfnyzxwgxwt	
40	4	sxwbufgxwyzhixwt	
41	4	sfxwgxwdhiexwt	
42	4	syzxwbuxwfxwgxwyzhxwyzxwixwyzt	
43	4	sdxwehxwixwt	
44	4	sxwPxwRyzxwfgxwyzhixwt	
45	4	sxwyzbuyzhixwfxwgxwt	
46	4	sxwhixwyzfyzxwgxwt	
47	4	sxwyzExwfnxwgxwhixwtxwPxwR	
48	4	sxwyzxwdbuxwpvxwlmyznoyzxwqrnxwqret	
49	6	jlxwmxwFxwyzxwGxwnyzxwExwKoxwpvxwk	
50	5	jnxwoxwFGqrxwPxwRxwAZpvxwlyzmxwk	
51	5	jAxwZqxwrpvxwlExwKmhxwixwnxwoxwk	
52	5	jxwqrxwnxwPxwRxwoxwAxwZlxwpvmxwdexwbuxwk	
53	6	jxwyzxwlyzxwExwKhmxwqyzrxwnoyzxwpvxwk	
54	6	jlxwmxwnxwExwKopvxwk	
55	6	jxwAZxwlmxwqrFxwGnxwopvk	
56	5	jnxwoxwqAxwZyzrpvFxwGxwPxwRlmkxwPbuR	
57	5	jxwEKqrxwpvlmExwKxwnxwok	
58	5	jxwyzxwqrxwExwKpvxwAxwZnstxwoxwlmk	
59	6	jlxwmnxwFGoyzxwPxwRxwpvxwk	
60	5	jxwnoxwqrAxwExwKyzxwZpvxwlmxwk	
61	5	jqrxwpvlmAxwZyzhxwFxwGxwnxwok	
62	6	jlxwmxwnxwFxwyzxwGbuPxwRodeAZpvxwk	
63	6	jxwyzlxwEKmdxwnxwbuAZoyzxwpvk	
64	5	jdxwyzepvxwbuxwqrExwKxwpvAxwZlmyzxwk	
65	5	jxwqrxwlxwFxwGdpvmAxwZxwPxwRnxwok	
66	5	jxwqxwrxwpvAxwyzZlxwdpvmEKxwnyzxwoxwk	
67	5	jxwdexwnbuxwoxwPxwRxwqrAxwZlxwpvmxwk	
68	6	jlxwmxwqxwstExwKrnoxwpvk	
69	6	jdxwexwyzqxwFGryzlxwpvxwmk	
70	6	jyzxwnoxwlxwmyzqxwPxwRxwyzrxwpvk	
71	6	jxwdxweyzxwqxwryzlExwyzKxwpvxwmxwk	
72	6	jxwdxwexwyzqxwryzlxwpvmxwk	

Table 3.2: Testing dataset.

Appendix 4

PARAMETERS OF THE LEARNED HHMM

The set of parameters of the learned HHMM, λ with $\lambda = \{\lambda^{q^d}\}_{d\in\{1,\ldots,D\}} = \{\{A^{q^d}\}_{d\in\{1,\ldots,D-1\}}, \{\Pi^{q^d}\}_{d\in\{1,\ldots,D-1\}},$
$\{B^{q^D}\}\}$ is given in this section.

Initial probabilities, $\{\Pi^{q^d}\}_{d\in\{1,\ldots,D-1\}}$ for the model are given in Table 4.1.

Emission probabilities of production states at level $D = 3$, B^{q^3}, are given in Table 4.3. Since each state of level 3 emits one and only one action with probability 1, for simplicity we symbolise each state at of level $D = 3$ with its corresponding action.

The transition probability matrix for level $d = 2$, A^{q^1} is given in Table 4.2.

Learned transition probability matrices for level $d = 3$, A^{q^2} are given in Fig. 4.1, 4.2 and 4.3. To cover state occurrences and state transitions not present in the examples of the training dataset, each transition matrix of level $d = 3$, A^{q^2} is extended to a $n \times n$ matrix, $A^{q^2,mod}$ where $n = |V_{T,C}|$ the number of key actions of the task. The added entries represent the task key actions missing from A^{q^2}. All transition probabilities which are either not defined in matrix $A^{q^2,mod}$

d	k	i	$\pi^{q_k^d}(q_i^{d+1})$	q_i^{d+1}
1		1	0.166	
1		2	0.166	
1		3	0.166	
1		4	0.166	
1		5	0.166	
1		6	0.166	
2	1	1	1.000	a
2	2	1	1.000	a
2	3	19	1.000	s
2	4	19	1.000	s
2	5	10	1.000	j
2	6	10	1.000	j

Table 4.1: Initial probabilities of the HHMM.

	q_1^2	q_2^2	q_3^2	q_4^2	q_5^2	q_6^2	q_{end}^2
q_1^2	0.142	0.142	0.142	0.142	0.142	0.142	0.142
q_2^2	0.142	0.142	0.142	0.142	0.142	0.142	0.142
q_3^2	0.142	0.142	0.142	0.142	0.142	0.142	0.142
q_4^2	0.142	0.142	0.142	0.142	0.142	0.142	0.142
q_5^2	0.142	0.142	0.142	0.142	0.142	0.142	0.142
q_6^2	0.142	0.142	0.142	0.142	0.142	0.142	0.142

Table 4.2: Transition probabilities for states at level $d = 2$.

or equal to zero are set to a small value, 1×10^{-4}. Inference is performed using matrix $A^{q^2,mod}$.

i	σ	$b^{q_i^D}(\sigma)$ $D=3$
1	a	1.000
2	b	1.000
3	c	1.000
4	d	1.000
5	e	1.000
6	f	1.000
7	g	1.000
8	h	1.000
9	i	1.000
10	j	1.000
11	k	1.000
12	l	1.000
13	m	1.000
14	n	1.000
15	o	1.000
16	p	1.000
17	q	1.000
18	r	1.000
19	s	1.000
20	t	1.000
21	u	1.000
22	v	1.000
23	A	1.000
24	E	1.000
25	F	1.000
26	G	1.000
27	K	1.000
28	P	1.000
29	R	1.000
30	Z	1.000

Table 4.3: Emission probabilities of production states at level 3.

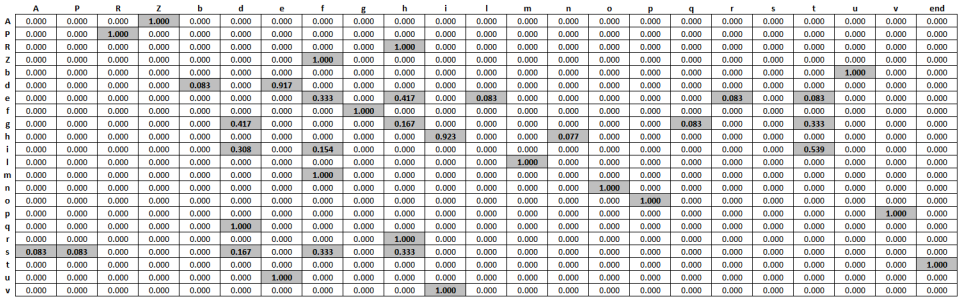
	a	b	c	l	m	n	o	p	q	r	s	t	u	v	end
a	0.000	**1.000**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
b	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**1.000**	0.000	0.000
c	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**1.000**
l	0.000	0.000	0.000	0.000	**1.000**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
m	0.000	0.000	**0.500**	0.000	0.000	**0.500**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
n	0.000	0.000	0.000	0.000	0.000	0.000	**1.000**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
o	0.000	0.000	**0.333**	0.000	0.000	0.000	0.000	**0.667**	0.000	0.000	0.000	0.000	0.000	0.000	0.000
p	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**1.000**	0.000
q	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**1.000**	0.000	0.000	0.000	0.000	0.000
r	0.000	0.000	**1.000**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
s	0.000	0.000	0.000	0.000	0.000	**1.000**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
t	0.000	0.000	**1.000**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
u	0.000	0.000	**0.143**	**0.143**	**0.143**	**0.143**	0.000	0.000	**0.286**	0.000	**0.143**	0.000	0.000	0.000	0.000
v	0.000	0.000	**0.500**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**0.500**	0.000	0.000	0.000

(a)

	A	F	G	K	a	c	d	f	h	i	l	m	n	p	q	r	s	t	v	end
A	0.000	0.000	0.000	**1.000**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
F	0.000	0.000	**0.500**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**0.500**	0.000	0.000	0.000	0.000	0.000	0.000
G	**0.500**	0.000	0.000	0.000	0.000	0.000	0.000	**0.500**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
K	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**1.000**	0.000	0.000	0.000	0.000	0.000	0.000
a	0.000	**0.200**	0.000	0.000	0.000	0.000	0.000	0.000	**0.200**	0.000	**0.200**	0.000	0.000	0.000	**0.200**	0.000	**0.200**	0.000	0.000	0.000
c	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**1.000**
d	0.000	**1.000**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
f	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**1.000**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
h	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**1.000**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
i	0.000	0.000	0.000	0.000	0.000	**0.500**	0.000	0.000	0.000	0.000	0.000	0.000	**0.500**	0.000	0.000	0.000	0.000	0.000	0.000	0.000
l	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**1.000**	0.000	0.000	0.000	0.000	0.000
m	0.000	0.000	0.000	0.000	0.000	**1.000**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
n	0.000	0.000	0.000	0.000	0.000	**1.000**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
p	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**1.000**	0.000
q	0.000	0.000	0.000	0.000	0.000	0.000	**0.500**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**0.500**	0.000	0.000	0.000	0.000
r	0.000	0.000	0.000	0.000	0.000	**0.500**	0.000	0.000	0.000	0.000	0.000	**0.500**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
s	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**1.000**	0.000	0.000
t	0.000	0.000	0.000	0.000	0.000	**1.000**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
v	0.000	0.000	**0.500**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**0.500**	0.000	0.000	0.000	0.000

(b)

Figure 4.1: Learned transition matrices for the third level of the HHMM. **(a):** $A^{q_1^2}$, **(b):** $A^{q_2^2}$.



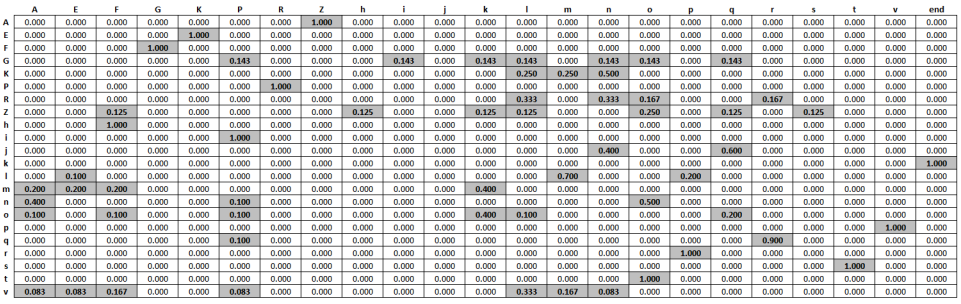
	A	P	R	Z	b	d	e	f	g	h	i	l	m	n	o	p	q	r	s	t	u	v	end
A	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
P	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
R	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Z	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
b	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000
d	0.000	0.000	0.000	0.000	0.083	0.000	0.917	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
e	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.333	0.000	0.417	0.000	0.083	0.000	0.000	0.000	0.000	0.000	0.083	0.000	0.083	0.000	0.000	0.000
f	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
g	0.000	0.000	0.000	0.000	0.000	0.417	0.000	0.000	0.000	0.167	0.000	0.000	0.000	0.000	0.000	0.000	0.083	0.000	0.000	0.333	0.000	0.000	0.000
h	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.923	0.000	0.000	0.077	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
i	0.000	0.000	0.000	0.000	0.000	0.308	0.000	0.154	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.539	0.000	0.000	0.000
l	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
m	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
n	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
o	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
p	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000
q	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
r	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
s	0.083	0.083	0.000	0.000	0.000	0.167	0.000	0.333	0.000	0.333	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
t	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000
u	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
v	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

(a)

	b	d	e	f	g	h	i	l	m	p	s	t	u	v	end
b	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000
d	0.000	0.000	0.833	0.000	0.000	0.000	0.000	0.167	0.000	0.000	0.000	0.000	0.000	0.000	0.000
e	0.000	0.000	0.000	0.000	0.000	0.167	0.000	0.500	0.000	0.000	0.000	0.333	0.000	0.000	0.000
f	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
g	0.000	0.000	0.000	0.000	0.000	0.400	0.000	0.000	0.000	0.400	0.000	0.200	0.000	0.000	0.000
h	0.000	0.000	0.000	0.000	0.000	0.000	0.572	0.000	0.000	0.429	0.000	0.000	0.000	0.000	0.000
i	0.000	0.286	0.000	0.000	0.000	0.000	0.000	0.000	0.143	0.143	0.000	0.429	0.000	0.000	0.000
l	0.167	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.833	0.000	0.000	0.000	0.000	0.000	0.000
m	0.000	0.000	0.167	0.000	0.000	0.000	0.333	0.000	0.000	0.000	0.000	0.500	0.000	0.000	0.000
p	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000
s	0.111	0.222	0.000	0.333	0.000	0.333	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
t	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000
u	0.000	0.000	0.000	0.500	0.000	0.500	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
v	0.000	0.333	0.000	0.167	0.000	0.000	0.167	0.333	0.000	0.000	0.000	0.000	0.000	0.000	0.000

(b)

Figure 4.2: Learned transition matrices for the third level of the HHMM. **(a):** $A^{q_3^2}$, **(b):** $A^{q_4^2}$.



	A	E	F	G	K	P	R	Z	h	i	j	k	l	m	n	o	p	q	r	s	t	v	end
A	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
E	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
F	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
G	0.000	0.000	0.000	0.000	0.000	0.143	0.000	0.000	0.000	0.143	0.000	0.143	0.143	0.000	0.143	0.143	0.000	0.143	0.000	0.000	0.000	0.000	0.000
K	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.250	0.250	0.500	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
P	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
R	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.333	0.000	0.333	0.167	0.000	0.000	0.167	0.000	0.000	0.000	0.000
Z	0.000	0.000	0.125	0.000	0.000	0.000	0.000	0.000	0.125	0.000	0.000	0.125	0.125	0.000	0.000	0.250	0.000	0.125	0.000	0.125	0.000	0.000	0.000
h	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
i	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
j	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.400	0.000	0.000	0.600	0.000	0.000	0.000	0.000	0.000
k	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000
l	0.000	0.100	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.700	0.000	0.000	0.200	0.000	0.000	0.000	0.000	0.000	0.000
m	0.200	0.200	0.200	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.400	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
n	0.400	0.000	0.000	0.000	0.000	0.100	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.500	0.000	0.000	0.000	0.000	0.000	0.000	0.000
o	0.100	0.000	0.100	0.000	0.000	0.100	0.000	0.000	0.000	0.000	0.000	0.400	0.100	0.000	0.000	0.000	0.000	0.200	0.000	0.000	0.000	0.000	0.000
p	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000
q	0.000	0.000	0.000	0.000	0.000	0.100	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.900	0.000	0.000	0.000	0.000	0.000
r	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000
s	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000
t	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
v	0.083	0.083	0.167	0.000	0.000	0.083	0.000	0.000	0.000	0.000	0.000	0.000	0.333	0.167	0.083	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

(a)

	A	E	F	G	K	P	R	Z	b	d	e	j	k	l	m	n	o	p	q	r	s	t	u	v	end
A	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.833	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.167	0.000	0.000	0.000	0.000	0.000	0.000
E	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
F	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
G	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.200	0.000	0.000	0.000	0.400	0.200	0.000	0.200	0.000	0.000	0.000	0.000	0.000
K	0.500	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.500	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
P	0.000	0.000	0.000	0.000	0.000	0.000	0.667	0.000	0.333	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
R	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.333	0.000	0.000	0.000	0.000	0.333	0.000	0.333	0.000	0.000	0.000	0.000	0.000
Z	0.000	0.000	0.000	0.000	0.000	0.167	0.000	0.000	0.000	0.000	0.000	0.000	0.333	0.000	0.000	0.000	0.167	0.167	0.000	0.000	0.167	0.000	0.000	0.000	0.000
b	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000
d	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
e	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
j	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.818	0.000	0.091	0.000	0.091	0.000	0.000	0.000	0.000	0.000	0.000	0.000
k	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000
l	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
m	0.182	0.000	0.091	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.546	0.000	0.000	0.182	0.000	0.000	0.000	0.000	0.000	0.000
n	0.000	0.000	0.091	0.000	0.000	0.000	0.000	0.182	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.727	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
o	0.091	0.000	0.091	0.000	0.000	0.000	0.000	0.000	0.000	0.091	0.000	0.000	0.000	0.182	0.000	0.000	0.000	0.546	0.000	0.000	0.000	0.000	0.000	0.000	0.000
p	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000
q	0.000	0.000	0.250	0.000	0.000	0.000	0.000	0.250	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.500	0.000	0.000	0.000	0.000	0.000
r	0.000	0.250	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.250	0.000	0.000	0.500	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
s	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000
t	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
u	0.250	0.000	0.250	0.000	0.000	0.000	0.250	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.250	0.000	0.000	0.000	0.000	0.000	0.000
v	0.091	0.091	0.000	0.000	0.000	0.091	0.000	0.000	0.091	0.000	0.000	0.000	0.546	0.000	0.000	0.091	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

(b)

Figure 4.3: Learned transition matrices for the third level of the HHMM. **(a):** $A^{q_5^2}$, **(b):** $A^{q_6^2}$.

Appendix 5

TRACKING PARAMETERS

For the dynamics model of the tracking algorithm used in this Thesis, given by the equation

$$\mathbf{x}_{t+1} = A\mathbf{x}_t + B\mathbf{x}_{t-1} + C\mathbf{u}_t \tag{5.1}$$

where $\mathbf{u}_t \sim \mathcal{N}(0, \Sigma)$, coefficients A, B and C as well as the covariance matrix, Σ are empirically defined as follows:

$$A = \begin{pmatrix} 2.0 & 0 & 0 \\ 0 & 2.0 & 0 \\ 0 & 0 & 2.0 \end{pmatrix} \tag{5.2}$$

$$B = \begin{pmatrix} -1.0 & 0 & 0 \\ 0 & -1.0 & 0 \\ 0 & 0 & -1.0 \end{pmatrix} \tag{5.3}$$

$$C = \begin{pmatrix} 1.0 & 0 & 0 \\ 0 & 1.0 & 0 \\ 0 & 0 & 1.0 \end{pmatrix} \tag{5.4}$$

$$\Sigma = \begin{pmatrix} 1.0 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 0.001 \end{pmatrix} \tag{5.5}$$

Appendix 6

VAN DER CORPUT SEQUENCE ESTIMATION

This chapter explains how the van der Corput sequence of numbers [van der Corput, 1935a, van der Corput, 1935b] is generated.

For a subsequence of natural numbers, $S = \{1, 2, 3, ...\nu\}$ a base is first chosen, b, with $b \in \mathbb{N}$. Each of the numbers of S is first converted to its base-b form, such that, for every $n \in S$ it is:

$$n = \sum_{j=0}^{m} a_j(n) b^j, \tag{6.1}$$

where $a_j(n)$ is the j^{th} digit of n in its base-b form and m is defined as follows:

$$m = min(k \in \mathbb{N}) : \frac{n}{b^k} \geq 1. \tag{6.2}$$

Then the van der Corput sequence, $\Phi_b(n)$, is defined as follows:

$$\Phi_b(n) = \sum_{j=0}^{m} a_j(n) b^{-j-1}. \tag{6.3}$$

REFERENCES

[pit, 1985] (1985). Geological and Geotechnical Maps. In *A Manual of Geology for Civil Engineers*, pages 137–180. World Scientific.

[Akerkar and Sajja, 2010] Akerkar, R. and Sajja, P. (2010). *Knowledge-Based Systems*. Jones and Bartlett Publishers, 1st edition.

[Allen, 1983] Allen, J. (1983). *Maintaining knowledge about temporal intervals*. PhD thesis.

[Amato and Di Lecce, 2011] Amato, A. and Di Lecce, V. (2011). Semantic classification of human behaviors in video surveillance systems. *W. Trans. on Comp.*, 10(10):343352.

[Baos et al., 2010] Baos, O., Pomares, H., and Rojas, I. (2010). Novel method for feature-set ranking applied to physical activity recognition. In Garca-Pedrajas, N., Herrera, F., Fyfe, C., Bentez, J. M., and Ali, M., editors, *Trends in Applied Intelligent Systems*, number 6097 in Lecture Notes in Computer Science, pages 637–642. Springer Berlin Heidelberg.

[Bay et al., 2008] Bay, H., Ess, A., Tuytelaars, T., and Van Gool, L. (2008). Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, 110(3):346359.

[Ben-Gal, 2005] Ben-Gal, I. (2005). Outlier detection. In Maimon, O.

and Rokach, L., editors, *Data Mining and Knowledge Discovery Handbook*, pages 131–146. Springer US.

[Bettadapura et al., 2013] Bettadapura, V., Schindler, G., Ploetz, T., and Essa, I. (2013). Augmenting bag-of-words: Data-driven discovery of temporal and structural information for activity recognition. In *2013 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2619–2626.

[Beymer and Konolige, 1999] Beymer, D. and Konolige, K. (1999). Real-time tracking of multiple people using continuous detection. Artificial intelligence center, SRI international report, Menlo Park, CA, USA.

[Bishop, 2007] Bishop, C. M. (2007). *Pattern Recognition and Machine Learning*. Springer.

[Blei et al., 2003] Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:9931022.

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

[Bobick and Wilson, 1997] Bobick, A. and Wilson, A. (1997). A state-based approach to the representation and recognition of gesture. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(12):1325–1337.

[Bodor et al., 2003] Bodor, R., Jackson, B., and Papanikolopoulos, N. (2003). Vision-based human tracking and activity recognition. In *Pro-*

ceedings of the 11th Mediterranean Conference on Control and Automation, pages 18–20. Kostrzewa Joseph.

[Bosch et al., 2007] Bosch, A., Zisserman, A., and Munoz, X. (2007). Representing shape with a spatial pyramid kernel. *Image Processing*, 5:401–408.

[Bradski, 1998] Bradski, G. R. (1998). Computer vision face tracking for use in a perceptual user interface. In *Workshop on Applications of Computer Vision*, pages 214–219, Princeton, NJ.

[Brand, 1996] Brand, M. (1996). Understanding manipulation in video. In *Proceedings of the Second International Conference on Automatic Face and Gesture Recognition, 1996*, pages 94–99.

[Brand et al., 1997] Brand, M., Oliver, N., and Pentland, A. (1997). Coupled hidden markov models for complex action recognition. In *, 1997 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1997. Proceedings*, pages 994 –999.

[Breiman, 2001] Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

[Bui, 2004] Bui, H. H. (2004). Hierarchical hidden markov models with general state hierarchy. *IN AAAI 2004*, pages 324—329.

[Bui et al., 2004] Bui, H. H., Phung, D. Q., and Venkatesh, S. (2004). Hierarchical hidden markov models with general state hierarchy. In *Proceedings of the 19th national conference on Artifical intelligence*, AAAI 04, page 324329, San Jose, California. AAAI Press.

[Campbell, 1988] Campbell, D. J. (1988). Task complexity: A review and analysis. *The Academy of Management Review*, 13(1):40–52.

[Carrillo and Lpez-Lpez, 2010] Carrillo, M. and Lpez-Lpez, A. (2010). Concept based representations as complement of bag of words in information retrieval. In Papadopoulos, H., Andreou, A. S., and Bramer, M., editors, *Artificial Intelligence Applications and Innovations*, volume 339, pages 154–161. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Chang et al., 2001] Chang, S.-F., Zhong, D., and Kumar, R. (2001). Real-time content-based adaptive streaming of sports videos. In *IEEE Workshop on Content-Based Access of Image and Video Libraries, 2001. (CBAIVL 2001)*, pages 139–146.

[Charniak and Goldman, 1993] Charniak, E. and Goldman, R. P. (1993). A bayesian model of plan recognition. *Artificial Intelligence*, 64(1):53–79.

[Chen et al., 2004] Chen, D., Malkin, R., and Yang, J. (2004). Multimodal detection of human interaction events in a nursing home environment. In *Proceedings of the 6th international conference on Multimodal interfaces*, ICMI '04, page 8289, New York, NY, USA. ACM.

[Chen et al., 2009] Chen, X., Hu, X., and Shen, X. (2009). Spatial weighting for bag-of-visual-words and its application in content-based image retrieval. In Theeramunkong, T., Kijsirikul, B., Cercone, N., and Ho, T.-B., editors, *Advances in Knowledge Discovery and Data Mining*, number 5476 in Lecture Notes in Computer Science, pages 867–874. Springer Berlin Heidelberg.

[Cho et al., 2004] Cho, K., Cho, H., and Um, K. (2004). Human action

recognition by inference of stochastic regular grammars. In Fred, A., Caelli, T. M., Duin, R. P. W., Campilho, A. C., and Ridder, D. d., editors, *Structural, Syntactic, and Statistical Pattern Recognition*, number 3138 in Lecture Notes in Computer Science, pages 388–396. Springer Berlin Heidelberg.

[Cho et al., 2006] Cho, K., Cho, H., and Um, K. (2006). Inferring stochastic regular grammar with nearness information for human action recognition. In Campilho, A. and Kamel, M., editors, *Image Analysis and Recognition*, number 4142 in Lecture Notes in Computer Science, pages 193–204. Springer Berlin Heidelberg.

[Choi and Choi, 1993] Choi, C. K. and Choi, I. H. (1993). An expert system for selecting types of bridges. *Computers & Structures*, 48(2):183–192.

[Chomat and Crowley, 1999] Chomat, O. and Crowley, J. (1999). Probabilistic recognition of activity using local appearance. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 2, pages –109 Vol. 2.

[Chomsky, 1956] Chomsky, N. (1956). Three models for the description of language. *IRE Transactions on Information Theory*, 2(3):113–124.

[Chotimongkol, 2008] Chotimongkol, A. (2008). *Learning the Structure of Task-Oriented Conversations from the Corpus of In-Domain Dialogs*. PhD thesis, Carnegie Mellon University, Pittsburgh, USA.

[Cielniak et al., 2003] Cielniak, G., Bennewitz, M., and Burgard, W. (2003). Where is . . . ? Learning and utilizing motion patterns of

persons with mobile robots. *In 18th Int. Joint Conf on Artificial Intelligence (IJCAI)*, pages 909—914.

[Clark et al., 2008] Clark, R. E., Feldon, D. F., Van Merrienboer, J. J., Yates, K. A., and Early, S. (2008). Cognitive task analysis. In *Handbook of research on educational communications and technology*, pages 578–593. Lawrence Erlbaum Associates, Mahwah, NJ, 3rd edition.

[Cohn et al., 2003] Cohn, A. G., Magee, D. R., Galata, A., Hogg, D. C., and Hazarika, S. M. (2003). Towards an architecture for cognitive vision using qualitative spatio-temporal representations and abduction. In Freksa, C., Brauer, W., Habel, C., and Wender, K. F., editors, *Spatial Cognition III*, number 2685 in Lecture Notes in Computer Science, pages 232–248. Springer Berlin Heidelberg.

[Comaniciu et al., 2000] Comaniciu, D., Ramesh, V., and Meer, P. (2000). Real-time tracking of non-rigid objects using mean shift. In *IEEE Conference on Computer Vision and Pattern Recognition, 2000. Proceedings*, volume 2, pages 142–149.

[Cortes and Vapnik, 1995] Cortes, C. and Vapnik, V. (1995). Support-Vector networks. *Machine Learning*, 20:273—297.

[Costello, 2008] Costello, E. (2008). *American Sign Language Dictionary*. Diversified Publishing, 2 edition.

[Criminisi et al., 2012] Criminisi, A., Shotton, J., and Konukoglu, E. (2012). Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning. *Foundations and Trends in Computer Graphics and Vision*, 7(2–3):81227.

[Damen and Hogg, 2007] Damen, D. and Hogg, D. (2007). Bicycle theft detection. In *International Crime Science Conference (CS2)*, London.

[Damen and Hogg, 2009] Damen, D. and Hogg, D. (2009). Recognizing linked events: Searching the space of feasible explanations. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2009*, pages 927–934.

[Deerwester et al., 1990] Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391407.

[Dollar et al., 2005] Dollar, P., Rabaud, V., Cottrell, G., and Belongie, S. (2005). Behavior recognition via sparse spatio-temporal features. In *Proceedings of the 14th International Conference on Computer Communications and Networks*, ICCCN '05, pages 65–72, Washington, DC, USA. IEEE Computer Society.

[Dubin, 2004] Dubin, D. (2004). The most influential paper gerard salton never wrote. *Library Trends*, 52(4):748–764.

[Duchenne et al., 2009] Duchenne, O., Laptev, I., Sivic, J., Bach, F., and Ponce, J. (2009). Automatic annotation of human actions in video. In *2009 IEEE 12th International Conference on Computer Vision*, pages 1491–1498.

[Duda et al., 2000] Duda, R. O., Hart, P. E., and Stork, D. G. (2000). *Pattern Classification*. John Wiley & Sons, Inc., New York, NY, USA, 2nd edition.

[Efron and Tibshirani, 1994] Efron, B. and Tibshirani, R. (1994). *An Introduction to the Bootstrap*. Monographs on Statistics & Applied Probability. Chapman & Hall/CRC, 1 edition.

[Efros et al., 2003] Efros, A., Berg, A., Mori, G., and Malik, J. (2003). Recognizing action at a distance. In *Ninth IEEE International Conference on Computer Vision, 2003. Proceedings*, pages 726–733 vol.2.

[Fahoum, 2010] Fahoum, J. (2010). *Bridge Design Manual*. American Association of State Highway and Transportation Officials, Washington.

[Fernyhough et al., 2000] Fernyhough, J., Cohn, A., and Hogg, D. (2000). Constructing qualitative event models automatically from video input. *Image and Vision Computing*, 18(2):81–103.

[Fine et al., 1998] Fine, S., Singer, Y., and Tishby, N. (1998). The Hierarchical Hidden Markov Model: Analysis and Applications. *Machine Learning*, 32(1):41–62.

[Forsyth and Ponce, 2011] Forsyth, D. A. and Ponce, J. (2011). *Computer Vision: A Modern Approach*. Pearson Education, Limited, 2nd edition.

[Föstner and Gülch, 1987] Föstner, M. and Gülch, E. (1987). A fast operator for detection and precise location of distinct points, corners and centers of circular features. In *Proceedings of the ISPRS Intercommission Workshop on Fast Processing of Photogrammetric Data*, pages 281–305.

[Fouse et al., 2011] Fouse, A., Weibel, N., Hutchins, E., and Hollan, J. D. (2011). ChronoViz: a system for supporting navigation of time-

coded data. In *Part 1 Proceedings of the 2011 annual conference extended abstracts on Human factors in computing systems*, CHI EA '11, page 299304, New York, NY, USA. ACM.

[Frejus, 2009] Frejus, O. (2009). Website dedicated to the disaster of malpasset. `http://frejus59.fr/en`. [Online; accessed 17 Mar 2013].

[Galata et al., 2001] Galata, A., Johnson, N., and Hogg, D. (2001). Learning variable length Markov models of behaviour. *Computer Vision and Image Understanding*, 81:398—413.

[Gall et al., 2012] Gall, J., Razavi, N., and Van Gool, L. (2012). An introduction to random forests for multi-class object detection. *Lecture Notes in Computer Science*, 7474:243–263.

[Genuer et al., 2010] Genuer, R., Poggi, J.-M., and Tuleau-Malot, C. (2010). Variable selection using random forests. *Pattern Recognition Letters*, 31(14):2225–2236.

[Gilbert et al., 2009] Gilbert, A., Illingworth, J., and Bowden, R. (2009). Fast realistic multi-action recognition using mined dense spatio-temporal features. In *2009 IEEE 12th International Conference on Computer Vision*, pages 925–931.

[Gill and Hicks, 2006] Gill, T. G. and Hicks, R. C. (2006). Task complexity and informing science: A synthesis. *Informing Science Journal*, 9:1–30.

[Gini, 1912] Gini, C. (1912). *Variabilitá mutabilitá (Variability and Mutability)*. Tipogr. di P. Cuppini.

[Guerra-Filho and Aloimonos, 2006] Guerra-Filho, G. and Aloimonos, Y. (2006). Learning parallel grammar systems for a human activity language. Technical Report CS-TR-4837, University of Maryland.

[Gupta et al., 2009] Gupta, A., Srinivasan, P., Shi, J., and Davis, L. (2009). Understanding videos, constructing plots learning a visually grounded storyline model from annotated videos. In *IEEE Conference on Computer Vision and Pattern Recognition, 2009. CVPR 2009*, pages 2012–2019.

[Guyon and Pereira, 1995] Guyon, I. and Pereira, F. (1995). Design of a linguistic postprocessor using variable memory length markov models. In *Proceedings of the Third International Conference on Document Analysis and Recognition, 1995*, volume 1, pages 454–457 vol.1.

[Hamid, 2008] Hamid, M. R. (2008). *A computational framework for unsupervised analysis of everyday human activities*. PhD Thesis. Georgia Institute of Technology.

[Hamid et al., 2005] Hamid, R., Johnson, A., Batta, S., Bobick, A., Isbell, C., and Coleman, G. (2005). Detection and explanation of anomalous activities: Representing activities as bags of event n-Grams. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 1031–1038 vol. 1.

[Hamid et al., 2007] Hamid, R., Maddi, S., Bobick, A., and Essa, I. (2007). Structure from statistics - unsupervised activity analysis using suffix trees. In *Proceedings of the IEEE Conference on Computer Vision (ICCV)*, volume 1, pages 1–8, Los Alamitos, CA, USA.

[Hamid et al., 2009] Hamid, R., Maddi, S., Johnson, A., Bobick, A., Essa, I., and Isbell, C. (2009). A novel sequence representation for unsupervised analysis of human activities. *Artificial Intelligence*, 173(14):1221–1244.

[Harris and Stephens, 1988] Harris, C. and Stephens, M. (1988). A combined corner and edge detector. In *In Proc. of Fourth Alvey Vision Conference*, pages 147–151.

[Hillebrand, 2011] Hillebrand, E. (2011). Poverty, growth and inequality over the next 50 years. pages 159–190.

[Ho, 2002] Ho, T. K. (2002). A data complexity analysis of comparative advantages of decision forest constructors. *Pattern Analysis & Applications*, 5(2):102–112.

[Hodges and Pollack, 2007] Hodges, M. R. and Pollack, M. E. (2007). An 'object-use fingerprint': the use of electronic sensors for human identification. In *Proceedings of the 9th international conference on Ubiquitous computing*, UbiComp '07, page 289303, Berlin, Heidelberg. Springer-Verlag.

[Hofmann, 1999] Hofmann, T. (1999). Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '99, page 5057, New York, NY, USA. ACM.

[Hong et al., 2002] Hong, N. K., Chang, S.-P., and Lee, S.-C. (2002). Development of ANN-based preliminary structural design systems for cable-stayed bridges. *Advances in Engineering Software*, 33(2):85–96.

[Ikizler and Duygulu, 2009] Ikizler, N. and Duygulu, P. (2009). Histogram of oriented rectangles: A new pose descriptor for human action recognition. *Image and Vision Computing*, 27(10):1515–1526.

[Ikizler and Forsyth, 2007] Ikizler, N. and Forsyth, D. (2007). Searching video for complex activities with finite state models. In *IEEE Conference on Computer Vision and Pattern Recognition, 2007. CVPR '07*, pages 1–8.

[Inomata et al., 2009] Inomata, T., Naya, F., Kuwahara, N., Hattori, F., and Kogure, K. (2009). Activity recognition from interactions with objects using Dynamic Bayesian Network. In *Proceedings of the 3rd ACM International Workshop on Context-Awareness for Self-Managing Systems*, pages 39–42, Nara, Japan.

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

[Jehn et al., 1999] Jehn, K. A., Northcraft, G. B., and Neale, M. A. (1999). Why differences make a difference: A field study of diversity, conflict and performance in workgroups. *Administrative Science Quarterly*, 44(4):741–763.

[John et al., 2002] John, B., Vera, A., Matessa, M., Freed, M., and Remington, R. (2002). Automating CPM-GOMS. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '02, page 147154, New York, NY, USA. ACM.

[Johns and Mahadevan, 2005] Johns, J. and Mahadevan, S. (2005). A

variational learning algorithm for the abstract hidden markov model. In *Proceedings of the 20th National Conference on Artificial Intelligence*, volume 1 of *AAAI'05*, page 914, Pittsburgh, Pennsylvania. AAAI Press.

[Johnson and Hogg, 1996] Johnson, N. and Hogg, D. (1996). Learning the distribution of object trajectories for event recognition. *Image and Vision Computing*, 14:583592.

[Joo and Chellappa, 2006] Joo, S.-W. and Chellappa, R. (2006). Attribute grammar-based event recognition and anomaly detection. In *Conference on Computer Vision and Pattern Recognition Workshop, 2006. CVPRW '06*, pages 107–114.

[Kaloskampis et al., 2011a] Kaloskampis, I., Hicks, Y., and Marshall, D. (2011a). Analysing engineering tasks using a hybrid machine vision and knowledge based system application. In *12th IAPR International Conference on Machine Vision Applications (MVA)*, volume 1, pages 495—498, Nara, Japan.

[Kaloskampis et al., 2011b] Kaloskampis, I., Hicks, Y., and Marshall, D. (2011b). Automatic analysis of composite activities in video sequences using key action discovery and hierarchical graphical models. In *Proceedings of 2nd IEEE Workshop on Analysis and Retrieval of Tracked Events and Motion in Imagery Streams (IEEE ARTEMIS 2011)*, pages 890– 897, Barcelona, Spain.

[Kaloskampis et al., 2011c] Kaloskampis, I., Hicks, Y., and Marshall, D. (2011c). Reinforcing conceptual engineering design with a hybrid computer vision, machine learning and knowledge based system framework.

In *2011 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 3242 –3249, Anchorage, AK, USA.

[Kieras, 1994] Kieras, D. (1994). A guide to GOMS task analysis. In *Handbook of HCI*. Elsevier, Amsterdam, The Netherlands, 2nd edition.

[Kira and Rendell, 1992] Kira, K. and Rendell, L. A. (1992). A practical approach to feature selection. In *Proceedings of the ninth international workshop on Machine learning*, ML92, page 249256, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Kirwan and Ainsworth, 1992] Kirwan, B. and Ainsworth, L. K. (1992). *A Guide To Task Analysis: The Task Analysis Working Group*. CRC Press.

[Kishore et al., 2004] Kishore, R., Agrawal, M., and Rao, H. R. (2004). Determinants of sourcing during technology growth and maturity: An empirical study of e-commerce sourcing. *J. Manage. Inf. Syst.*, 21(3):4782.

[Kitani et al., 2007] Kitani, K., Sato, Y., and Sugimoto, A. (2007). Recovering the basic structure of human activities from a video-based symbol string. In *IEEE Workshop on Motion and Video Computing, 2007. WMVC '07*, pages 9–17.

[Knuth, 1968] Knuth, D. E. (1968). Semantics of context-free languages. *Mathematical systems theory*, 2(2):127–145.

[Kohavi and Provost, 1998] Kohavi, R. and Provost, F. (1998). Glossary of terms. *Machine Learning*, 30(2-3):271274.

[Krahnstoever et al., 2005] Krahnstoever, N., Rittscher, J., Tu, P., Chean, K., and Tomlinson, T. (2005). Activity recognition using visual tracking and RFID. In *Seventh IEEE Workshops on Application of Computer Vision, 2005. WACV/MOTIONS '05 Volume 1*, volume 1, pages 494–500.

[Kriegel et al., 2011] Kriegel, H.-P., Krger, P., Sander, J., and Zimek, A. (2011). Density-based clustering. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(3):231240.

[Krusse and Heiser, 2001] Krusse, W. G. I. and Heiser, J. G. (2001). *Computer Forensics: Incident Response Essentials*. Addison-Wesley, Boston, USA, 1 edition.

[Kuettel et al., 2010] Kuettel, D., Breitenstein, M., Van Gool, L., and Ferrari, V. (2010). What's going on? discovering spatio-temporal dependencies in dynamic scenes. In *2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1951–1958.

[Laerhoveni, 2012] Laerhoveni, K. V. (2012). An introduction to activity rrecognition. `http://www.ess.tu-darmstadt.de/sites/default/files/act_rec/actrec_1.html`. [Online; accessed 25 Mar 2013].

[Lama et al., 2013] Lama, A., Wrbel, Z., and Dziech, A. (2013). Depth estimation in image sequences in single-camera video surveillance systems. In Dziech, A. and Czyewski, A., editors, *Multimedia Communications, Services and Security*, number 368 in Communications in Computer and Information Science, pages 121–129. Springer Berlin Heidelberg.

[Lane and Schooler, 2004] Lane, S. M. and Schooler, J. W. (2004).

Skimming the surface. verbal overshadowing of analogical retrieval. *Psychological science*, 15(11):715–719. PMID: 15482442.

[Laptev and Lindeberg, 2003] Laptev, I. and Lindeberg, T. (2003). Space-time interest points. In *Ninth IEEE International Conference on Computer Vision, 2003. Proceedings*, volume 1, pages 432–439.

[Laptev et al., 2008] Laptev, I., Marszalek, M., Schmid, C., and Rozenfeld, B. (2008). Learning realistic human actions from movies. pages 1–8, Anchorage, AK, USA. IEEE.

[Lavee et al., 2009] Lavee, G., Rivlin, E., and Rudzsky, M. (2009). Understanding video events: A survey of methods for automatic interpretation of semantic occurrences in video. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 39(5):489–504.

[Laxton et al., 2007] Laxton, B., Lim, J., and Kriegman, D. (2007). Leveraging temporal, contextual and ordering constraints for recognizing complex activities in video. In *IEEE Conference on Computer Vision and Pattern Recognition, 2007. CVPR '07*, pages 1–8.

[Lebanon et al., 2007] Lebanon, G., Mao, Y., and Dillon, J. (2007). The locally weighted bag of words framework for document representation. *J. Mach. Learn. Res.*, 8:24052441.

[Leibe et al., 2005] Leibe, B., Seemann, E., and Schiele, B. (2005). Pedestrian detection in crowded scenes. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005. CVPR 2005*, volume 1, pages 878–885 vol. 1.

[Leistner et al., 2009] Leistner, C., Saffari, A., Santner, J., and Bischof, H. (2009). Semi-Supervised random forests. In *IEEE International Conference on Computer Vision (ICCV)*, pages 506–513.

[Lepetit et al., 2005] Lepetit, V., Lagger, P., and Fua, P. (2005). Randomized trees for Real-Time keypoint recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 775–781, Washington, DC, USA.

[Lewis, 1982] Lewis, C. (1982). Using the "Thinking aloud" method in cognitive interface design. Technical report RC-9265, IBM.

[Liao, 2006] Liao, L. (2006). *Location-Based Activity Recognition, PhD Thesis*. PhD thesis, University of Washington, Washington, DC, USA.

[Lindeberg, 1997] Lindeberg, T. (1997). On automatic selection of temporal scales in time-causal scale-space. In Sommer, G. and Koenderink, J. J., editors, *Algebraic Frames for the Perception-Action Cycle*, number 1315 in Lecture Notes in Computer Science, pages 94–113. Springer Berlin Heidelberg.

[Lindeberg, 1998] Lindeberg, T. (1998). Feature detection with automatic scale selection. *Int. J. Comput. Vision*, 30(2):79–116.

[Lowe, 1999] Lowe, D. (1999). Object recognition from local Scale-Invariant features. volume 2, pages 1150–1157 vol.2.

[Lowe, 2004] Lowe, D. G. (2004). Distinctive image features from Scale-Invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110.

[MacQueen, 1967] MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. volume 1, pages 281–297. University of California Press.

[Maldonado and Weber, 2009] Maldonado, S. and Weber, R. (2009). A wrapper method for feature selection using support vector machines. *Information Sciences*, 179(13):2208–2217.

[Malekly et al., 2010] Malekly, H., Meysam Mousavi, S., and Hashemi, H. (2010). A fuzzy integrated methodology for evaluating conceptual bridge design. *Expert Systems with Applications*, 37(7):4910–4920.

[Manohar et al., 2006] Manohar, V., Boonstra, M., Korzhova, V., Soundararajan, P., Goldgof, D., Kasturi, R., Prasad, S., Raju, H., Bowers, R., and Garofolo, J. (2006). Pets vs. vace evaluation programs: A comparative study. In *In Proceedings 9th IEEE International Workshop Performance Evaluation of Tracking and Surveillance*.

[Mashood et al., 2007] Mashood, P. K., Krishnamoorthy, C. S., and Ramamurthy, K. (2007). KB-GA-Based hybrid system for layout planning of multistory buildings. *Journal of Computing in Civil Engineering*, 21(4):229–337.

[McCowan et al., 2005] McCowan, I., Gatica-Perez, D., Bengio, S., Lathoud, G., Barnard, M., and Zhang, D. (2005). Automatic analysis of multimodal group actions in meetings. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):305–317.

[Menn, 1990] Menn, C. (1990). *Prestressed Concrete Bridges*. Birkhauser Verlag, Basel-Boston-Berlin.

[Merriënboer, 1997] Merriënboer, J. J. G. v. (1997). *Training Complex Cognitive Skills: A Four-Component Instructional Design Model for Technical Training*. Educational Technology.

[Mikolajczyk and Schmid, 2005] Mikolajczyk, K. and Schmid, C. (2005). A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630.

[Miller and Attwood, 2001] Miller, C. J. and Attwood, T. K. (2001). PSST... the probabilistic sequence search tool. volume 0, page 33, Los Alamitos, CA, USA. IEEE Computer Society.

[Moore, 1991] Moore, C. J. (1991). *An expert system for the conceptual design of bridges*. PhD Thesis, Cardiff University.

[Moore and Miles, 1991] Moore, C. J. and Miles, J. C. (1991). The development and verification of a user oriented KBS for the conceptual design of bridges. *Civil Engineering and Environmental Systems*, 8(2):81–86.

[Moore et al., 1997] Moore, C. J., Miles, J. C., and Rees, D. W. G. (1997). Decision support for conceptual bridge design. *Artificial Intelligence in Engineering*, 11(3):259–272.

[Moore and Essa, 2001] Moore, D. and Essa, I. (2001). Recognizing multi tasked activities using stochastic context-free grammar. In *Proceedings IEEE CVPR, Workshop on Models vs. Exemplars*, pages 770–776, Kauai, Hawaii, USA.

[Moore and Essa, 2002] Moore, D. and Essa, I. (2002). Recognizing multitasked activities from video using stochastic Context-Free grammar. pages 770—776.

[Moosmann et al., 2006] Moosmann, F., Triggs, B., and Jurie, F. (2006). Fast discriminative visual codebooks using randomized clustering forests. In *Conference on Neural Information Processing Systems (NIPS)*, pages 985–992.

[Murphy and Paskin, 2001] Murphy, K. P. and Paskin, M. A. (2001). Linear time inference in hierarchical HMMs. *In Proceedings of Neural Information Processing Systems*, pages 833–840.

[Naaman et al., 2004] Naaman, M., Harada, S., Wang, Q., Garcia-Molina, H., and Paepcke, A. (2004). Context data in geo-referenced digital photo collections. In *Proceedings of the 12th annual ACM international conference on Multimedia*, MULTIMEDIA '04, page 196203, New York, NY, USA. ACM.

[Naaman et al., 2005] Naaman, M., Yeh, R. B., Garcia-Molina, H., and Paepcke, A. (2005). Leveraging context to resolve identity in photo albums. In *Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries*, JCDL '05, page 178187, New York, NY, USA. ACM.

[Natarajan and Nevatia, 2007] Natarajan, P. and Nevatia, R. (2007). Coupled hidden semi markov models for activity recognition. In *IEEE Workshop on Motion and Video Computing, 2007. WMVC '07*, pages 10–10.

[Nguyen and Venkatesh, 2005] Nguyen, N. and Venkatesh, S. (2005). Discovery of activity structures using the hierarchical hidden markov model.

[Nguyen et al., 2003] Nguyen, N. T., Bui, H. H., Venkatesh, S., and West, G. (2003). Recognising and monitoring High-Level behaviours in

complex spatial environments. *In Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2:620—625.

[Nguyen et al., 2005] Nguyen, N. T., Phung, D. Q., Venkatesh, S., and Bui, H. (2005). Learning and detecting activities from movement trajectories using the Hierarchical Hidden Markov Model. *In Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2:955—960.

[Niebles et al., 2010] Niebles, J. C., Chen, C., and Fei-Fei, L. (2010). Modeling temporal structure of decomposable motion segments for activity classification. In *Proceedings of the 11th European Conference on Computer Vision (ECCV)*, ECCV'10, pages 392–405, Heraklion, Crete, Greece.

[Niebles et al., 2006] Niebles, J. C., Wang, H., and Fei-Fei, L. (2006). Unsupervised learning of human action categories using spatial-temporal words. In *In Proc. British Machine Vision Conference (BMVC)*, pages 127.1–127.10, Edinburgh. BMVA Press.

[Nimtawat and Nanakorn, 2009] Nimtawat, A. and Nanakorn, P. (2009). Automated layout design of beam-slab floors using a genetic algorithm. *Computers & Structures*, 87(21-22):1308–1330.

[Ogale et al., 2007] Ogale, A. S., Karapurkar, A., and Aloimonos, Y. (2007). View-invariant modeling and recognition of human actions using grammars. In Vidal, R., Heyden, A., and Ma, Y., editors, *Dynamical Vision*, number 4358 in Lecture Notes in Computer Science, pages 115–126. Springer Berlin Heidelberg.

[Oliver et al., 2002] Oliver, N., Horvitz, E., and Garg, A. (2002). Layered representations for human activity recognition. In *Proceedings of the IEEE 4th International Conference on Multimodal Interfaces*, pages 3–8.

[Oliver et al., 2000] Oliver, N., Rosario, B., and Pentland, A. (2000). A Bayesian computer vision system for modeling human interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence,*, 22(8):831–843.

[Osentoski et al., 2004] Osentoski, S., Manfredi, V., and Mahadevan, S. (2004). Learning hierarchical models of activity. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 891–896.

[Over, 2013] Over, P. (2013). TREC video retrieval evaluation: TRECVID. `http://trecvid.nist.gov/`. [Online; accessed 30 Aug 2013].

[Padoy et al., 2009] Padoy, N., Mateus, D., Weinland, D., Berger, M. O., and Navab, N. (2009). Workflow monitoring based on 3D motion features. In *2009 IEEE 12th International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 585–592.

[Pahl et al., 2007] Pahl, G., Beitz, W., Feldhusen, J., and Grote, K. (2007). *Engineering design: a systematic approach*. Springer.

[Papadimitriou et al., 2003] Papadimitriou, S., Kitagawa, H., Gibbons, P., and Faloutsos, C. (2003). LOCI: fast outlier detection using the local correlation integral. In *19th International Conference on Data Engineering, 2003. Proceedings*, pages 315–326.

[Parzen, 1962] Parzen, E. (1962). On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076.

[Patterson et al., 2003] Patterson, D. J., Liao, L., Fox, D., and Kautz, H. (2003). Inferring high-level behavior from low-level sensors. In Dey, A. K., Schmidt, A., and McCarthy, J. F., editors, *UbiComp 2003: Ubiquitous Computing*, number 2864 in Lecture Notes in Computer Science, pages 73–89. Springer Berlin Heidelberg.

[Pellegrini, 2008] Pellegrini, J. (2008). Proposed cable-stay bridge deemed a hazard to migratory birds. `http://www.niagarafallsreview.ca/2008/04/23/peace-bridge-authority-goes-to-plan-b`. [Online; accessed 17 Mar 2013].

[Pentland, 1998] Pentland, A. (1998). Smart rooms, smart clothes. In *Proceedings of the Fourteenth International Conference on Pattern Recognition*, volume 2, pages 949–953 vol.2.

[Pérez et al., 2002] Pérez, P., Hue, C., Vermaak, J., and Gangnet, M. (2002). Color-based probabilistic tracking. *In Proceedings of the European Conference on Computer Vision (ECCV)*, 1:661—675.

[Philbey et al., 1993] Philbey, B. T., Miles, C., and Miles, J. C. (1993). User-interface design for a conceptual bridge design expert system. *Computing Systems in Engineering*, 4(23):235–241.

[Provost and Fawcett, 2001] Provost, F. and Fawcett, T. (2001). Robust classification for imprecise environments. *Machine Learning*, 42(3):203–231.

[Rabiner, 1989] Rabiner, L. (1989). A tutorial on Hidden Markov Models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

[Ratanamahatana and Keogh, 2004] Ratanamahatana, C. and Keogh, E. (2004). Making time-series classification more accurate using learned constraints. In *Proceedings of SIAM International Conference on Data Mining*, pages 11–22, Lake Buena Vista, Florida, USA.

[Remington et al., 2012] Remington, R., Folk, C. L., and Boehm-Davis, D. A. (2012). *Introduction to Humans in Engineered Systems*. John Wiley & Sons.

[Reynard et al., 1996] Reynard, D., Wildenberg, A., Blake, A., and Marchant, J. (1996). Learning dynamics of complex motions from image sequences. In Buxton, B. and Cipolla, R., editors, *Computer Vision ECCV '96*, number 1064 in Lecture Notes in Computer Science, pages 357–368. Springer Berlin Heidelberg.

[Ribeiro and Santos-Victor, 2005] Ribeiro, P. C. and Santos-Victor, J. (2005). Human activity recognition from video: modeling, feature selection and classification architecture. In *in International Workshop on Human Activity Recognition and Modeling (HAREM*.

[Rosenblatt, 1956] Rosenblatt, M. (1956). Remarks on some nonparametric estimates of a density function. *The Annals of Mathematical Statistics*, 27(3):832–837.

[Ryoo and Aggarwal, 2009] Ryoo, M. S. and Aggarwal, J. (2009). Spatio-temporal relationship match: Video structure comparison for

recognition of complex human activities. In *2009 IEEE 12th International Conference on Computer Vision*, pages 1593–1600.

[Sahaf et al., 2011] Sahaf, Y., Krishnan, N. C., and Cook, D. J. (2011). Defining the complexity of an activity. In *Workshops at the Twenty-Fifth AAAI Conference on Artificial Intelligence*.

[Salton, 1971] Salton, G. (1971). *The SMART Retrieval System - Experiments in Automatic Document Processing*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

[Salton and Lesk, 1968] Salton, G. and Lesk, M. E. (1968). Computer evaluation of indexing and text processing. *J. ACM*, 15(1):836.

[Salton and McGill, 1986] Salton, G. and McGill, M. J. (1986). *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA.

[Salton et al., 1975] Salton, G., Wong, A., and Yang, C. S. (1975). A vector space model for automatic indexing. *Commun. ACM*, 18(11):613620.

[Schooler et al., 1993] Schooler, J. W., Ohlsson, S., and Brooks, K. (1993). Thoughts beyond words: When language overshadows insight. *Journal of Experimental Psychology: General*, 122(2):166–183.

[Schuldt et al., 2004] Schuldt, C., Laptev, I., and Caputo, B. (2004). Recognizing human actions: a local SVM approach. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004*, volume 3, pages 32–36.

[Schwarzwald et al., 2004] Schwarzwald, J., Koslowsky, M., and Ochana-Levin, T. (2004). Usage of and compliance with power tactics in routine versus nonroutine work settings. *Journal of Business and Psychology*, 18(3):385–402.

[Shcherbakov, 2009] Shcherbakov, A. P. (2009). A two-camera based algorithm for 3D reconstruction of a visual scene. probabilistic approach. *Journal of Computer and Systems Sciences International*, 48(1):131–138.

[Shechtman and Irani, 2005] Shechtman, E. and Irani, M. (2005). Space-time behavior based correlation. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005. CVPR 2005*, volume 1, pages 405–412 vol. 1.

[Shi et al., 2006] Shi, Y., Bobick, A., and Essa, I. (2006). Learning temporal sequence model from partially labeled data. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 1631–1638, Los Alamitos, CA, USA. IEEE Computer Society.

[Shi et al., 2004a] Shi, Y., Huang, Y., Minnen, D., Bobick, A., and Essa, I. (2004a). Propagation networks for recognition of partially ordered sequential action. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 862–869.

[Shi et al., 2004b] Shi, Y., Huang, Y., Minnen, D., Bobick, A., and Essa, I. (2004b). Propagation networks for recognizing partially ordered sequential activity. `http://www.cc.gatech.edu/cpl/`

projects/monsoon/PropagationNet/PropagationNet.htm. [Online; accessed 19 Nov 2012].

[Sibson, 1973] Sibson, R. (1973). SLINK: an optimally efficient algorithm for the single-link cluster method. *The Computer Journal*, 16(1):30–34.

[Sisk et al., 2003] Sisk, G., Miles, J., and Moore, C. (2003). Designer centered development of GA-Based DSS for conceptual design of buildings. *Journal of Computing in Civil Engineering*, 17(3):159–166.

[Smeulders et al., 2000] Smeulders, A. W. M., Worring, M., Santini, S., Gupta, A., and Jain, R. (2000). Content-based image retrieval at the end of the early years. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(12):13491380.

[Spiegel and O'Donnell, 1997] Spiegel, E. and O'Donnell, C. J. (1997). *Incidence algebras*. Marcel Dekker Incorporated.

[Sridhar et al., 2008] Sridhar, M., Cohn, A. G., and Hogg, D. C. (2008). Learning functional Object-Categories from a relational Spatio-Temporal representation. In *Frontiers in Artificial Intelligence and Applications*, volume 178, pages 606–610. IOS Press.

[Sridhar et al., 2010] Sridhar, M., Cohn, A. G., and Hogg, D. C. (2010). Unsupervised learning of event classes from video. pages 1631–1638, Atlanta, Georgia, USA.

[Starner and Pentland, 1995] Starner, T. and Pentland, A. (1995). Real-time american sign language recognition from video using hidden markov models. In *, International Symposium on Computer Vision, 1995. Proceedings*, pages 265–270.

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

[Steinhaus, 1956] Steinhaus, H. (1956). Sur la division des corp materiels en parties. *Bulletin of the Polish Academy of Sciences*, 4(12):801–804.

[Torralba et al., 2007] Torralba, A., Murphy, K. P., and Freeman, W. T. (2007). Sharing visual features for multiclass and multiview object detection. *IEEE Transactions in Pattern Analysis and Machine Intelligence (PAMI)*, 29(5):854869.

[Troitsky, 2000] Troitsky, M. S. (2000). Conceptual bridge design. In Chen, W.-F. and Duan, L., editors, *Bridge Engineering Handbook*, pages 1–19. CRC Press LLC, Boca Raton, FL.

[Truyen et al., 2006] Truyen, T., Phung, D., Venkatesh, S., and Bui, H. (2006). AdaBoost.MRF: boosted markov random forests and application to multilevel activity recognition. volume 2, pages 1686–1693.

[Turaga et al., 2008] Turaga, P., Chellappa, R., Subrahmanian, V., and Udrea, O. (2008). Machine recognition of human activities: A survey. *Circuits and Systems for Video Technology, IEEE Transactions on*, 18(11):1473–1488.

[United Nations, 2010] United Nations (2010). *Human Development Report: The Real Wealth of Nations: Pathways to Human Development: 2010*. Palgrave Macmillan, London, UK.

[van der Corput, 1935a] van der Corput, J. G. (1935a). Verteilungsfunktionen I. *Proceedings Akademie van Wetenschappen*, 38:813–821.

[van der Corput, 1935b] van der Corput, J. G. (1935b). Verteilungsfunktionen II. *Proceedings Akademie van Wetenschappen*, 38:1058–1066.

[Vasconcelos and Lippman, 1998] Vasconcelos, N. and Lippman, A. (1998). Learning mixture hierarchies. In *In Proceedings of Advances in Neural Information Processing Systems (NIPS'98*, pages 606–612. MIT Press.

[Vaswani et al., 2005] Vaswani, N., Chowdhury, A. R., and Chellappa, R. (2005). "Shape activity": A continuous state HMM for Moving/Deforming shapes with application to abnormal activity detection. *IEEE Trans. on Image Processing*, 14(10):16031616.

[Vishwakarma and Agrawal, 2012] Vishwakarma, S. and Agrawal, A. (2012). A survey on activity recognition and behavior understanding in video surveillance. *The Visual Computer*, pages 1–27.

[Vogler and Metaxas, 1999] Vogler, C. and Metaxas, D. (1999). Parallel hidden markov models for american sign language recognition. In *The Proceedings of the Seventh IEEE International Conference on Computer Vision, 1999*, volume 1, pages 116–122.

[Voulodimos et al., 2012] Voulodimos, A., Kosmopoulos, D., Vasileiou, G., Sardis, E., Anagnostopoulos, V., Lalos, C., Doulamis, A., and Varvarigou, T. (2012). A threefold dataset for activity and workflow recognition in complex industrial environments. *IEEE MultiMedia*, 19(3):42–52.

[Wang et al., 2009] Wang, X., Ma, X., and Grimson, W. E. (2009). Unsupervised activity perception in crowded and complicated scenes using

hierarchical Bayesian models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(3):539–555.

[Wang et al., 2006] Wang, X., Tieu, K., and Grimson, E. (2006). Learning semantic scene models by trajectory analysis. In *Proceedings of the 9th European conference on Computer Vision - Volume Part III*, ECCV'06, page 110123, Berlin, Heidelberg. Springer-Verlag.

[Wong et al., 2007] Wong, K.-Y. K., Kim, T.-K., and Cipolla, R. (2007). Learning motion categories using both semantic and structural information. In *IEEE Conference on Computer Vision and Pattern Recognition, 2007. CVPR '07*, pages 1–6.

[Wood, 1986] Wood, R. E. (1986). Task complexity: Definition of the construct. *Organizational Behavior and Human Decision Processes*, 37(1):60–82.

[Wren and Pentland, 1998] Wren, C. and Pentland, A. (1998). Dynamic models of human motion. In *Third IEEE International Conference on Automatic Face and Gesture Recognition, 1998. Proceedings*, pages 22–27.

[Wyatt et al., 2005] Wyatt, D., Philipose, M., and Choudhury, T. (2005). Unsupervised activity recognition using automatically mined common sense. In *In AAAI*, page 2127.

[Xiang and Gong, 2006] Xiang, T. and Gong, S. (2006). Beyond tracking: Modelling activity and understanding behaviour. *International Journal of Computer Vision*, 67(1):21–51.

[Xie, 2005] Xie, L. (2005). *Unsupervised Pattern Discovery for Multi-*

media Sequences. PhD thesis, Columbia University, New York, NY, USA.

[Xie et al., 2002] Xie, L., Chang, S., Divakaran, A., and Sun, H. (2002). Structure analysis of soccer video with hidden markov models. Orlando, FL, USA.

[Xie et al., 2008] Xie, L., Sundaram, H., and Campbell, M. (2008). Event mining in multimedia streams. *Proceedings of the IEEE*, 96(4):623–647.

[Xie et al., 2004] Xie, L., Xu, P., Chang, S., Divakaran, A., and Sun, H. (2004). Structure analysis of soccer video with domain knowledge and hidden markov models. *Pattern Recognition Letters*, 25(7):767–775.

[Yamato et al., 1992] Yamato, J., Ohya, J., and Ishii, K. (1992). Recognizing human action in time-sequential images using hidden markov model. In *1992 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1992. Proceedings CVPR '92*, pages 379–385.

[Yao and Fei-Fei, 2010] Yao, B. and Fei-Fei, L. (2010). Grouplet: A structured image representation for recognizing human and object interactions. In *2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9–16.

[Yeh et al., 2012] Yeh, C.-C., Lin, F., and Hsu, C.-Y. (2012). A hybrid KMV model, random forests and rough set theory approach for credit rating. *Knowledge-Based Systems*, 33:166–172.

[Yu et al., 2010] Yu, M., Naqvi, S., and Chambers, J. (2010). A robust fall detection system for the elderly in a smart room. In *2010 IEEE*

International Conference on Acoustics Speech and Signal Processing (ICASSP), pages 1666 –1669.

[Zelnik-Manor and Irani, 2001] Zelnik-Manor, L. and Irani, M. (2001). Event-based analysis of video. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2001. CVPR 2001*, volume 2, pages II–123–II–130 vol.2.

[Zhang et al., 2008] Zhang, Z., Huang, K., and Tan, T. (2008). Multi-thread parsing for recognizing complex events in videos. In Forsyth, D., Torr, P., and Zisserman, A., editors, *Computer Vision ECCV 2008*, number 5304 in Lecture Notes in Computer Science, pages 738–751. Springer Berlin Heidelberg.

[Zhang et al., 2011] Zhang, Z., Tan, T., and Huang, K. (2011). An extended grammar system for learning and recognizing complex visual events. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(2):240–255.