

# Activity Monitoring: Continuous Recognition and Performance Evaluation

A dissertation submitted to the  
SWISS FEDERAL INSTITUTE OF TECHNOLOGY  
ZURICH  
for the degree of  
Doctor of Sciences

presented by  
Jamie A. Ward  
BEng CS&E (hons.) Edinburgh  
born 24th March 1979  
citizen of United Kingdom

accepted on the recommendation of  
Prof. Dr. Gerhard Tröster, examiner  
Prof. Dr. Hans W. Gellersen and Prof. Dr. Paul Lukowicz, co-examiner

© Jamie A. Ward 2006

Activity Monitoring: Continuous Recognition and Performance Evaluation

First edition 2006

Published by the Swiss Federal Institute of Technology (ETH), Zürich,  
Switzerland ( Diss. No. 16520)

ISBN 3-909386-63-6

Printed by Lulu.com

Copies may be ordered online from <http://www.lulu.com>

# Acknowledgements

This work would not have been possible without the facilities and support offered by my adviser Prof. Gerhard Tröster and the ETH Wearable Computing Group which he heads. I thank him for the opportunity he has given me to work at ETH in such an interesting research field. I would also like to offer warm thanks to Paul Lukowicz. His presence in and around the Lab through the years has been an invaluable source of both support and inspiration.

I wish to thank all those other colleagues at ETH who, either through direct assistance or in discussion, have helped with this work in some way. Oliver Amft, Nagendra Bhargava, Thomas Stiefmeier, Holger Junker, Mathias Stäger, Urs Anliker, Tünde Kirstein, Patrick de la Hamette, Stijn Ossevoort and Fabrizio Macaluso all deserve credit. As do my test subjects Thomas von Büren, Roberto Barbieri, David Bannach and Veronica Housen (not forgetting of course - though I wasn't aware of him at the time - Veronica's little boy Teo).

Special thanks go to the most important person at ETH: Ruth Zähringer, our institute secretary. On that note, a special thanks to Mrs N. Banergee of Partho Stores, without whose daily portions of quality Indian food I would surely have starved.

Outside ETH, I thank Thad Starner (whose encouragement over the years has been most supportive) and Amin Atrash of Georgia Tech., and Kai Kunze and Georg Ogris of UMIT, Austria.

Most importantly, I thank my family - my mum and dad, gran and pappa - for all the love and support they have given me over the years. Marina, too, deserves a thankyou for putting up with me over the last few months of thesis write-up (and Michaela, my flatmate, for the four chaotic years of 'work-in-progress').

I dedicate this work to the memory of my late grandfather Thomas Ward (senior). It was under his guidance, almost twenty years ago, that I first picked up a soldering iron and began tinkering with bits of wire and transistors. Somewhere along the line that tinkering got me onto computers. Where it'll get me next I do not know.

Jamie A. Ward

Zürich, July 2006



# Contents

<b>Abstract</b>	<b>ix</b>
<b>Zusammenfassung</b>	<b>x</b>
<b>1. Introduction</b>	<b>1</b>
1.1. The need for activity monitoring . . . . .	2
1.1.1. A wearable solution . . . . .	2
1.2. Aims of the thesis . . . . .	3
1.2.1. Continuous activity recognition . . . . .	3
1.2.2. Performance evaluation (and optimisation) . . . . .	4
1.3. Related work . . . . .	5
1.4. Outline of the thesis . . . . .	9
<b>2. Choice of test scenario</b>	<b>11</b>
2.1. Problem analysis . . . . .	12
2.2. Experimental setup . . . . .	12
2.2.1. Procedure . . . . .	14
2.2.2. Data collection system . . . . .	15
2.3. Analysis of the data . . . . .	17
2.3.1. Acceleration analysis . . . . .	18
2.3.2. Sound analysis . . . . .	22
2.4. Conclusion . . . . .	24
<b>3. Activity recognition using sound and acceleration</b>	<b>25</b>
3.1. Introduction . . . . .	26
3.2. Recognition method . . . . .	27
3.2.1. Recognition using sound . . . . .	27
3.2.2. Recognition using acceleration . . . . .	29
3.3. Experiment . . . . .	34
3.3.1. Setting parameters . . . . .	34
3.3.2. Evaluation method . . . . .	34
3.4. Isolation results . . . . .	35
3.5. Conclusion . . . . .	35
<b>4. Segmentation using two microphones</b>	<b>37</b>
4.1. The segmentation problem . . . . .	38
4.2. Two microphone sound segmentation . . . . .	38
4.3. Experiment . . . . .	40

4.4.	Results . . . . .	41
4.4.1.	Analysis of results . . . . .	42
4.5.	Conclusion . . . . .	45
<b>5.</b>	<b>Continuous recognition using classifier fusion</b>	<b>47</b>
5.1.	Introduction . . . . .	48
5.2.	Recognition method . . . . .	49
5.2.1.	Frame-by-frame sound classification using LDA . . . . .	49
5.2.2.	Sound-based segmentation . . . . .	50
5.2.3.	Sound classification . . . . .	51
5.2.4.	Acceleration recognition . . . . .	52
5.2.5.	Comparison of top choices (COMP) . . . . .	52
5.2.6.	Fusion using class rankings . . . . .	53
5.3.	Experimental setup . . . . .	55
5.3.1.	Training and testing . . . . .	55
5.4.	Recognition in isolation . . . . .	56
5.5.	Continuous recognition . . . . .	58
5.5.1.	Segmentation evaluation method . . . . .	59
5.5.2.	Segmentation results . . . . .	60
5.5.3.	Segmentation and classification results . . . . .	60
5.5.4.	Frame-by-frame results . . . . .	63
5.5.5.	Analysis of frame-by-frame results . . . . .	67
5.5.6.	Event results . . . . .	67
5.5.7.	Analysis of event results . . . . .	68
5.5.8.	Combined time and event evaluation . . . . .	68
5.5.9.	Analysis of combined time and event evaluation . . . . .	70
5.6.	Lessons learnt . . . . .	73
5.6.1.	Concluding remarks . . . . .	74
<b>6.</b>	<b>Recognition using wrist-worn sensors</b>	<b>75</b>
6.1.	Introduction . . . . .	76
6.2.	Recognition methods . . . . .	76
6.2.1.	Sliding window segmentation . . . . .	77
6.2.2.	Sound recognition using LDA . . . . .	77
6.2.3.	Acceleration recognition . . . . .	77
6.2.4.	Fusion of classifiers: COMP and LR . . . . .	78
6.3.	Experiment . . . . .	79
6.3.1.	Setting parameters . . . . .	79
6.4.	Results . . . . .	80
6.4.1.	Preliminary study (using 40Hz LDA and HMM) . . . . .	80
6.4.2.	Frame-by-frame results (using 40Hz LDA and HMM) . . . . .	80
6.4.3.	Selecting a lower frequency LDA . . . . .	82
6.4.4.	Frame-by-frame results (using 2Hz LDA and NB) . . . . .	83

6.4.5.	Analysis of computation times . . . . .	85
6.5.	Discussion . . . . .	85
6.5.1.	COMP versus. LR . . . . .	88
6.5.2.	Conclusion . . . . .	89
<b>7.</b>	<b>Evaluation and optimization of performance</b>	<b>91</b>
7.1.	Introduction . . . . .	92
7.1.1.	Chapter contributions and organisation . . . . .	92
7.2.	Motivation . . . . .	93
7.2.1.	Frame based analysis . . . . .	93
7.2.2.	Event analysis . . . . .	95
7.3.	Problem specification and existing methods . . . . .	95
7.3.1.	Definition of performance evaluation . . . . .	96
7.3.2.	General considerations . . . . .	97
7.3.3.	Requirements for activity recognition . . . . .	97
7.4.	Error characterisation and representation . . . . .	99
7.4.1.	Event analysis . . . . .	100
7.4.2.	Segment analysis . . . . .	102
7.5.	Discussion . . . . .	107
7.5.1.	Application of method to worked example . . . . .	107
7.5.2.	Significance and Limitations . . . . .	110
7.5.3.	Work in related fields . . . . .	111
7.6.	Conclusion . . . . .	112
<b>8.</b>	<b>SET optimization for continuous activity recognition</b>	<b>113</b>
8.1.	SET analysis of the wood workshop . . . . .	114
8.1.1.	(Re-)analysis of results . . . . .	115
8.2.	Parameter optimisation with SET . . . . .	118
8.3.	Conclusion . . . . .	120
<b>9.</b>	<b>Conclusion</b>	<b>121</b>
9.1.	Continuous activity recognition . . . . .	122
9.2.	Performance evaluation (and optimisation) . . . . .	125
	<b>Glossary</b>	<b>127</b>
	<b>Bibliography</b>	<b>131</b>
	<b>Curriculum Vitae</b>	<b>141</b>



# Abstract

Wearable computers promise the ability to access information and computing resources directly from miniature devices embedded in our clothing. The problem lies in how to access the most relevant information without disrupting whatever task it is we are doing. Most existing interfaces, such as keyboards and touch pads, require direct interaction. This is both a physical and cognitive distraction. The problem is particularly acute for the mobile maintenance worker who must access information, such as on-line manuals or schematics, quickly and with minimal distraction.

One solution is a wearable computer that monitors the user's 'context' - information such as activity, location and environment. Being 'context aware', the wearable would be better placed to offer relevant information to the user as and when it is needed.

In this work we focus on recognising one of the most important parts of context: user activity. The contributions of the thesis are twofold. First, we present a method for recognising hand activities from a sequence using body worn sensors. Second, in evaluating this method, we present a generalised strategy for characterising the performance of activity recognition systems.

We define a set of typical hand and tool activities in a woodwork assembly scenario. We evaluate two methods for detecting and recognising these activities using a combination of body-worn microphones and accelerometers. The first method uses two separately placed microphones on the user's arm to locate the source of an activity. Whenever a sound is made close to the wrist, an interesting activity is assumed and classification is carried out using both acceleration and sound. The second method requires only wrist-worn sensors. It recognises activities by classifying sound and acceleration over a sliding window. The classifications from each of the sensor types are then compared, and a final result is given depending on how well they agree.

In the second part of the thesis we introduce a strategy for evaluating the performance of continuous activity recognition systems. Like any area of scientific research, activity recognition requires standard methods and measures of performance. These are the tools with which researchers can compare and evaluate different systems, thus allowing the field to advance. Continuous activity recognition, however, has a number of performance issues which existing evaluation strategies - borrowed from related fields such as speech recognition - fail to capture. We explore these issues in depth and propose a new strategy of performance evaluation based on the complete characterisation of error types common to the activity recognition problem.

Finally, we bring the two main topics of the thesis together. Using the results from our continuous recognition work we show the improvements of the proposed performance evaluation strategy over existing approaches.

# Zusammenfassung

'Wearable Systems' - am Körper tragbare Systeme - zeigen einen Weg auf, die Nützlichkeit von Computern weit über die vertraute Rolle als Hilfsmittel in Büros hinaus zu erhöhen. Gegenwärtige Technologien wie PDAs (Personal Digital Assistants) ermöglichen beispielsweise mobilen Technikern den Zugang zu einem grossen Spektrum an Informationen über ihre zu verrichtende Tätigkeit. Ein entscheidender Nachteil dieser Technologie ist allerdings, dass keine adäquate Schnittstelle für die Interaktion zwischen Techniker und PDA existiert, die nicht von der eigentlichen Tätigkeit ablenkt. Das Gebiet 'Context Awareness' versucht dieses Problem zu lösen, indem Computer in die Lage versetzt werden, Informationen über den Nutzer und dessen Status in eine automatische, proaktive Interaktion zu überführen. Ein wichtiges Teilgebiet dieser Bestrebung ist das 'Activity Monitoring' - die Aktivitätserkennung. Im Vergleich zu benachbarten Gebieten, wie der Spracherkennung, in der es neben Wörtern Unterteilungen wie Silben und Phoneme gibt, existiert auf dem Gebiet der menschlichen Aktivitätserkennung keine solche ausgeprägte Taxonomie. Dies gilt im Besonderen für Aktivitäten der Hand. Sie sind in der Regel sehr facettenreich bezüglich ihrer Ausführung, haben eine variable Ausführungsdauer und sind oft durchsetzt mit anderen irrelevanten Aktivitäten.

Der Beitrag der vorliegenden Arbeit ist zweigeteilt. Zunächst präsentieren wir eine Methode zur Erkennung von kontinuierlichen Sequenzen von Handaktivitäten unter Verwendung von am Körper getragenen Sensoren. Weiterhin beschreiben wir eine generalisierbare Strategie, um die Leistung von Aktivitätserkennungssystemen zu charakterisieren.

Für den ersten Teil der Arbeit definieren wir eine Untermenge von typischen Handaktivitäten, die sich während der Arbeit in einer Werkstatt bei der Holz- bzw. Metallverarbeitung ergeben. Diese Aktivitäten umfassen, wie auch andere Interaktionen mit der Hand, sowohl Bewegungs- als auch Audiocharakteristika. Daher stützt sich unsere Erkennungsmethode auf Daten, die mit Mikrofonen und Beschleunigungssensoren aufgenommen werden. Wir präsentieren zwei Methoden, um für uns relevante Aktivitäten in einem kontinuierlichen Datenstrom zu erkennen. Die erste Methode basiert auf zwei Mikrofonen, die am Oberarm und am Handgelenk des Benutzers bestigit sind, um die akustische Quelle der jeweiligen Aktivität zu lokalisieren. Wenn aktivitätsbezogene Geräusche in der Nähe des Handgelenks erkannt werden, so wird die Aktivitätserkennung auf Beschleunigungs- und Audiodaten gestartet. Eine zweite Methode der Aktivitätserkennung benutzt eine einfache Fensterfunktion auf den Daten. Die Klassifizierungen beider Sensordomänen wird innerhalb eines Fensters verglichen und bei entsprechend wahrscheinlicher Übereinstimmung wird das entsprechende Ergebnis berechnet. Im Falle einer

zu niedriger Übereinstimmung ergibt sich die Klassifizierung NULL bzw. keine Aktivität wird als erkannt zurückgegeben. Die letztere Methode hat gegenüber der ersten den Vorteil, dass sie nur Sensoren am Handgelenk benötigt und damit mehr Komfort während des Tragens für den Nutzer bietet.

Im zweiten Teil der vorliegenden Arbeit führen wir eine Strategie zur Bewertung der Leistung von Aktivitätserkennungssystemen ein. Wie ein jedes Gebiet der wissenschaftlichen Forschung, so braucht auch die Aktivitätserkennung standardisierte Methoden und Masse für deren Leistungsfähigkeit. Dadurch erst werden Forscher in die Lage versetzt, Systeme zu vergleichen und diese zu optimieren, um die Arbeit auf dem jeweiligen Gebiet insgesamt voranzutreiben. Kontinuierliche Aktivitätserkennung allerdings weist einige Defizite diesbezüglich auf. Existierende Bewertungsansätze - zumeist an solche in benachbarten Gebieten wie der Spracherkennung angelehnt - sind nicht mächtig genug. Wir untersuchen die zugrundeliegenden Schwierigkeiten tiefgründig und stellen eine neue Strategie der Leistungsbewertung vor. Diese umfasst eine solide Charakterisierung der Fehlertypen, die bei Aktivitätserkennungssystemen geläufig sind.

Zum Abschluss bringen wir die beiden Hauptteile der Arbeit zusammen. Wir bewerten unseren Ansatz zur kontinuierlichen Aktivitätserkennung und zeigen wie die vorgestellte Bewertungsstrategie ein tieferes Verständnis der Leistungsfähigkeit des Erkennungssystems als alle existierenden Methoden bietet.



# 1

## Introduction

*This chapter provides a brief motivation on the need for activity monitoring, specifically as part of a context aware wearable system for use in assembly and maintenance tasks. The two areas covered in this thesis, namely the recognition of activities using on-body microphones and accelerometers, and the general problem of evaluating performance in activity recognition, are introduced together with the main aims of the thesis. Finally, a brief review of the literature relevant to this work is presented.*

## 1.1. The need for activity monitoring

For the office worker, computers have become the primary tool for information access. The on-site maintenance worker, however, has greater difficulty in accessing this resource. Manuals, schematics, system status, and updated instructions may be readily available on-line via wireless networks. This information can be accessed using a laptop, or a Personal Digital Assistants (PDA), but in order to do so, the worker must shift attention away from the task at hand. For example, to access a specific schematic through a PDA, an aircraft repair technician needs to interrupt his work, retrieve his PDA from a pocket or bag, navigate the PDA's interface, read the desired information, and finally stow it away again before resuming work. Equipping the worker with a head-up display, speech input, or one-handed keyboard, helps reduce distraction from the physical task. However, to navigate the interface and find the required information, a potentially distracting amount of cognitive effort is still required.

### 1.1.1. A wearable solution

Wearable computing researchers have come up with body mounted computers and novel user interfaces to address the physical distractions involved in this problem. From the cognitive side, pro-active systems that automatically retrieve the right information at the right time have been proposed [1]. The core of such systems is the ability to follow the progress of the task and automatically recognise *what* is being done and *when*. Thus, a manual on performing the removal of a turbine blade from an engine can be automatically brought up to the head mounted display of an aircraft maintenance worker as soon as he begins with this specific part of the repair procedure.

The key to the success of this is the monitoring of specific activities as they happen. Three main approaches for this have been investigated:

1. Augmenting the environment (e.g the aircraft engine) with sensors which can tell the system which part is being accessed by the user and what is happening to it. This has the advantage of being computationally inexpensive, and potentially very reliable. For extensive and detailed recognition however, such a system would require a large number of parts to be augmented, something which is either infeasible or too expensive. Recently, however, this approach has seen growing interest with the advent of Radio Frequency Identification (RFID) tags. These are cheap to manufacture, and have been already widely deployed as product identification tags by some well known high street stores.
2. Video analysis. This is the classical AI approach which attempts to imitate the way humans perceive the world (which is mostly visual). Of

the three approaches a video system with appropriately placed cameras provides the most information, and in principle could facilitate the most detailed analysis. However, the signal processing involved in extracting this information can be computationally intensive and, in the general case, remains an open problem. This is particularly true for setups with varying, dynamic lighting conditions and occlusions.

3. Body worn sensors to monitor user activity. This is a relative new approach that has emerged as a key research focus in wearable computing. The most popular variant relies on motion sensors (accelerometers and gyroscopes) attached to relevant parts of the body, in particular hands and arms, to provide information about user motions. Other sensors which are commonly used include microphones, GPS (Global Positioning System), and light sensors. These provide additional information about the user's environment or location which might help with the activity recognition task. By augmenting the user rather than the environment, the cost of augmentation can often be reduced (there are more engines than maintenance workers). When a small number of simple sensors are used, the signal processing and extraction of information is potentially less complex than in the case of vision. (This is not necessarily so when a large number of sensors, with high data rates, are used.)

The choice of which approach to use depends on the specific application and its associated requirements. In many cases a combination of two, or even all three, of the above might be appropriate. In this work we focus on the use of body worn sensors. Specifically, we look at the combination of accelerometers and microphones mounted on the wrist and upper arm.

## 1.2. Aims of the thesis

We deal with the problem of developing an activity monitoring system using body-worn sensors. Despite the growing research interest in this area, many problems remain unsolved. This is particularly true for the continuous, real-world recognition case. The first problem this thesis deals with is how to recognise useful activities from a continuous sequence of mostly unwanted motions. The second problem, once we have a solution to the recognition problem, is how to tell how good this solution is - how do we measure its performance.

### 1.2.1. Continuous activity recognition

Given a continuous sequence of real-world activities, how can we recognise which activity is being carried out at any given time? The general case -

recognising all possible activities - is a clearly intractable task: there are just too many possibilities. If we constrain the problem space, however, the problem becomes more manageable.

In this work we restrict the activities of interest to those performed in an assembly scenario - specifically a wood workshop. These all involve the interaction of the hand with some tool: a hammer, a saw, a drill, etc. Such activities usually involve characteristic motion sequences and have distinct sounds. We therefore base our study on microphones and accelerometers mounted on two locations: the wrist and upper arm.

The recognition problem is then dealt with in two steps: (1) segmentation of the interesting activities from a continuous sequence (when do they occur, and for how long?); and (2) classification of the selected segments (which activities have been detected?).

The main questions on this topic that are addressed in this thesis are:

1. How do we detect interesting activities from a stream of (mostly) uninteresting activities using microphones and accelerometers?
2. How accurately can we recognise specific activities using these sensors - is performance improved by combining them?
3. If the recognition system is to be used in a working environment, the wearability of its implementation becomes an important issue. It must be small and unobtrusive. A single body-worn device would meet these requirements - similar, for example, to a wristwatch. But would the limitations of such a device allow adequate recognition performance?

### 1.2.2. Performance evaluation (and optimisation)

Once we have developed a potential solution to the recognition problem, how do we then evaluate how good it is? If we have several solutions, how do we compare them? These questions are important not only for helping developers fine-tune or optimise their systems to a given application, but also to allow different researchers to compare and contrast their approaches - a necessity, it might be argued, if the field is to mature. To-date, no standard method exists for the specific problem of activity recognition - nor indeed for the more general topic of context recognition.

There are three main requirements for any performance evaluation strategy. The first is the establishment of open datasets to be used as benchmarks.<sup>1</sup> The second is in finding appropriate measure(s) for capturing the most critical information about the problem domain. And the third, perhaps most important requirement, is the acceptance of both of these as standard by the research community.

---

<sup>1</sup>For example, see the dedicated workshop on this topic at Pervasive '04 [2].

This thesis deals with the the second requirement. Specifically, we address the following questions:

1. What is the most relevant performance information for continuous activity recognition?
2. If existing measures fail to capture this information, is there scope for a new proposal specific to the problem of activity recognition?

### 1.3. Related work

Many wearable systems explore context awareness and pro-active involvement as means of reducing the cognitive load on the user, an early example this being the work of Abowd *et al.* [3].

**Vision** Much work has been devoted to motion and activity recognition using computer vision. For an extensive survey on the visual analysis of human motion, see Moeslund and Granum [4]. Full body recognition has been studied by Kale *et al.* [5], by Aggarwal and Ali [6], and by Rittscher and Blake [7]. Torralba *et al* [8], among others, have used wearable vision to recognise location and objects. The thesis of Clarkson [9] investigated the long term use of wearable vision to track user location and situation. A topic which has been extensively studied in the wearable domain is that of gesture and hand recognition. Over the years many works have appeared in this direction, for example Starner [10], Vogler [11], Wilson and Bobick [12], Schlenzig [13] and Rehg [14]. Though powerful vision can suffer in the mobile and wearable domains from drawbacks such as occlusion and changes in lighting conditions as users move around. For many recognition tasks the computation complexity - and the associated power consumption - is often beyond what current wearable hardware can support.

**Body-fixed accelerometers** Non-visual, body fixed sensors (BFS), in particular accelerometers, have been employed for many years in the analysis of body posture and activity, such as in the work of Bussmann *et al.* [15]. The primary area of application has been ambulatory monitoring in a clinical setting, collecting data to assist in assessment of patients with a wide range of conditions, such as Parkinson's disease, or injury rehabilitation (for example, see Bonato [16], or Salarian *et al* [17]). Several works have been proposed to monitor the walking patterns of elderly people (Najafi *et al.* [18], or Sekine *et al.* [19]) . They have also been used to study working environments. Uiterwaal *et al.* [20] performed a feasibility study on the long term monitoring of ambulatory activities in maintenance and messenger work.

Using two uniaxial accelerometers - one radial at the chest, the other tangential at the thigh - Veltink *et al.* [21] were able to evaluate the feasibility of distinguishing postures, such as standing, sitting and lying; they also attempted to distinguish these from the dynamic activities of walking, using stairs and cycling. Developments on this, all with the goal of ambulatory activity recognition, have since been carried out by Aminian *et al.* [22], and more recently by Wetzler *et al.* [23]. For an overview on the use of different BFSs (particularly for clinical applications) see the report by Aminian and Najafi [24].

In the wearable domain recognition of ambulatory activities has been dealt with by a number of researchers. This is usually part of an attempt at recognising user context (for example, Mantyjarvi *et al.* [25], Randell and Muller [26], Van-Laerhoven and Cakmakci [27]).

Vildjiounaite *et al.* [28] used a combination of accelerometers, magnetic sensors and maps to detect user location and walking behaviour inside a building.

Recently, Westeyn *et al.* [29] used wearable accelerometers to spot certain behavioural activities in autistic children, and Bao and Intille [30] to recognise multiple full-body activities. Chambers *et al.* [31] investigated the recognition of certain Kung Fu moves by augmenting visual recordings with wrist-worn accelerometer data.

Of more intricate hand activities, such as interaction with objects, or gesticulation, there have been several works using accelerometers - generally involving sensors either on the objects being manipulated, as presented by Antifakos *et al.* [32], or embedded in special gloves, as shown by Fang *et al.* [33].

In the thesis of Junker [34], the topic of continuous activity recognition using body-worn accelerometers is dealt with in depth. Of particular interest in this work is the introduction of a method for segmenting interesting activities (specifically, closed motions with a clear start and stop points) from a continuous stream.

**Sound and sensor combination** Pelton *et al.* [35] investigated the use of sound for analysing situations, such as detecting which location the user is in - bedroom, street, church, etc. Büchler [36] presented a method of using sound analysis to improve the performance of hearing aids. Recently, Scott *et al.* [37] used sound for fine grained location detection within a building. And in [38], Chen *et al.* used sound - rather distastefully for some - to detect activities in a bathroom.

Clarkson and Pentland used a combination of audio and video to infer situation based on short-term events (such as opening and closing doors) [39].

This combination has also been used by Chibelushi *et al.* for people recognition [40].

Using body-worn accelerometers to estimate user posture and sound for speaker detection, Kern *et al.* [41] was able to build a system that helped annotate video recordings in a meeting scenario. In a more general setting, Kern [42] went on to annotate long term recordings of daily life. Physical activity ‘cues’ were provided by acceleration, social environment ‘cues’ by sound, and location information using GPS. The focus of this work, however, was to infer information about a user’s interuptability rather than specific recognition of activities.

Wu and Siegel [43] used a combination of accelerometers and microphones to provide complementary information about defects in material surfaces.

**Other approaches** Van Laerhoven and Gellersen [44] presented a cost-effective and low-power alternative to accelerometers that used multiple tilt-switches. This setup was used in [45] for observing and logging human activity. From the same group, Schmidt *et al.* [46] used sensors built into surfaces - such as table tops - to detect loads placed on them. This was part of a general work to augment the user’s environment with ‘smart’ devices [47].

As already noted RFID tags, as they shrink in size and cost, are growing in popularity. One instance of their use in the wearables domain is that by Philipose *et al.* [48]. In this work an RFID reader is integrated into a glove, allowing context to be inferred from the interactions the user has with different tagged objects.

**Fusion** Fusion of multiple information sources is a well studied and diverse field covering many different disciplines. Within the domain of activity recognition, fusion of multiple sensors stems largely from the intuitive notion that two well placed sensors say more about an activity than merely one alone. Just as sensors can be anything from real hardware, such as the microphones and accelerometers used in this work, to virtual sensors existing only as an abstraction in software. So too ‘fusion’ can be anything from the concatenation of different feature vectors from a single sensor, to some probabilistic inference from a diverse set of independent sensors.

Two commonly used approaches for fusing the data from two or more sensors are ‘feature fusion’ and ‘classifier fusion’.

**Feature fusion** This is where data from different sensors is first combined and then fed into a single classifier. For simple recognition problems using sensors with similar feature sets, a naive Bayes classifier often gives the best performance (for example, see Rish [49] or Domingos and Pazzani [50]). For more complex problems with diverse sensors and feature sets, stochastic techniques

like Hidden Markov Models (HMM), a specific type of Dynamic Bayesian Network (DBN), can be used (Rabiner [51] gives the ‘classic’ text on HMMs, but for a more recent overview of DBNs, see Murphy [52]). Variants on these, such as Coupled HMMs, allow for a greater modeling power for different features. These have been used to combine audio and video for speech recognition by Nefian *et al.* [53, 54] and by Potamianos *et al.* [55].

**Classifier fusion** This approach involves fusing the results from several, independent classifiers. Often classifiers perform best when they are specialised for a specific subset of classes. The approach taken in classifier fusion is to gather the results of several such ‘specialist’ classifiers and pool their resources to provide a more accurate overall result (see Kittler *et al.* [56] or Xu *et al.* [57]). There are three basic approaches to classifier fusion: ‘hard’ fusion using comparison of top classifier results, ‘soft’ fusion using class likelihood estimates, and, somewhere in between, fusion using class rankings. Each of these are discussed below.

The simplest classifier combination is to compare the top decisions of each classifier, throwing out any results where there is disagreement. The problem with this is that it disregards any specialisation that one classifier might have over another.

The second approach at classifier fusion is to make use of the class probabilities. Assuming the classifiers are able to provide a continuous output, such as likelihood values, or class distances, the fusion step will have a far richer source of information with which to make an overall decision. Variants of HMMs can be used to do this (again, see Murphy [52]). One stochastic approach which is particularly suited to this problem is Dempster-Shafer (DS) combination. DS has been used in a variety of work, for example, by Denoeux [58], Le Hgarat-Masclé [59]. Of particular interest is the use of DS for sensor fusion in the thesis of Huadong Wu [60] and by Wu and Siegel in [61, 62]. A good overview is also provided by Sentz and Ferson [63]. One drawback of these methods is the high computation cost. It can also be difficult to find sufficient data for training.

The third approach at classifier fusion is to combine the classifier rankings. Rather than incorporate the specific class probability or likelihood values from each classifier, this approach just uses the rankings of these values. An advantage of this is the reduced computation required. Several alternative methods for doing this were explored in the works of Ho [64, 65].

In this work we use the third approach - fusion of classifier rankings. We extend the methods covered by Ho [64] and apply them to the specific problem of fusing sound and acceleration classifiers.

**Performance evaluation** While working on different recognition problems and looking at related publications we have found that existing evaluation measures, mostly taken from related fields such as automatic speech recognition (ASR), information retrieval (IR), and vision related fields such as optical character recognition (OCR), often fail to adequately reflect the specific problems of the context recognition task, in particular with the non segmented, continuous case - e.g. continuous recognition of activities.

Even within some of the well-established domains there is a lack of consensus on standard methods of evaluating performance. Often the methods used tend to vary from researcher to researcher. In the vision domain, for example, Hoover *et al.* [66] commented that “the comparative framework is itself a research issue,” a notion echoed by Müller in 1999 [67], and again by Zhang in 2001[68]. In ASR the problem of finding a proper scientific evaluation has been mulled over since Pierce’s 1969 paper “Wither Speech Recognition” [69], through Moore in 1977[70], to Greenberg in 2001[71]. In the general topic of time series data mining, Keogh & Kasetty[72] make similar criticisms. Today, researchers in these fields continue to develop new or adapt existing measures to suit the particular problems which they face. It is true that no single evaluation measure can capture all of the different facets relevant to each domain, but generally the combined use of several well-chosen standard measures is desirable - and is recommended by nearly all of the above commentaries.

This has prompted us to investigate the particular problems of evaluation measures in continuous context recognition, and to propose an alternative strategy for evaluation which combines the strengths of several existing methods without throwing away critical information relevant to the context problem.

## 1.4. Outline of the thesis

The thesis is structured into two main parts: Chapters 2 to 6 deal with the continuous activity recognition problem; and Chapter 7 and 8 with performance evaluation. The outline of each chapter is as follows:

**Chapter 2** introduces the testbench scenario, the use of tools in a wood workshop, upon which the remainder of the work presented in this thesis is based. The scenario - and the method of using body-worn microphones and accelerometers to monitor it - is justified by analysing the different activities, and highlighting their diversity as representative of more general activities involving interaction of the hand with tools.

**Chapter 3** establishes the feasibility of the proposed microphone and accelerometer based recognition. The activities are recognised in isolation

using LDA-based classification on the sound, and HMM classification on features extracted from the accelerometers.

**Chapter 4** presents a method for segmenting activities from a continuous sequence using two separately placed microphones. We investigate the use of this method for different types of activity, and for different test subjects.

**Chapter 5** takes the ideas of the previous chapters and brings them together in a multi-subject analysis of the continuous activity recognition problem. This uses a variant of the sound based segmentation introduced in Chapter 4. A key contribution of this chapter is the use of fusion methods to combine the sound and acceleration classifiers for improved performance.

**Chapter 6** wraps up the topic of continuous recognition by adapting the methods from Chapter 5 to show how they might be used on a single wrist-worn device.

**Chapter 7** analyses the problem of evaluating performance in continuous activity recognition. The key contribution of this chapter is in the development of a more thorough method for assessing performance in continuous activity recognition systems. We identify four categories of error - *fragmenting*, *merge*, *overflow* and *underfill* - which we argue are important for activity recognition tasks, but which existing methods fail to capture. We introduce a method of performance characterisation based on the notion of *segments*, and show how this can be tabulated using the so-called *Segment Error Table (SET)*.

**Chapter 8** brings the two main parts of the thesis together. We apply the evaluation strategy of Chapter 7 to the earlier results of chapters 5 and 6. We discuss the significance of these findings, and show how the evaluation strategy can also be used for system optimisation.

**Chapter 9** concludes the thesis with a review of the findings presented. We discuss the limitations of the work and its relevance to other fields, as well as giving suggestions for future work.

# 2

## Choice of test scenario

*This chapter investigates the use of sound and acceleration for the purpose of recognising activities in an assembly scenario. The specific scenario used is that of a user performing carpentry tasks in a wood workshop. We introduce a multi-subject database of the scenario, recorded using body-worn microphones and 3-axis accelerometers, from which the algorithms introduced in the later chapters of this thesis can be evaluated. This scenario, though limited to a specific application, provides a small but diverse collection of activities with the potential to provide more general insight into recognition tasks involving the hand - specifically when interacting with some tool.*

## 2.1. Problem analysis

We wish to explore the use of on-body sensors to recognise a user’s activities. To ground our work we have chosen to examine the activities performed during an assembly task in a wood workshop. Specifically, this involves the use of five different hand tools (hammer, saw, sanding paper, file and screwdriver), the use of two machine tools (grinder and drill), and the use of fixed bench apparatus (vise and drawer).

These activities, though limited here to a specific scenario, are fairly diverse. In some respects they can be said to provide insight into a wide range of activities using the hand and some object or tool. Common to many activities they produce both a variety of different sound signatures and a range of characteristic motions. Hammering, for example, is characterised by the rise and fall of the arm, accompanied on impact by a loud bang. Use of the saw produces a more regular sound, directly correlated with the back and forth movements of the arm. The hand twists associated with using a screwdriver are generally accompanied by correlated, but much quieter sounds. In contrast, the use of a drilling machine produces a loud, continuous sound regardless of what the user does with his or her arm. The arm motion associated with drill use is itself well-defined, but is more or less independent from the sound being made. Even more extreme, the opening and closing of a drawer produces a characteristic sound, while its corresponding motion can vary from a well-defined push, to a simple nudge of the elbow or leg.

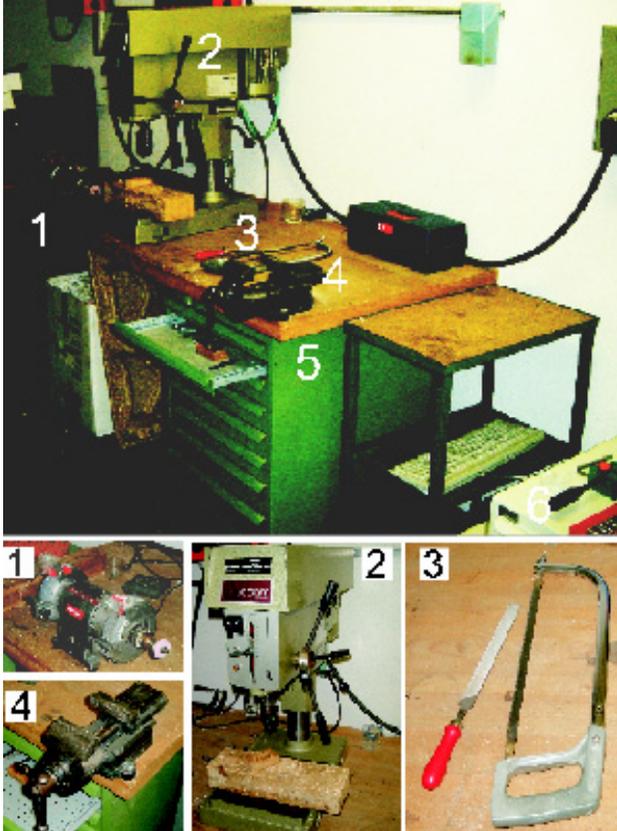
Following from the observation that hand motions and accompanying sounds can provide characteristic information on certain activities, microphones and accelerometers were chosen as the system sensors. These were used to record a multi-subject dataset based on a wood workshop scenario, which shall be described further in the following section.

## 2.2. Experimental setup

Performing live experiments using “real-world” assembly or maintenance tasks is inadvisable due to the cost, safety concerns, and the difficulties in getting repeatable measurements under experimental conditions. As a consequence we decided to focus on an “artificial” task performed at the wood workshop of our lab (see Figure 2.1). The task consisted of assembling a simple object made of two pieces of wood and a piece of metal. The task required several processing steps using different tools; these were intermingled with actions typically exhibited in any real world assembly task, such as walking from one place to another or retrieving an item from a drawer.

No	action
1	take the wood out of the drawer
2	put the wood into the vise
3	take out the saw
4	saw
5	put the saw into the drawer
6	take the wood out of the vise
7	drill
8	get the nail and the hammer
9	hammer
10	put away hammer, get driver, get screw
11	drive the screw in
12	put away the driver
13	pick up the metal
14	grind
15	put away the metal, pick up the wood
16	put the wood into the vise
17	take the file out of the drawer
18	file
19	put away the file, take the sandpaper
20	sand
21	take the wood out of the vise

**Table 2.1:** *Steps of workshop assembly task*



**Figure 2.1:** *The wood workshop (top) with closeups of (1) grinder, (2) drill, (3) file and saw, (4) vise, and (5) cabinet with drawers.*

### 2.2.1. Procedure

The exact sequence of actions is listed in Table 2.1. The task was to recognise nine selected actions: use of hand tools such as hammer, saw, sandpaper, file and screwdriver; use of fixed machine tools such as grinder, drill and vise; and finally the use of two different types of drawer. To be ignored, or assigned to the garbage class, are instances of the user moving between activities and of interactions with other people in the shop.

For practical reasons, the individual processing steps were only executed long enough to obtain an adequate sample of the activity. This policy did not require the complete execution of any one task (e.g. the wood was not

completely sawn), allowing us to complete the experiment in a reasonable amount of time. However this protocol influenced only the duration of each activity and not the manner in which it was performed.

Five subjects were employed: one female, four male; all of normal build and in the range of 25-35 years old. All were right handed. Each subject performed the sequence in repetition between three and six times producing a total of  $(3+3+4+4+6)=20$  recordings. Some subjects performed more repetitions than others because of a combination of technical problems in recording data and the availability of subjects. Each sequence lasted five minutes on average. The total experiment time, therefore, was around 1 hour and 40 minutes (6014 seconds).

For each recording the activity to be performed was prompted automatically by a computer, which an observer read out vocally to the subject. The exact timing of each activity was recorded by the computer when the observer pressed a key at the beginning and end of the activity. Any errors in these semi-automatic annotations were later corrected by visual inspection of the data and listening to the recorded audio. This provided the ground truth from which all subsequent training and evaluations were based.

No labelling scheme of a continuous system can be perfect. The definitions of activity start and stop during ground truth annotation might be judged differently by different observers. Differences again arise depending on which sources are used (visual, sound, or even acceleration signals). For these experiments, therefore, a set of criteria on what constitutes each particular activity class - when it starts and stops - was used during the labelling process. The intention was not to provide the definitive definitions on the classes, but rather to help maintain consistency in the labelling between each of the different recordings.

### 2.2.2. Data collection system

Data collection was carried out using the ETH PadNET sensor network [73] equipped with two 3-axis accelerometers connected to a body-worn computer, and two Sony mono microphones connected to a MiniDisk recorder. The sensors were positioned on the dominant wrist and upper arm of each subject, with both an accelerometer node and microphone at each location, as shown in Figure 2.2 (further details on these sensors and their placements are given below.) These recordings were later ported to a desktop PC for processing.

#### 3-axis accelerometers

The wrist was regarded as a natural choice of sensor location - a close proximity to the hand, but without the obtrusiveness that a hand or glove mounted sensor might present. The relation between upper and lower arm movement



**Figure 2.2:** *The sensor type and placement on a test subject: (1,2) microphone and 3-axis acceleration sensors. Inset shows one of the PadNET sensor nodes used for collecting acceleration data. Sound collection is done using two mono microphones mounted alongside the PadNET nodes (only the wrist mic can be seen on this image (1)).*

was felt to reveal useful additional information on the posture and dynamics of certain hand activities, thus motivating selection of the upper arm location.

The reason for choosing 3-axis accelerometers was based on the observation that the arm and wrist tend to have more degrees of freedom than, say, the torso, where standard 2-axis sensors are traditionally used (for example, in ambulatory monitoring work [21, 22, 23]).

The x-axis on the wrist is defined by drawing a line along the forearm towards the index finger. The y-axis follows the direction of the extended thumb. When the arms are straight, flush with the torso, and with palms facing inwards, the x-axis of the upper arm follows the same line along the arm as the wrist x-axis. For this same position, the y-axis on the upper arm faces forward and stands perpendicular to the body.

Each PadNET sensor node consist of two modules. The main module incor-

porates a MSP430149 low power, 16-bit mixed signal microprocessor (MPU) from Texas Instruments running at 6MHz maximum clock speed. The current version reads up to three analog sensor signals including amplification and filtering and handles the communication between modules through dedicated I/O pins. The sensors themselves (see inset on Figure 2.2) are hosted on an even smaller “sensor-module” that can be either placed directly on the main module or connected through wires. The sensor modules were based on a 3-axis accelerometer package consisting of two ADXL202E devices from Analog Devices. The analog signals from the sensor were lowpass filtered in hardware with a  $f_{cutoff} = 50Hz$ , 2nd-order, Sallen Key filter and digitised at 12-bit resolution using a sample rate of 100Hz (as used previously in [73]).<sup>1</sup>

### Microphones

The wrist was chosen as a convenient location to detect sounds made by hand activities, and so this was where the first microphone was placed. Based on the hypothesis that two separately placed microphones can give a rough indication of sound source location (elaborated further in Chapter 4), a second microphone was also used. This was placed, during an initial study [74], on the user’s chest. Unfortunately this location meant that the distance between mics would change as the user performed the tasks. Instead, a microphone on the upper arm, just above the elbow, and a fixed distance from the wrist, was used.

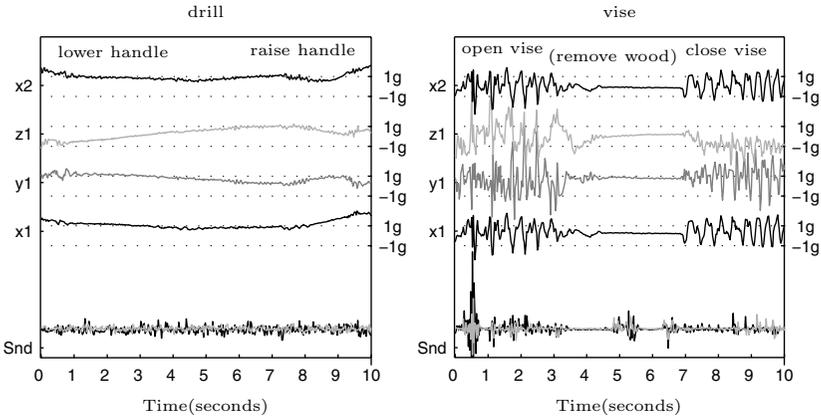
The two channels of recorded sound - wrist and arm - were initially sampled at 48kHz, but were then downsampled to 2kHz (with appropriate filtering in software) for use by the sound processing algorithms. In an earlier work on sound classification of activities [75], a sample rate of 4.8kHz was shown to be sufficient in capturing the most distinguishing frequencies of interest for common non-speech sounds. On further analysis of the sounds produced by the woodshop activities, we found that the most dominant signal frequencies were to be found below 1kHz. Thus a 2kHz sample rate was deemed to be sufficient.

### 2.3. Analysis of the data

The purpose of the following section is to investigate both the usefulness of the proposed method and the suitability of the chosen scenario. By looking at the recorded data, we categorise the different signals into more general activity, and sound, *types*.

---

<sup>1</sup>Note that some aliasing is likely but was not found to affect the experiments described.



**Figure 2.3:** Example data from drilling (left) and vise (right), both 10 seconds long. The signals shown are the  $x$ -axis arm acceleration ( $x_2$ ), 3-axis of wrist acceleration ( $x_1, y_1, z_1$ ), and sound (Snd - wrist in black, arm in grey). The acceleration signals, though offset from one another for readability, are each shown relative to gravity ( $\pm 1g$ ). Note the smooth acceleration signals as the drill is lowered, and then, after about 7s, raised again. The corresponding sound signal, mostly from the machine motor, remains fairly constant throughout. Starting from the jammed shut position, the vise is forcefully opened (note the peaks at 0.5 seconds), the wood is removed, and the vise is then slowly wound back to the closed position. Most of the sounds from each turn are a result of ‘chunks’ produced whenever the handle is loosened.

### 2.3.1. Acceleration analysis

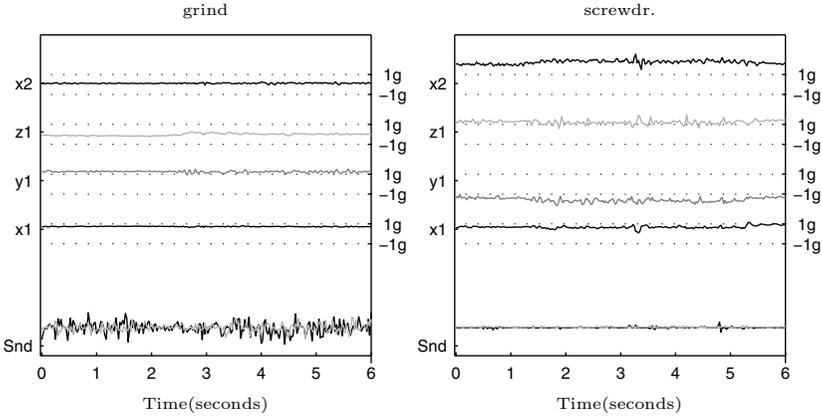
The maintenance activities that we are concerned with all involve the interaction of a subject’s hand with an object or tool. When analysing these interactions, a human observer might note certain characteristics of motion that are representative of that type of activity. If activity types can be distinguished visually, using plots of signal data, then any attempt at machine recognition is likely to be more successful.

The following analysis of the wood workshop data is conducted using visual inspection of the accelerometer signals. Of the nine activities at least seven types of motion involving hand-tool interaction were found:

1. *Well-defined, smooth motion* - as observed in the use of a drilling machine, an example of which is shown in the left plot of Figure 2.3. Such activities will have well defined motions which are dictated by the physical mechanics of the tool. The lowering of the drill handle, for example,

leaves very little room for variations other than the speed at which it is executed. This is also true for the rise of the handle at the end of a drilling sequence. The time length of a single sequence is usually in the order of a few seconds.

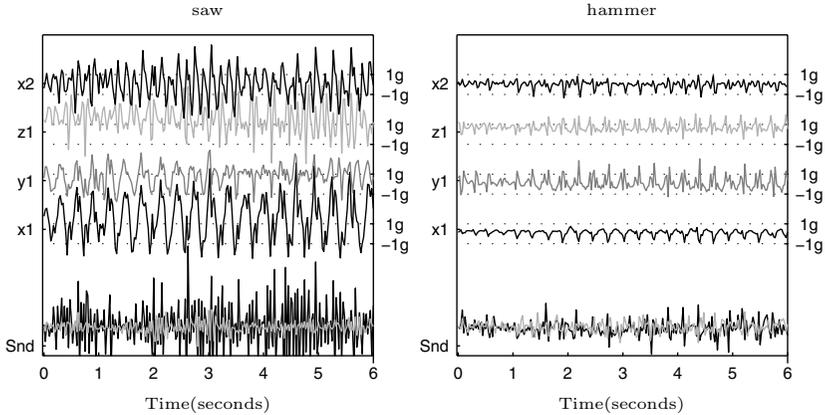
2. *(Almost) well-defined, continuous motion* - as observed in the tightening and loosening of a vise, an example of which is shown in the right plot of Figure 2.3. Though such activities have a well-defined motion dictated by the physical mechanics of the tool, there exists scope for a user to manipulate the tool in a variety of ways. With tightening the vise, for example, the user might lightly turn the handle with their finger at first, but move on to using their whole hand as the movement becomes tighter. Such activities are generally continuous, with a change in direction only at the beginning and end of the sequence - where the user tightens or releases the vise, for example.
3. *Steady hold (with possible vibration)* - such as observed with the use of a grinding machine while the user holds a piece of metal to the grindstone. Almost no movement is observed, as the signals in the left plot of Figure 2.4 show, and the hand is held in a fairly steady position, with only very occasional movement.
4. *Well-defined, repetitive twisting motion* - as observed in the use of a screwdriver. The user's hand and arm move the tool in a specific twisting manner. For any screw position one of the wrist axis always remain still while the other two show small notches in acceleration as the driver is twisted. In all of the examples in this dataset the screw is driven downwards, with respect to gravity. Consequently, the x-axis shows fewer, and less pronounced, notches than y and z.
5. *Well-defined, repetitive linear motion*. These activities can involve a steady back-and-forth motion, such as is observed with the use of a saw; or a steady rise of the arm, followed by a sharp fall, ending in an abrupt halt, as observed during hammering (Figure 2.5). Both of these activities are characterised by repetitive motions. Sawing, for example, usually involves more than one back-and-forth movement; hammering generally involves more than one strike, etc.
6. *Ill-defined, repetitive linear motion*. Such as is observed in the use of a hand-file or sandpaper (Figure 2.6). Again characterised by repetition, these activities have the additional complexity of allowing the user more freedom of movement, thus making them difficult to define. Sanding, for example, might involve regular side-to-side movements, alternating with back and forth movements. Consequently, these activities are harder to



**Figure 2.4:** Example data from grinder (left) and screwing (right). (Signals shown are the  $x$ -axis arm acceleration ( $x2$ ), 3-axis of wrist acceleration ( $x1, y1, z1$ ), and sound ( $Snd$  - wrist in black, arm in grey). Acceleration signals shown with respect to gravity ( $\pm 1g$ .) Note the lack of hand movement, and the continual sound (from the machine), when using the grinder. The notches of sound shown in the grinder example correspond to whenever the metal makes contact with the grindstone. The screwdriver example, in comparison, is almost silent. The small notches in the  $z1$  and  $y1$  axis reflect the twists of the driver. The  $x1$  axis shows slightly fewer notches than  $z1$  and  $y1$ , reflecting the fact that this is the axis pointing into the screw. More pronounced with this activity, however, is the effect of gravity on each of the axis - this reflects the specific position of the wrist and arm.

model than the well-defined ones such as hammering. In many of the experiments here, these activities are carried out in a way which can be regarded as very similar to that of sawing. Some confusion between recognition of these classes is therefore to be expected.

7. *Ill-defined, one-off motion* - as observed in the opening and closing of a drawer. The object being manipulated is fixed, with a well-defined motion dictated by its mechanics, but the manner of interaction can vary. The well-defined motion of the drawer, for example, might correlate perfectly with the motion of a user's hand. It might, however, be nudged shut by an elbow or foot - leaving no useful signals for a hand mounted sensor to detect. Such vagaries make this activity type difficult to model. For the experiment reported here, therefore, subjects were asked to use only their right hands. Owing to their shape, a 90 degree turn of the wrist is required in order to grasp the drawer handles. This turn is

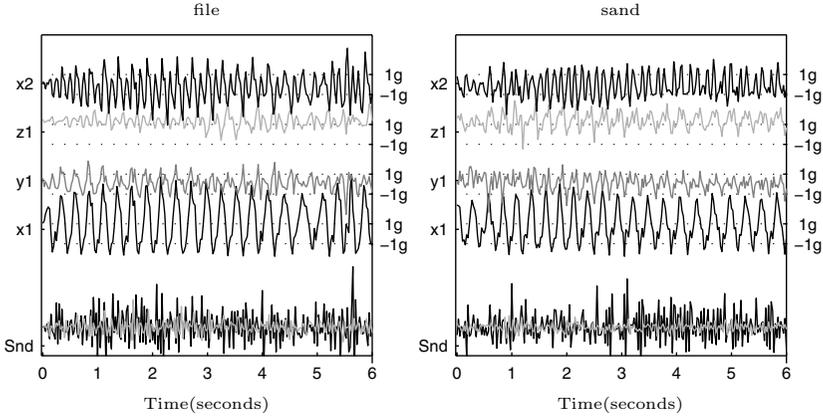


**Figure 2.5:** *Example data from sawing (left) and hammering (right). (Signals shown are the x-axis arm acceleration ( $x_2$ ), 3-axis of wrist acceleration ( $x_1, y_1, z_1$ ), and sound (Snd - wrist in black, arm in grey). Acceleration signals shown with respect to gravity ( $\pm 1g$ .) The sawing action produces a loud, regular sound and highly correlated, slightly serrated, acceleration signals. The hammer action is also very regular and well-correlated with sound: the smooth raise and quick fall; then, on striking the surface, a sudden halt and a loud peak in sound.*

reflected in the acceleration examples of Figure 2.7 by a 'dip' (a change in the  $1g$  influence of gravity) at the beginning and end of each open and close sequence.

A further category is the non-interesting, or *NULL* activities. These include everything from walking between activities, scratching one's nose, to picking up something from the floor. This category is the most difficult to model - especially if using only acceleration signals. This is because of the huge variety of activities consigned to *NULL*. We cannot model all possible activities - at least not using any method that requires real data for training. The required dataset, though a pattern recognition researcher's dream, would be extremely large and consequently difficult, if not infeasible, to collect.

An analysis of the sound signals, particularly the differences between the two microphone locations, might provide a solution to this problem - at the very least allowing the selection of potentially interesting activities to be narrowed down.

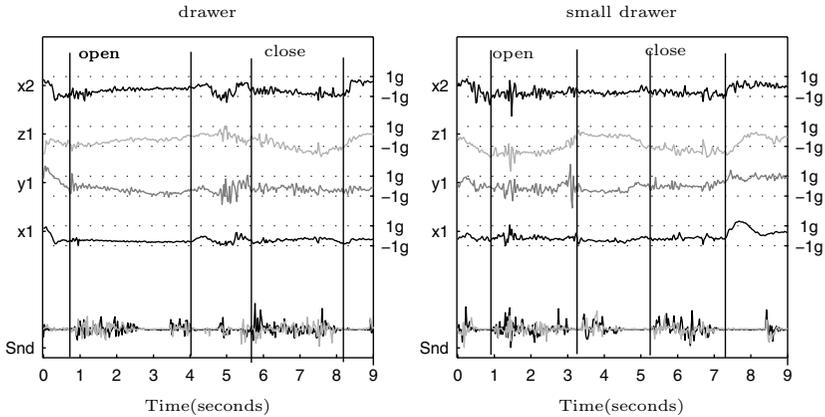


**Figure 2.6:** Example data from filing (left) and sanding (right), both with a similar back-and-forth motion. (Signals shown are the  $x$ -axis arm acceleration ( $x_2$ ), 3-axis of wrist acceleration ( $x_1, y_1, z_1$ ), and sound (Snd - wrist in black, arm in grey). Acceleration signals shown with respect to gravity ( $\pm 1g$ ).) Note, these signals might look regular, but are in fact potentially very variable - particularly with sanding. For example, the dominant wrist axis might change from  $x$  to  $y$  at any one time depending on the positioning of the object being worked on

### 2.3.2. Sound analysis

Considering that many of the motions relevant to assembly and maintenance scenarios are associated with distinct sounds, it makes intuitive sense to use sound analysis as a complementary method for recognition. Looking at the plots of data, we distinguish four different types of sound:

1. *Sounds made by a hand-tool* - these sounds are directly correlated with user hand motion. They include sounds such as sawing, hammering, filing, and sanding. Such actions are generally repetitive, and produce short sections of quasi-stationary sound: that is, sounds that have a constant and distinctive frequency spectrum signature over a given section of time (e.g. all strokes of the hammer have identical, or very similar, signatures to one another). This means that they are particularly attractive to recognition methods using spectral pattern matching. In addition these sounds are much louder than the background noise (they are dominant) and are likely to be much louder at the hand (where the action is happening) than elsewhere on the body.



**Figure 2.7:** Example data from the open and close sequences of two different drawers. (Signals shown are the  $x$ -axis arm acceleration ( $x_2$ ), 3-axis of wrist acceleration ( $x_1, y_1, z_1$ ), and sound (Snd - wrist in black, arm in grey). Acceleration signals shown with respect to gravity ( $\pm 1g$ .) The only notable difference between these two drawer types is their location - below the desk (for tools), or above (for screws and nails). This is reflected in the differing effects of gravity on the signals. Otherwise, the sequences follow a similar pattern: a dip (in acceleration) as the subject turns their hand to manipulate the drawer handle; a steady motion as the drawer is opened (with a corresponding 'rolling' sound); a flip of the wrist again as they manipulate objects within the drawer; and the same in reverse for drawer closing.

2. *Semi-autonomous* - these sounds, though not directly correlated to hand motion, are caused by machines or tools which the hand might come close to. This includes sound produced by a machine, such as the drill or grinder. Although ideal quasi-stationary sounds, sounds in this class may not necessarily be dominant and tend to have a less distinct intensity difference between the hand and the upper arm (for example, when a user moves their hand away from the machine during operation). The sound of a drawer opening and closing might also be included in this category.
3. *Quiet sounds* - these are generated by activities which are almost silent, or have no distinctive sound characteristic. An example of this type is the use of the screwdriver (see sound signals in right plot of Figure 2.4). In many instances, the vise too can be regarded as quiet. This is particularly so when a subject winds the vise handle in a careful and controlled manner, thus avoiding the 'clunking' sounds produced

whenever the handle drops.

4. *Autonomous sounds*. These are sounds generated by activities not driven by the user's hands (e.g loud background noises or the user speaking). These sounds are classed in this work as *NULL*.

The vast majority of relevant actions in assembly and maintenance are associated with handtool sounds and semi-autonomous sounds. In principle, these sounds should be easy to identify by utilising the differences in signal strength between the wrist and the arm microphones - this idea shall be treated in greater detail in Chapter 4. In addition, If extracted appropriately the sounds may be treated as quasi-stationary and can be reliably classified using simple spectrum pattern matching techniques.

## 2.4. Conclusion

The woodshop dataset presented here, though limited to 9 activities in a constrained scenario, represent a wide range of different hand and tool interactions. By analysing the sound and acceleration signals recorded from a user's hand and arm, a recognition system based on these sensor types seems feasible. We explore this hypothesis in greater detail in the following chapters.

# 3

## Activity recognition using sound and acceleration

*We now present a method for recognising activities using sound and acceleration. We demonstrate the feasibility of using two classifiers - LDA for sound, and HMM for acceleration - for this task. The classifiers are evaluated in isolation, against hand annotated data, using the multisubject dataset introduced in the previous chapter.*

### 3.1. Introduction

The previous chapter highlighted the feasibility of using sound and acceleration from body worn sensors to recognise activities in an assembly scenario. We now present a concrete method for performing this recognition.

For sound, a spectral pattern matching classifier is used - specifically, a minimum distance classifier based on linear discriminant analysis (LDA). This follows from the recommendation of Chapter 2 that most of the sounds involved in an assembly scenario may be treated as quasi-stationary.

A standard method of classifying quasi-stationary sounds is to perform pattern matching on the frequency spectrum using a fast fourier transform (FFT). FFTs have high a dimension output though, so some means of reducing this is often required - particularly for a system aiming towards a low power implementation. Stäger *et al.* [75] compared two such feature reduction techniques - Principle Component Analysis (PCA) and Linear Discriminant Analysis (LDA) (see Duda *et al* [76]). Of these, LDA performed favourably for the everyday sound recognition tasks dealt with in that work. Because we have a similar recognition task here, the same approach - using LDA - is also used.

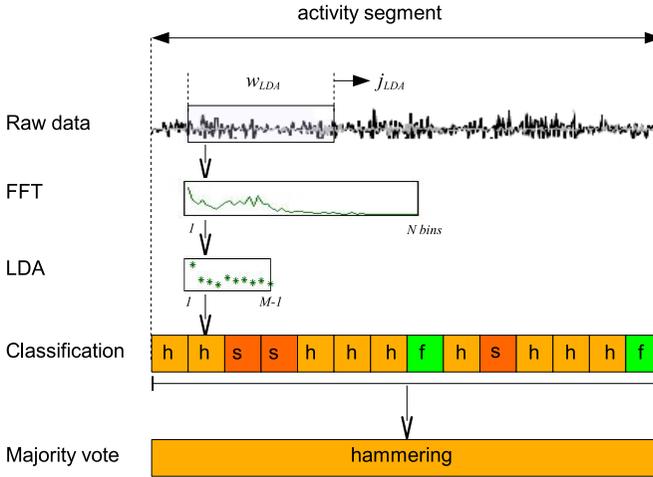
For acceleration, a time series classifier is used - specifically, Hidden Markov Models (HMM). The acceleration data, as shown in Chapter 2, is less characterised by frequency and more by particular sequences of motions. In the wood workshop scenario introduced in the last chapter, the ‘activity’ of hammering is defined by a repetitive sequence of smaller motions - lifting the hammer, then falling to strike. Such a sequence is an ideal candidate for HMM based classification.

HMMs are often used in pattern matching topics for modeling sequences. An advantage of HMMs, being probabilistic, is their resilience to noise caused by imperfect training and inaccurate sensors. They are capable of modelling important properties of activities such as time variance (the same activity can be repeated at varying speeds) and repetition (an activity containing some motion which can be repeated any number of times).<sup>1</sup>

Most commonly used in speech recognition, HMMs are gaining ground in other fields too. In the work of Starner [78] HMMs were used to recognise communication gestures, in Cohen *et al.* [79] for emotion recognition from facial expressions. In Potamianos *et al.* [55] they were used to assist with the automatic speechreading of impaired speech. HMMs have also been used by

---

<sup>1</sup>There have been many tutorials on HMMs, the most cited being the 1986 work by Rabiner [51]. A more recent recommendation would be Bilmes [77]. Murphy [52] includes an excellent treatment of HMMs, within the more general topic of Dynamic Bayesian Networks (DBN) (the same author is also responsible for the Matlab toolkit that is used later on in this chapter.)



**Figure 3.1:** Steps of LDA classification on sliding window. FFT features with high  $N$  dimension are reduced by LDA to  $M - 1$  dimension (where  $M$  is the number of classes in the system). The reduced vectors are then classified. Majority vote is taken over all the classifications within a given segment of data. Note: the ‘activity segment’ referred to here is any (variable length) section of data taken from, for example, the annotated ground truth.

various researchers in the topic of activity recognition. Recently, for example, Westeyn *et al.* [29] used HMMs for their work with wearable accelerometers.

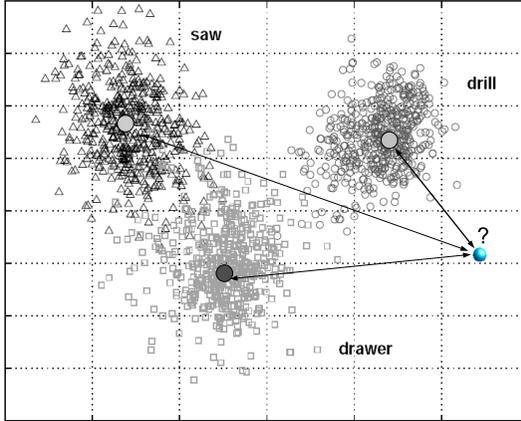
The implementation of the two classifiers - LDA and HMM - are described in more detail in the following sections. These are then evaluated on isolated data from the wood workshop scenario introduced in Chapter 2.

## 3.2. Recognition method

Recognition of both sound and acceleration data is carried out in two main steps: feature extraction, where the data is represented in a suitable form for distinguishing different classes; and classification, where a decision is made as to which class the data belongs.

### 3.2.1. Recognition using sound

The sound recognition process is illustrated in the example of Figure 3.1. The basic scheme operates on individual sound segments (or *frames*) of length  $w_{lda}$  which are sampled at  $f_s$ . The classification algorithm is ‘run’ across the data



**Figure 3.2:** *LDA minimum distance classification for a  $M = 3$  class example (reduced here to  $\#M - 1 = 2$  dimensions). Incoming LDA vector (marked with ‘?’) is shown with distances to the means of the training data clusters for each of the three classes. The minimum distance for the example shown here is to class ‘drill’.*

in increments of  $j_{lda}$ . At each step, an LDA vector is generated and classified. The classification for an entire data segment is then generated from majority vote of all constituent frame classifications. These steps are given in detail below.

### Feature extraction using FFT & LDA

The features used are the spectral components of each  $w_{lda}$  obtained by Fast Fourier Transformation (FFT). This produces  $N = \frac{f_s}{2} \cdot w_{lda}$  dimension feature vectors. Rather than attempting to classify such large  $N$ -dimension vectors directly, Linear Discriminant Analysis (LDA) is employed to derive an optimal projection of the data into a smaller,  $M - 1$  dimensional feature space (where  $M$  is the number of classes).

Transformation of each  $N$ -dimension FFT vector to an  $M - 1$  LDA vector is done by a single  $(N \times M - 1)$  matrix multiplication. This matrix is calculated from training data.

### LDA classification

The resulting LDA vector is then compared with the LDA class mean vectors (also obtained from training). Classification is performed simply by choosing the class mean which has the minimum Euclidean distance from the test

feature vector<sup>2</sup>. As an example of this, Figure 3.2 shows possible training data clusterings and their means for a simple 3 class system.

Finally, for classification of an entire segment, a majority vote is taken over all the constituent frame classifications.

### 3.2.2. Recognition using acceleration

Acceleration is processed by first extracting suitable features. The sequence of features for each segment of data is then ‘run’ through each of the class HMMs. These, in turn, return the likelihood of whether the sequence might have been ‘generated’ by that model. Classification is carried out by choosing the class of the model with the highest likelihood.

#### Feature extraction

In Chapter 2 it was shown that many of the different activities can be recognised by simply looking at plots of their data. These activities are often temporal in nature, with sequences of specific ‘sub-motions’: hammering, for example, was highlighted in the introduction as being a particular case of this. Drilling, too, involves a sequence of clear sub activities. These are enacted over a longer time period than hammering; the slow change in acceleration as the drill is lowered then raised again (see Figure 2.3).

A time-series modeling method, such as HMM, should have no problem identifying activities like these. In fact, for many of the activities only a small number of features is actually required. In this work the x-axis from both wrist and arm provide enough information to resolve most ambiguities between classes. For an example of this, refer to the grinder versus screwdriver plots in Figure 2.4 from the previous chapter.

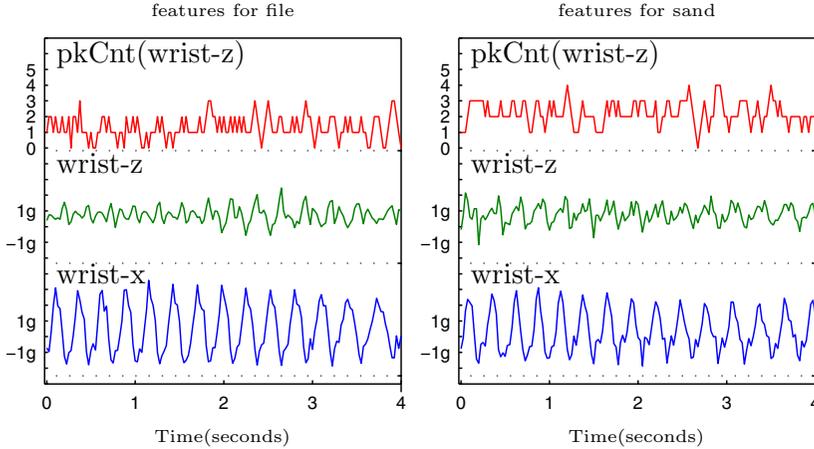
**Peak based features** Some of the activities, however, are not so easily characterised. Consider the similarity of signals for use of the saw, file and sandpaper in Figure 2.5 and Figure 2.6 from the previous chapter. These classes do have subtle differences, but not ones that are easily recognised by a time series analysis.

The use of sandpaper, for example, involves many subtle motions in the z-axis which are not as frequent as those during filing. This corresponds to the more ‘circular’ motion often employed during sanding, compared to the regular back and forth motion of using a file. To capture this effect, an approximation of a frequency based feature is used.

This is achieved using absolute counts on the number of peaks within a sliding window. Specifically, a  $w_{acc}$  window is slid across the data (sample

---

<sup>2</sup>A nearest neighbour (NN) approach might also be used. For this study both NN and Euclidean distance gave similar results.



**Figure 3.3:** *Filing versus sanding: (top) peak count feature for the z axis from wrist; (middle and bottom) the raw acceleration shown for the z and x axis. Note that, despite the similarities of the raw signals for the two activities, the peak count of z records a higher value for sanding than it does for filing.*

by sample). At each step the number of peaks found is returned. Figure 3.3 shows an example of this peak count feature for the z-axis (from the wrist sensor) for filing and sanding. Also shown, for comparison, is the raw x and z-axis signals. The actual  $w_{acc}$  used here is 100ms.

**Gravity invariance** One concern with using only ‘raw’ accelerometer data is the influence of gravity. Gravity is useful for characterising particular postures (such as the case with drilling using a fixed machine), but this also means that it constrains the characterisation of hand held tools. Hammering, for example, might be misleadingly characterised by the particular orientation used in training (striking downwards).

To combat this, a ‘gravity independent’ feature is desirable. Such a feature is approximated using the mean amplitude of all the peaks within a  $w_{acc}$  sliding window. Signal peaks are again used because they often represent the most important parts of a sequence - sudden changes in acceleration. For activities involving strong acceleration changes (the end motions of the saw, for example) the amplitude of these peaks will be much larger than the  $\pm 1g$  effect of gravity. Thus the mean over the short  $w_{acc}$  window will give an indication of large changes in acceleration with only minimal influence from gravity.

Specifically, then, the following features 14 features are used in this work:

- raw x-axis signals from wrist and arm
- count of peaks within  $w_{acc} = 100ms$  sliding window (on x,y and z for both wrist and arm)
- mean of peak amplitudes within  $w_{acc} = 100ms$  sliding window (on x,y and z for both wrist and arm)

These features are additionally downsampled from 100Hz to 40Hz. To avoid complications with large variances in the learning algorithms, they are also standardized across each experiment set. This effectively means that each test feature is subject to both an offset and a scaling factor obtained from training data.

### HMM classification

An HMM is defined for each activity class. Because the features used are continuous, the HMM observations are modeled using Gaussian probability distributions. For most class models, a mixtures of Gaussians are used. This provides a more expressive method of modeling the features.

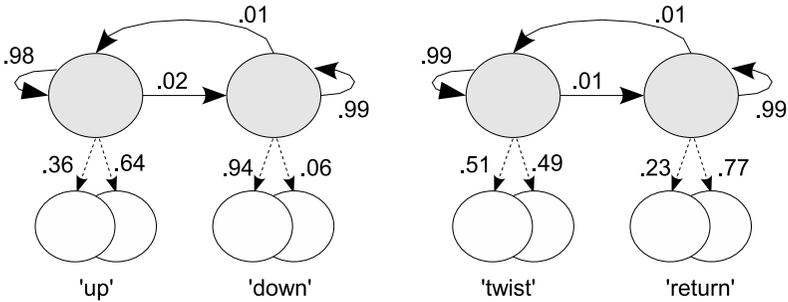
Defining an HMM involves setting three sets of parameters : the number of states each model is to have, how many gaussian mixtures it uses to represent each feature, and the prior probabilities for the transition matrices. The final setting of parameters - state transition probabilities, observation likelihoods and gaussian mixture settings - is done by training on data using the Baum Welch (*forwards-backwards*) algorithm (see Rabiner [51] for details).

Deciding on the number of states and the number of gaussians to use can be a tricky problem. Often the decision is made on a ‘rule of thumb’ basis: so long as there are sufficient states to describe the data - but not too many so as to avoid overfitting - the training step will usually take care of the details. This is the approach taken here.

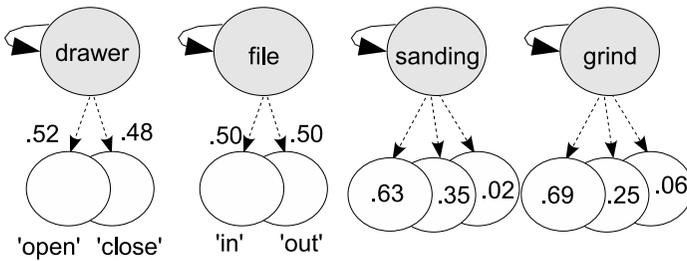
The models used in this work were all tailored on the motion analysis presented in Chapter 2. The states (and possible state transitions) are shown together with the number of gaussian mixtures, for each of the class models, in Figures 3.4, 3.5 and 3.6. Also shown are example transition and mixture probabilities (obtained from one of the training sets).

The implementation of the HMM learning and inference routines for this experiment was provided courtesy of Kevin P. Murphy’s HMM Toolbox for Matlab [80].

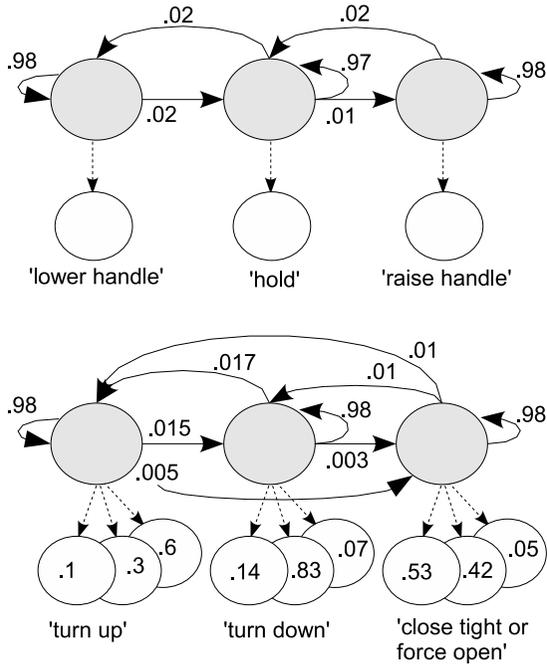
Classification is carried out by running the forwards algorithm (part of forwards-backwards algorithm) across the segmented dataset features for each HMM. This produces a log likelihood for each class model. The winning class is the one which produces the highest likelihood.



**Figure 3.4:** HMM states and gaussian mixtures for hammer (left) and screw-driving (right). Both are 2 class ergodic models.



**Figure 3.5:** HMM state and gaussian mixtures for drawer, file, sanding and grinder. The saw class (omitted) uses an identical model to 'file'. All of these classes use a single state. The 3 gaussians used for sanding and grind reflect the greater degree of freedom involved with these activities - whereas file usually only involves 'in' or 'out' motions, sanding can involve sideways motions too.



**Figure 3.6:** HMM state and gaussian mixtures for drill (top) and vise (bottom). These are the most complex of all the models used - each 3 states, with drill having 1 observation gaussian, and vise using 3.

### 3.3. Experiment

Data from the five-subject wood workshop scenario, as introduced in Chapter 2, is used to evaluate the methods. To recapitulate, this involves nine activities performed in a well-defined sequence, including use of: hammer, saw, file, vise, grinder, drill, sandpaper, screwdriver and drawer.

The sound data used here was recorded from a microphone mounted on each subject’s wrist. The accelerometer data was taken from both the 3-axis accelerometers on each subject’s wrist and upper arm.

#### 3.3.1. Setting parameters

In the work by Stäger *et al.* [75] the LDA algorithm was found to perform well using a sample rate of  $f_s = 4.8kHz$  and a window size of roughly  $w_{lda} = 50ms$ . This produced an FFT of  $N = \frac{256}{2} = 128$  bins.

In the current work, it was found that  $f_s$  could be reduced to as low as 2kHz without losing performance. This was obtained by increasing the sliding window size to  $w_{lda} = 100ms$ . To allow a suitable overlap, the window jump distance was set at  $j_{lda} = 25ms$ .

Because the results presented here are evaluated in isolation, the segments over which the classification is applied vary depending on the length of each event in the annotated ground truth.

#### 3.3.2. Evaluation method

The recognition methods are evaluated in isolation. That is, wherever the ground truth denotes a positive activity segment, a result is returned for HMM classification, and a result is returned for LDA. Because neither classifier can return a *NULL* or non-positive classification, there are only two possible outcomes: correct, or substitution (incorrect).

The metric used to summarise the results is class relative *isolation accuracy*, defined for each class  $c$  as:

$$isolation\ accuracy(c) = \frac{correct_c}{total_c} \quad (3.1)$$

with the number of  $correct_c$  and  $total_c$  isolated segments in the dataset.

Both the LDA and HMMs require prior training from data. Throughout this work, the data from the five subjects was divided and evaluated using the leave-one-out method for each of the user-dependent, user-independent, and user-adapted cases. These are implemented as follows:

1. *User-dependent*, where one set is put aside for testing, and the remaining sets from the same subject used for training.

2. *User-independent*, where data from the subject under test is evaluated using training data provided by the other subjects. This is the most severe test - evaluating the system's response to a never-before seen subject.
3. *User-adapted*, where one set is put aside for testing, and all remaining sets from all subjects are used for training. This case emulates situations where the system is partially trained for the user.

### 3.4. Isolation results

Table 3.1 shows results for the (a) user-dependent, (b) user-independent, and (c) user-adapted cases.

In the user-dependent case (a), both sound and acceleration produce very promising results, with average accuracies of around 96% and 94% respectively.

For LDA on sound, the worst performing class is screwdriving at 85%. This same class sees a much improved performance when acceleration is used (95%). The acceleration classifier has difficulty with the motion of filing (75%), but sound classification returns a more acceptable 95%.

When the LDA and HMM classifiers are trained using data from previously unseen subjects (b), there is an expected dip in overall sound and acceleration performance, 77% and 72% respectively. The activities which are particularly hard hit are those with typically high variation between subjects - filing, sawing and sanding. It is worth noting that the drill, with very little allowance for variation of use, maintains a solid 100% accuracy for both classifiers. In between, activities with stable characteristics of useage (the acceleration signature of screwdriving, or the sound of hammering), produce nearly similar results to the user-dependent case.

When a sample of the test subject's data is incorporated into training (c), a compromise between the above two training methods is obtained, with averages around 86% and 89% for sound and acceleration respectively.

### 3.5. Conclusion

This chapter evaluates the implementation of two systems capable of recognizing typical activities in an assembly task. One system uses data from a wrist worn microphone, the other uses wrist and arm mounted 3-axis accelerometers. Using the dataset of Chapter 2, the performance of the two recognition systems were evaluated in isolation (using hand segmented data). The best case result of 96% for sound and 94% for acceleration was obtained using user-dependent training. For user-independent training, where the systems were tested on never before seen data, recognition rates averaging around 77% for



# 4

## Segmentation using two microphones <sup>1</sup>

*Before attempting classification, a continuous sequence of activities must first be segmented into those which are potentially ‘interesting’ and those which are not.*

*In this chapter we present a method for detecting activities from a continuous sequence using two microphones. The algorithm is based on utilising the difference in sound intensity recorded by microphones placed on the upper arm and on the wrist. We show that the method is feasible for detecting activities involving the interaction of the hand with tools or machinery where noise is produced close to the hand.*

---

<sup>1</sup>This chapter is based on work originally presented in [81]

## 4.1. The segmentation problem

The previous chapter showed that classification of activities in isolation - where the data has been partitioned by hand - is a manageable problem. Real activity sequences are not as simple to recognise. For one, the start and end times of the activities are not known. Additionally, the bulk of activities occurring in a continuous sequence may not be interesting for the recogniser. These belong to the so-called garbage, or *NULL* class, and include, for the wood workshop scenario, activities such as standing still, talking to other people in the room, and walking back and forth.

Before recognition in a continuous sequence can be carried out, therefore, the data must be separated into those segments which are potentially ‘interesting’ activities, and those which are not (*NULL*). This process is known as *segmentation*.

This chapter presents a method of segmentation that uses the differences in sound intensity recorded at two different microphone positions. Specifically, the wrist and the upper arm.

The approach is based on the observation of Chapter 2 that all the activities of interest produce some kind of noise close to the hand. While this is certainly not true for arbitrary human activities, in the case of an assembly environment, it is a reasonable assumption.

Using the multi-subject dataset introduced in Chapter 2, two main criteria are investigated:

- *suitability of 2-microphone sound intensity analysis to separate ‘interesting’ activity sounds from ‘uninteresting’, or NULL*
- *robustness of 2-microphone sound intensity analysis for different subjects*

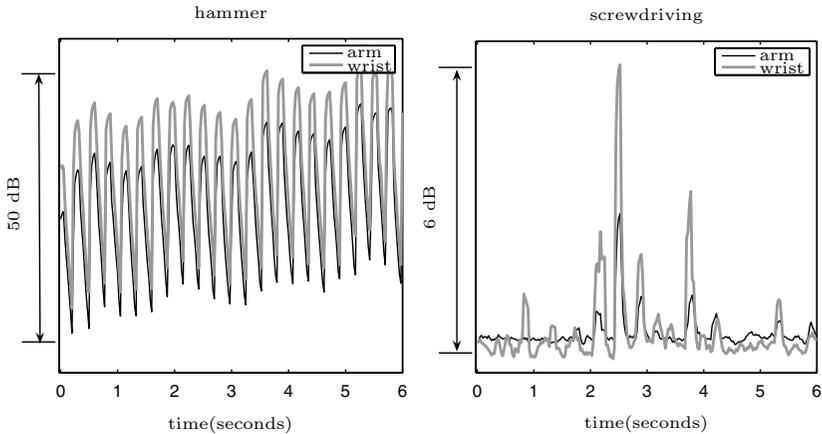
Dealing with the first criteria, the nine assembly tasks are divided into groups based on the four different sound types highlighted in Chapter 2. These are then analysed using ROC curves to determine the suitability of the method for spotting each of these types of sound.

For the second criteria, separate ROC curves are produced and analysed for each of the five subjects (1 female and 4 male).

## 4.2. Two microphone sound segmentation

Partitioning cues are obtained from an analysis of the difference in sound intensity from two different microphone positions: one on the wrist, the other on the upper arm.

The sound intensities,  $I_1$  from the wrist and  $I_2$  from the arm, are approximated by calculating the signal energies over a sliding window and then



**Figure 4.1:** *Audio profile of hammering and screwdriving activities from wrist and upper arm microphones. An approximation of the intensities are shown in dB. Note the stark difference between the wrist and upper arm measured signals during hammering. The differences for screwdriving, a much quieter activity, are smaller - though the wrist intensity does rise whenever the screwdriver ‘clicks’ with the screw being turned.*

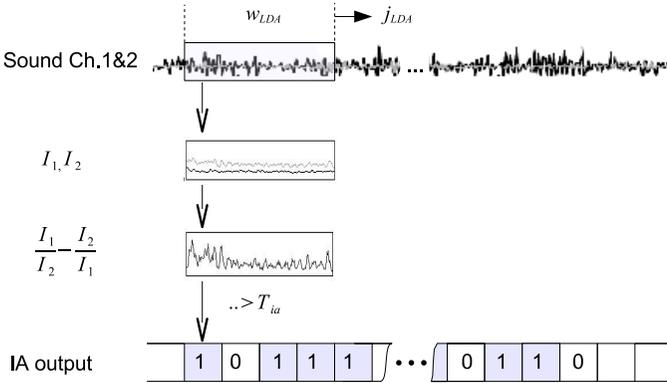
dividing by the window length. Though the intensity values obtained in this may be inexact, they give a good enough approximation for the work described here.

Figure 4.1 shows an example of the intensity signals for the activities hammering and screwdriving. For ease of comparison, the signals are shown on the decibel scale (dB) - this is obtained by taking the  $\log_{10}$  of  $I_1$  and of  $I_2$ . The loud hammer sound, when produced close to the hand, show a much higher intensity at the wrist microphone than it does at the arm. Even the relatively quiet activity of screwdriving shows a higher wrist intensity whenever a noise is made close to the hand.

### Intensity analysis algorithm (IA)

The intensity analysis algorithm (IA) makes use of the fact that the intensity of a sound signal is inversely proportional to the square of the distance from its source.

Two microphones - (1) on the wrist, and (2) on the upper arm - will register two signal intensities ( $I_1$  and  $I_2$ ) whose ratio  $I_1/I_2$  depends on the absolute distance of the source from the user. Assuming that the sound source is located at distance  $d$  from the first microphone and  $d + \delta$  from the second,



**Figure 4.2:** IA algorithm

the ratio of the intensities is proportional to:

$$\frac{I_1}{I_2} \simeq \frac{(d + \delta)^2}{d^2} = \frac{d^2 + d\delta + \delta^2}{d^2} = 1 + \frac{\delta}{d} + \frac{\delta^2}{d^2} \quad (4.1)$$

Sound originating far from the user,  $d \gg \delta$ , will result in  $\frac{I_1}{I_2} \simeq 1$ . Whereas sound originating close to the user's hand,  $d \simeq \delta$ , will result in  $\frac{I_1}{I_2} > 1$ . Thus, the ratio  $\frac{I_1}{I_2}$  provides an indicator of whether a sound was generated from the action of the user's hand.

Based on this, the following sliding window algorithm is performed on data from the two audio channels. This process is also illustrated in Figure 4.2

1. Calculate  $I_1$  and  $I_2$  over a window frame of length  $w_{ia}$  seconds.
2. For each frame, calculate  $I_1/I_2 - I_2/I_1$ : zero indicating a far off (or exactly equidistant) sound, while a positive value indicating a sound closer to mic.1 (wrist)
3. Select those frames where  $I_1/I_2 - I_2/I_1$  passes a threshold  $T_{ia}$
4. Move the window forward by  $j_{ia}$  seconds and repeat.

### 4.3. Experiment

The five subject dataset from Chapter 2 was again used here. To recapitulate, this involved 4 males (A to D) and 1 female (E). Each subject repeated a five minute task using tools in a wood workshop between 3 and 6 times. This

produced a total of  $(6+4+4+3+3)=20$  recordings. The final dataset is 6013.7 seconds long (approximately 1 hour and 40 minutes).

The nine activities were separated into four distinct sound categories:

- handheld - sounds produced by the use of some object (or tool) held in the user’s hand.
- machine - sounds produced by an external machine with which the user might interact.
- quiet - activities which do not produce much noise, or produce noise away from the user’s hand, and
- *NULL* - the ‘garbage’ class of background noises and silences.

These categories are shown in Table 4.3. The total duration (in seconds) that each category represents in the dataset is also shown.

category	activities	total time(s)
handheld	hammer, saw, sandpaper, file	1119
machine	drill, grinder, drawer	1178
quiet	screwdriver, vise	938
<i>NULL</i>	”	2778

**Table 4.1:** *Four categories of sound, with constituent activities and total time represented in dataset. Total dataset size is 6013.7 seconds. Note that NULL makes up nearly half of the entire dataset size.*

In keeping with the parameters for sound classification (see Chapter 2), the audio streams were sampled at 2kHz. Also in keeping with the earlier work, the window parameters were set to  $w_{ia} = 100ms$  and  $j_{ia} = 25ms$ .

## 4.4. Results

The IA threshold,  $T_{ia}$ , was swept between the range -5 to 5. For each  $T_{ia}$ , the IA algorithm was run across the dataset, picking out frames of interest and leaving the rest as *NULL*.

This is a two class problem: positive activity classes versus *NULL*. The measures used for evaluation, therefore, are the true positive rate (*recall*) and false positive rate *fp*:

$$recall = True\ Positive\ Rate = \frac{True\ Positives}{Total\ Positives} \quad (4.2)$$

$$fp = \text{False Positive Rate} = \frac{\text{False Positives}}{\text{Total NULL}} \quad (4.3)$$

Recall measures how well the system returns positive activities from the dataset. If the method detected all of the ground truth frames which were labelled as activities, for example, then the recall would be 100%. False positive rate measures the amount of *NULL* labelled frames which have been wrongly detected. An ideal system would have an *fp* of 0 (as well as a recall of 1).

In order to gain a better idea of how the algorithm works over the range of threshold values, graphs plotting the Receiver Operator Characteristic (ROC) were used. ROC curves plot recall against *fp* for a sweep of some decision parameter (in this case  $T_{ia}$ ). For a good overview of ROC, see the tutorial by Fawcett [82].

The results for the five subjects - A to E - are shown in Figure 4.3. The recall and *fp*, as defined above, for the entire dataset are shown in the curves marked 'all'. Points falling on the diagonal indicate a random result; the further above the diagonal (i.e. high recall, low *fp*), the better the result.

For this evaluation, however, it is also desirable to look at how well the different sound categories are detected. To do this, the recall measure was adapted to only consider frames within the ground truth labelled segments for each category *c*: handheld, machine or quiet<sup>2</sup>.

$$\text{recall}(c) = \frac{\text{True Positives}(c)}{\text{Total Positives}(c)} \quad (4.4)$$

The modified curves are also shown in Figure 4.3. Note that the only difference between these is their recall - the *fp* rate remains the same for each value of  $T_{ia}$ .

Figure 4.4 summarises the results by taking the mean of the curves over the subjects. Also shown is the standard deviation of the recall.

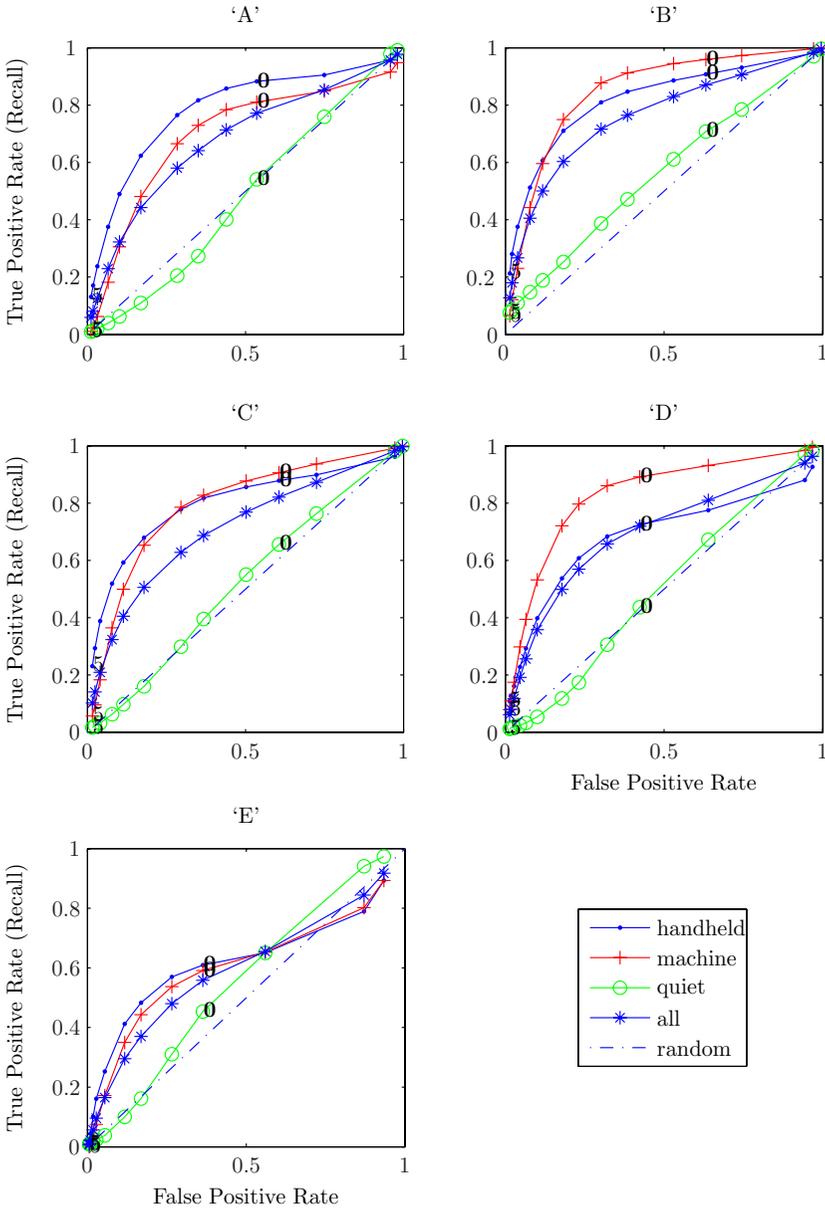
#### 4.4.1. Analysis of results

As might be expected, the IA algorithm picks out both handheld and machine activities consistently better than quiet activities.

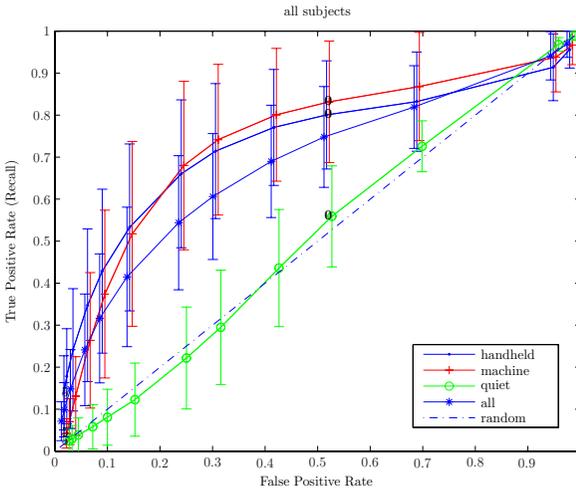
For most subjects, the machine sounds produce the best results of all. One reason for this is that machines tend to produce a constant sound when on. Additionally, one of the experimental constraints was that subjects kept their hand close to the machine at all times during operation. In reality, it might be expected that such strong performance will fall on relaxation of this constraint.

---

<sup>2</sup>This is done using the hand labelled ground truth - at no stage here do we attempt to classify activities.



**Figure 4.3:** ROC graphs for the five subjects over a sweep of  $T_{ia}$  from  $-5$  to  $5$ . Each plots the overall (marked ‘all’) recall vs. false positive rate (fp). Also shown are recall plots for the activity sets (handheld, machine, quiet) against the same fp values. A random result is marked by the diagonal where  $fp=recall$ . The optimal result lies closest to the top left of each ROC.



**Figure 4.4:** ROC graphs over a sweep of  $T_{ia}$  from  $-5$  to  $5$ . Each graph plots the overall (marked 'all') recall vs. false positive rate (fp). Also shown are the recall plots for each activity set (handheld, machine, quiet) for the same fp values. The values shown here are taken from the mean over all subjects. Additionally shown is the standard deviation of recall over all subjects. A random result is marked by the diagonal where  $fp=recall$ . The optimal result is the one lying closest to the top left corner of each ROC. Note how machine sounds and handheld sounds perform consistently better than the quiet activities.

With the handheld tools, this constraint need not be enforced - it is inherent in the manner of use. It can be expected therefore that the performance reported here would be an accurate prediction of a real world application (as shown, to some extent, by a smaller standard deviation for handheld on Figure 4.4.)

The results do vary somewhat between users, in particular the only female subject (E) shows a relatively poor response. This might be explained by the somewhat awkward setup involved with placement of the microphones.<sup>3</sup> A more comfortable arrangement of sensor placement - using smaller, more wearable apparatus - is a definite requirement if such systems are to be used in real situations.

In general however, the graphs show that depending on the application requirements - recall vs. fp tradeoff - a suitable IA threshold parameter might be chosen independent of the user.

## 4.5. Conclusion

In this chapter, ROC curves were used to evaluate the feasibility of using two microphones to segment continuous hand activities from *NULL*. The following conclusions were made: the segmentation scheme is suitable for use in detection of handheld tool activities (provided a sound is made during use); it is also suitable for detecting activities involving interaction with a (noisy) machine - provided that the user's hand comes into contact with machine. Finally it was found that, depending on the desired recall - fp trade-off, the algorithm parameters can be set independent of any particular user.

---

<sup>3</sup>It turned out, unknown to the authors at the time, that subject E was in fact pregnant - a factor that no-doubt influenced the recordings for this subject!



# 5

## Continuous recognition using classifier fusion<sup>1</sup>

*We now present a complete solution to the problem of continuous recognition. The method of sound based segmentation, introduced in Chapter 4, is improved here by the addition of classifier information and output smoothing. We apply sound and acceleration classification to these segments using the methods presented in Chapter 3. We then present four variants of classifier fusion - comparison, highest rank, borda count and a method using logistic regression - and show how these improve on the performance of the individual classifiers. Once again, we evaluate all of the steps and methods using the wood workshop multi-subject database.*

---

<sup>1</sup>This chapter is based on work presented in [83]

## 5.1. Introduction

In Chapter 3 we have shown that activity recognition in the isolation case - where the beginning and ending of activities are known - can be achieved with good accuracy. Similar findings to these are corroborated by the work of other groups, e.g. Bao & Intille [30].

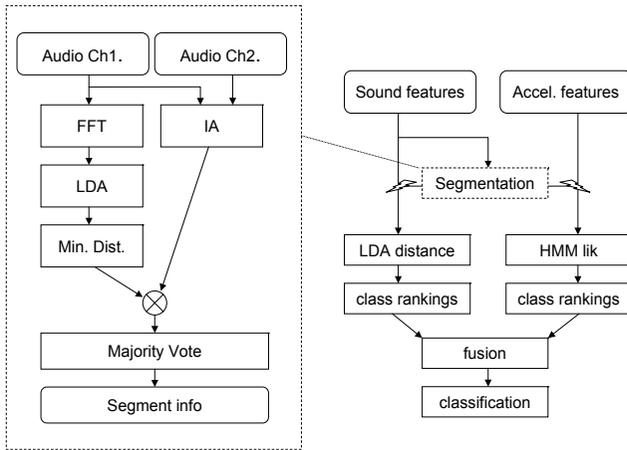
In the continuous case, however, where the start and stop times of activities are not known, reliable recognition is an elusive goal. The first problem encountered is that of segmentation: finding the start and stop times of an activity event from a stream of (mostly useless) “garbage” movements. The second problem, once a candidate segment has been found, is with classification. This is a difficult problem for the continuous case because, unlike recognition in isolation, a possible result is garbage, or *NULL*.

*NULL* poses a particular problem for continuous activity recognition. Any realistic dataset of human activities involves segments of random, non-relevant activities. These can involve many diverse movements, usually interspersed with those we are interested in. The wood workshop scenario, for example, includes extra activities such as moving tools around, swinging arms, or scratching. This diversity means that it is difficult to define a catch-all classifier for the *NULL* ‘class’ - at least not in the same way that a distinctive class, such as hammering, can be defined.

In the first part of this chapter we solve the problem of segmentation by making use of a variation on the intensity analysis “trick” (IA) introduced in Chapter 4. Taking the ratio of signal intensities recorded from microphones placed on the wrist and upper arm, IA makes it possible to identify sounds made close to a subject’s hand. The method developed here improves on the earlier work by classifying each of the (short) frames detected by IA and then smoothing over these with a longer sliding window to produce the desired segments.

The event classification problem - differentiating activities of interest from *NULL*, as well as one another - is then solved using classifier fusion. We treat segments as isolated events on which both sound and acceleration classification is performed separately. These separate classifications are then fused. This step is particularly important for removing false positives resulting from over sensitivity of the sound based segmentation. This follows from the premise that it is unlikely that two completely different sensing modalities - sound and motion - will agree on a false classification. Four different methods of fusion are evaluated: comparison of top choices (COMP), highest rank, Borda count, and a method using logistic regression.

The methods are all evaluated using the multi-subject dataset of the wood workshop scenario detailed in Chapter 2. User-dependent, user-independent, and user-adaptive training, to assess robustness of the methods to changes in user, are evaluated for both isolation and continuous recognition.



**Figure 5.1:** *Recognition algorithm: sound-based segmentation (left); overall recognition process (right)*

## 5.2. Recognition method

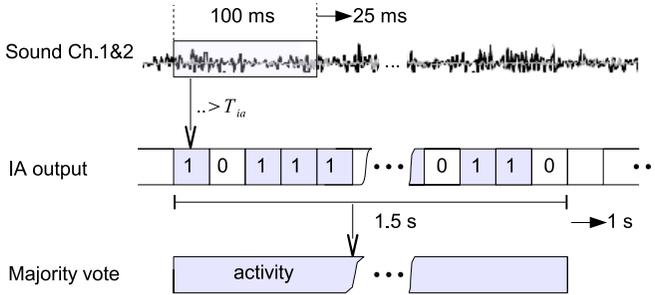
An overview of the recognition process is given in Figure 5.1. The right side of the figure shows the overall algorithm for continuous recognition. First, the sound and acceleration features are broken up into segments. Then, LDA class distance calculation for sound, and HMM likelihood calculation for acceleration (see Chapter 3) is carried out on these segments. The distance and likelihood values are converted into class rankings, which are then combined using one of the fusion methods. The fused rankings are then classified for each segment by selecting the highest ranking class.

The processing of the audio channels (from the wrist and arm microphones) for segmentation, and LDA classification, is shown in the left-hand box of Figure 5.1.

All of the steps in this process are explained in more detail in the following sections.

### 5.2.1. Frame-by-frame sound classification using LDA

Sound classification is carried out exactly as introduced in Chapter 2. The audio stream is sampled at  $f_s = 2\text{kHz}$  from the wrist worn microphone. The signal is then processed by a jumping  $w_{lda} = 100\text{ms}$  window (or *frame*), which



**Figure 5.2:** First attempt at segment smoothing: IA uses two channels of sound to select ‘interesting’ frames (made here by ‘1’). A majority vote is then taken over a large (1.5 seconds) window of these frames. This window advances by 1 second and the process repeats.

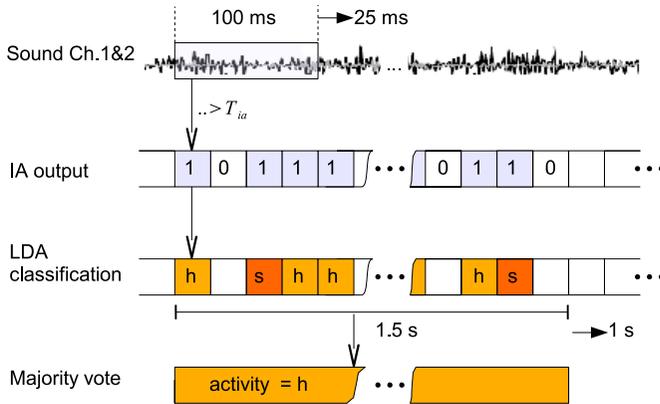
advances in  $j_{lda} = 25ms$  increments, producing an output of 40 frames per second. A Fast Fourier Transform (FFT) is carried out on each 100ms window, generating an output vector of 100 bins. The dimensionality is then reduced using Linear Discriminant Analysis (LDA) to  $\#Classes - 1$ .

### 5.2.2. Sound-based segmentation

The initial approach to segmentation was as presented in Chapter 4. That is: apply the IA algorithm, with a window of  $w_{ia} = 100ms$  and jump of  $j_{ia} = 25ms$ , across a sweep of different thresholds, highlighting those frames of interest for LDA classification and marking the rest as *NULL*. This tended to produce a somewhat fragmented result with wildly varying partition sizes. To combat this, two different methods of “smoothing” using variations of the majority vote were applied. In each of these, a window of 1.5 seconds was moved over the data in one second increments. This relatively large window was chosen to reflect the typical timescale of the activities of interest.

The first approach at smoothing, shown in Figure 5.2, was to run a two-class majority vote window directly over the output of the IA algorithm. This process has the effect that in any given window, the class with the most number of frames (either “interesting” or *NULL*), wins and takes all the frames within that window. In the (rare) event of a tie, the *NULL* class is assigned.

The second approach, and the one chosen for the remainder of the work, is to perform a majority vote over already classified frames. The left box of Figure 5.1 shows an outline of this algorithm, as does the example illustration of Figure 5.3. A frame-by-frame LDA classification is performed on those frames selected by IA; those not selected by IA are “classified” as *NULL*.



**Figure 5.3:** *The segmentation algorithm used in this work: IA uses two channels of sound to select ‘interesting’ frames. These are then classified by LDA. The final result is taken by majority vote over all classification results in a 1.5 second window. In the example shown, activity ‘h’ has the majority. However, if any class fails to take more than  $1/\#Classes$  of the total frame count, the output is assigned to NULL.*

Then a majority vote window is moved over all the frames. This process differs from the previous approach in that in order to “win” a window, a class has to have both more frames accounted to it than any other non-NULL class, and more than  $1/\#Classes$  of the total number of frames. If no positive class wins, NULL is assigned.

The results from all three of these approaches, and the reason for choosing multi-class majority vote, is explored further in the results section 5.5.1.

### 5.2.3. Sound classification

Segments are defined as a sequence of one or more contiguous same-class windows. Because of the window sizes used in the segmentation scheme, the segment lengths can vary from a minimum of 1.5 seconds to any additional multiple of 1 second. Classification of these segments is carried out in a similar way to the isolation case, with one result for the entire segment.

When higher level information about a segment is required, such as the likelihood of each possible class, then the problem is not as straightforward. One approach is to build a histogram entry for each class over the frame-by-frame classifications, thus providing an estimate of class probability. However, this method throws out potentially useful information provided by the LDA frame-by-frame classification. Another approach, and that adopted in this

work, is to take the LDA distance values for each class and calculate their mean over all the frames. This provides a single set of class distance values for each segment. How these distances are then used for classifier fusion is elaborated in the later sections on fusion.

#### 5.2.4. Acceleration recognition

The classification of acceleration is carried out in the same way as introduced in Chapter 2. A combination of features are used to feed the HMM models. These are calculated from sliding a 100ms window over the 100Hz sampled, x,y, and z axis of the wrist and arm data. The specific features used, for each window, are: a count of the number of peaks, the mean amplitude of these peaks, and the raw x-axis data from both wrist and arm.

The features are then globally standardised - such that each dataset has zero mean and a standard deviation of one. Finally, they are re-sampled (and appropriately lowpass filtered) to produce an output of 40 frames per second.

HMM classification is then carried out on segments of these features. The HMMs used are those described in Chapter 2. The segments themselves are those defined by the sound processing algorithm described above. This means that for each segment there is a list of HMM likelihoods for each class in addition to the list of class distances produced by the sound classification.

#### 5.2.5. Comparison of top choices (COMP)

Ideally segments are non-*NULL*, but because segmentation is rarely perfect there are likely to be instances of false positives - sequences of *NULL* which the recognition algorithm falsely returned as positive. As stated in the introduction, classification of *NULL* is a difficult problem for any one classifier, and as such neither the sound nor acceleration methods used here are able to detect this ‘class’ on its own. Instead, on encountering a *NULL* segment, each method will return a result from its closest fitting class. As these two methods are based on completely different sensing modalities - sound and acceleration - the chance that they will agree on such a false classification is low. In fact, given 9 classes, the probability of such an occurrence happening randomly is  $9^{-2} = 0.0123$ , or about 1.2%.

The first approach at fusion is simply to compare the output of the two classifiers. The final decision labels from each of the sound and acceleration classifiers for a given segment are taken, compared, and returned as valid if they agree. Those segments where the classifiers disagree are classified as *NULL*.

### 5.2.6. Fusion using class rankings

There can be cases where the correct class is not the top choice of a classifier, but may be a close second. If judged only by its top choice, the result would be an error. A more tolerant approach considers the levels of confidence a classifier has for each possible class. By comparing their respective levels of likelihood, for example, two (or more) classifiers might be able to come to a more accurate conclusion.

A problem arises when trying to combine two different classifiers that use different measures of class confidence - two measures which might be incompatible. This is the situation when trying to combine measures based on LDA distance and those based on HMM class likelihoods. It is conceivable that these measures might be converted into probabilities and then fused using some Bayesian method, but such an approach would require additional training in order to perform such a conversion. Additionally, with the view to a future distributed wearable sensing system, such calculations might be expensive - both computationally and, considering any future expansion on the number of classes, communication bandwidth. A mid-range solution, therefore, is to consider the class rankings. This is a computationally simple approach which lends itself to modular system design (for example, if additional classes and classifiers are added at a later stage).

The first step is to use confidence measures (distance or likelihood) to assign a ranking to each candidate class. Each classifier issues a list of such rankings for every class. This is compared with the rankings from the other classifiers. A final decision is made based on this comparison. To ensure that a decision is possible, rankings must be given in a strict linear ordering, with "1" being the highest.

From the acceleration HMMs, an ascending rank can be produced directly from the inverse log likelihood of each class model - the largest likelihood being assigned the highest rank. For sound, the approach is slightly different. First, the LDA class distances for each frame in the segment are calculated. The mean of these is then taken, and ranking is assigned according to the criteria of shortest distance. Where there is a tie between classes, the ranking can be assigned randomly or, as used here, by reverting to prior class preferences.

Three different methods of fusion using class rankings are used, all adapted from the work of Tin Kam Ho *et. al.* in [65]: highest rank, Borda count, and logistic regression. The implementation of each of these methods is described below (for a detailed treatment, however, see the referenced work):

#### Highest rank (HR)

For any given input, take the rankings assigned to each class by the classifiers and chooses the highest value. For example, if the sound classifier assigns

“drilling” with rank “2” and the acceleration classifier gives it rank “1”, the highest rank method will return rank “1.”

This method is particularly suited to cases where for each class there is at least one classifier that is capable of recognising it with high accuracy. It is also suitable for systems with a small number of classifiers - more classifiers might produce too many ties between class rankings.

### Borda count

The Borda count is a group consensus function - the mapping from a set of individual rankings to a combined ranking (first used in the study of political elections [84]). It is a generalisation of majority vote: for each class it is the sum of the number of classes ranked below it by each classifier. The output is taken from ranking the magnitude of these sums, e.g. highest Borda count is assigned the highest rank.

Borda count is simple to implement, but it retains the drawback of all fusion mechanisms mentioned so far in that it treats all classifiers equally. To address this shortcoming, a method based on logistic regression was employed to approximate weightings for each classifier combination.

### Logistic regression (LR)

It is possible to train weights for different ranking combinations. Thus if one classifier is ‘known’ to give erroneously high rankings to a particular class, the fusion mechanism could weigh down the significance of its occurrence - relying more on what the other classifiers have to say.

A problem with this approach is that calculating and storing weights for every possible combination of rankings, and for every class, can be computationally expensive. Finding enough combinations in the data set with which to train these weights can also be troublesome. These problems are particularly acute for large numbers of classes and classifiers. One solution, therefore, is to estimate the weights using a linear function.

Specifically, a function is defined which computes the likelihood of whether a class is correct or not for a given set of rankings. That is, it estimates  $P(\text{true}|c, X)$  for each class  $c$  and set of input rankings  $X$ .

This is the approach taken with LR. For each class the following function is computed:

$$L_c(X) = \alpha + \sum_{i=1}^m \beta_i x_i \quad (5.1)$$

where  $X = [x_1, x_2, \dots, x_m]$  are the rankings assigned by each of the  $m$  classifiers; and  $\alpha$  and  $\beta$  are the logistic regression coefficients for that class. These coefficients are computed by applying a suitable regression fit to training data.

(This data is taken from all the ranking combinations which produced correct results during classifier isolation tests.)

In this work, LR is implemented as follows. At each step, the two classifiers individually assign a rank to each class:  $x_s$  from sound and  $x_a$  from acceleration. With  $X = [x_s, x_a]$ , the class likelihood is obtained by estimating  $L_c(X)$ . This is repeated for all classes. The final ranking is then obtained by sorting these likelihoods.

One problem with the implementation in this work is that none of the constituent classifiers are able to detect *NULL*. To solve this, LR allows thresholding on  $L_c(X)$ , thus enabling a *NULL* classification if the combination is an unlikely one. Class thresholds for each input combination are calculated using:

$$L_c^N(X) = \alpha^N + \sum_{i=1}^m \beta_i^N x_i \quad (5.2)$$

where  $\alpha^N$  and  $\beta^N$  are the coefficients trained from those ranking combinations which *do not* lead to correct classifications in the training set.

Thus, if  $L_c^N(X) > L_c(X)$ , then the ranking combination  $X$ , for class  $c$ , would be considered unlikely. In such an instance, *NULL* would be assigned a higher rank than  $c$ . Following from this, *NULL* is assigned top rank if all classes fall below their respective thresholds.

## 5.3. Experimental setup

The sequence of actions used is the same as that listed in Chapter 2. As a brief recap, the task was to recognise nine selected actions: use of hand tools such as hammer, saw, sanding paper, file and screwdriver; use of fixed machine tools such as grinder, drill and vise; and finally the use of two different types of drawer. To be ignored, or assigned as garbage class, are instances of the user moving between activities and of interactions with other people in the shop.

Five subjects were employed (one female, four male), each performing the sequence in repetition between three and six times producing a total of  $(3+3+4+4+6)=20$  recordings. Some subjects performed more repetitions than others because of a combination of technical problems in recording data and the availability of subjects. Each sequence lasted five minutes on average.

### 5.3.1. Training and testing

Where training was required, such as for the LDA, HMMs and LR, the data from the five subjects was divided and evaluated using the leave-one-out

method for each of the user-dependent, user-independent, and user-adapted cases. We describe these cases here:

1. The user-dependent case is where one set is put aside for testing, and the remaining sets from the same subject are used for training.
2. The user-independent case is where data from the subject under test is evaluated using only training data provided by the other subjects. This is the most severe test - evaluating the system's response to a never-before seen subject.
3. The user-adapted case is where one set is put aside for testing, and all remaining sets from all subjects are used for training. This case emulates situations where the system is partially trained for the user.

These were applied consistently throughout the work, and results for each are given where appropriate.

## 5.4. Recognition in isolation

The positive, or non-*NULL*, events in the ground truth are evaluated in isolation for each of the different recognition methods. The metric used to evaluate these is *event-based recognition accuracy*, defined as  $\frac{\text{correct}_c}{\text{total}_c}$ , with  $\text{correct}_c$  and  $\text{total}_c$  referring to the number of events for each non-*NULL* class  $c$ .

The results are given in Table 5.1 for (a) user-dependent, (b) independent, and (c) adapted. Usually in isolation tests *NULL* is not defined as a possible outcome. It is, however, included in this evaluation because of the fact that the COMP and LR methods are able to return *deleted* events, which are effectively *NULL* classifications. In practice, all of the LR errors in this study are substitutions. But for the COMP method, nearly all incorrect events are in fact deletions. The exceptions to this occur in 5.1(b), where the few substitution errors are shown alongside the accuracy, in brackets, as percentages of  $\text{total}_c$ .

As shown in Table 5.1(a), most cases with user-dependent training produce very strong results for the individual sound and acceleration classifiers (averaging over 90%). Any substitution errors that do exist in each of these methods are then completely removed when the classifier decisions are compared (the COMP method), albeit at the expense of introducing deletions. The ranking fusion methods fare much better - with Borda count recognizing five classes perfectly, and four with only a single event error.

When the same methods are applied to data from subjects not in the training set (user-independent Table 5.1(b)), there is an expected drop in the recognition rates of the constituent sound and acceleration classifiers. Activities such as using the drill or drawer continue to register almost perfect

Class	$total_c$	sound	accel.	COMP	HR	Borda	LR
hammer	20	100	100	100	100	100	100
sawing	20	100	90	90	90	95	100
filing	20	95	75	70	95	100	95
drill	20	100	100	100	100	100	95
sand	20	95	95	90	95	95	95
grind	20	100	90	90	100	100	100
screw	20	85	95	85	95	95	95
vise	160	87.5	99.4	87	99.4	100	99.4
drawer	440	98.2	99.1	98	99.3	99.3	99.3
Average%		95.6	93.7	90.1	97.1	98.3	97.6

(a) User-dependent isolation accuracies

Class	$total_c$	sound	accel.	COMP	HR	Borda	LR
hammer	20	90	85	75	70	75	85
sawing	20	75	45	35	35	70	80
filing	20	25	25	10(10)	10	50	60
drill	20	100	100	100	95	100	95
sand	20	60	70	35(5)	60	80	75
grind	20	85	35	30(5)	90	90	95
screw	20	85	95	85	95	95	95
vise	160	79.4	96.9	78(1)	97.5	99.4	97.5
drawer	440	95	96.4	92.1	99.1	98.6	98.2
Average%		77.2	72	60	86.3	84.2	86.7

(b) User-independent isolation accuracies

Class	$total_c$	sound	accel.	COMP	HR	Borda	LR
hammer	20	100	100	100	85	85	95
sawing	20	85	65	60	60	75	90
filing	20	60	70	35	50	90	85
drill	20	100	100	100	100	100	100
sand	20	60	100	60	90	90	95
grind	20	95	75	70	100	95	100
screw	20	90	95	90	95	95	95
vise	160	85.6	96.9	83.8	97.5	98.8	96.9
drawer	440	96.4	98.9	95.7	99.6	99.3	99.6
Average%		85.8	88.9	77.2	86.3	92.0	95.2

(c) User-adapted isolation accuracies

**Table 5.1:** *Isolation accuracies for sound, acceleration, and the four combination methods. (The sound and acceleration results are the same as those presented in Table 3.1, reprinted here for ease of comparison.) The absolute number of events for each class is given by  $total_c$ . Accuracies are given as percentages  $\frac{correct_c}{total_c} \%$ , where  $correct_c$  refers to the correct number of events for each class. The average percentage accuracy for each method is also given. Note: the few substitution errors for COMP in (b) are shown alongside the accuracy, in brackets, as percentages of  $total_c$ . The COMP method is unique in that, with the exception of those highlighted, all its errors are deletions. Note, too, that the fusion methods all show an overall improvement over sound and acceleration.*

results on both sensors. This is largely due to the specific movements which one must perform to complete them and the correspondingly person-independent sounds which they produce. Some activities, such as driving a screw and using a vise, yield poor results in the sound sensor data but are clearly recognisable in the accelerometer data. Again this is largely due to the very unique and specific person-independent motions which one must perform to use these tools.

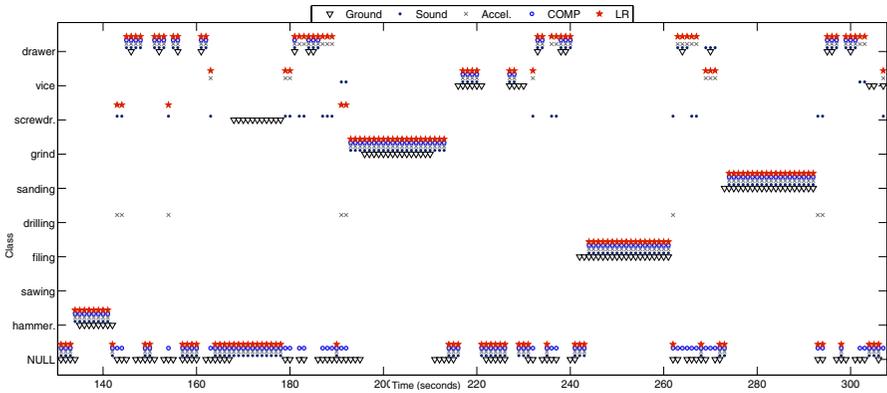
Comparison of classifier results fares less well in the user-independent experiment. Although the number of substitution errors is very low (but not quite zero for some classes), the large discrepancies in performance of the constituent classifiers ensures that the possibility of agreement is almost as low as the possibility of disagreement. This effect causes a large number of deletions - particularly for filing, sawing, sanding and grinding.

The ranking methods - in particular the LR, and to a lesser extent Borda - resolve the classes extremely well. Of particular note is the case of filing: although 60% (12/20) accuracy is not ideal, it is an enormous improvement on the 25% of the constituent classifiers.

Finally, when a sample of the user's data is incorporated into the training set for a user-adapted test Table 5.1(c), the results improve once again. Again the LR method performs best - nearly as well as in the user-dependent case.

## 5.5. Continuous recognition

Defining appropriate evaluation metrics is difficult in continuous activity recognition research [82]. There is no application independent solution to this problem [85]. Often the continuous recognition task requires discrimination of relatively rare activities from a default *NULL* activity that constitutes the majority of the time in the data. In addition, there may be more than one type of error in a system, such as posed by multi-class continuous recognition, and the common metric of accuracy can be misleading [86]. Further problems arise when one wishes to evaluate continuous recognition with ill-defined, often fragmented and variable length class boundaries. Similar problems exist in computer vision. Although ways of processing variable length boundaries exist (in 2D graphics [87], for example), it is common for researchers simply to show typical output figures (as commented by Hoover et al. [66]). A typical output of the work in this thesis is shown in Figure 5.4. Although these results can be compared (and evaluated) visually against the hand-labelled ground truth, for large datasets it is desirable to have some automatic way of doing this.



**Figure 5.4:** Section of a typical output sequence (approx. 3 minutes). Ground truth is plotted alongside the sound and acceleration classifications, together with two approaches at fusing these - comparison (COMP) and logistic regression (LR). User-dependent training is used. Note how sound and acceleration are prone to frequent insertion errors, a situation notably improved by COMP and LR. The question posed is: how are such results best quantified and evaluated?

### 5.5.1. Segmentation evaluation method

The following shows how well positive activities in a continuous stream are identified and segmented from *NULL* for each of the different methods. There are four possible outcomes: those returning positive activities, *true positive (TP)* and *false positive (FP)*; and those returning *NULL*, *true negative (TN)* and *false negative (FN)*. As the continuous recognition methods are all aimed at detecting TP activities, *NULL* is simply what is left behind and TN is consequently regarded as less critical than other outcomes. This is a similar view to that held by the Information Retrieval (IR) community, where the evaluation focus is on the positive results that are returned - how many of these are correct, and what proportion of the total existing positives they make up - rather than the remaining (often more numerous) negative results. The metrics chosen, therefore, are those used in IR, namely *precision* (also known as *positive prediction value*) and *recall* (*sensitivity*, or *true positive rate*):

$$\text{recall} = \frac{\text{true positive time}}{\text{total positive time}} = \frac{TP}{TP + FN} \quad (5.3)$$

$$\text{precision} = \frac{\text{true positive time}}{\text{predicted positive time}} = \frac{TP}{TP + FP} \quad (5.4)$$

A *precision-recall* (PR) graph can be plotted to show the effects of different parameters when tuning a recogniser [88].

An alternative representation would be the Receiver Operator Characteristic, or *ROC*, which plots recall against *false positive rate*, or *fp* (defined as  $fp = \frac{FP}{FP+TN}$ ). Unlike the use of *fp* in Chapter 4, however, a later requirement for the current analysis will be to present class relative results (section 5.5.4). For this *fp* is not well suited because of the dominating effect *NULL* - which is 46% of the entire dataset - has on it. This goes back to the IR argument, given above, in favour of using metrics that focus on positive, non-*NULL* results. For this reason, precision (and hence PR), rather than *fp* (and ROC), is used throughout the remainder of this work.

### 5.5.2. Segmentation results

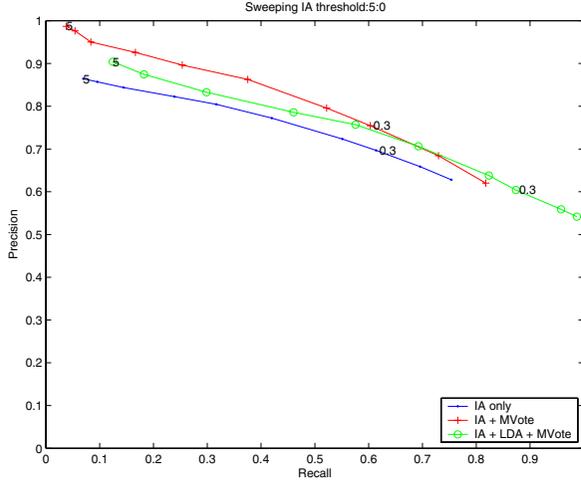
In evaluating segmentation there are two parameters that can be varied: intensity analysis threshold  $T_{ia}$ , and the majority vote window size. Of these,  $T_{ia}$  has the most significant effect. In Chapter 4, a sweep range of  $[-5, +5]$  was used for  $T_{ia}$ . This study uses values in the range  $[0, +5]$ , specifically 0, 0.1, 0.3, 0.5, 1, 1.5, 2, 3, and 5 (only positive values were considered useful). For each threshold, the total, correct, and predicted times are calculated and summed over all the test data sets. PR curves are then generated for each of the three segmentation schemes: IA selection on its own, IA smoothed with a majority vote, and IA+LDA smoothed with majority vote.

The IA alone gives the worst segmentation performance. Its prediction output is heavily fragmented, with many false negative frames and scatterings of false positives. Figure 5.5 shows this performance across the range of thresholds. When a majority vote is run over the IA selected frames, however, many of the spurious frames are smoothed away. Again this is reflected in the improved PR performance.

The third approach, IA+LDA+majority vote, does not, on first glance, seem to perform much better. More complicated than the previous two methods, it involves three steps: selecting frames using IA, applying LDA classification, and then running a multi-class majority vote window over these classified frames. The segmentation results are not immediately improved - in fact, for high precision, the IA+majority vote approach is still preferable. However, when considering that the later recognition stages aim to use fusion as a means of reducing insertions, a lower precision at the segmentation stage can be tolerated. With this in mind, what becomes preferable is a high recall. As it is the only method able to deliver recall rates above 80%, IA+LDA+majority vote is chosen for use in the subsequent work.

### 5.5.3. Segmentation and classification results

The sound and acceleration classifiers are applied to the partitioned segments. The four fusion algorithms are then applied over these.



**Figure 5.5:** PR comparison of 3 different segmentation schemes. Best performance (assuming equal precision-recall cost), is found towards the top right corner. Here we assume precision can be improved at a later stage, and are therefore more interested in high recall. For this, the IA+LDA+majority vote method is chosen.

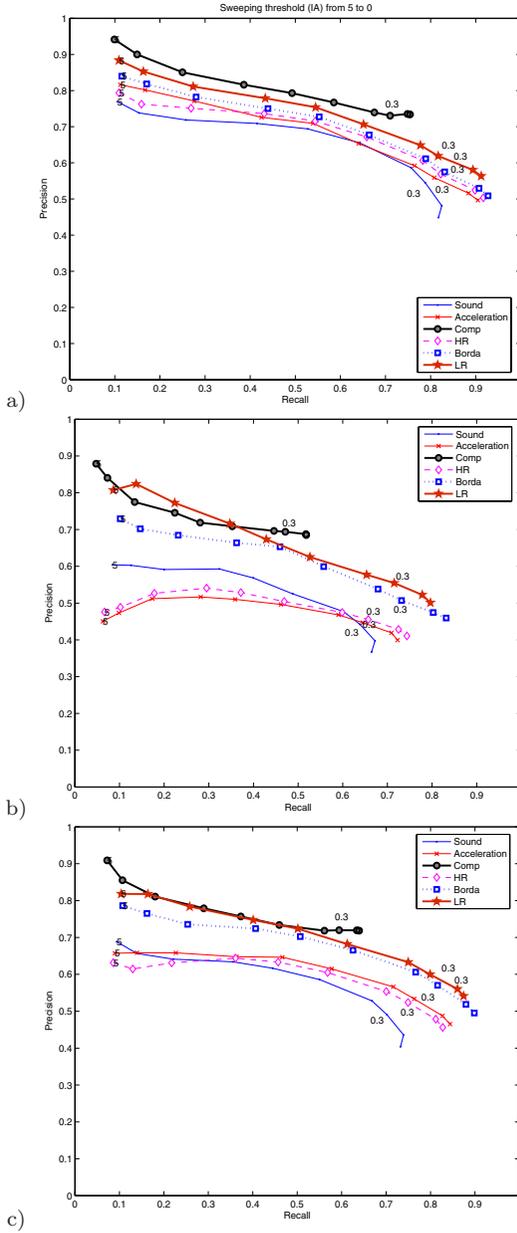
In the following analysis, the definitions of precision and recall are modified slightly. This is done to encapsulate the notion that in a setup with more than two classes, a TP data point is not just non-NULL, but is either a *correct* classification, or a *substitution* error. The revised definitions of *correct recall* and *correct precision* are given as:

$$\text{correct recall} = \frac{\text{correct positive time}}{\text{total positive time}} = \frac{\text{correct}}{TP + FN} \quad (5.5)$$

$$\text{correct precision} = \frac{\text{correct positive time}}{\text{predicted positive time}} = \frac{\text{correct}}{TP + FP} \quad (5.6)$$

Counts of correct, TP, FN and FP are summed up from the whole dataset. This is done for each value of  $T_{ia}$ . The corresponding correct precision and correct recall values are then plotted. Figure 5.6 shows the curves for each method in the (a) user-dependent, (b) user-independent, and (c) user-adapted setups.

The main conclusion to be drawn from these graphs is that the performance is fairly consistent between each of the classifiers and fusion methods. This is particularly so within the desired middle-to-top right hand region of



**Figure 5.6:** Correct PR comparison of the classifiers with the combination schemes for user-dependent(a), independent(b) and adapted(c). Best case (assuming equal precision-recall costs) is found at the top right corner. So for high recall, choose LR; for precision, choose COMP.

the PR graphs. Of the ranking fusion methods, LR always performs better than Borda, which performs better than HR. All of these are improvements over the individual sound and accelerometer classifiers. COMP, usually with much lower recall than the other methods, maintains the highest precision in this region. Also noteworthy is the conclusion that  $T_{ia} = 0.3$  yields consistency within a suitably close operating region for each of the methods. This helps legitimise further comparisons that require a fixed  $T_{ia}$ .

#### 5.5.4. Frame-by-frame results

With  $T_{ia}$  set at 0.3, the results can be analysed in more detail. The first step is to calculate time-based confusion matrices. A confusion matrix shows how well a system performs by charting the output that should have been achieved (rows) against the system's predictions (columns). Correct results, therefore, are recorded in the  $c : c$  diagonal (with  $c =$  any class), with substitutions off from this. In this work, *NULL* is treated as a special class, so when *NULL* is substituted by another class, the error is a FP; likewise, where a class is substituted by *NULL*, the error is a FN. Table 5.2 shows a selection of matrices for the user-dependent evaluation of the wood workshop dataset. The entries here are counts of the total time (in seconds), over the entire dataset, for each correct class, substitution, FP, and FN.

Presenting all of these matrices for each of the different methods and training configurations - there are 18 in all - might lead to some confusion. As a means to compare the different methods, confusion matrices are simply too unwieldy. On the other hand, using a single metric, such as accuracy, simplifies things too much - and can lead to an equally unhelpful comparison. Instead, further analysis of these results is conducted using two complimentary summaries of the most pertinent results.

The first summary uses a visualisation of the overall performance measures that were given at the bottom of the matrices in Table 5.2. Specifically, the substitution, FN, FP, TN and correct positive counts are recorded and shown in graphical form as percentages of the total dataset size. Figure 5.7 shows these 'error bar' summaries for each of the different methods and training setups. This provides a convenient summary of the main findings from the confusion matrices and allows easy comparison of results from the different recognition methods.

For comparing performance of individual classes, a second approach is used. This evaluates each class using *class relative precision* and *class relative recall*: recall is defined as  $\frac{\text{correct}_c}{\text{total}_c}$ , and precision as  $\frac{\text{correct}_c}{\text{predicted}_c}$ , where  $\text{correct}_c$  is the total correct time,  $\text{total}_c$  the total ground truth time, and  $\text{predicted}_c$  the total predicted time, for each class  $c$ . The precision and recall rates for each positive class are shown in Table 5.3. Also shown is the average of these values over all classes. As an additional indicator of performance *NULL* is

Class	Total(s)	hammer	saw	file	drill	sand	grind	screw	vise	drawer	NULL	%R
hammer	195.7	<b>179.7</b>						0.6	3.1	0.5	11.7	91.84
saw	306.4		<b>276.4</b>	18.0		2.7		4.4	3.0		1.8	90.21
file	304.6		9.2	<b>233.1</b>		26.1	11.0	11.4	5.0	4.8	4.1	76.52
drill	241.5				<b>229.5</b>	6.0	4.5				1.6	95.00
sand	313.0	0.6	4.5	17.8		<b>255.9</b>	1.5	1.7	2.0	13.0	16.1	81.76
grind	277.7					1.5	<b>230.5</b>	1.4	0.2		44.0	83.02
screw	260.4							<b>136.5</b>	7.9		116.0	52.41
vise	678.1	0.8	6.6	1.5	1.9	1.2		55.2	<b>439.1</b>	1.5	170.2	64.76
drawer	658.8	1.2	2.8	1.2	2.3	7.5		25.0	5.9	<b>569.5</b>	43.4	86.44
Null	2777.5	61.5	17.1	18.1	191.3	81.8	88.8	431.8	332.8	629.2	<b>925.0</b>	33.30

<b>Sound</b>	Total: 6013.7	FN: 409.0 6.8%	FP: 1852.5 30.8%	Subst.: 277.1 4.6%	cTP: 2550.2 42.4%	cTP+TN: 3475.2 Accuracy: 57.8%
--------------	---------------	-------------------	---------------------	-----------------------	----------------------	-----------------------------------

Class	Total(s)	hammer	saw	file	drill	sand	grind	screw	vise	drawer	NULL	%R
hammer	195.7	<b>181.1</b>			0.1		2.1	0.2	0.6		11.7	92.56
saw	306.4		<b>276.8</b>	23.5		3.0			1.3		1.8	90.33
file	304.6		52.5	<b>244.4</b>	0.5				3.2		4.1	80.23
drill	241.5				<b>239.9</b>						1.6	99.35
sand	313.0			15.0	0.7	<b>272.6</b>			8.7		16.1	87.08
grind	277.7				44.2		<b>174.6</b>		14.8		44.0	62.89
screw	260.4		1.3		2.4			<b>137.2</b>	3.4		116.0	52.69
vise	678.1				2.2	0.3	1.2	0.1	<b>500.1</b>	3.9	170.2	73.75
drawer	658.8	2.9			0.4	0.1	0.3		34.8	<b>576.8</b>	43.4	87.55
Null	2777.5	44.6	16.0	15.8	292.8	19.6	84.7	20.6	452.5	906.0	<b>925.0</b>	33.30

<b>Accel.</b>	Total: 6013.7	FN: 409.0 6.8%	FP: 1852.5 30.8%	Subst.: 223.8 3.7%	cTP: 2603.5 43.3%	cTP+TN: 3528.5 Accuracy: 58.7%
---------------	---------------	-------------------	---------------------	-----------------------	----------------------	-----------------------------------

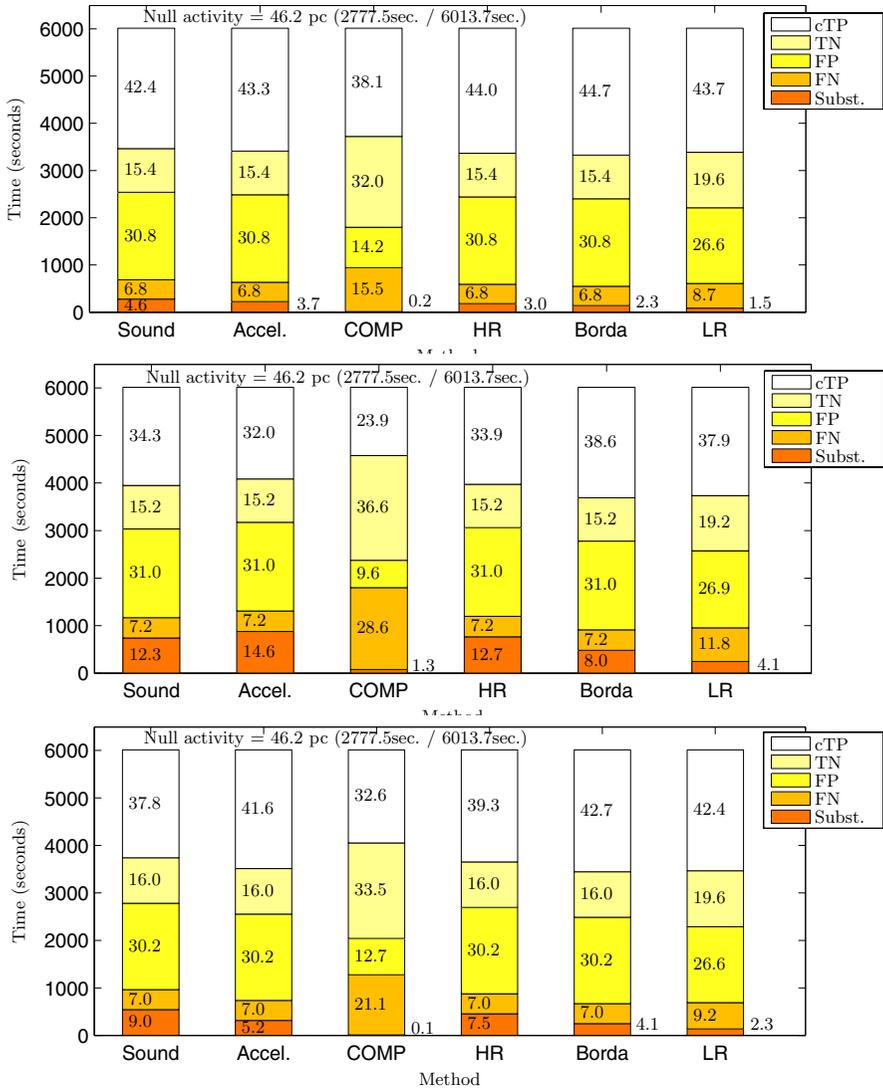
Class	Total(s)	hammer	saw	file	drill	sand	grind	screw	vise	drawer	NULL	%R
hammer	195.7	<b>179.7</b>						0.2			15.8	91.84
saw	306.4		<b>269.5</b>	6.0							30.8	87.97
file	304.6		1.5	<b>199.1</b>							104.0	65.38
drill	241.5				<b>229.5</b>						12.1	95.00
sand	313.0					<b>241.4</b>					71.6	77.13
grind	277.7						<b>171.0</b>				106.7	61.59
screw	260.4							<b>132.7</b>	0.9		126.8	50.95
vise	678.1							0.1	<b>414.5</b>		263.4	61.14
drawer	658.8					0.1			3.4	<b>453.1</b>	202.2	68.78
Null	2777.5	11.1	12.6	6.4	129.2	18.0	42.2	20.6	184.9	427.8	<b>1924.6</b>	69.29

<b>COMP</b>	Total: 6013.7	FN: 933.2 15.5%	FP: 852.9 14.2%	Subst.: 12.3 0.2%	cTP: 2290.6 38.1%	cTP+TN: 4215.3 Accuracy: 70.1%
-------------	---------------	--------------------	--------------------	----------------------	----------------------	-----------------------------------

Class	Total(s)	hammer	saw	file	drill	sand	grind	screw	vise	drawer	NULL	%R
hammer	195.7	<b>179.7</b>						0.3	0.6		15.1	91.84
saw	306.4		<b>284.9</b>	7.5					1.2		12.8	92.97
file	304.6		10.5	<b>250.9</b>					2.9	0.3	40.1	82.38
drill	241.5				<b>231.0</b>						10.6	95.63
sand	313.0		4.5	8.9		<b>260.9</b>			5.0		33.7	83.36
grind	277.7				12.2		<b>209.1</b>		1.6		54.8	75.31
screw	260.4							<b>138.4</b>	2.2		119.8	53.16
vise	678.1							0.1	<b>496.1</b>	4.4	177.5	73.16
drawer	658.8					0.1			25.7	<b>575.9</b>	57.0	87.42
Null	2777.5	14.1	15.2	10.3	15	18.0	76.2	31.4	401.6	884.9	<b>1175.8</b>	42.33

<b>LR</b>	Total: 6013.7	FN: 521.3 8.7%	FP: 1601.7 26.6%	Subst.: 87.9 1.5%	cTP: 2627.0 43.7%	cTP+TN: 3802.8 Accuracy: 63.2%
-----------	---------------	-------------------	---------------------	----------------------	----------------------	-----------------------------------

**Table 5.2:** Confusion matrices for sound, acceleration, COMP and LR, summed over all datasets, with  $T_{ia}=0.3$  and using user-dependent training. All matrix entries are given in seconds(s). For each class,  $c$ , the class relative recall ( $\%R$ ) =  $\frac{\text{correct}_c}{\text{total}_c}$  is shown. The times and percentages of false negative (FN), false positive (FP), substitution (Subst.) and correct true positive (cTP) are also shown, as is the overall correct time (cTP+cTN), or accuracy. This information is represented more clearly, and for all user training setups, in the barcharts of Figure 5.7.



**Figure 5.7:** Graphical summary of confusion matrix data for user-dependent (top), user-independent (middle) and user-adapted (bottom) cases. Totals of the substitution, false negative (FN) and false positive (FP) error times are given as percentages of the total dataset time, together with true negative (TN) and correct positive times (cTP). Total count of NULL time in dataset is 46%. Note the extremely low substitution count and better overall performance of COMP. This comes at the expense of a slightly larger FN than the other fusion methods.

Class (s)	Sound		Accel.		COMP		LR	
	%R	%P	%R	%P	%R	%P	%R	%P
hammer (196)	92	74	93	79	92	94	92	93
saw (306)	90	87	90	80	88	95	93	90
file (305)	77	80	80	82	65	94	82	90
drill (242)	95	54	99	41	95	64	96	59
sand (313)	82	67	87	92	77	93	83	94
grind (278)	83	69	63	66	62	80	75	73
screwd.(260)	52	20	53	87	51	86	53	81
vise (678)	65	55	74	49	61	69	73	53
drawer (659)	86	47	88	39	69	51	87	39
Pos.Average%	76	62	76	68	73	79	78	74
NULL(2778)	33	69	33	69	69	67	42	69

(a) User-dependent

Class (s)	Sound		Accel.		COMP		LR	
	%R	%P	%R	%P	%R	%P	%R	%P
hammer (196)	83	66	76	59	67	93	84	77
saw (306)	71	75	53	51	36	84	78	77
file (305)	29	46	19	39	7	34	23	46
drill (242)	91	47	99	28	92	62	93	62
sand (313)	48	35	51	66	31	89	50	67
grind (278)	72	57	26	45	19	74	82	66
screwd.(260)	46	14	50	86	48	86	50	79
vise (678)	55	54	71	38	47	79	71	62
drawer (659)	81	46	72	38	54	53	89	37
Pos.Average%	61	51	55	52	48	71	66	63
NULL(2778)	33	68	33	68	79	56	42	62

(b) User-independent

Class (s)	Sound		Accel.		COMP		LR	
	%R	%P	%R	%P	%R	%P	%R	%P
hammer (196)	85	62	92	81	85	94	91	83
saw (306)	78	79	61	81	49	97	85	88
file (305)	48	52	58	66	23	88	49	89
drill (242)	94	56	99	46	94	64	94	64
sand (313)	49	42	85	75	43	93	85	76
grind (278)	78	64	82	54	78	72	82	70
screwd.(260)	49	18	51	87	51	87	51	80
vise (678)	65	56	74	56	61	80	74	65
drawer (659)	84	47	89	38	68	52	92	37
Pos.Average%	67	55	73	65	63	79	75	72
NULL(2778)	35	69	35	69	72	61	43	68

(c) User-adapted

**Table 5.3:** Continuous % recall( $R$ ) and precision( $P$ ) for each positive class, and the average of these; also given are the  $R$  &  $P$  values for NULL ( $T_{ia} = 0.3$ ,  $s$  = total time in seconds) Note the best case for positive overall precision and recall is found with LR

included as a special ‘class’. Although the terms recall and precision are used here for *NULL*, the recall of *NULL* is more accurately referred to as the system *specificity* =  $\frac{TN}{TN+FP}$ . Likewise, precision of *NULL* is more accurately referred to as the *negative prediction value* (NPV) =  $\frac{TN}{TN+FN}$ .

### 5.5.5. Analysis of frame-by-frame results

Based on the results in Table 5.3 and in 5.7 the following observations can be made:

- Recognition performance is improved by fusion. For almost all classes the results using fusion are better than those where only a sound or acceleration classifier is used. One exception is with screwdriving, where performance is slightly lower than can be achieved by acceleration alone. An explanation for this is the influence of extremely poor sound performance for this class.

Of particular note is the ability of the fusion methods to reduce substitution errors. Figure 5.7 clearly shows an overall substitution reduction from around 3.7% of the total time in the acceleration classifier to as low as 1.5% for LR, and even 0.2% for COMP (in the user dependent study).

- User independence. When using LR with user independent training, machine tools such as drill, grinder, vise and drawer can be recognised fairly well. With handheld tools, such as saw and hammer, recognition performance drops by roughly 10%. Filing and sanding recognition performs even worse. This is almost certainly due to the greater variety of ways these activities can be carried out, particularly between different users.
- Performance of *NULL*. As the system has been tailored for recognition of positive events, it is not surprising that *NULL*, when treated as a class in its own right, is recognised poorly (e.g. 69/42 P/R for LR in Table 5.3 (a)). A compromise result would be to use COMP, which returns a P/R of 69/67. The advantage of COMP is fewer false positives (FP), at the expense of more false negatives (FN). This is particularly evident when considering the user-independent case where a low recall rate for positive classes is paired with the highest precision of all the methods. It also has the highest recall of *NULL* (specificity) at 79%.

### 5.5.6. Event results

For many applications frame-by-frame performance is of little significance. Of more interest is the detection of events. These take place on a time scale of at least several seconds or hundreds of frames. For instance, when referring

to “hammering,” we consider the the entire hammering sequence and not just each individual stroke. The corresponding definition of an event is a continuous time segment throughout which the system has returned the same classification.

Evaluation of event performance is similar to the strategy used in speech and character recognition. Importance is placed on the ordering of letters and words, rather than the specific time they are uttered. Table 5.4 presents event based results using the standard metrics of insertion and deletion. We reduce each evaluation to a two-class problem, i.e. one class against all others combined. Thus any predicted instance of a class that does not overlap with a same class event in the ground truth is marked as an insertion. Any ground truth instance of a class that has no corresponding prediction of that same class is marked as a deletion.

### 5.5.7. Analysis of event results

Table 5.4 helps confirm many of the observations from the earlier frame-by-frame analysis. Across all user training cases, fusion drastically reduces the number of insertions for most classes. For user-independent, the low recall and high precision of COMP is confirmed with a high number of deletions - in worst case, filing with 17 deletions out of 20 events - but with few insertions. For fewer deletions, the LR method is a better choice.

### 5.5.8. Combined time and event evaluation

There is some information which the two tables, 5.3 and 5.4 fail to capture. As an example the sanding activity in (a), has a recall of 83% (an error of 17% existing class time) yet only one deletion ( $1/20=5\%$  of existing class events). Is this because the deleted event is longer than the others, or is it because the other sanding events do not completely cover their ground truth? The answer is generally a bit of both. In this case, most of the blame lies with the later cause. Such mismatches in event timing constitute a considerable portion of the total frame by frame errors in the experiments described in this paper. We have also found them to be common in other similar work [89, 90]. As a consequence we conclude our results presentation with a closer look at timing issues.

We first solidify the notion of timing errors through the concepts of Overfill and Underfill:

- Overfill (t) - FP frames forming part of correct event which strayed over its segment borders.
- Underfill (t) - FN frames left when correct event does not completely cover its borders

Class (T)	Sound		Accel.		COMP		LR	
	I	D	I	D	I	D	I	D
hammer (20)	33	0	18	0	0	0	2	0
saw (20)	17	0	15	0	1	0	7	0
file (20)	20	0	17	1	2	1	8	0
drill (20)	40	0	83	0	2	0	8	0
sand (20)	62	1	2	1	0	2	0	1
grind (20)	13	0	24	5	2	6	9	2
screwd. (20)	293	4	8	3	8	4	14	3
vise(160)	131	18	168	15	38	35	146	15
drawer (440)	47	8	86	31	14	110	85	30
NULL (740)	33	299	33	299	35	86	41	242

(a) User-dependent

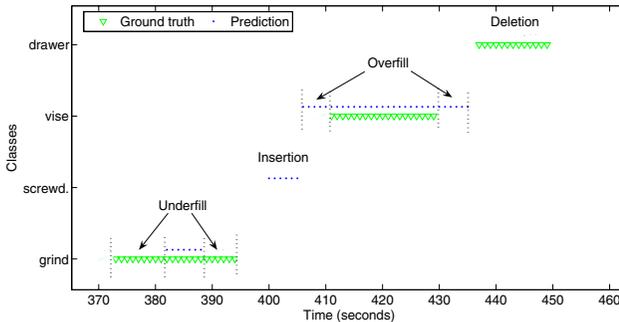
Class (T)	Sound		Accel.		COMP		LR	
	I	D	I	D	I	D	I	D
hammer (20)	46	0	44	2	0	3	20	1
saw (20)	32	0	25	7	4	11	15	0
file (20)	27	9	13	14	4	17	19	14
drill (20)	71	0	175	0	3	0	5	0
sand (20)	76	7	25	7	2	11	20	8
grind (20)	42	1	39	12	3	14	21	1
screwd. (20)	322	3	5	3	5	3	12	3
vise(160)	132	32	223	20	6	55	80	19
drawer (440)	57	22	105	90	13	169	123	16
NULL (740)	32	311	32	311	31	27	50	232

(b) User-independent

Class (T)	Sound		Accel.		COMP		LR	
	I	D	I	D	I	D	I	D
hammer (20)	59	0	17	0	0	1	15	0
saw (20)	26	0	7	6	0	8	7	1
file (20)	25	4	17	4	3	10	4	5
drill (20)	34	0	79	0	1	0	1	0
sand (20)	70	6	15	2	0	7	11	2
grind (20)	28	1	58	1	2	1	7	1
screwd. (20)	285	3	4	3	4	3	11	3
vise(160)	126	26	101	19	3	44	68	19
drawer (440)	51	15	93	22	17	111	121	8
NULL (740)	33	291	33	291	38	43	43	229

(c) User-adapted

**Table 5.4:** Class relative event errors for each class:  $T = Total$ ,  $I = Insertions$ ,  $D = Deletions$ . Note the generally low deletion count of LR versus the low insertion count of COMP



**Figure 5.8:** *Examples of Underfill, Insertion, Overfill and Deletion errors. Note that ground truth is only shown for the four classes (grind, screwd., vise and drawer). NULL, though not marked, makes up the remainder of the time shown.*

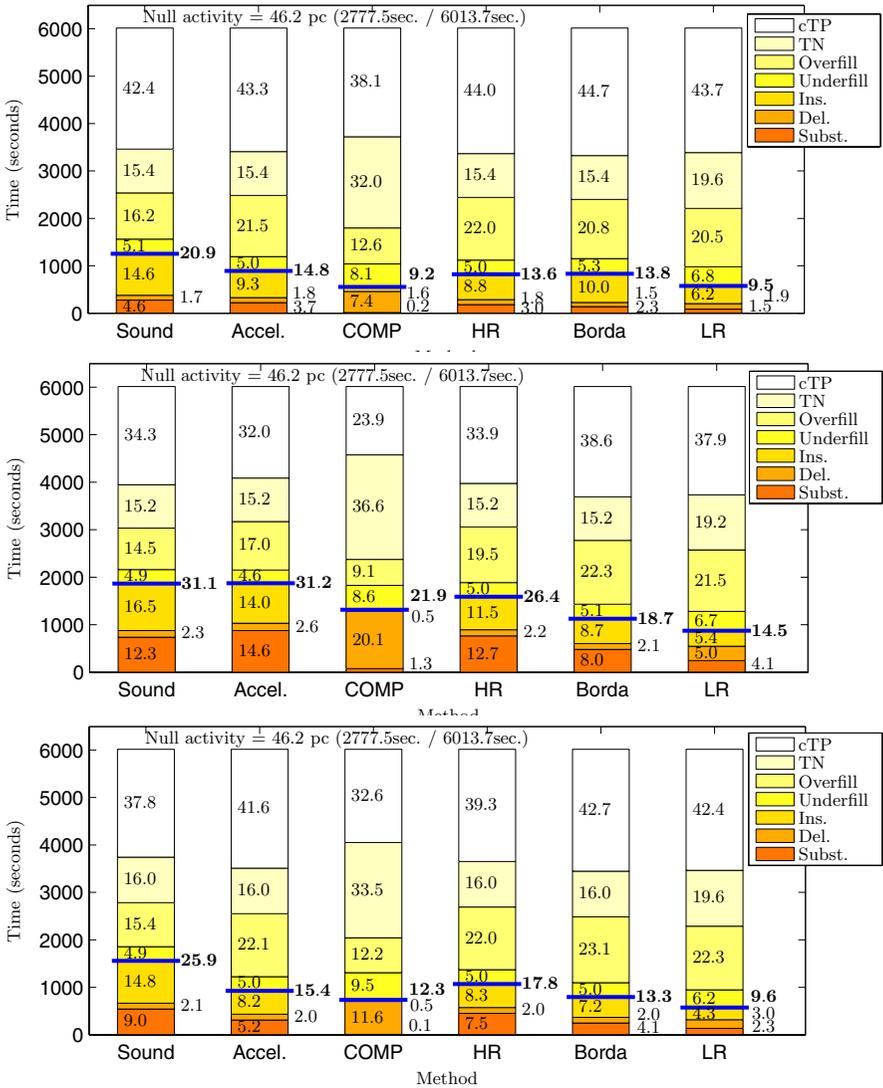
Examples of these are illustrated in Figure 5.8. We use the above definitions to recalculate the evaluation presented in Figure 5.7. This leads to some frames previously considered false positive to become Overfill. Similarly some FN frames are reevaluated as Underfill. Note that substitutions, correct positive and TN frames are not affected. Thus the recalculation essentially subdivides FP and FN time into 'timing error' components, which have no influence on event recognition, and 'serious error' components, which do.

Figure 5.9 shows the results of such a recalculation. Here *serious error level (SEL)* is denoted by a thick line. This includes substitution time in addition to the serious error components of FP and FN. Errors below the SEL would be considered part of an error for an event-based recognition system. Errors above this line are timing errors and would be of more concern to a frame-by-frame recognition system. This analysis can be considered a combined time and event evaluation.

### 5.5.9. Analysis of combined time and event evaluation

The decision as to which method is 'best' depends on the application to which it is aimed. The presented visualisation of results merely provides a greater depth of information which in turn allows the application designer to then make this choice. From these results, however, the following observations can be made:

1. The correct positive entry indicates the amount of time that a correct activity was recognised. Similarly, TN indicates the percentage of time where the system correctly recognised that no activity was happening.



**Figure 5.9:** Continuous results with respect to total time: correct positive (cTP), true negative (TN), Overfill, Underfill, Insertion time (Ins.), Deletion time (Del.) and substitution time (Subst.) for the user-dependent (top), user-independent (middle) and user-adapted (bottom) cases. Serious error level (SEL) is marked by horizontal bar. Note the more consistently good SEL performance of the LR method compared to the other fusion methods

The sum of these two percentages indicate the standard frame-by-frame accuracy of the system.

At a glance we see that the recognition system is not suitable for tasks requiring a high degree of frame accuracy. However, if our goal was a time-critical recognition system, COMP provides the best performance, with 70.1% (38.1%+32.0%), 60.5% (23.9%+36.6%), and 66.1% (32.6%+33.5%) accuracy for the user-dependent, user-independent, and user-adapted cases, respectively.

2. Looking at the charts, we see that 46% of the frames had no activity. The size of the *NULL* class is important in judging the performance of a system. In many continuous recognition tasks, over 90% of the time may be *NULL*. Thus, the TN portion of the column provides an implicit understanding of the type of problem being addressed. With high TN as a criteria, COMP would again be the top choice.
3. The underfill and overfill portions of the column provide an intuition of how “crisp” the recognition method is at determining activity boundaries. High levels of overfill and underfill indicate that the recognition system has difficulty determining the beginning and end of an activity, or that it breaks an activity up into smaller fragments. Thus for timing sensitive tasks a researcher might once again choose COMP to minimise these errors.
4. The substitution, deletion, and insertion portions of the columns represent “serious errors” where the activity is completely mis-recognised. Ideally, these errors should be minimised for a system intended to recognise activities as discrete events. The best performance in minimising such errors - particularly in the user independent and adapted cases - is achieved by the logistic regression (LR) method (9.5%, 14.5% and 9.6% for the cases, respectively). In the user dependent case, COMP performs slightly better on this score (9.2%), however unlike LR, this method does not respond well to changes in the training setups.
5. Some tasks require a more detailed analysis of the “serious errors”. If the goal is to minimise substitution and insertion errors, according to the charts of Figure 5.9, COMP would be the most suited. If, on the other hand, it is more critical not to miss important events, keeping deletions to a minimum, one of the ranking fusion methods would be more appropriate.

## 5.6. Lessons learnt

This chapter presented a method of recognising activities in a wood workshop using a heterogeneous distributed on-body sensor network consisting of microphones and accelerometers. To conclude, we summarise the lessons learnt.

### Using two body worn microphones to segment continuous activities

Provided that the activities of interest are associated with a sound produced closer to the hand than to the upper arm, the strategy of using intensity differences between two separately placed microphones works relatively well for the detection of the activities. However, the strategy tends to produce short, fragmented segments. Smoothing is required to segment the data into useful events of 1-2 seconds in length. In this experiment, a successful approach classified the sound data individually in each 100ms frame using LDA and smoothed the results with a larger majority decision sliding window of 1 second. The sensitivity (recall) of this segmentation can be adjusted using a threshold on the intensity ratio difference  $T_{ia}$ . Further classification using separate sound and accelerometer based classifiers can then be performed on the discovered segments. The performance of these classifiers is affected directly by the setting of  $T_{ia}$ , and the classifiers can be tailored for specific application requirements by adjusting this parameter. For the experiments described here, this value was fixed so as to maximise the performance of positive class recognition.

**Classifier fusion using sound and acceleration** Hand activities involving both a motion and complementary sound component are well recognised by fusion of the individual classifiers. For the assembly scenario investigated, the following was found:

- A simple fusion method based on comparison of outputs (COMP) provides a ‘cautious’ recognition, preferring low instances of falsely recognised activities, and almost no substitution errors (0.2% for user-dependent, to 1.3% for user-independent), at the expense of more deletions than either of the constituent classifiers.
- More advanced fusion methods, based on a combination of class rankings (Borda & HR), are better at detecting all positive activities, at the expense of insertions.
- The logistic regression (LR) fusion method provides a compromise in performance. This method can be trained to identify common combinations, and it produces a *NULL* result in the event of unlikely combinations. LR results in fewer insertions than Borda & HR and fewer

deletions than COMP. In terms of recall and precision over positive activities, LR gives the best overall performance, ranging from 78% recall and 74% precision for the user dependent case and 66% recall and 63% precision for the user-independent case.

Note: by altering  $T_{ia}$ , the exact ratio of insertions to deletions can be adjusted according to application requirements, but in general the above holds for any fixed  $T_{ia}$  (see Figure 5.6).

**Recognition robustness across different users** In user independent testing, the individual audio and gesture classifiers performed poorly compared to the user dependent scenario. However, the fused classifiers - particularly those based on class ranking - experienced only a slight drop in performance when switching from user dependent to independent. Activities that allow little variation, such as the use of machine tools or tools affixed to the bench, are barely affected by changes in user. Other activities such as the use of sandpaper, or file, allow more variation between users and consequently perform less well in user independent testing.

### 5.6.1. Concluding remarks

During the course of evaluating these methods, it was felt necessary to introduce two new performance measures, overfill and underfill, to capture information which existing measures fail to deal with in a suitable way for this problem domain. The problems which prompted the introduction of these measures, among others, shall be investigated in greater detail later in Chapter 7.

Firstly though, Chapter 6 concludes the work on recognition methods with a look at applying some of the methods applied here to a system using only wrist-worn sensors.

# 6

## Recognition using wrist-worn sensors<sup>1</sup>

*The recognition strategy presented thus far suffers from two limitations: (1) it requires sensor placement on two separate body locations; and (2) it uses algorithms that might be regarded as computationally expensive, particularly for implementation on a wearable device.*

*This chapter rounds off the work on continuous activity recognition by evaluating the feasibility of a low power, wrist-worn version of the microphone and accelerometer system. Specifically, it introduces a method of segmentation based on a simple sliding window. This relies on classifier fusion - again using COMP and LR - to distinguish activities from NULL. The sound classifier, based on LDA, is adapted here for a lower frequency operation (from 40Hz down to 2Hz). In addition, the acceleration classification, previously using HMM, is developed here to work with the much simpler, single state, Naive Bayes (NB).*

---

<sup>1</sup>This chapter is based, in part, on work presented in [91]

## 6.1. Introduction

If a body worn activity recognition system is to be genuinely useful, it must be physically unobtrusive to the person wearing it. This is particularly important in a maintenance or assembly scenario, where workers - and particularly their hands - are constantly moving. The hand and arm based segmentation scheme presented in Chapter 4, which was the cornerstone of the continuous recognition work in Chapter 5, would only be feasible in a real scenario if the sensors were small enough to be integrated into the subject's clothing. Even then, it relies on the subject wearing long-sleeved clothing - not always desirable in a working environment.

A system based on a single wrist-worn device would be more acceptable. That is, provided the implementation was no bigger than that of a standard wristwatch. With such a form factor however, implementation issues such as on-device processing come in to play. With limited room for a battery, low power consumption becomes extremely important.

The algorithms that were used previously - Linear Discriminant Analysis (LDA) for sound, and Hidden Markov Models (HMM) for acceleration - can be computationally expensive. For a wrist-worn microprocessor, this translates into a need for more power.

This chapter presents a solution to both of these issues - wearability and computation complexity. It tackles the first by using sound and accelerometer data recorded only from a wrist worn location. Because the 2-mic sound segmentation scheme can no longer be applied, a method using a simple sliding window is used instead. This relies more heavily on the findings of Chapter 5 that *NULL* segments can be identified through fusion of classifiers even though the individual classifiers can not. The fusion methods used are comparison of top ranks (COMP) and logistic regression (LR), as introduced in the previous chapter.

The second issue, computation complexity, is tackled by replacing the HMM with an Naive Bayes (NB) classifier and reducing the LDA calculation frequency from the original 40Hz to 2Hz.

## 6.2. Recognition methods

Recognition is carried out by fusing the output of two classifiers (sound and acceleration) over a fixed-width sliding window. Two classifier strategies were implemented: the first based on the LDA and HMM methods used in previous chapters; and the second, with an aim to reducing computation requirements, based on a lower frequency LDA and a Naive Bayes (NB) classifier.

The fusion methods used for both these approaches are: comparison of top results (COMP) and logistic regression (LR), as introduced in Chapter 5.

### 6.2.1. Sliding window segmentation

A window of length  $w_{len}$  is moved across the data in increments of  $w_{jmp}$ . This window is treated as a segment upon which the following algorithms are applied.

### 6.2.2. Sound recognition using LDA

As recommended in Chapter 3, LDA classification is applied to 100ms windows, or *frames*. For each frame an FFT is applied, producing a 100 bin vector. This vector is then reduced to 8 dimensions ( $\#Classes - 1$ ) by multiplication with the LDA transform matrix. The LDA vector is then compared to its class means (obtained from training) to produce a list of class distances. All of this is done for each frame in 25ms increments.

To produce a single result for the  $w_{len}$  segment, the constituent LDA distance vectors must be combined. This is done by taking the mean of the LDA distance vectors, for each class, over  $w_{len}$ <sup>2</sup>. The resulting vector of class distances is then sorted, shortest distance first, for conversion to class rankings.

### LDA at reduced rate

Most of the activities in the wood workshop scenario produce sounds that are *quasi stationary* for the timescales of those activities (see Chapter 2). Specifically, they keep a constant sound signature for times of one second or more. Even a sound which does not last long, such as that made by the fall of a hammer, is usually repeated often enough that most of its short duration ‘hits’ may still be detected by low frequency sampling (one or two samples per second, for example).

Running an LDA calculation every 25ms, a rate of  $40\text{Hz}$ , might therefore be regarded as a computational overhead. A reduction of this rate is desirable, provided the recognition performance does not suffer too much.

A variety of different LDA classification rates  $R_{LDA}$ , from 0.25Hz to 40Hz, were evaluated. The results of this trial are given in section 6.4.3.

### 6.2.3. Acceleration recognition

To simplify the feature extraction stage, the short feature windows that were used in Chapter 3 are replaced here with longer, 1 second windows. The continuous features which are used are:

- a count of the number of peaks in x, y and z over a 1s window

---

<sup>2</sup>Earlier published work used a sum of distances [91]. Whether mean or sum is used, the difference, when applied here, is minimal.

- mean amplitude of peaks in x, y and z over a 1 s window
- mean and variance of x over a 1 s window
- raw x data

One of the reasons for using a 1 second window is to remove the need to work with 100Hz accelerometer data (required previously when 100ms windows were used). Here we use only data sampled at 40Hz.

Another reason is that a one second window roughly reflects the shortest activity period of interest. Features such as mean and variance of the  $x$ -axis data, when calculated over this window, help characterise whole activities rather than just their parts. Whereas this is not a major concern for a sequencing classifier, such as HMM, it becomes important if a single state classifier, such as NB, is to be used.

### HMM classification

The acceleration data is classified using the HMMs (Hidden Markov Models) introduced in Chapter 3. For each segment, the HMMs are run over the 40Hz feature data. These produce class likelihoods, from which rankings for fusion can be obtained.

### NB classification

Unlike the HMM, which can have multiple states and several Gaussians for each feature, NB has a single state and models every feature with a single Gaussian.

The input features, sampled at 40Hz, are first combined over the duration of the segment ( $w_{len}$ ). This is done by simply taking a mean value for each feature. The NB likelihood for each class is then calculated using these combined features. Thus, NB is only applied once for each segment.

#### 6.2.4. Fusion of classifiers: COMP and LR

The two fusion methods - COMP and LR - are applied to the classifier outputs for each segment. These methods were introduced in Chapter 5. The other ranking fusion methods introduced in that chapter - Borda count and highest rank (HR) - are not used here because, unlike LR, their basic implementations do not allow assignment of *NULL*. An overview of COMP and LR, with details on changes made to the *NULL* thresholding for LR, is given below.

## COMP

COMP simply compares the two classifier rankings and returns only those classes which are assigned top rank by both. If the classifiers do not agree on their top ranks, *NULL* is returned.

## LR

LR (logistic regression) uses a linear function to estimate the likelihood of a class for a given set of rankings. For each class  $c$ , the ranking pair  $X$ , as output by the sound and acceleration classifiers, is fed into the likelihood function  $L_c(X)$  (defined in 5.2.6). The combined ranking is then obtained by sorting the different class likelihoods.

The problem here is that none of the input classifiers are able to give rankings for *NULL*. In Chapter 5 this problem was overcome by introducing thresholds  $L_c^N(X)$ . Thus *NULL* is introduced by giving it a higher rank than those classes which have fallen below their threshold.

As with  $L_c(X)$ , the coefficients of the thresholds are obtained using isolated classifier results from the training set. The results used are those ranking combinations that, if chosen, produce false classifications.

In practice, this method works well provided the training data includes enough instances of typical combinations. In previous work, it was found that the training for some classes did not produce thresholds that were strong enough. This is indicated by the large number of false positives produced by those classes - in particular, vise and drawer. Looking back to Table 5.2, this can be seen by comparing the entries for vise and drawer in the LR matrix with those in the COMP matrix.

In this work, therefore, the thresholds were tightened manually so that vise and drawer are only chosen if *both* classifiers assign them top rank.

## 6.3. Experiment

Data from the five-subject wood workshop scenario, as introduced in Chapter 2, is used to evaluate the methods. Because only wrist information is of use, the upper arm data is simply discarded.

### 6.3.1. Setting parameters

The system was initially evaluated across sweeps of the two segmentation parameters, window length  $w_{len}$  and jump length  $w_{jmp}$ . From these sweeps, setting both  $w_{len}$  and  $w_{jmp}$  to 2 seconds was found to produce favourable results. Intuitively, the suitability of such a large window stems from the fact that all activities of interest in these experiments occur at a timescale

of at least several seconds. All further analysis was carried out with these parameters set.

The LDA, HMM and NB methods all require training of parameters using data. This was carried out in a user-dependent, leave-one-out fashion. This is where, for each user, one set is put aside for testing while the remaining sets (from the same user) are used for training.

The LR coefficients are also trained using leave-one-out. The training data for this is obtained from the results of isolation tests on the constituent classifiers.

## 6.4. Results

### 6.4.1. Preliminary study (using 40Hz LDA and HMM)

For each successive 2 second segment the HMM was run on the accelerometer data and LDA minimum distance on the audio. This was done for all 20 sets of data. Typical results from one of these sets is plotted in Figure 6.1, with classifier predictions compared alongside the hand-labelled ground truth.

Lacking any ability to distinguish valid activities from *NULL*, the constituent classifiers, as expected, produce much noise. With LDA tending to treat *NULL* as a quiet class, such as screwdriving; and HMM usually giving random misclassifications. For the non-*NULL* classes, however, both classifiers seem to perform well.

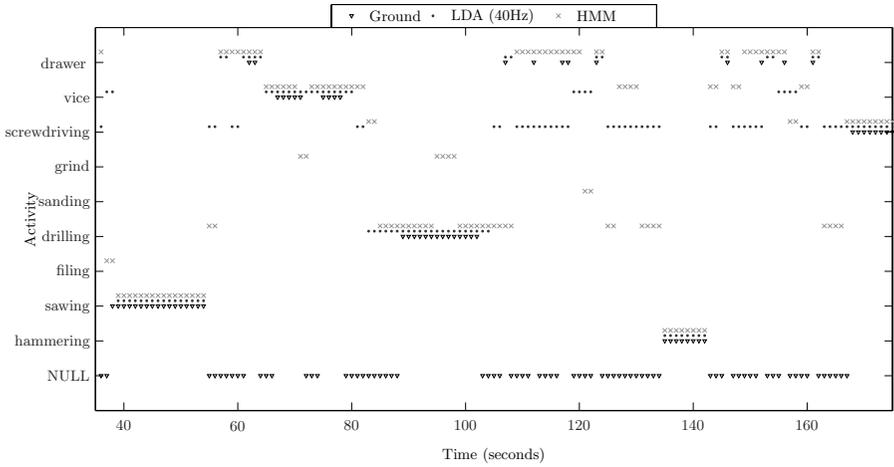
Using the output rankings of these classifiers, the COMP and LR fusion methods were then evaluated.

On first run, the LR method was found to produce a large number of insertions - primarily from the class 'screwdriving'. This is possibly because screwdriving is a comparatively 'silent' class, and as the training data consists mostly of noisy, positive class examples (at no stage do we use *NULL* labelled data for training), it winds up becoming a 'catch all' class for *NULL*. This problem was easily solved by raising the threshold  $L^N(X)$  for screwdriving so that only classifier agreement is allowed (as is already done for vise and drawer).

As can be seen in Figure 6.2, the situation is dramatically improved when COMP and LR are applied. These not only introduce *NULL* 'classifications', but also help eliminate substitution errors.

### 6.4.2. Frame-by-frame results (using 40Hz LDA and HMM)

The confusion matrices for the HMM & 40Hz LDA based methods, summed across all test datasets, are shown in Table 6.1. Recognition rate (recall), indicating how well the system returns true frames, is given at the end of each class row. To the bottom of each matrix, a smaller table shows summary



**Figure 6.1:** Plot of a typical classifier output sequence showing the acceleration predictions (using HMM), the sound predictions (using 40Hz LDA) and the ground truth. Each marker on these plots represents a single 2 second window. Note the inability of either method to detect NULL

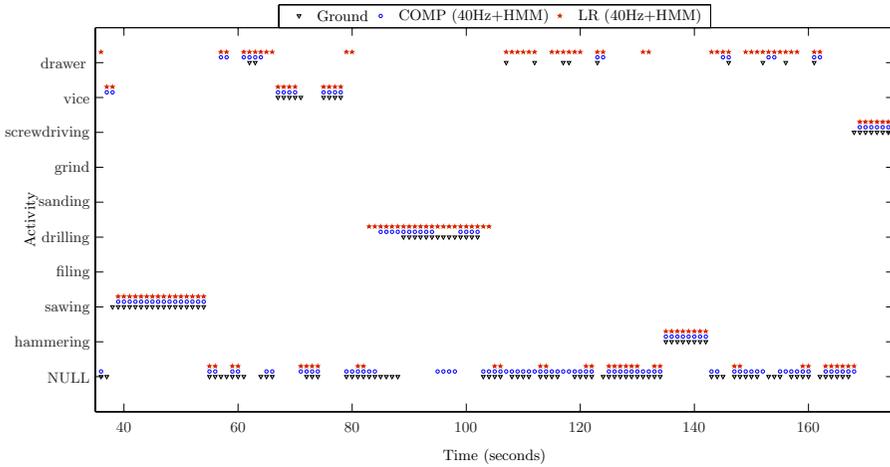
measures as both absolute times, and as percentages of the total experiment time. Specifically, these are the amount of false positive (FP), false negative (FN), substitution (Subst.), correct true positive (cTP) and total correct (or, as a percentage, accuracy) times for the experiment.

The first thing that might be noted from these matrices is that both fusion steps manage to increase the ‘recall’ of *NULL* from 0% in the original classifiers, to as much as 65% (for COMP).

Fusion also helps to reduce the amount of substitution errors. Where sound produces 8.6%, and acceleration 10.1%, LR manages to bring the substitutions down to 1.5%. COMP, more impressively, returns only 0.5% substitutions.

Overall, the accuracies of 68% for COMP and 72% for LR seem reasonable. However, as discussed in Chapter 5, frame-by-frame confusion matrix evaluation does not account for ‘fuzzy’ boundaries - a common phenomena in activity recognition. To combat this, *overflow* and *underfill*, are used. To recapitulate, the definitions of these are:

- overflow: when a continuous sequence of correct prediction frames slip over the ground truth boundary to cover *NULL*;
- underfill: the time left when a continuous sequence of correct prediction



**Figure 6.2:** Plot of the COMP and LR fusion predictions versus ground truth. The same typical example and input classifiers from Figure 6.1 are used here.

frames does not completely cover the corresponding ground truth.

The introduction of overfill and underfill means that the confusion matrix errors of FP and FN are now subdivided into four: FP is divided into the less serious error of overfill time and the more serious *insertion time* (Ins); FN is divided into underfill time and *deletion time* (Del).

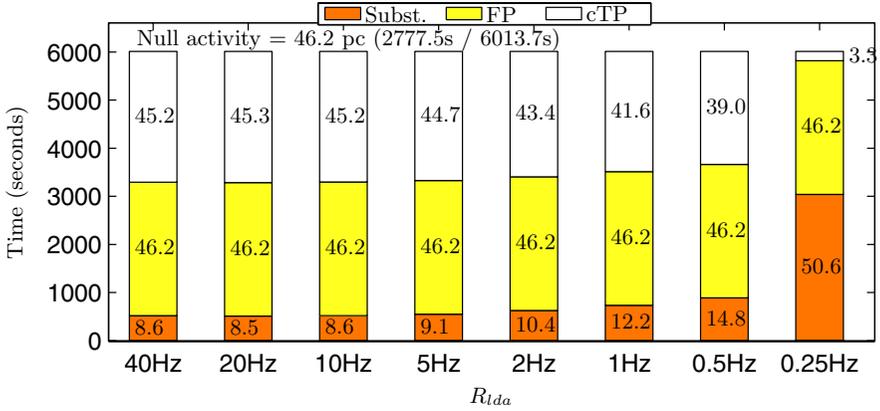
The results using these are summarised in the visualisation of Figure 6.4. This shows the total overfill and underfill, together with Subst., Del, Ins, cTP and TN, as percentages of the overall experiment time.

If using confusion matrix based error rate, defined as  $100\% - \textit{Accuracy}$ , LR would have a rate of  $100 - 72 = 28\%$ . With the amount of deletion time taking up only 3.2% of the LR result, and insertions 2.7%, the ‘serious’ errors of this method seem much less than the confusion matrix analysis would suggest. In fact, together with substitutions, the total amount of these errors amounts to ‘only’ 7.3%. COMP scores similarly, with a serious error level of only 6.2% (compared to its 32% error rate using the confusion matrix).

### 6.4.3. Selecting a lower frequency LDA

Before showing the results from the alternative algorithms, we first investigate a suitable classification frequency for the low power LDA.

A sweep of possible LDA classification frequencies was carried out. Specifically, values of  $R_{lda}$  were tried between 40Hz and 0.25Hz. From each of these,



**Figure 6.3:** Classifier results for LDA across a range of classification frequencies  $R_{lda}$  (40 times per second down to once every 4 seconds.) Substitution time (Subst.), false positive time (FP) and correct true positive time, are shown for each frequency. Because the classifier cannot detect NULL, the maximum possible correct time is 53.8%. Note how the results degrade quickly below 2Hz.

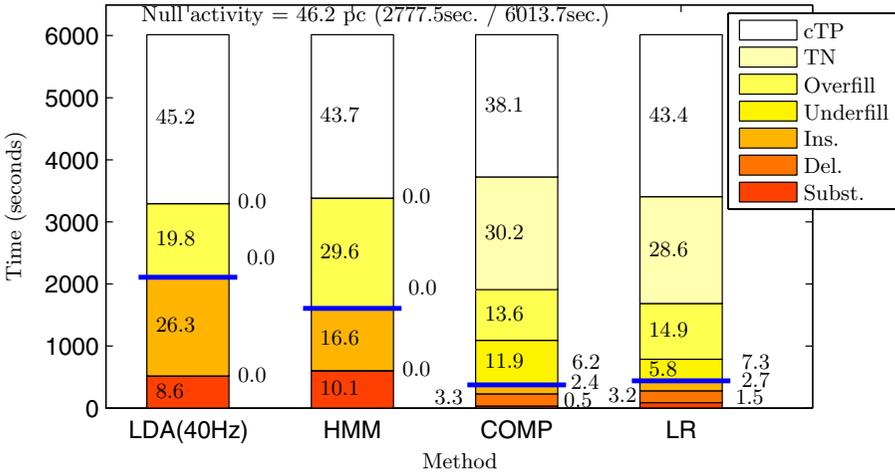
the recognition results - correct positive, false positive and substitution times - were calculated over the entire test dataset. (Because LDA is unable to detect *NULL*, false negative and true negative times are not shown here.) The results are shown in Figure 6.3.

This sweep shows that as the rate  $R_{lda}$  is reduced, the performance degrades, slowly at first, until about 0.5Hz. After that, because of the limitation imposed by the 2 second segmentation window, the classifier is unable to produce any result. Based on this, a suitable low frequency setting with an acceptable level of performance would be 2Hz. This implies a reduction in computational effort by about 20 - with only a 1.8% increase in substitution time. Thus, the 2Hz LDA is used for the remainder of the low power evaluation.

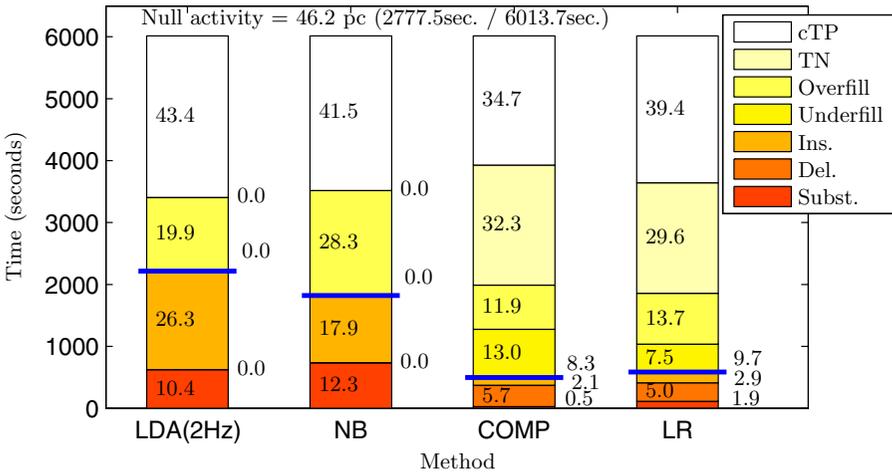
#### 6.4.4. Frame-by-frame results (using 2Hz LDA and NB)

The results from the 2Hz LDA were compiled together with the NB on acceleration. Again, COMP and LR were run on top. The results from this are shown in Figure 6.5. (For completeness, we also show the corresponding confusion matrices in Table 6.2).

As already shown, despite having a classification frequency 20 times lower,



**Figure 6.4:** Breakdown of errors as a percentage of total experiment time for LDA(40Hz), HMM and the COMP and LR fusion methods: correct true positive (cTP), true negative (TN), overfill, underfill, insertion (Ins.), deletion (Del.) and substitution (Subst.) times; also given is the 'serious error' level.



**Figure 6.5:** Breakdown of errors as a percentage of total experiment time for LDA(2Hz), NB and the COMP and LR fusion methods: correct true positive (cTP), true negative (TN), overfill, underfill, insertion (Ins.), deletion (Del.) and substitution (Subst.) times; also given is the 'serious error' level.

the 2Hz LDA produces only 1.8% more substitution errors than its 40Hz variant. Likewise, the NB method produces only 2.2% more substitutions than HMM.

After fusion, these errors fall to almost the same level as those produced when the HMM and 40Hz LDA are used. The only real difference is the increase in deletion errors for the 2Hz LDA based methods (COMP deletions rise by 2.4%, LR by 1.8% ).

The total percentage of ‘serious errors’ for COMP and LR are 8.3% and 9.7% of the total experiment time. If using only the confusion matrix based measures from Table 6.2, then the respective error rates would be 33.1% (COMP) and 30.9% (LR).

#### 6.4.5. Analysis of computation times

As a measure of computation complexity, the time taken for each of the algorithms to run in matlab was recorded and presented in Table 6.3. These recordings were made over the total dataset size of 6013.7 seconds (approximately 1 hour and 40 minutes). Specifically, the absolute time spent for each calculation was summed up and divided by the dataset length. This indicates the average amount of time required for that calculation for every one second of data. The results shown are all given in miliseconds (ms).

Although the measurements given here are subject to both the specifics of the Matlab implementation and the speed of the processor used (Intel Pentium M, 1200 MHz), they do give an indication of the relative times for each of the methods. We can confirm, for example, that the 40Hz LDA is around 17 times more computationally expensive than the 2Hz. (Which is roughly in line with what we expect when the same algorithm is run 20 times more frequently.) We also see that the HMM requires around 14 times more computation time than NB. The most complex of the fusion algorithms, LR, requires almost negligible computation time compared to the classifiers (0.08ms for every 1000ms of data). Overall, the 2Hz + NB method is nearly 15 times less computationally expensive than the 40Hz + LDA.

## 6.5. Discussion

Though the individual recognition performance for each of the two sensor types perform quite poorly on their own, once combined their results improve dramatically. Neither sensor classifier is able to detect *NULL*, but when combined over a sliding 2 second window, they can. Using COMP (with an HMM and 40Hz LDA) as much as 65% of the *NULL* time is retrieved. For LR, around 62% of is retrieved.

Class	Total(s)	hammer	saw	file	drill	sand	grind	screw	vise	drawer	NULL	%R
hammer	195.7	<b>169.5</b>						18.7	6.7	0.8		86.61
saw	306.4		<b>263.2</b>	18.0		4.0		8.0	13.0	0.2		85.90
file	304.6			<b>236.8</b>		33.9	1	18.0	2.5	3.4		77.74
drill	241.5				<b>228.5</b>		1	1.0				94.62
sand	313.0		6.0	15.9		<b>256.0</b>		23.7	0.9	10.5		81.78
grind	277.7					2.9	<b>274.5</b>	0.3				98.86
screw	260.4							<b>249.0</b>	9.8	1.6		95.64
vise	678.1		0.3					101.3	<b>571.6</b>	4.8		84.31
drawer	658.8					0.5		165.7	23.7	<b>468.8</b>		71.17
NULL	2777.5	6.5	8.8	7.3	111.5	12.7	69.5	1372.8	486.7	701.8		0

<b>LDA(40Hz)</b> (sound)	Total: 6013.7	FP: 2777.5 46.2%	FN: 0.0 0.0%	Subst.: 518.1 8.6%	cTP: 2718.1 45.2%	cTP+TN: 2718.1 Accuracy: <b>45.2%</b>
-----------------------------	---------------	---------------------	-----------------	-----------------------	----------------------	--

Class	Total(s)	hammer	saw	file	drill	sand	grind	screw	vise	drawer	NULL	%R
hammer	195.7	<b>157.8</b>				1	4.5	2.0	18.4	3.0		80.64
saw	306.4	2.4	<b>205.3</b>	52.3	2.2	23.6	0.3		20.3			67.01
file	304.6		41.1	<b>227.6</b>		19.1			16.2	0.7		74.72
drill	241.5				<b>182.4</b>	4.0	34.2	0.8	11.6	8.6		75.53
sand	313.0	10.5	27.1	27.0	0.6	<b>206.2</b>		3.8	36.8	1.1		65.86
grind	277.7		4.2		4.4		<b>255.7</b>		10.8	2.5		92.08
screw	260.4		16.9		0.7		0.2	<b>231.6</b>	1.9	9.0		88.96
vise	678.1		63.4		1.2	7.7	8.1	5.3	<b>578.6</b>	13.7		85.33
drawer	658.8		11.8		4.1	5.4	9.5	42.3	<b>585.7</b>			88.90
NULL	2777.5	221.0	8.5	19.1	196.3	7.5	160.3	119.0	750.1	1295.7		0

<b>HMM</b> (acceleration)	Total: 6013.7	FP: 2777.5 46.2%	FN: 0.0 0.0%	Subst.: 605.3 10.1%	cTP: 2630.9 43.7%	cTP+TN: 2630.9 Accuracy: <b>43.7%</b>
------------------------------	---------------	---------------------	-----------------	------------------------	----------------------	--

Class	Total(s)	hammer	saw	file	drill	sand	grind	screw	vise	drawer	NULL	%R
hammer	195.7	<b>141.5</b>									54.2	72.32
saw	306.4		<b>192.7</b>	8.0					3.4		102.3	62.88
file	304.6			<b>186.5</b>		2.0					116.1	61.21
drill	241.5				<b>172.4</b>						69.1	71.39
sand	313.0		4.0	2.0		<b>172.6</b>		1.8			132.6	55.13
grind	277.7						<b>255.0</b>				22.7	91.81
screw	260.4							<b>227.6</b>	0.1	1.6	31.1	87.42
vise	678.1						2.0	<b>498.7</b>	1.5	175.9		73.55
drawer	658.8						2.2	4.4	<b>441.9</b>	210.3		67.08
NULL	2777.5	4.5	1.3	3.5	83.6	1.4	39.0	88.4	209.1	531.0	<b>1815.7</b>	65.37

<b>COMP</b> (40Hz & HMM)	Total: 6013.7	FP: 961.8 16.0%	FN: 914.3 15.2%	Subst.: 33.0 0.5%	cTP: 2288.9 38.1%	cTP+TN: 4104.6 Accuracy: <b>68.3%</b>
-----------------------------	---------------	--------------------	--------------------	----------------------	----------------------	--

Class	Total(s)	hammer	saw	file	drill	sand	grind	screw	vise	drawer	NULL	%R
hammer	195.7	<b>177.9</b>							0.6		17.2	90.91
saw	306.4		<b>269.4</b>	14.0		2.0			2.6		18.4	87.94
file	304.6		1.4	<b>237.6</b>		29.6					35.9	78.01
drill	241.5				<b>232.3</b>	2.0	4.2				3.0	96.19
sand	313.0		6.0	10.9		<b>248.9</b>		1.8			45.5	79.50
grind	277.7					2.7	<b>274.7</b>				0.3	98.94
screw	260.4							<b>224.5</b>	0.1	1.6	34.2	86.23
vise	678.1		0.3					2.0	<b>500.7</b>	1.5	173.5	73.85
drawer	658.8							2.2	4.4	<b>441.9</b>	210.2	67.08
NULL	2777.5	14.1	11.1	5.5	117.7	8.8	75.1	85.5	209.3	531.0	<b>1719.5</b>	61.91

<b>LR</b> (40Hz & HMM)	Total: 6013.7	FP: 1058.0 17.6%	FN: 538.1 8.9%	Subst.: 89.9 1.5%	cTP: 2608.1 43.4%	cTP+TN: 4327.6 Accuracy: <b>72.0%</b>
---------------------------	---------------	---------------------	-------------------	----------------------	----------------------	--

**Table 6.1:** Confusion matrices for the 40Hz LDA, HMM, COMP and LR with jumping window of 2 seconds. All times are given in seconds. Class relative recall %R is shown for each class. At the bottom of each matrix, a summary table gives times and percentages of false negative (FN), false positive (FP), substitution (Subst.), correct true positive (cTP), and overall correct (cTP+cTN).

Class	Total(s)	hammer	saw	file	drill	sand	grind	screw	vise	drawer	NULL	%R
hammer	195.7	<b>173.4</b>				0.6		14.3	4.7	2.6		88.61
saw	306.4		<b>242.5</b>	22.0		1		6.2	18.6	7.1		79.13
file	304.6		2.0	<b>231.9</b>		29.4	1	17.2	7.1	7.1		76.12
drill	241.5				<b>226.4</b>	10.1	2.0	1.0		2.0		93.75
sand	313.0		2.0	29.9		<b>241.9</b>		22.1	4.0	13.2		77.28
grind	277.7			2.0		4.6	<b>270.9</b>	0.2				97.54
screw	260.4			4.0		2.0		<b>243.0</b>	9.4			93.33
vise	678.1	2.8	1.5			1.1		123.1	<b>541.8</b>	7.8		79.90
drawer	658.8				0.2	0.8		181.9	38.5	<b>437.4</b>		66.40
NULL	2777.5	12.6	12.0	4.3	101.4	36.4	67.1	1303.7	531.2	708.8		0

<b>LDA(2Hz)</b> (sound)	Total: 6013.7	FP: 2777.5 46.2%	FN: 0.0 0.0%	Subst.: 627.0 10.4%	cTP: 2609.2 43.4%	cTP+TN: 2609.2 Accuracy: <b>43.4%</b>
----------------------------	---------------	---------------------	-----------------	------------------------	----------------------	--

Class	Total(s)	hammer	saw	file	drill	sand	grind	screw	vise	drawer	NULL	%R
hammer	195.7	<b>149.1</b>		2.4	8.6	12.0	3.3	12.0	2.3	6.0		76.22
saw	306.4	1.8	<b>195.6</b>	73.0	2.5	21.6		11.8	0.1			63.85
file	304.6		59.8	<b>227.8</b>	1.3	6.0		7.9	1.8			74.77
drill	241.5				<b>192.5</b>	4.0	34.0		5.0	6.0		79.71
sand	313.0	2.0	27.2	26.8	2.8	<b>230.5</b>			21.9	1.8		73.63
grind	277.7	0.3			38.1		<b>230.5</b>		4.1	4.7		83.00
screw	260.4	15.0			4.8			<b>221.8</b>	1.1	17.7		85.18
vise	678.1	114.4	5.5	9.9	20.5	15.7	4.2	7.7	<b>451.0</b>	49.1		66.51
drawer	658.8	16.4	0.1	0.3	8.0		4.1	14.7	15.8	<b>599.4</b>		90.98
NULL	2777.5	28	21.7	23.9	359.6	6.2	97.9	170.2	348.4	1469.5		0

<b>NB</b> (acceleration)	Total: 6013.7	FP: 2777.5 46.2%	FN: 0.0 0.0%	Subst.: 738.0 12.3%	cTP: 2498.2 41.5%	cTP+TN: 2498.2 Accuracy: <b>41.5%</b>
-----------------------------	---------------	---------------------	-----------------	------------------------	----------------------	--

Class	Total(s)	hammer	saw	file	drill	sand	grind	screw	vise	drawer	NULL	%R
hammer	195.7	<b>141.1</b>									54.6	72.12
saw	306.4		<b>177.2</b>	1					3.0		116.2	57.82
file	304.6			<b>183.0</b>					0.5		121.1	6.6
drill	241.5				<b>178.4</b>						63.1	73.88
sand	313.0		2.0	5.9		<b>185.6</b>			0.9		118.6	59.30
grind	277.7						<b>228.5</b>				49.2	82.28
screw	260.4							<b>211.8</b>			48.6	81.34
vise	678.1							2.8	<b>371.6</b>	1.4	302.2	54.81
drawer	658.8							2.1	0.5	<b>407.7</b>	248.6	61.88
NULL	2777.5	6.9	2.8	1.2	87.6	2.4	37.5	75.3	134.9	488.9	194	69.85

<b>COMP</b> (2Hz & NB)	Total: 6013.7	FP: 837.5 13.9%	FN: 1122.2 18.7%	Subst.: 29.1 0.5%	cTP: 2084.9 34.7%	cTP+TN: 4024.8 Accuracy: <b>66.9%</b>
---------------------------	---------------	--------------------	---------------------	----------------------	----------------------	--

Class	Total(s)	hammer	saw	file	drill	sand	grind	screw	vise	drawer	NULL	%R
hammer	195.7	<b>177.3</b>							0.2		18.2	90.61
saw	306.4		<b>242.6</b>	16.0		6.0			5.5		36.3	79.18
file	304.6		19.4	<b>198.2</b>		13.4			0.5		73.1	65.05
drill	241.5	2.0			<b>228.5</b>		4.0				7.0	94.62
sand	313.0		2.0	18.9		<b>229.8</b>			0.9		61.4	73.42
grind	277.7				0.3		<b>272.9</b>				4.4	98.29
screw	260.4			4.0		2.0		<b>201.8</b>			50.6	77.50
vise	678.1	1.0	3.0			0.8		2.8	<b>413.4</b>	1.4	255.6	60.97
drawer	658.8				0.2			2.1	4.1	<b>407.7</b>	244.8	61.88
NULL	2777.5	19.7	12.9	3.0	122.9	6.7	71.1	75.3	194.8	488.9	1782.1	64.16

<b>LR</b> (2Hz & NB)	Total: 6013.7	FP: 995.4 16.6%	FN: 751.3 12.5%	Subst.: 112.7 1.9%	cTP: 2372.3 39.4%	cTP+TN: 4154.4 Accuracy: <b>69.1%</b>
-------------------------	---------------	--------------------	--------------------	-----------------------	----------------------	--

**Table 6.2:** Confusion matrices for the 2Hz LDA, NB, COMP and LR with jumping window of 2 seconds. All times are given in seconds. Class relative recall %R is shown for each class. At the bottom of each matrix, a summary table gives times and percentages of false negative (FN), false positive (FP), substitution (Subst.), correct true positive (cTP), and overall correct (cTP+cTN).

		40Hz LDA + HMM	2Hz LDA + NB
Sound	<i>FFT</i>	8.07	0.45
	<i>LDA transform</i>	1.34	0.07
	<i>LDA distance calc.</i>	4.46	0.23
	<b>total</b>	<b>13.87</b>	<b>0.84</b>
Accel.	<i>Calc. features</i>	1.92	1.92
	<i>Calc. likelihoods</i>	32.10	0.55
	<b>total</b>	<b>34.02</b>	<b>2.47</b>
LR fusion		0.08	0.08
<b>All methods total</b>		<b>47.97</b>	<b>3.39</b>

**Table 6.3:** Average computation times (in ms) for each method applied to 1 second of data. The total dataset evaluated was approx. 1 hour and 40 minutes long. Computations were done in Matlab on an IBM X40 laptop, (Pentium M, 1200MHz). Note how the 40Hz LDA + HMM based methods require almost 15 times more computation time than the 2Hz LDA + NB.

Using the low power implementation, running LDA at 2Hz and naive Bayes instead of HMM, this ability to detect *NULL* actually seems to improve: COMP returns nearly 70% and LR 64%.

The overall accuracy is, however, slightly lower. The serious positive class errors - substitutions, insertions and deletions - increase for each method by about 2% in the low power study. Specifically, the total error levels are 8.3% for COMP and 9.7% for LR. This is compared to 6.2% for COMP and 7.3% for LR in the 40Hz LDA + HMM study.

### 6.5.1. COMP versus. LR

As with any comparison between recognition systems, it is unwise to make claims as to the absolute superiority of one method over the other - the differences between COMP and LR, for example, should be highlighted in view of whatever performance criteria is most important to the application. The LR for 2Hz LDA+NB has a higher overall accuracy than COMP (67% versus 69%), but at the expense of having more ‘serious errors’, such as insertions, deletions and substitutions. Some applications, time-critical safety monitoring of dangerous activities, for example, might regard a false negative as being much worse than a false positive - regardless of whether it is an underfill or deletion. In which case the LR method, as used here, might be regarded preferable. If an application viewed substitution errors as being particularly critical, however, then COMP would be the method to choose.

The parameters which have for the purposes of this work been set to some ‘optimal’ value, such as the *NULL* thresholds for LR, can alter the

nature of the results by raising or lowering the chance of returning a *NULL*. It is, for example, possible to tailor the LR method to have exactly the same performance as COMP if one raises the threshold  $L^N(X)$  of all classes to just under the  $L(x)$  value for matching top rank classifier results. This was shown here already for the screwdriver, vise and drawer classes. This ability means that although more complex to implement - particularly as it requires training - LR is more versatile in terms of performance optimisation than the basic comparison.

To properly analyse the LR method in greater detail, a sweep of its class thresholds would be required. This might be shown and compared to other methods using ROC or PR curves. Such an analysis is beyond the scope of this chapter and is therefore left for future work.

### 6.5.2. Conclusion

We have shown that it is possible to recognise wood workshop activities using a wrist worn 3-axis accelerometer and a microphone. Recognition is achieved for each sensor using a standard sliding window based approach. Alone, neither sensor can detect a *NULL* gesture, but when fused together, this becomes possible.

The 40Hz LDA and HMM classifiers that were used in previous chapters perform well with with this setup. However, with the view to implementing an on-device, wearable, recognition system, these algorithms might be regarded as computationally expensive. Running the LDA at 2Hz and classifying the acceleration segments using a simple naive Bayes classifier produces comparable results for much less cost. Although requiring almost 15 times less computation than 40Hz LDA+HMM, this approach loses only a couple of percentage points in accuracy.

The highest overall accuracy for the 2Hz LDA, naive Bayes method, as described here, is just over 69% (using LR fusion). By discounting overfill and underfill, the 'serious error level' for this method is 9.7% of the total experiment time. The comparable results using the full 40Hz LDA and HMM is 72% accuracy and serious error level 7.3%.



# 7

## Evaluation and optimization of performance<sup>1</sup>

*Evaluating the performance of a continuous context recognition system can be a challenging problem. To-date there is no widely accepted standard for dealing with this, and methods and measures are usually taken from related fields such as speech and vision.*

*In this chapter we attempt to identify and characterise the errors typical to continuous context recognition. We introduce a means of quantifying these errors in an unambiguous manner. In an initial investigation, we score the errors in an example taken from previous work, and discuss the advantages that the proposed method provides over two of the most commonly used approaches.*

---

<sup>1</sup>This chapter is based on work to appear in [92]

## 7.1. Introduction

In previous chapters, several different methods of recognising continuous activities were presented and compared. To help evaluate these a number of standard measures were used, such as precision and recall, calculated from the timewise, frame-by-frame, comparison of prediction sequences with their corresponding ground truth; and counts of insertions and deletions based on comparisons where the unit of measure is an activity event. Each strategy was chosen with the aim of providing the most relevant and critical information for analysis of the systems being presented. A combination of different evaluation strategies, such as used in the later result sections of Chapter 5, can give a fuller account of the information that might be necessary, both for the designer optimising a system, and for the developer charged with finding the most suitable recognition solution for an application.

Unfortunately, as also highlighted in Chapter 5, existing strategies of performance evaluation fail to capture - or obscure - some of the information that might be useful for the designer of an activity (or context) recognition system. One aspect of a typical continuous context recognition system which existing evaluation strategies fail to address is the problem associated with imprecise, variable duration event boundaries. This was initially tackled by the introduction of measures such as *overflow* and *underfill*, representing cases where recognised events spill over, or fail to cover, the boundaries marked by ground truth (see 5.5.8). Though sufficient for the purposes of the described experiments, these measures do not go far enough in more general categorisation of context performance. For one, there is no standard method in context for dealing with events that become fragmented into several smaller events; nor is there a method for dealing with cases where several events become merged into one.

### 7.1.1. Chapter contributions and organisation

The chapter is divided into four main parts. In 7.2 we motivate the work by highlighting the problems of existing measures when used on a context recognition example taken from previously published work. In 7.3 we provide a detailed analysis of these problems, together with an overview of evaluation methods in related fields and the issues involved with their use. Section 7.4 introduces our proposed methods to combat these problems. These are then, in 7.5.1, applied to the original example and used to fuel the discussion on how they might be used in practice.

## 7.2. Motivation

As a motivation for this work, consider Figure 7.1. Plot (a) is an example of output from a multi-class, continuous activity recognition task which was carried out on a mock assembly scenario in the wood workshop of our lab [74]. The plot shows hand-labelled ground truth for five activities which we attempted to recognise in this experiment: use of a grinder, file, screwdriver, vice and drawer. The time where no relevant activity was performed is recorded as *NULL*. Plotted above the ground truth are the recognition system's predictions. This data is output on a timewise frame by frame basis, with each frame being one second in length.

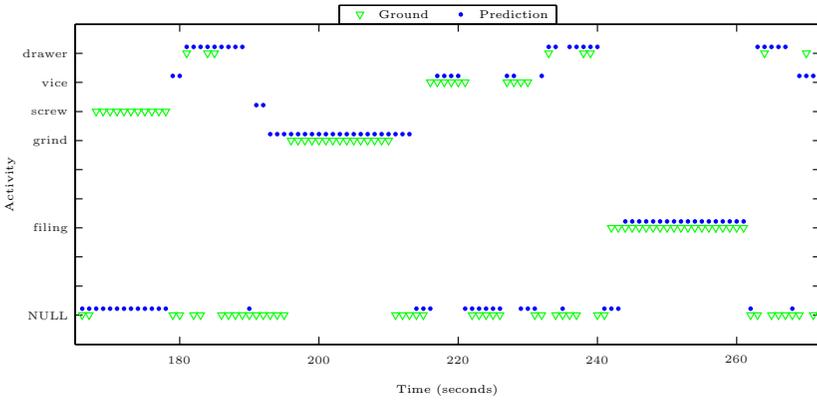
Visually, most of the non-*NULL* activities in (a) seem to correlate well with their ground truth: there are few insertions and only one completely deleted activity. The middle (b) and bottom (c) plots of Figure 7.1 tell a different tale: the abundance of insertions in (b) and the heavily fragmented output of (c) both contribute to the conclusion that these results are much poorer than that of (a). This is assuming, of course, that we are more interested in the correct ordering and contiguous nature of our prediction events than we are about their specific time durations.

### 7.2.1. Frame based analysis

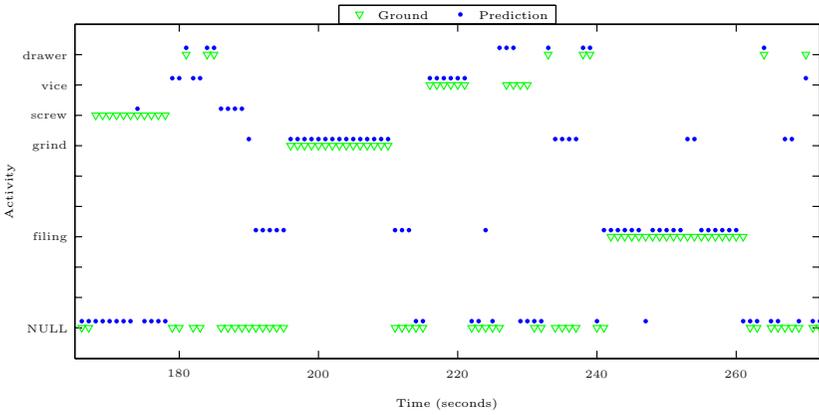
When evaluating timewise prediction sequences a standard practice is to make frame-by-frame comparisons with ground truth. Counts of correct and incorrect frames can then be tallied for each class and entered into a confusion matrix [76]. From this a number of standard performance measures can be calculated, the most common of which is accuracy (overall correct rate) but also increasingly the dual metrics precision and recall.

However, frame-by-frame analysis has its limitations. These can be seen by referring again to the examples of Figure 7.1. The frame-by-frame confusion matrices for these, simplified to the summation of positive classes vs. *NULL*, are shown in Table 7.1. Note how the accuracies calculated for each of the examples tell nothing about their differences - they are all identical. In addition the confusion matrices for (a) and (b) are also very similar and tell us nothing about, for example, the prevalence of insertion errors in (b).

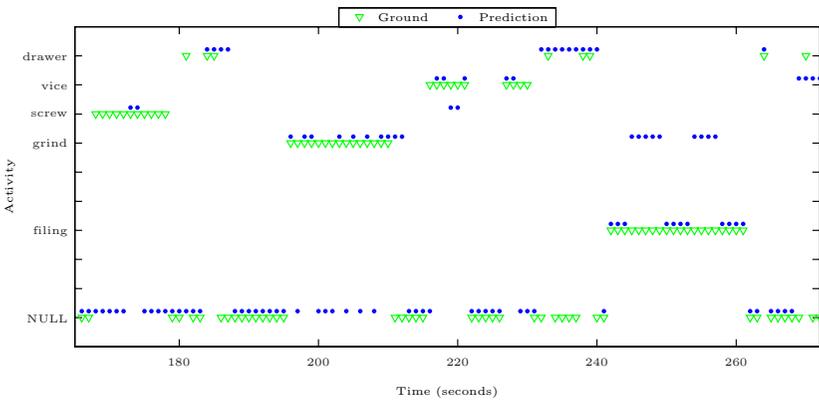
These results are not wrong - the numbers of frame errors in all three examples are in fact equal. What the visual analysis shows, and the frame analysis does not show, is that every positive frame forms part of an *event* - a contiguous sequence of same class frames. When judged from an event perspective then the distribution of frame errors becomes more important. Many of the false positives in (a), for example, are joined to otherwise correctly classified sequences; however, in (b) they tend to form part of *event insertions* - an arguably more serious misclassification.



a)



b)



c)

**Figure 7.1:** Multi-class continuous activity problem, examples (a–c). All examples have identical accuracy in frame-by-frame comparison. Note the prevalence of inserted events in b, and fragmented events in c.

### 7.2.2. Event analysis

Researchers in the fields of optical character recognition (OCR) [93, 94] and automatic speech recognition (ASR) both commonly employ counts of insertion ( $I_e$ ), deletion ( $D_e$ ) and substitution ( $S_e$ ) event errors to measure performance. This is a standard approach which is also used in context recognition.

When these scores are calculated for the examples (see Table 7.2), the differences between (a) and (b) become much more apparent. This time (a) clearly has a lower insertion count than (b). However, if we look at example (c) - again a very different output from (a) - we are once again disappointed: its deletion, substitution and insertion counts are identical to those of (a).

There are two main problems underlying these results, neither of which are highlighted by any of the commonly used evaluation methods. The first is that many of the events are fragmented: several smaller segments, although correctly classified, only sparsely cover parts of the ground truth. Some of these segments are separated by small fragments of *NULL* (frame deletions); while others, such as the ‘filing’ event, are fragmented by insertions of another class (frame substitutions).

The second problem is that events can be merged together: two or more events of the same class can be recognised as a single large event. In the examples given here this happens on two occasions, both involving the ‘drawer’ class (the first two instances of (a), and the third and fourth instances in (c)). In each case this error affects two closely occurring events. For purposes of evaluation the fact that these two separate events have been merged is simply ignored. They are both treated as correct. In other instances it might be decided (by the system designer) that merging two separate events constitutes one correct event and one deleted event.

In both of these cases, fragmenting and merging, it is clear that there are several ways one might choose to score the results, and here lies the problem: there is no standard definition for such errors. The existing designations of  $D_e$ ,  $I_e$  and  $S_e$  were developed for fields such as OCR which enjoy well-defined, discrete events. In continuous activity recognition, as highlighted by the examples given here, this is not always the case.

## 7.3. Problem specification and existing methods

In order to develop more appropriate evaluation metrics the problems illustrated in the previous section should first be formulated in a more systematic way. This section begins with a definition of the performance evaluation task. From this definition we identify specific characteristics of performance that are common to continuous activity recognition.

a)	Predictions		b)	Predictions		c)	Predictions		Total
	Pos.	Null		Pos.	Null		Pos.	Null	
P	<b>46</b> (1)	17	P	<b>45</b> (5)	14	P	<b>32</b> (12)	20	64
N	27	<b>16</b>	N	26	<b>17</b>	N	13	<b>30</b>	43
$acc_s = 57.9\%$			$acc_s = 57.9\%$			$acc_s = 57.9\%$			107

**Table 7.1:** Performance using standard methods: Frame errors using binary confusion matrices of positive (P) vs. NULL (N) frames, where rows denote the ground truth and columns the output predictions. Positive substitutions are entered in brackets alongside True Positives (TP) in these matrices. Accuracy is calculated as:  $acc_f = \frac{TP+TN-subst.}{T_f}$ , with the total frames in each example being  $T_f = 107$ .

	$I_e$	$D_e$	$S_e$	$err_e$
a)	3	1	1	<b>45.5%</b>
b)	10	0	2	<b>109.0%</b>
c)	3	1	1	<b>45.5%</b>

**Table 7.2:** Event errors are given as insertion ( $I_e$ ), deletion ( $D_e$ ) and substitution ( $S_e$ ) counts. The event error rate is  $err_e = \frac{I_e+D_e+S_e}{T_e}$ , with total number of positive events,  $T_e = 11$

### 7.3.1. Definition of performance evaluation

In the most general classification problem we have  $n$  classes ( $c_1, c_2, \dots, c_n$ ) without a designated class for *NULL*. The ground truth consists of a number of  $m$  distinct events ( $e_1, e_2, \dots, e_m$ ), each mapping to one of the  $n$  classes. We assume the system to be time discrete with the smallest considered time unit being a frame. In most cases, a frame would correspond to the length of the sensor sampling window.

An ideal classifier would be one where every ground truth event,  $e_i$ , has a start time, stop time and label matching an event in the prediction sequence. Correspondingly, all constituent frames would also match.

Unfortunately such perfect alignment is rare. A typical recognition system deletes, inserts, and substitutes data. In addition, even for correctly correlated data, the start and stop frames of the recognised sequence might be shifted in time. The problem of evaluating such imperfect classification is equivalent to that of finding an appropriate similarity metric for the comparison of two time series. As we see it, this problem can be tackled on three levels:

1. *Frame by frame.* For each pair of corresponding time frames  $f$  (from the ground truth) and  $\bar{f}$  (from the recognition system output) we perform a simple comparison of the class labels.

2. *Event-based.* Determine how many of the  $m$  ground truth events ( $e_1, e_2, \dots, e_m$ ) are accurately reflected in the  $\bar{m}$  events  $\bar{e}_1, \bar{e}_2 \dots \bar{e}_{\bar{m}}$  produced by the recognition system. The difficulty of event based evaluation stems from the fact that neither the number of events nor their start and end points are necessarily identical in the ground truth and the recogniser output.
3. *Hybrid frame and event based.* A frame by frame comparison that takes into account the events to which individual frames are a part. Thus frame errors that merely cause the start and end points of events to be shifted are treated differently from frame errors that contribute to the deletion and insertion of events. This type of evaluation only makes sense if some prior event analysis has been carried out.

### 7.3.2. General considerations

Given two time series there can be no such thing as an optimal measure of similarity that holds for all applications. As a consequence there is no optimal, problem independent performance evaluation. Different application domains are subject to different performance criteria. In speech recognition, for example, it is more important that the system recognises what words have been spoken, and in which order, rather than how long it took to utter them. Consequently, methods that emphasise correct ordering over the specific duration of symbols are used to evaluate these systems. An input to a real-time system, on the other hand, would need to be extremely time sensitive. As such an evaluation metric that emphasises timing errors and delays, i.e. based on a direct timewise comparison, would be more appropriate.

For every domain, a specific metric must be chosen that characterises and highlights the most critical types of error. This means that evaluation methods that are successful in one field need not necessarily be so in another. Applying methods from one domain to another only makes sense if both domains have the same types of dominant error; in addition, similar relevance should be assigned to equivalent errors.

### 7.3.3. Requirements for activity recognition

The study of activity recognition encompasses a wide range of problems including standard modes of locomotion (walking, standing, running, etc.) [6, 41, 95, 96], tracking of specific procedures (e.g. assembly tasks [97]), and the detection of changes in environmental conditions [30, 41]. While each of these problems have their own characteristic and relevant error types, there are a number of things that most continuous activity recognition tasks have in common:

***Large variability in event length*** In many activity recognition tasks event length can vary by an order of magnitude or more. A wood workshop assembly example includes such activities as sawing, which can take minutes, as well as taking or putting away tools, which can take just a few seconds. Similarly, when recognising modes of locomotion, there can be instances of long uninterrupted walks, as well as instances of a user making only a few steps. Direct frame-by-frame evaluation can be misleading in these cases.

***Fragmented events*** Long lasting events are often interrupted by the occurrence of short events. Thus a long sawing sequence might include one or two interruptions or an instance of the user changing the saw. A long walk might include a few short stops. A recognition system designed to spot such situations will also be prone to false fragmentation. As an example, a slight irregularity in the sawing motion might be falsely interpreted as an interruption, or a short instance of an entirely different activity. Fragmentation breaks what should be one long event in the ground truth into several smaller events in the recogniser output.

***Event merging*** Trying to avoid false fragmentation can lead to a system that tends to overlook genuinely fragmented outputs. Thus two events of the same class, separated by a short event of another class, might be merged into a single long event of the first class. This in a sense is a 'double deletion' because it deletes the short event in the middle, and causes the two events of the outer class to become one.

***Lack of well defined NULL class*** Many activity recognition tasks aim to spot a small set of interesting activities or situations while regarding the rest as instances of a 'garbage' or *NULL* class. A *NULL* class has the same function as the pauses in speech, or spaces in character recognition. The problem is that many activity recognition tasks have a *NULL* class that is complex and difficult to model. In the assembly task, for example, any motion made between the specific tool activities falls into this class. This includes everything from scratching one's head to unpacking a chocolate bar. As a consequence the *NULL* class model tends to be 'greedy', so that any unusual segment in an event (e.g. strange motion while sawing) tends to create a *NULL* event, thus contributing to the fragmentation problem.

***Fuzzy event boundaries*** When collecting large sets of 'real life' data it is often impossible to perfectly time ground truth labels by hand. The definitions for start and stop times of an event are often arbitrary and imprecise. This is particularly so in domains such as activity recognition, where the notion of an 'event' is often difficult to define - at which point

does a walking event end and a running event begin? This leads to timing errors in the recognition, even if the system can be said to be working perfectly. Similarly, in tasks where interesting events are separated by a greedy *NULL*, the lack of a well defined *NULL* model will inevitably result in some incursion into the boundaries of the correct events.

The importance of these different issues is dependent on the specific application for which the system is being evaluated. However, we believe that for most activity recognition tasks one or more of these issues is important and that they should be taken into account when evaluating these systems.

## 7.4. Error characterisation and representation

Following from the above observations we now present a characterisation of the critical error types in continuous activity recognition. Specifically we propose an approach that (1) includes event mergers and fragmentation as errors in their own right; and (2) provides information about event timing errors. This section presents both a definition of the proposed errors, and a precise method on how to score them. We then show how this information can be tabulated for presentation of a system's results. Additionally, we show how the methods can be tailored for dealing with activity recognition systems that treat *NULL* as a special case.

Our evaluation method is based on partitioning the signal stream into what we call *segments*. As an example, Figure 7.2 shows a three class recognition problem broken up into 14 segments (denoted by the vertical dotted lines). A segment is a variable-duration, contiguous sequence of frames during which neither prediction nor ground truth label change. That is, each boundary of a segment is defined by either the boundary of a ground truth, or of a prediction event.

From the point of view of performance evaluation such a segment definition has two advantages. The first is that there are no ambiguities in comparison: each segment can either have the prediction and the ground truth fully agree, or fully disagree. The second advantage is that from an analysis of these segments an exhaustive definition of the event and timing errors, appropriate to context recognition, can be derived. This strategy has three main steps:

1. Create the segment sequence and note each segment as matching or non matching. A *match* being when both the ground truth segment and its corresponding prediction segment have the same class label.
2. Use segment match information to score events and event timing errors. Prediction and ground truth events are scored separately. The flowchart of Figure 7.3 shows the algorithm to do this for ground truth events,

with possible outputs of fragmenting  $F$ , deletion  $D$ , underfill  $U$ , correct, and *no label* (a single matching segment event to which none of the other designations apply). Prediction events are scored using the same algorithm, but with outputs: merge ( $M$ ), insertion( $I$ ) and overfill ( $O$ ).

3. Score the segment errors. The Figure 7.4 shows how this is done. Each non-matching segment is assigned an error pair based on the ground and prediction events to which it forms part.

In Figure 7.2 we take a single, three-class example and show the results from each of the event, timing and segment scoring algorithms. Note that, because we are initially dealing with the general multiple class problem (we do not give special consideration to *NULL*), all errors are substitutions between the three classes ‘A’, ‘B’ and ‘C’. The error categories (event, timing and segment) and how to score them, are described in greater detail in the following sections.

#### 7.4.1. Event analysis

There are four types of event error, each falling into one of two divisions depending on whether they are part of the ground truth or of the prediction sequence. A positive error in the *prediction* sequence can be defined as either:

***Insertion*** - a prediction event containing no matches, or

***Merge*** - a prediction event containing more than one match.

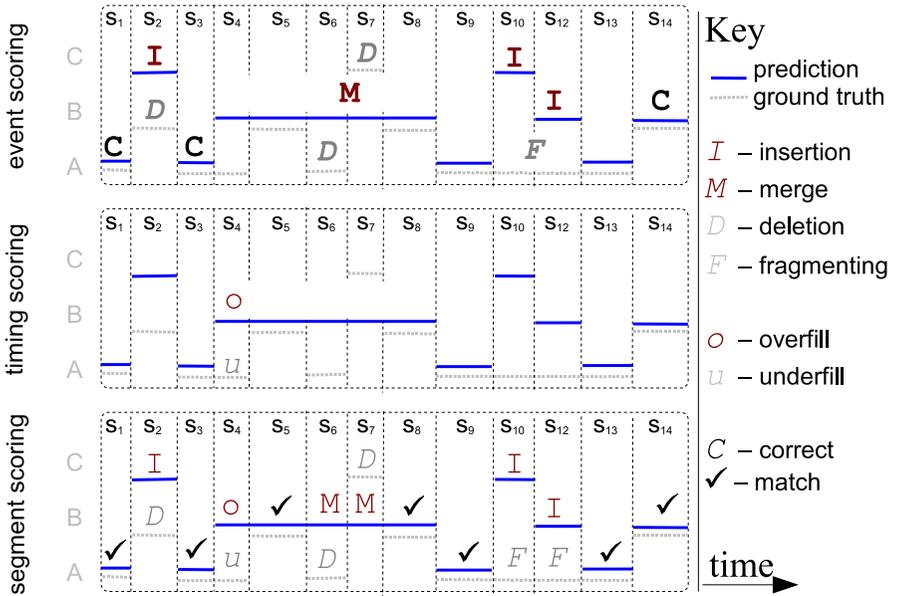
A *negative* error, the failure to detect all or part of an event in the ground truth, is defined as either:

***Deletion*** - a ground truth event containing no matches, or

***Fragmentation*** - a ground truth event containing more than one match.

Correct is only assigned when a corresponding prediction and ground truth event is free from *all* of the above errors. There are some cases where a single-segment, matched event is not assigned any designation (for example, see the merged ground events  $s_5$  and  $s_8$  in Figure 7.2). On an event analysis these events cannot be said to be correct - but neither can they be called errors. Instead, we just call them *segment matches*.

Positive and negative errors are related: an insertion in the prediction sequence, for example, can result in the deletion or fragmentation of an event in the ground truth. This relationship is not always one-to-one however: a fragmentation might be caused by more than one insertion, possible of different classes. It is for this reason that the event level scoring is carried out on two sequences; on the ground truth (negative errors) and prediction (positive errors).



**Figure 7.2:** Some possible error combinations for a single, three class ( $A, B, C$ ) example (the special class NULL is not considered here): upper diagram shows event errors, middle diagram shows event timing errors, and lower diagram shows segment error pairs. Dotted vertical lines show how the sequence is broken up into segments  $s_{1..14}$ . Note how the merge event, covering  $s_4$  to  $s_8$ , is made up of  $OU$ , match and  $MD$  segments.

## Event timing

Often an event might be judged correct (or merged, or fragmented) but fail to align completely with its boundaries. The prediction event might spill over the ground truth boundaries; or it might fall short of them. For these cases, we introduce two *event timing* error categories that can be applied to an event in addition to a correct, merge<sup>2</sup>, or fragmenting score:

**Underfill** - *ground truth* event not completely covered by prediction.

**Overfill** - *prediction* event which spills over its ground truth boundary.

Four further sub-categories of timing errors might also be used: *delay*, noting an underfill at the beginning of an event; *shortening* an underfill at the end; *preemption*, an overfill at the beginning; and *prolongation*, an overfill at the end. In this thesis we focus only on the two main categories of overfill and underfill and leave treatment of the sub-categories for future work.

The algorithm for assigning both event errors and event timing errors is shown in Figure 7.3.

## Event error and timing error representation

Counts of the four types of event error - insertion, deletion, merge and fragmentation - can be summed up for each class and presented in a simple table, one entry for each error type and each class. Similarly, counts of the timing event errors - overfill and underfill - can also be summed up and presented, in a separate table, alongside the specific time lengths (or number of frames) associated with them.

### 7.4.2. Segment analysis

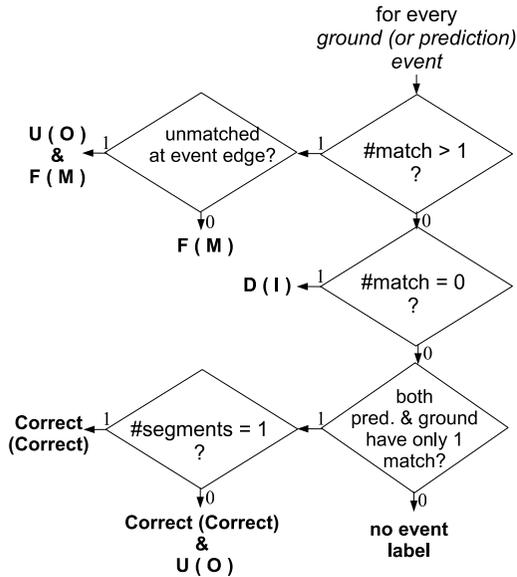
One aspect of performance that event based scoring does not capture is the absolute time duration (in terms of frames or seconds) for each type of error. Additionally, subtle information such as the cause-effect relationship between prediction and ground truth errors is not captured. It can be shown that the following pairings are possible:

1. An event is deleted by insertions, merging, or overfilling of another class
2. An event is underfilled by either an overfill or an insertion of another class
3. An event is fragmented by insertion(s) of another class.

---

<sup>2</sup>Segment  $s_4$  of Figure 7.2 shows one such example of this.

## Event and timing error algorithm



**Figure 7.3:** Algorithm for assigning error labels to each ground truth event, and to each prediction event: for processing ground truth events, use  $F, D$  and  $U$ , for fragmenting, deletion and underfill; for processing prediction events, use bracketed labels  $(M)$ ,  $(I)$  and  $(O)$ , referring to merge, insertion and overfill errors respectively; correct or no label can be assigned to both.  $\#segments$  refers to the number of segments that make up an event,  $\#match$  refers to the number of matching segments in that event, with a match defined as a segment where ground truth and prediction agree.

Rarely do event level comparisons allow a one-to-one relation between the prediction and ground truth. One deletion, for example, might be the result of a combination of an overfill plus several different insertions. Segments do allow such a relation. By definition, every segment forms part of exactly one prediction event and one ground truth event.

The specific combination of event and timing errors for each ground truth and prediction can therefore be used to define the segment error type, as detailed in Figure 7.4. In total there are six possible error types for non-matching segments based on the event combinations. Three of these involve segments forming part (or all) of event errors:

***Insertion-Deletion (ID)*** - forms part (or all) of an inserted prediction and a deleted ground truth

***Insertion-Fragmenting (IF)*** - an inserted prediction that lies somewhere between two matching segments of a fragmented ground truth

***Merge-Deletion (MD)*** - forms part of a merge prediction, occurring somewhere between two matching segments of the same prediction event, causing a deletion in the ground truth

The remaining three error designations involve segments that form part of timing (or both timing and event) errors:

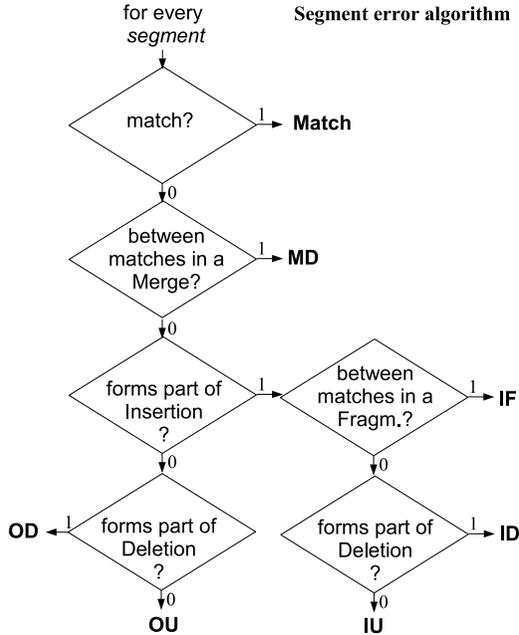
***Insertion-Underfill (IU)*** - forms part (or all) of an inserted prediction and an underfilled ground truth (only assigned if the segment has not already been classified as IF)

***Overfill-Deletion (OD)*** - forms part (or all) of an overfilled prediction and a deleted ground truth (only assigned if the segment has not already been classified as MD)

***Overfill-Underfill (OU)*** - forms part (or all) of an overfilled prediction and an underfilled ground truth

The general rule is to take a non-matching segment, and name it according to the constituent event or timing error designations. Thus a segment that forms part of an insertion event in the prediction sequence, and a deletion in the ground truth, is classed as an *Insertion-Deletion (ID)*; the part of an insertion event which causes an underfilled segment in the ground truth is called an *Insertion-Underfill (IU)*. Similarly, a segment that forms part of an overfill timing error in the prediction and a deletion in the ground truth is classed as *Overfill-Deletion (OD)*; if the ground truth is merely an underfill, the classification is *Overfill-Underfill (OU)*.

Two exceptions to this rule occur when a timing error is assigned in addition to a merge or fragmentation event error. If a non-matching segment lies



**Figure 7.4:** Algorithm for assigning error pair labels to a segment based on its constituent event error designations: Match defined as a segment where ground truth and prediction agree; MD=merge-deletion, IF=insertion-fragmentation, ID=insertion-deletion, OD=overflow-deletion, OU=overflow-underfill and IU=insertion-underfill.

between two matching segments of a merge event it is, by definition of merge, partly responsible for a deletion and should be called *Merge-Deletion (MD)*. However, if the predicted merge event is also an overflow and the segment lies outside of the matching segments then it should be referred to as either an *OD*, or *OU* segment (depending on the state of the corresponding ground event).

Similarly, if a non-matching segment lies between two matching segments of a fragmented event it can only be caused by an insertion in the prediction sequence and should be called *Insertion-Fragmenting (IF)*. However, if the fragmented ground event is also an underfill, and the segment lies outside of the matching segments, then it should be referred to as either an *IU*, or *OU* segment.

These pairings are codified and presented in Table 7.3, which we name the Segment Error Table (SET). Prediction errors (insertion, overflow and merge) form the rows, while ground truth errors (deletion, underfill and fragmentation) make up the columns of this table.

	Deletion	Underfill	Fragmentation
Insertion	ID	IU	IF
Overfill	OD	OU	
Merge	MD		

**Table 7.3:** Possible segment error designations: rows represent prediction segment errors, columns ground truth errors; ID=insertion-deletion, OD=overfill-deletion, MD=merge-deletion, IU=insertion-underfill, OU=overfill-underfill and IF=insertion-fragmentation

	D	U	F	$D_N$	$U_N$	$F_N$		D	U	F	N	
I	ID	IU	IF	$ID_N$	$IU_N$	$IF_N$		I	ID	IU	IF	I
O	OD	OU		$OD_N$	$OU_N$			O	OD	OU		O
M	MD			$MD_N$				M	MD			M
$I_N$	$I_N^D$	$I_N^U$	$I_N^F$					N	D	U	F	
$O_N$	$O_N^D$	$O_N^U$										
$M_N$	$M_N^D$											

**Table 7.4:** Segment Error Table with NULL(N) as special case: full table (left) and reduced version (right)

Analysis of segments provides an unambiguous assessment of errors. In the simplest analysis, segment counts of the six different error types, ID, IU, IF, OD, OU, and MD are made and filled into the table. Additional information about the absolute time length, or frame counts, of these segments can also be included. This combined segment and frame count SET provides a representation of errors that combines the temporal resolution of frame-by-frame evaluation with the descriptive power of event level evaluation.

**NULL as a special case**

We can expand the table thus described to handle NULL as separate from the other classes - a separation required for most activity recognition tasks. This is achieved by the addition of rows and columns denoting the six error combinations with respect to NULL, as shown to the left of Table 7.4. The SET to the top left corner of the expanded table now only contains information regarding substitution errors between non-NULL positive classes. The top right section of the table then gives a breakdown of false positive errors, while the bottom section gives information about false negative errors.

In many continuous recognition scenarios we are not interested in whether a ground segment labelled NULL has been completely deleted, fragmented or underfilled; likewise we are not interested in whether a positive class deletion was caused by an insertion or an overfilling of NULL. In such situations the

error designations can be combined to produce a reduced table, as shown to the right of Table 7.4. For convenience we drop the 'N' suffix and the dual error designation from the errors involving *NULL*, referring to them directly as *I*, *O*, *M*, *D*, *U* and *F*. The remaining *substitution* errors retain the dual *OU*, *IU*, *etc.*, designators.

### Multi-class SET

While the basic SET provides much more information about system performance than event level analysis alone, it still lacks the exact relation between errors of different classes as traditionally represented by confusion matrices. Where more detail is required regarding specific (or all) classes in a multi-class system, SET can easily be extended in a similar manner to that already shown for *NULL* in Table 7.4. By adding three additional columns (*D,U* and *F*) and three additional rows (*I,O* and *M*) for each class, SET can be extended for any number of classes.

## 7.5. Discussion

### 7.5.1. Application of method to worked example

We now apply the described error characterisations to our examples from Section 7.2 and give examples of how the event, timing and SET representations might look.

#### Event and timing results

Treating *NULL* as a special case, we present counts of the non-*NULL* class insertions, deletions, merge and fragmentation errors, for the examples of Figure 7.1, in Table 7.5. Comparing these counts with the insertion and deletion counts of the earlier event analysis in Table 7.2, we can draw much the same conclusions. Notably, the new method allows us to clearly see the additional merge and fragmentation errors in example (c). The poor timing performance of (a), with many overfilled events in comparison to the other examples, is also now evident.

The information regarding class substitution errors, however, has been lost in this representation - they are dissolved into pairs, such as insertion/deletion. The lack of a one-to-one relationship between prediction and ground truth errors makes the idea of a 'substitution event' difficult to define at the event level. Instead, we defer to the segment analysis to provide this information.

<table border="1" style="border-collapse: collapse; width: 100%;"> <tr><td style="text-align: center;">#events</td></tr> <tr><td>I' 4</td></tr> <tr><td>D' 2</td></tr> <tr><td>M 1</td></tr> <tr><td>F 0</td></tr> </table>	#events	I' 4	D' 2	M 1	F 0	<table border="1" style="border-collapse: collapse; width: 100%;"> <tr><td style="text-align: center;">#timing(#frames)</td></tr> <tr><td>Overfill 8 (18)</td></tr> <tr><td>Underfill 4 (6)</td></tr> </table>	#timing(#frames)	Overfill 8 (18)	Underfill 4 (6)	<table border="1" style="border-collapse: collapse; width: 100%;"> <tr><td style="text-align: center;">#events</td></tr> <tr><td>I' 12</td></tr> <tr><td>D' 2</td></tr> <tr><td>M 0</td></tr> <tr><td>F 1</td></tr> </table>	#events	I' 12	D' 2	M 0	F 1	<table border="1" style="border-collapse: collapse; width: 100%;"> <tr><td style="text-align: center;">#timing(#frames)</td></tr> <tr><td>Overfill 1 (1)</td></tr> <tr><td>Underfill 3 (11)</td></tr> </table>	#timing(#frames)	Overfill 1 (1)	Underfill 3 (11)	<table border="1" style="border-collapse: collapse; width: 100%;"> <tr><td style="text-align: center;">#events</td></tr> <tr><td>I' 4</td></tr> <tr><td>D' 2</td></tr> <tr><td>M 1</td></tr> <tr><td>F 3</td></tr> </table>	#events	I' 4	D' 2	M 1	F 3	<table border="1" style="border-collapse: collapse; width: 100%;"> <tr><td style="text-align: center;">#timing(#frames)</td></tr> <tr><td>Overfill 4 (6)</td></tr> <tr><td>Underfill 4 (12)</td></tr> </table>	#timing(#frames)	Overfill 4 (6)	Underfill 4 (12)
#events																													
I' 4																													
D' 2																													
M 1																													
F 0																													
#timing(#frames)																													
Overfill 8 (18)																													
Underfill 4 (6)																													
#events																													
I' 12																													
D' 2																													
M 0																													
F 1																													
#timing(#frames)																													
Overfill 1 (1)																													
Underfill 3 (11)																													
#events																													
I' 4																													
D' 2																													
M 1																													
F 3																													
#timing(#frames)																													
Overfill 4 (6)																													
Underfill 4 (12)																													
a)		b)		c)																									

**Table 7.5:** Event errors (for Positive, non-NULL classes only), I'=Insertion, D'=Deletion, M=Merge and F=Fragmentation; and event timing errors, Overfill and Underfill. Number of timing event errors are given together with the corresponding frame counts

### Segment (and frame-by-frame) results

The segment and frame errors for the examples are presented in Table 7.6. Now clearly visible is the higher proportion of segments which form timing errors in (a) (Underfilling by NULL, U and Overfill onto NULL, O), than in the other examples. Correspondingly, a higher proportion forming event errors is found in (b) and (c). These examples contain fragmenting errors whereas (a) does not. Of merger errors there are only two instances - in (a) and (c) - each of which involves only a single merge of two 'drawer' events.

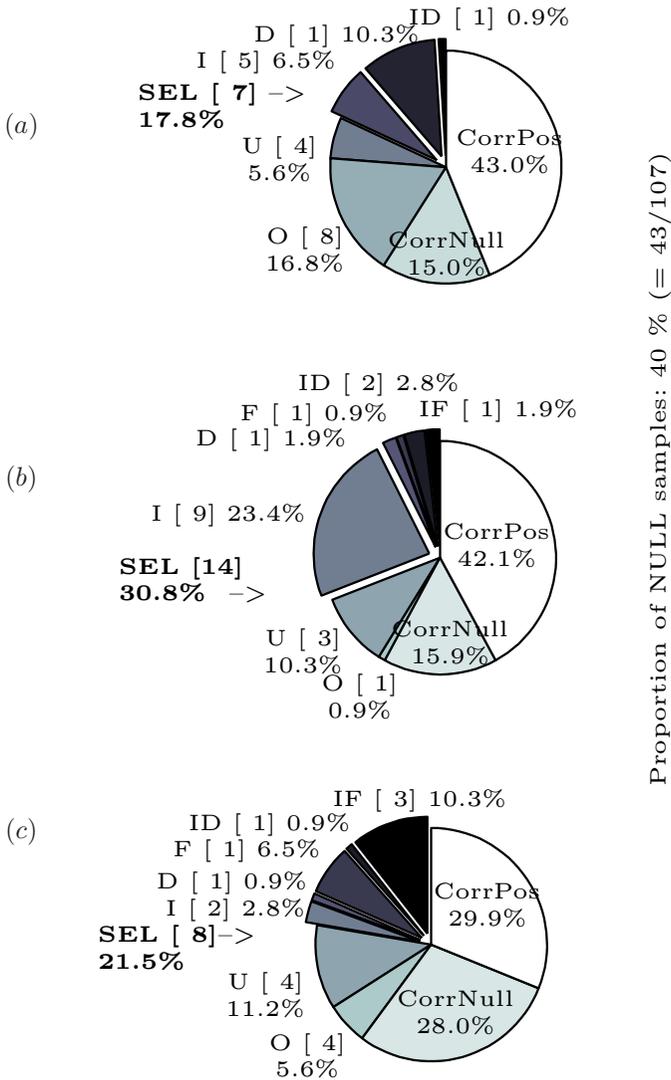
These measures correspond to what a visual inspection of the output might confirm, and provide much more information than a basic frame-by-frame comparison.

### Visualisation of SET

The pie-charts of Figure 7.5 give one possible manner in which the SET information might be presented. Another approach, perhaps better for system comparisons, is to stack the errors in a barchart. For readability these need not necessarily show all of the different error types in the same graph - for example, the substitution errors might be treated together as a single error type, with perhaps a separate graph for all the others.

<table border="1" style="border-collapse: collapse; width: 100%;"> <tr><td style="text-align: center;">#segments (#frames)</td></tr> <tr><td style="text-align: center;">D U F N</td></tr> <tr><td>I 1(1)</td></tr> <tr><td>O 8(18)</td></tr> <tr><td>M 1(2)</td></tr> <tr><td>N 1(11) 4(6)</td></tr> </table>	#segments (#frames)	D U F N	I 1(1)	O 8(18)	M 1(2)	N 1(11) 4(6)	<table border="1" style="border-collapse: collapse; width: 100%;"> <tr><td style="text-align: center;">#segments (#frames)</td></tr> <tr><td style="text-align: center;">D U F N</td></tr> <tr><td>I 2(3)</td></tr> <tr><td>O 1(2)</td></tr> <tr><td>M 9(25)</td></tr> <tr><td>N 1(1)</td></tr> <tr><td>N 1(2) 3(11) 1(1)</td></tr> </table>	#segments (#frames)	D U F N	I 2(3)	O 1(2)	M 9(25)	N 1(1)	N 1(2) 3(11) 1(1)	<table border="1" style="border-collapse: collapse; width: 100%;"> <tr><td style="text-align: center;">#segments (#frames)</td></tr> <tr><td style="text-align: center;">D U F N</td></tr> <tr><td>I 1(1)</td></tr> <tr><td>O 3(11)</td></tr> <tr><td>M 2(3)</td></tr> <tr><td>N 4(6)</td></tr> <tr><td>N 1(4)</td></tr> <tr><td>N 1(1) 4(12) 1(7)</td></tr> </table>	#segments (#frames)	D U F N	I 1(1)	O 3(11)	M 2(3)	N 4(6)	N 1(4)	N 1(1) 4(12) 1(7)
#segments (#frames)																							
D U F N																							
I 1(1)																							
O 8(18)																							
M 1(2)																							
N 1(11) 4(6)																							
#segments (#frames)																							
D U F N																							
I 2(3)																							
O 1(2)																							
M 9(25)																							
N 1(1)																							
N 1(2) 3(11) 1(1)																							
#segments (#frames)																							
D U F N																							
I 1(1)																							
O 3(11)																							
M 2(3)																							
N 4(6)																							
N 1(4)																							
N 1(1) 4(12) 1(7)																							
a)	b)	c)																					

**Table 7.6:** SETs for positive classes (P) vs. NULL for the examples in Figure 7.1, with counts of segment errors and corresponding number of frames



**Figure 7.5:** Pie chart visualisation of SET information for the three examples (top (a), middle (b) and bottom (c)). Errors given as a percentage of the total time (frames), with the number of segments given [in square brackets]. Exploded segments represent serious errors, the total of which is marked SEL

### 7.5.2. Significance and Limitations

As shown by the examples above, our scheme has three advantages over standard methods of performance evaluation in activity recognition:

1. It introduces the notion of segments as the largest continuous time slices in which no ambiguities occur in scoring the correctness of the predictions
2. Based on this notion it leads to an *unambiguous*, objective characterisation of event level error
3. It makes explicit different sources of error (timing, fragmentation, merging) which are ignored in conventional evaluation methods, even though they are widespread in activity recognition systems.

The main limitation of the method concerns events with a large time shift between ground truth and the prediction. A prediction that is shifted by so much that it has no overlap with the corresponding ground truth will be scored as an insertion, and the corresponding ground truth event as a deletion.

What require further investigation are the benefits of this additional error information. These are dependent on the application for which the recognition system is to be used. For a safety critical system, such as an accident avoidance monitor in an industrial setting, timing may be regarded as critical, thus making the minimisation of overfill and underfill of recognised activities desirable. On the other hand, for a system interested only in which activities are carried out such errors would be less critical. Imagine, for example, a system monitoring the sequence of events as a mechanic repairs part of an aircraft engine. What is important here is that the number of insertions and deletions is kept low - that the system does not miss out any activities, and that it gets the sequence correct. If further information on the count of specific activities is required (how many bolts have been removed from the engine) then errors such as fragmenting and merge errors must also be kept to a minimum.

For a conclusive proof of the value of the information provided by our method an elaborate empirical study is needed. Such a study would need to consider a wide range of applications and preferably look at previously published activity recognition experiments and re-score their results using the above method.

For a meaningful study access to data from different groups would be required and the associated effort would be beyond the scope of this work. This is clearly a limitation and means that no authoritative statement can be made about the value of the additional error information. Nonetheless such benefits are very plausible. Considering the undisputed benefit of an objective scoring method we believe that this work consist a valuable contribution to the community.

### 7.5.3. Work in related fields

Some of the problem domains closest to continuous activity recognition are line detection in 2D Graphics [87] and video analysis [98, 99]. Consider the case of a 2D line: the ground truth indicates a single line, but the recognition system might return a sequence of shorter lines. Further, these might overlap with the ground line, or be slightly offset from it. Different approaches have been suggested to tackle this problem of fragmentation. One suggestion is to redefine the error measures to incorporate fragmented events as some lower weighted correct event[87].

Some decision function based on a measure of closeness might also be used; perhaps utilising fuzzy error margins (as suggested at [100]). However this approach, as with weighting, requires the introduction of further parameters which only serve to further complicate the evaluation process. In addition, all of these approaches aim to “cover up” the problem rather than finding a way of presenting it as a result in itself.

In extreme cases, particularly in the vision domain, the problem of finding a suitable measure is sidestepped altogether in favour of showing typical example images (as commented by Hoover *et al.* [66] and by Müller [67]). This is an approach which has - out of necessity for lack of a standard measure - been used by researchers publishing in the activity domain. The trouble is that, although valid for establishing the feasibility of a method with a small number of samples, it does not scale up well to comparative studies with large databases.

### Time series matching methods

More generally, the performance evaluation problem can be viewed as the matching of two time series - the prediction output with a trusted ground truth. Time-series similarity methods are used in an extremely wide variety of domains - astronomy, finance, chemistry, robotics, etc., to mention only a few. Even more vast is the number of performance measures that are introduced for every specific application (Keogh & Kasetty[72] give an extensive overview). Some of the more common similarity measures are generally based on dynamic time warping (DTW)[101], or methods using longest common subsequences (LCS)[102]. Another useful method, as introduced by Perng *et al.*[103] utilises ‘landmarks’ in the data, applying several different transformations (shifting, time warping, etc.) to approximate a more human perception of similarity. Though useful in measuring similarity, these methods do not provide a clear means of measuring phenomena such as event fragmenting and merging.

Rather than selecting some measure of ‘similarity’, or parametrized boundary decision to fit existing error designations, we aim to characterise and

present the errors as they are - in a quantifiable way which corresponds closely to that of the human observer.

## 7.6. Conclusion

In this chapter we present a non-ambiguous scoring of event errors in a continuous activity recognition system. Observing the lack of a one-to-one relationship between events in the ground truth and those in the prediction sequence, we target errors in these two sequences separately: specifically, we define positive errors as *insertion* ( $I$ ) and *merge* ( $M$ ) events by the prediction sequence; and negative errors as *deleted* ( $D$ ) and *fragmented* ( $D$ ) events in the ground truth. Complementary to these, we introduce timing event categories which score whether a prediction event overfills its ground truth, or a ground event is underfilled by its prediction.

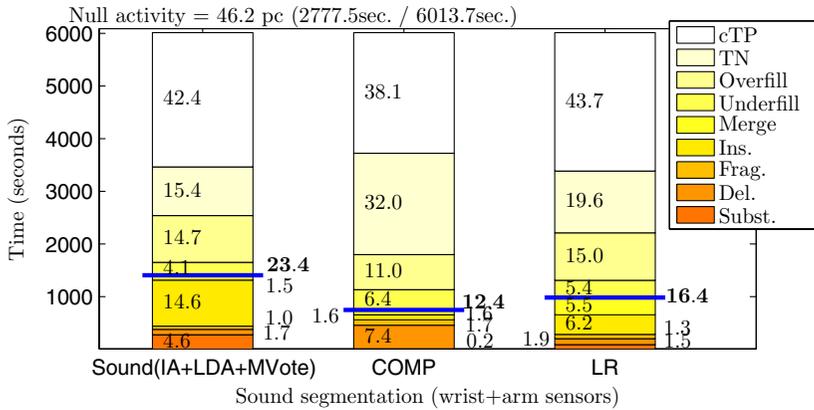
We introduce a timewise method of comparison based on the idea of segments - a segment being a contiguous section of time where neither ground truth nor prediction changes. This allows the representation of an unambiguous one-to-one relation between ground and prediction segments, which we have shown to produce a maximum of six possible error combinations, each assigned depending on the nature of the events to which each segment forms part: ID, IU, IF, OD, OU, and MD. These error pairings can be represented in the so-called *Segment Error Table* ( $SET$ ), with scoring on the number of segments, and their corresponding time durations (or number of frames).

The aim of this chapter is primarily to highlight some of the problems in performance evaluation of context recognition systems, and to suggest a way in which these might be dealt with. The proposed methods of event and segment analysis, and the use of the  $SET$ , are intended as starting points from which further discussion in the community can commence. The examples given here are also intended as a preliminary study, and further evaluation of these methods on a wider range of context problems is desirable. Though it is the author's belief that these methods will find general use, it is likely that some revision will be necessary on encountering specific problems not envisaged here.

# 8

## SET optimization for continuous activity recognition

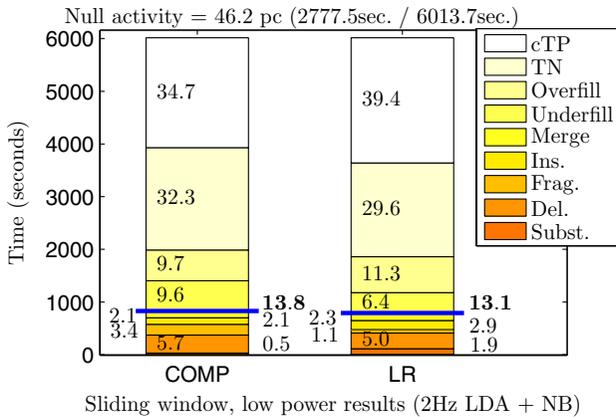
*This chapter provides a synthesis of some of the main ideas presented in the thesis. In particular it provides a re-assessment of the earlier activity recognition methods using the ideas introduced in Chapter 7. Specifically, the Segment Error Table (SET) strategy for performance evaluation is applied to a selection of the earlier results. The aim here is to provoke a discussion on the utility of SET as a tool for system optimization.*



**Figure 8.1:** Visualisation of selected results using sound based segmentation, and wrist+arm based sensors, from Chapter 5: sound only (IA+LDA+MVote), COMP and LR. Correct positive (cTP), true negative (TN), overfill, underfill, insertion (Ins.), deletion (Del.) and substitution times (Subst.) given as before. This time Merge and Fragmenting (Frag.) errors are included as part of the revised Serious Error Level (SEL - shown in bold). Note how merge accounts for 5.5% of LR - previously, in Figure 5.9, this was part of ‘overfill’.

## 8.1. SET analysis of the wood workshop

The time based results of the work described in Chapter 5 and 6 were presented using visualisations based on confusion matrices. In those chapters, the results were also given with overfill and underfill. However, the presented versions of overfill and underfill incorporated frames which could now be classified as merge and fragmenting errors. For the purposes of the evaluation dealt with in the thesis so far, these errors might be regarded as having little importance, and the consequent combination of merge time with overfill, and fragmenting time with underfill, could be viewed as acceptable. However, with the considerations of Chapter 7 in mind, the definition of *serious error level (SEL)* given in the earlier chapters should now be revised to incorporate merge and fragmenting. In doing so, a slight reduction in the levels of overfill and underfill, with a corresponding rise in serious error can be expected. The revised barcharts, using the full breakdown of overfill, underfill, merge, fragmenting, insertion, deletion and substitution timing errors, are given in Figure 8.1 and 8.2 for selected user-dependent results of Chapter 5 and 6 respectively.



**Figure 8.2:** Visualisation of selected results using the low power, wrist only, sliding window segmentation from Chapter 6 (2Hz LDA+NB): comparison classifiers (COMP) and LR fusion. Again, Merge and Fragmenting errors are shown contributing towards the Serious Error Level (SEL). Note how fragmenting errors take up nearly 3.4% of COMP - in the earlier result of Figure 6.5 this was considered part of ‘underfill’.

### 8.1.1. (Re-)analysis of results

Though the SEL rises as expected, it rises fairly consistently with all the methods, and the same broad conclusions can be drawn as given earlier, that is:

1. Basic recognition can be achieved using sound alone. Using the two microphone (IA+LDA+MVote) algorithm, the SEL is 23%. Of this, 4.6% is substitution time. The majority of errors are caused by insertions (14.6% of the time). Both these error types can be reduced by fusion.
2. Applying different fusion methods to combine sound and acceleration improves this performance considerably. Comparison of outputs (COMP) provides a ‘cautious’ recognition, preferring low instances of falsely recognised activities, and almost no substitution errors. This comes at the cost of a high number of deletions (7.4% for Figure 8.1). Logistic regression (LR) provides a compromise, reducing the deletion time (1.9%) and still keeping a low level of substitutions (1.5%) - but at a cost of an increase in the other false positive errors.
3. Using only wrist-worn sensors, and a simple sliding window segmentation which utilises both sound and acceleration, similar performance is observed to the sound segmented method.

COMP (2Hz LDA + NB)

	D	U	F	N
I	6 <sub>(7.9)</sub>	19 <sub>(19.8)</sub>	0 <sub>(0.0)</sub>	56 <sub>(124.3)</sub>
O	1 <sub>(0.8)</sub>	2 <sub>(0.6)</sub>		567 <sub>(584.1)</sub>
M	0 <sub>(0.0)</sub>			41 <sub>(129.2)</sub>
N	179 <sub>(343.3)</sub>	460 <sub>(574.9)</sub>	63 <sub>(204.0)</sub>	

LR (2Hz LDA + NB)

	D	U	F	N
I	13 <sub>(27.1)</sub>	35 <sub>(51.9)</sub>	14 <sub>(32.0)</sub>	84 <sub>(173.8)</sub>
O	1 <sub>(0.8)</sub>	3 <sub>(0.9)</sub>		611 <sub>(681.6)</sub>
M	0 <sub>(0.0)</sub>			47 <sub>(140.0)</sub>
N	172 <sub>(299.5)</sub>	397 <sub>(387.8)</sub>	26 <sub>(64.0)</sub>	

**Table 8.1:** SET tables of selected results using the low power, wrist only, sliding window segmentation from Chapter 6: COMP and LR fusion. Each entry is the number of segments, with corresponding time (in s) given in brackets. (The total experiment time being 6013s.) This is the information upon which the visualisation of Figure 8.2 is based. The substitution error pairs, ID, IU, IF, OD, OU and MD, are summarised for the visualisation in a single metric ‘Subst’. The entries ND (read, ‘deletion by NULL’), IN (‘insertion onto NULL’), and so on, are referred to as just ‘deletion’, ‘insertion’, ..., etc.

When merge and fragmenting errors are taken into account, these additional observations can be made:

1. The COMP. method of Figure 8.1 produces an equally low count of merge as it does insertion (both 1.6%). Its count of fragmenting is also low (1.7%), with the bulk of its errors being attributed to deletions (7.4%). This helps confirm the ‘cautious’ nature of this method.
2. In Figure 8.1 LR also produces a low count of fragmentation (1.3%), but with a much larger degree of merging (5.5%).
3. The wrist-worn COMP method of Figure 8.2 shows a similar result to that of Figure 8.1. One difference, however, is that it has a much larger count of fragmenting errors (3.4%).
4. The wrist-worn LR method seems to show the best timing results of all the methods thus far. The major difference from the sound segmented version is that it has less than half the amount of insertion and merge errors. (The main reason for this is the effect of the ‘hard’ thresholding on the troublesome classes drawer and vise that was used for this LR, but not for the sound segmented version.)

### Using the segment error table (SET)

More detail on these results can be obtained by referring to the original tables on which the visualisations are based. Table 8.1 shows the original SETs, reduced to treat NULL as a special case, for the wrist worn, sliding window results shown in Figure 8.2. These record the number of *segments* of each

error, with the absolute times (which were used for the visualisation) given in brackets.

Potentially useful information which can be seen here that is not available in the visualisation (because of space limitations) is the breakup of substitution errors. Specifically, we can see that the bulk of substitution errors for COMP are, in fact, IU errors. This is where one class infringes the border of another: a ‘substitution underfill’.

The LR method, on the other hand, contains instances of IF errors. These are insertions of one class right in the middle of an event of another class: ‘substitution fragmentation’. LR also has a greater number of ID errors - complete substitutions - where one class is deleted and another inserted. These are, for many systems, perhaps the most serious error.

### Class by class event analysis

Leaving the basic SET aside, we now further the analysis by looking at class by class event errors. This is shown, for the sliding window results for COMP and LR, in Table 8.2. The COMP method, for example, deletes many instances of drawer, yet very few of the other classes. These classes do however experience much fragmentation. This corresponds to the high fragmenting level in Figure 8.2. This fragmentation, though frequent, is mostly caused by ‘insertions of *NULL*’. Further, these usually have very short duration.

Based on this, it might be decided to use COMP for a system which requires a low number of false positive errors, but which tolerates the occasional, short duration fragmentation.

According to Table 8.2, LR produces just over half as many fragmenting errors as COMP. This is in keeping with the timing report of Figure 8.2. The count of insertions and merge errors, however, is nearly double that of COMP. This differs from what the timing report might suggest because of the short duration of many of these errors.

A system where this configuration would be most suited, therefore, would be one which has many longer duration events (1-2 seconds) that should be detected contiguously (with minimal fragmentation).

A further interesting observation is the results given for the drawer and vise classes of both methods in Table 8.2. These classes are unique in that they are both fairly short in duration and that they always occur in pairs: the opening of a drawer is always followed by a short pause before being closed again; and similarly the opening of a vise to place a piece of wood is usually accompanied by the vise being tightened again to hold it. Consequently, these are the only two classes which experience merge errors. In a similar way, as their duration is so short, these classes rarely exhibit fragmenting errors.

This is a facet of the dataset, and raises the (somewhat obvious) importance of selecting not only appropriate measures, but an appropriate dataset

Class	T	I	M	D	F	C
<i>NULL</i>	740	85	120	66	16	658
hammer	20	1	0	0	3	17
saw	20	1	0	1	6	13
file	20	7	0	1	10	9
drill	20	1	0	0	15	5
sand	20	0	0	0	6	14
grind	20	0	0	2	4	14
screw.d.	20	26	0	1	6	13
vise	160	16	7	33	5	122
drawer	440	14	34	139	0	301
Pos.	740	66	41	177	55	508

Class	T	I	M	D	F	C
<i>NULL</i>	740	42	113	86	14	640
hammer	20	4	0	0	0	20
saw	20	10	0	1	5	14
file	20	11	0	2	7	11
drill	20	6	0	0	3	17
sand	20	9	0	0	3	17
grind	20	2	0	0	1	19
screw.d.	20	26	0	1	6	13
vise	160	33	13	28	4	128
drawer	440	14	34	139	0	301
Pos.	740	115	47	171	29	540

**Table 8.2:** Event error tables for (left) COMP, and (right) LR methods corresponding to the frame results in Figure 8.2 (using wrist-worn, sliding window segmentation). Total number of events ( $T$ ), insertions ( $I$ ), merges ( $M$ ), deletions ( $D$ ), fragmentation ( $F$ ), and correct ( $C$ ) are given for each class. Also given are the results when *NULL* is treated as a class, and the total positive class results (*Pos.*) Note the greater number of fragmentation for ‘cautious’ COMP, and the higher number of insertions for LR.

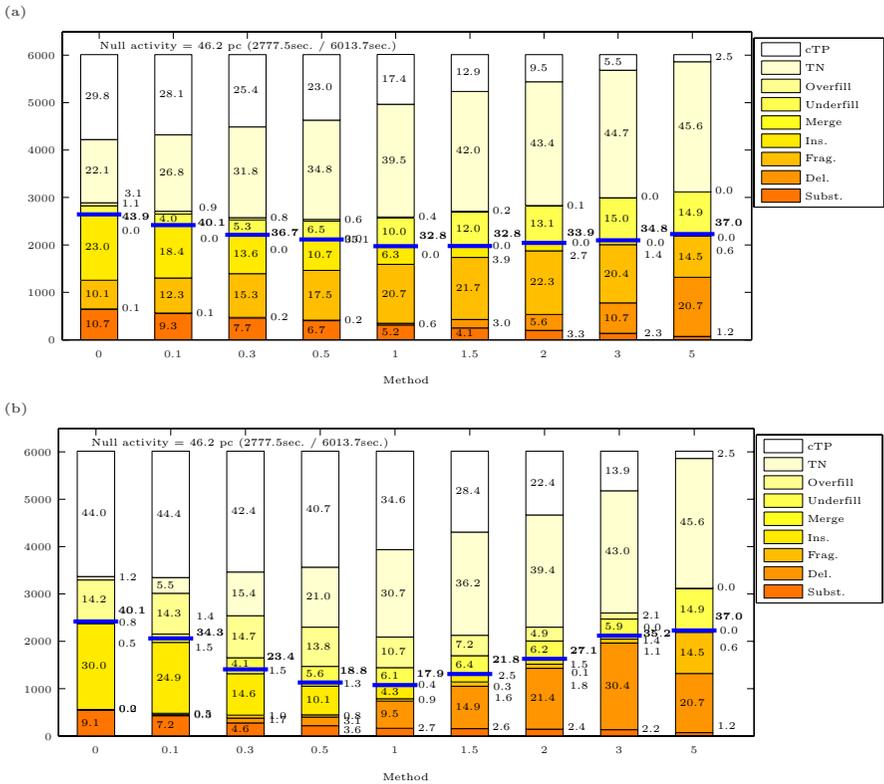
for whichever application is being developed. Any investigation into merge errors, therefore, would require a focus on short duration, frequently occurring classes such as these. Conversely, investigation of fragmenting errors requires classes of fairly long duration (such as represented here by the other activity classes).

## 8.2. Parameter optimisation with SET

In Chapter 5 the response of the sound based segmentation was optimised using a plot of the precision and recall across values across a sweep of the IA threshold  $T_{ia}$ . In the analysis presented, the main criteria was for a high recall - on the premise that later fusion would improve precision. However, the use of precision and recall meant that overflow and underfill were not considered. Equally, the effect of  $T_{ia}$  on levels of the error categories was not investigated.

As an example of how a SET analysis might be used to investigate the effects of system parameters, Figure 8.3 presents the error barcharts for two of the sound recognition setups used earlier in the thesis. These make a sweep across nine different settings of  $T_{ia}$ , from 0 to 5. The first graph (a) presents results for a partitioning and recognition method using only IA+LDA classification on frames. In (b), the frame classifications are smoothed over using a majority vote sliding window in a manner identical to that employed in 5.5.3.

Immediately it can be seen how the first method (a) has a consistently high level of fragmentation across all thresholds, in comparison to the low levels of (b). Even though (a) has very few deletions, it still retains less insertion time than the smoothed version of (b). These observations, not immediately evident in a precision-recall analysis, provide useful information that might allow a designer to further fine-tune system parameters for different application needs. For example, knowing that many of the false negatives in (a) actually form part of fragmentating errors rather than deletions, might allow choice of such a setup for an application where a fragmented output is more acceptable than one where events are deleted.



**Figure 8.3:** Sound analysis across a sweep of  $T_{ia}$ , for (a) using LDA classification, and (b) using LDA plus majority vote smoothing. Note the prevalence of fragmenting errors in (a) over (b)

### 8.3. Conclusion

Segment Error Table (SET) analysis provides a mechanism by which researchers of activity recognition can more completely assess the performance of their systems. This allows easier optimisation, or fine tuning, of system parameters for specific applications. The information provided by SET goes over and above that provided by more traditional methods, such as confusion matrices and standard ASR-based event counts.

Though the basic conclusions drawn in the earlier chapters of the thesis still hold, the use of SET provides a more thorough analysis of those results. It allows researchers to take issues such as fragmentation and merging into consideration and provides a means by which these can be summarised.

# 9

## Conclusion

*This chapter concludes the thesis with a summary for each of the two topics dealt with: continuous activity recognition and performance evaluation. For each topic we summarise the main contributions presented in the thesis. We discuss the limitations and relevance of the methods used and give a brief outlook of the ongoing and future work.*

## 9.1. Continuous activity recognition

### Contributions of the thesis

In addressing the three primary questions of 1.2.2, we make the following contributions to the continuous activity recognition problem (much of the work presented here can also be found in [74, 81, 83, 91]):

1. *Classification of sound data from wrist-worn microphone.* Using Linear Discriminant Analysis (LDA) classification on isolated sound data, average recognition rates varied between 96% for user-dependent, and 77% for user-independent.
2. *Classification of motion data from arm and wrist mounted accelerometers.* Using Hidden Markov Models on extracted accelerometer features, average recognition rates (in isolation) varied between 94% for user-dependent, and 72% for user-independent.
3. *Detection of activities using sound from two microphones.* We present and evaluate a method which uses the differences in sound intensity from two microphones - one at the wrist, the other on the upper arm - to detect sounds produced close to the hand. This method works well providing the activities being performed produce some noise close to the user's hand. From an initial frame-by-frame evaluation, recall rates of around 70% versus 30% false positive were achieved during the use of handheld and machine tools.
4. *Continuous recognition using sound only.* The two microphone segmentation method is enhanced by first classifying the detected frames of sound (using LDA classification), and then running a larger (1.5 second), majority vote window over the data. This produces a smoother, contiguous segment output. With the LDA classification, this method provides a complete activity recognition system by itself. In the user-dependent case, average class-relative scores of 76% recall and 62% precision were obtained. For the user-independent case, however, recall dropped to 61% and precision to 51%.
5. *Fusion of classifiers in isolation.* The combined use of acceleration and sound provides a means of improving recognition performance substantially. We evaluate several different methods for doing this: a basic comparison of classifier outputs (COMP); and three methods based on fusion of classifier rankings - highest rank, Borda count, and a method based on Logistic Regression (LR). In isolated testing, the top performing method (LR) achieves average recognition rates of between 98% (user-dependent) and 87% (user-independent).

6. *Fusion of classifiers on continuous data.* By using the detected segments from sound analysis, we perform independent LDA and HMM classifications, and combine the results. Compared with basic sound recognition, the LR method reports improvements in accuracy of between 10% (user-dependent case) and 16% (user-independent case).
7. *Continuous recognition for wrist-worn device.* Finally we show the feasibility of building a continuous recognition system using only wrist-worn sensors. Instead of using the sound based segmentation strategy described above, we simply run a two second sliding window across the data, classifying the different sensors at each step. We then perform comparison of the classifications (using LR and COMP), to produce the final recognition output. We show that it is possible to replace the HMM and LDA with simpler, less computationally expensive, classifiers. Specifically, we use a Naive Bayes (NB) classifier for acceleration, and a reduced frequency LDA for sound. The resulting system, though requiring approximately 17 times less computation time than the previous system, has an accuracy of 69% (only 2.9% lower). Discarding overflow and underfill, the method produces only 9.7% ‘serious’ errors.

## Limitations and relevance

The experiments presented in this thesis are intended as an initial exploration of continuous activity recognition using on-body sensing. In particular, we focus on activities that correspond to characteristic gestures and sounds. While the experiments involved a single, “mock” scenario, they provide insight and possible direction for future wearable continuous activity recognition systems. The assembly procedure involved a diverse selection of realistic activities performed by several subjects, and these activities represent a broad range of different types of sound and acceleration signatures.

Clearly, the specific sound and acceleration combination used here might not be useful for gestures which produce little or no sound, such as in sign language (Brashear *et al.*, for example, tackle this task using accelerometers[104]).

Domains where there is both sound and motion, however, particularly those involving the use of some object or tool, are well suited to the methods presented here.

The wearable computing lab at ETH Zurich, together with UMIT Innsbruck, have used similar sound recognition methods for recognising household activities [89] and the analysis of chewing sounds [105]. Again with UMIT, accelerometers have been used to recognising key steps in bicycle repair [90], and everyday activities such as opening doors or answering the phone ([106]). We are currently working on a system that uses the same methods - acceleration and sound - to recognise activities in a cooking scenario. Given the

results of these studies, we are optimistic that the techniques presented here will be valuable in other domains.

### Future and ongoing work

We are pursuing this work in three main directions: (1) further algorithmic improvements, (2) the use of different sensor combinations, and (3) implementation and evaluation in “real-life” applications.

We wish to add a segmentation algorithm to the acceleration analysis and apply sensor fusion at both the classification and segmentation levels. Initial work in this direction is described in [91, 106]. We will also improve our features, particularly for acceleration, as it is clear that the information available from the arm and hand may be better combined for activity discrimination. More detailed analysis of the sub-sequences of actions that compose the wood workshop activities should also yield improvements in performance. For example, the components of using the drill press could be modelled as “switch on,” “drill,” and “switch off.” Recognising these sub-activities separately within the structure of an expectation grammar should improve the results of recognising the activity as a whole (such as suggested by Minnen *et al.*[107]).

We are studying the use of ultrasonic hand tracking as a substitute for sound analysis for signal segmentation. Initial results on the utility of ultrasonic hand tracking have been described by Ogris *et al.*[90]. RFID readers to identify tools and more complex inertial sensors such as gyros and magnetometers are being investigated as additions to the system described here.

Currently the groups at ETH and UMIT are involved in a number of projects where the concepts described in this thesis are being applied to “real-life” applications. In the WearIT@Work project, sponsored by the European Union, activity recognition is being implemented for a car assembly training task. Similar systems are planned for aircraft maintenance. In a project sponsored by the Austrian regional government of Tirol, recognition of household activities is being pursued using wrist mounted accelerometers, microphones, and other sensors. This system is envisioned as a component for assistive systems for the cognitively disabled.

While intended as an initial experiment in activity recognition using on-body sensing, this effort has provided tools and direction for several new projects and highlighted difficulties in both performing experiments and communicating results. As the field of on-body activity recognition matures, we hope to continue to reduce the complexity of the problem using multimodal sensing and data fusion.

## 9.2. Performance evaluation (and optimisation)

### Contributions of the thesis

Addressing the questions of 1.2.2 we present the following contributions to the problem of finding suitable performance measures for continuous context recognition. (The first four contributions also appear in [92], the main ideas of which have previously been presented at a workshop in [108]):

1. *Problems with existing evaluation methods.* We present an investigation into a typical recognition problem in continuous context recognition and show that many aspects of performance that might be regarded as important are not captured by existing evaluation methods.
2. *Scoring event errors.* We present a method for un-ambiguously scoring event errors in continuous context recognition. Specifically, we define error types according to whether they are a facet of the ground truth, such as a deletion or fragmentation (negative errors); or of the prediction sequence, such as an insertion or merge (positive errors).
3. *Scoring timing errors* We present a method of assigning timing errors to events, introducing the concepts of overfill and underfill to specifically record cases where a correct output event is somehow offset from its ground truth.
4. *Segment error tables (SET).* We introduce a timewise evaluation, analogous to the standard frame-by-frame prediction and ground truth comparison, which takes into account the event nature of typical context system outputs. Specifically, the method is based on segmentation of the prediction and ground truth sequences in such a way as to allow an unambiguous one-to-one relation between the positive and negative error types as defined by the event analysis. From this we have shown there to be a maximum of six possible segment error types, which we codify using the Segment Error Table (SET).
5. *Practical application of SET.* Finally, we show how SET might be used for optimising system parameters and for evaluating the overall performance. To show this, we apply the method to example results taken from the earlier chapters in the thesis. We show how the new analysis provides a deeper understanding of the earlier presented results.

### Limitations and relevance

Though the categorisation of performance errors presented in this work strives to be complete - at least as far as the problems of continuous activity recognition are concerned - there is a chance that some applications might experience

errors not captured by a SET analysis. However it is our belief, based on the experiences of our groups within the field and repeated indications from publications, that this strategy should capture the most critical errors generally experienced in continuous context recognition.

A key point introduced in this thesis is that existing methods for performance evaluation are often insufficient for context. If nothing else therefore, the minimum contribution of this work should be to stimulate discussion within the context recognition community on the problems of how to adequately evaluate new systems.

### **Future and ongoing work**

Showing a multi-class SET on its own might provide the fullest account of error information in a system, but, as with the problems of showing confusion matrices, the fullest amount of information is not always the easiest to read and has a tendency obscure relevant results. For this reason, a promising future work is in the development of measures to summarise key results from SET. These might be analogous to measures such as recall, precision, sensitivity, etc.

Finally, in order to fully evaluate and develop the use of the SET, we plan to apply the strategy to as wide a selection of context experiments as possible, through our own work in context (and specifically, activity) recognition, and in cooperation with other groups.

# Glossary

## Symbols

$\alpha$	LR coefficient
$\beta$	LR coefficient
$c$	class label
$\#Classes$	number of classes
$d$	distance
$\delta$	change in distance
$D_e$	count of ASR-style deletion events
$err_e$	ASR-style event error rate
$f_{cutoff}$	cutoff frequency
$f_s$	sample rate
$g$	gravity ( $\approx 10ms^{-2}$ )
$I_e$	count of ASR-style inserted events
$I_1, I_2$	signal intensity at (1) wrist and (2) upper arm
$j_{ia}$	sliding intensity analysis window jump length
$j_{lda}$	LDA sliding window jump length
$L(X)$	logistic regression function
$T_e$	total number of events
$T_f$	total number of frames
$T_{ia}$	intensity ratio difference threshold
$S_e$	count of ASR-style substitution events
$s_i$	segment
$R_{lda}$	LDA calculation frequency
$w_{ia}$	sliding intensity analysis window size
$w_{lda}$	LDA and FFT window size

## Abbreviations

ASR	automatic speech recognition
C	correct event
cTP	correct True Positive
COMP	agreement comparison method
D, Del.	deletion
F	fragmentation
FFT	fast fourier transform
FN	false negative frame count
FP	false positive frame count
<i>fp</i>	false positive rate
HMM	hidden markov models
HR	highest rank
Hz	hertz
I, Ins.	insertion
IA	intensity analysis (algorithm)
ID	insertion-deletion segment
IF	insertion-fragmenting segment
IR	information retrieval
IU	insertion-underfill segment
kNN	k-nearest neighbor classifier
LDA	linear discriminant analysis
LR	logistic regression
M	merge
MD	merge-deletion segment
MSET	multi-class segment error table
n.d.	not defined
N	<i>NULL</i> frame count
N.B.	Naive Bayes
O	overflow
OCR	optical character recognition
OD	overflow-deletion segment

OU	overflow-underfill segment
P	positive class frame count
%P	percentage precision
PC	personal computer
PCA	Principle Component Analysis
PkCnt	peak count
PR	precision-recall
%R	percentage recall
RFID	radio frequency identification
ROC	receiver operator characteristics
SET	segment error table
subst.	substitution frame count
TN	true negative frame count
TP	true positive frame count
U	underfill



## Bibliography

- [1] S. Feiner, B. MacIntyre, and D. Seligmann. Knowledge-based augmented reality. *Com. of the ACM*, 36(7):52–62, 1993.
- [2] H.Junker, J.A.Ward, P.Lukowicz, and G.Tröster. *Benchmarks and a Data Base for Context Recognition*. ISBN 3-9522686-2-3, 2004.
- [3] D. Abowd, A. K. Dey, R. Orr, and J. Brotherton. Context-awareness in wearable and ubiquitous computing. *Virtual Reality*, 3(3):200–211, 1998.
- [4] Thomas B. Moeslund and Erik Granum. A survey of computer vision-based human motion capture. 81(3):231–268, 2001.
- [5] A. Kale, A.N. Rajagopalan, N. Cuntoor, and V. Krueger. Gait-based recognition of humans using continuous hmms. In *Int'l Conf. on Face and Gesture Recognition*, 2002.
- [6] A. Ali J.K. Aggarwal. Segmentation and recognition of continuous human activity. In *IEEE Workshop on detection and recognition of Events in Video*, pages 28–35, Vancouver, Canada, 2001.
- [7] Jens Rittscher and Andrew Blake. Classification of human body motion. In *ICCV, proceedings*, pages 634–639, 1999.
- [8] Antonio Torralba, Kevin Murphy, William Freeman, and Mark Rubin. Context-based vision system for place and object recognition. In *Intl. Conf. on Computer Vision*, 2003.
- [9] Brian Clarkson. *Life Patterns: Structure from Wearable Sensors*. PhD thesis, Massachusetts Institute of Technology, 2002.
- [10] T. Starner, B. Schiele, and A. Pentland. Visual contextual awareness in wearable computing. In *Proc. IEEE Int'l Symp. on Wearable Comp.*, pages 50–57, Pittsburgh, PA, 1998.
- [11] C. Vogler and D. Metaxas. ASL recognition based on a coupling between HMMs and 3D motion analysis. In *ICCV*, Bombay, 1998.
- [12] A. D. Wilson and A. F. Bobick. Learning visual behavior for gesture analysis. In *Proc. IEEE Int'l. Symp. on Comp. Vis.*, Coral Gables, Florida, November 1995.

- [13] J. Schlenzig, E. Hunter, and R. Jain. Recursive identification of gesture inputs using hidden Markov models. In *2nd Conf. on Applications of Comp. Vision*, pages 187–194, Dec. 1994.
- [14] J. M. Rehg and T. Kanade. Digiteyes: vision-based human hand tracking. Technical report, Carnegie Mellon, Dec 1993.
- [15] J. B. J. Bussmann, W. L. J. Martens, J. H. M. Tulen, F.C. Schasfoort, H. J. G. van den Berg-Emons, and H.J. Stam. Measuring daily behavior using ambulatory accelerometry: The activity monitor. *Behavior Research Methods, Instrmts.+Comp.*, 33(3):349–356, 2001.
- [16] P. Bonato. Advances in wearable technology and applications in physical and medical rehabilitation. *J. NeuroEngineering and Rehabilitation*, 2(2), 2005.
- [17] A. Salarian, H. Russmann, F.J.G. Vingerhoets, C. Dehollain, Y. Blanc, P.R. Burkhard, and K Aminian. Gait assessment in parkinson’s disease: toward an ambulatory system for long-term monitoring. *Biomedical Engineering, IEEE Trans.*, 51(8), aug 2004.
- [18] B. Najafi, K. Aminian, A. Paraschiv-Ionescu, F. Loew, C.J. Bula, and P. Robert. Ambulatory system for human motion analysis using a kinematic sensor: monitoring of daily physical activity in the elderly. *Biomedical Engineering, IEEE Trans.*, 50(6), Jun 2003.
- [19] Masaki Sekine, Toshiyo Tamura, Toshiro Fujimoto, and Yasuhiro Fukui. Classification of walking pattern using acceleration waveform in elderly people. In *Proceedings of th 22nd EMBS*, 2000.
- [20] M. Uiterwaal, E.B. Glerum, H.J. Busser, and R.C. van Lummel. Ambulatory monitoring of physical activity in working situations, a validation study. *J Med Eng Technol.*, 22(4):168–72, 1998.
- [21] P.H. Veltink, H.B.J. Bussmann, W. de Vries, W.L.J. Martens, and R.C. van Lummel. Detection of static and dynamic activities using uniaxial accelerometers. *IEEE Trans. Rehab. Eng.*, 4(4):375–386, 1996.
- [22] K. Aminian, P. Robert, E.E. Buchser, B. Rutschmann, D. Hayoz, and M. Depairon. Physical activity monitoring based on accelerometry: validation and comparison with video observation. *Med Biol Eng Comput.*, 37:304–8, 1999.
- [23] M.L. Wetzler, J. R. Borderies, O. Bigaignon, P. Guillo, and P. Gosse. Validation of a two-axis accelerometer for monitoring patient activity during blood pressure or ecg holter monitoring. *Clinical and Pathological Studies*, 2003.

- [24] K. Aminian and B. Najafi. Capturing human motion using body-fixed sensors: outdoor measurement and clinical applications. *Computer Animation and Virtual Worlds*, 15:79–94, 2004.
- [25] J. Mantyjarvi, J. Himberg, and T. Seppanen. Recognizing human motion with multiple acceleration sensors. In *Proc. IEEE Int'l Conf. on Sys., Man. and Cybernetics*, volume 2, pages 747–752, 2001.
- [26] C. Randell and H. Muller. Context awareness by analysing accelerometer data. In *Proc. IEEE Int'l Symp. on Wearable Comp.*, pages 175–176, 2000.
- [27] K. Van-Laerhoven and O. Cakmakci. What shall we teach our pants? In *Proc. IEEE Int'l Symp. on Wearable Comp.*, pages 77–83, 2000.
- [28] Elena Vildjiounaite, Esko-Juhani Malm, Jouri Kaartinen, and Petteri Alahuhta. Location estimation indoors by means of small computing power devices, accelerometers, magnetic sensors and map knowledge. In *Pervasive*, 2002.
- [29] Tracy Westeyn, Kristin Vadas, Xuehai Bian, Thad Starner, and Gregory D. Abowd. Recognizing mimicked autistic self-stimulatory behaviors using hmms. In *Proc. IEEE Int'l Symp. on Wearable Comp.*, pages 164–169, 2005.
- [30] L. Bao and S. Intille. Activity recognition from user-annotated acceleration data. In *Pervasive, LNCS 3001*, 2004.
- [31] Graeme S. Chambers, Svetha Venkatesh, Geoff A. W. West, and Hung H. Bui. Hierarchical recognition of intentional human gestures for sports video annotation. In *Int'l Conf. on Pattern Recognition*, 2002.
- [32] S. Antifakos, F. Michahelles, and B. Schiele. Proactive instructions for furniture assembly. In *4th Int'l Conf. UbiComp*, page 351, Gteborg, Sweden, 2002.
- [33] G. Fang, W. Gao, and D. Zhao. Large vocabulary sign language recognition based on hierarchical decision trees. In *Intl. Conf. on Multimodal Interfaces*, Vancouver, BC, Nov. 2003.
- [34] Holger Junker. *Human Activity Recognition and Gesture Spotting with Body-Worn Sensors*. Hartung-Gorre, PhD thesis, ETH Zurich, 2005.
- [35] V. Peltonen, J. Tuomi, A. Klapuri, J. Huopaniemi, and T. Sorsa. Computational auditory scene recognition. In *Proc. IEEE Int'l Conf. on Acoustics, Speech, and Signal Processing*, volume 2, pages 1941–1944, May 2002.

- [36] M.C. Buehler. *Algorithms for Sound Classification in Hearing Instruments*. PhD thesis, ETH Zurich, 2002.
- [37] James Scott and Boris Dragovic. Audio location: Accurate low-cost location sensing. In *Pervasive, LNCS*, volume 3468, pages 1–18, 2005.
- [38] Jianfeng Chen, Alvin Harvey Kam, Jianmin Zhang, Ning Liu, and Louis Shue. Bathroom activity monitoring based on sound. In *Pervasive, LNCS*, volume 3468, pages 47–61, 2005.
- [39] B. Clarkson, N. Sawhney, and A. Pentland. Auditory context awareness in wearable computing. In *Workshop on Perceptual User Interfaces*, November 1998.
- [40] C.C. Chibelushi, F. Deravi, and J.S. Mason. Audio-visual person recognition: an evaluation of data fusion strategies. In *ECOS 97*, pages 26–30, April 1997.
- [41] N. Kern, H. Junker, P. Lukowicz, B. Schiele, and G. Tröster. Wearable sensing to annotate meeting recordings. *Personal and Ubiquitous Computing*, 2003.
- [42] Nicky Kern, Albrecht Schmidt, and Bernt Schiele. Recognizing context for annotating a live life recording. *Personal and Ubiquitous Computing*, 2005.
- [43] H. Wu and M. Siegel. Correlation of accelerometer and microphone data in the coin tap test. *IEEE Trans. Instrumentation and Measurement.*, 49(3):493–497, June 2000.
- [44] Kristof Van Laerhoven and H.W. Gellersen. Spine versus porcupine: a study in distributed wearable activity recognition. In *In Proc. Int'l. Symp. Wearable Computers*, pages 142–150. IEEE, 2004.
- [45] Kristof Van Laerhoven, H.W. Gellersen, and Yanni Malliaris. Long-term activity monitoring with a wearable sensor node. In *In Proc. of the third International Workshop on Body Sensor Nodes*, pages 171–174. IEEE, 2006.
- [46] A Schmidt, K Van Laerhoven, M Strohbach, A Friday, and HW Gellersen. Context acquisition based on load sensing. In *In Proc. of Ubicomp (LNCS)*, volume 2498, pages 333–351, Sep 2002.
- [47] A. Schmidt and K. Van Laerhoven. How to build smart appliances? *IEEE Personal Communications*, 8(4), Aug 2001.

- [48] Matthai Philipose, Kenneth P. Fishkin, Mike Perkowitz, Donald J. Patterson, Dieter Fox, Henry Kautz, and Dirk Hähnel. Inferring activities from interactions with objects. *IEEE Pervasive Computing*, pages 50–57, Oct 2004.
- [49] I. Rish. An empirical study of the naive bayes classifier. In *Proceedings of IJCAI*, 2001.
- [50] Pedro Domingos and Michael Pazzani. On the optimality of the simple bayesian classifier under zero-one loss. *Machine Learning*, 29(2), Nov 1997.
- [51] L.R. Rabiner and B.H. Juang. An introduction to hidden Markov models. *IEEE ASSP Magazine*, pages 4–16, Jan 1986.
- [52] Kevin P. Murphy. Dynamic bayesian networks. In M. Jordan, editor, *Probabilistic Graphical Models*. 2002.
- [53] A. Nefian, L. Liang, X. Pi, L. Xiaoxiang, C. Mao, and K.P. Murphy. A coupled hmm for audio-visual speech recognition. In *ICASSP, Int'l Conf on Acoustics, Speech and Signal Proc.*, pages 2013–2016. IEEE, 2002.
- [54] A. Nefian, L. Liang, X. Pi, X. Liu, and K. Murphy. Dynamic bayesian networks for audio-visual speech recognition. In *Journal of Applied Signal Processing*, pages 1–15. EURASIP, 2002.
- [55] G. Potamianos and C. Neti. Automatic speechreading of impaired speech. In *Int'l Conf. on Auditory-Visual Speech Proc.*, pages 177–182, 2001.
- [56] J. Kittler, M. Hatef, R.P.W. Duin, and J. Matas. On combining classifiers. *IEEE Trans. PAMI*, 20(3):226–239, March 1998.
- [57] L. Xu, A. Kryzak, and C.Y. Suen. Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE Trans. Systems, Man., and Cybernetics*, 22(3):418–435, May/June 1992.
- [58] T. Denoeux. A neural network classifier based on dempster-shafer theory. *IEEE transactions on Systems, Man and Cybernetics*, 30(2):131–150, 2000.
- [59] S. Le Hgarat-Masclé and C. Ottl D. Richard. Multiscale data fusion using demspster-shafer evidence theory. *IEEE transactions on Systems, Man and Cybernetics*, 10(1):9–22, 2003.

- [60] Huadong Wu. *Sensor Fusion Architecture for Context-Aware Computing*. PhD thesis, Carnegie Mellon, 2003.
- [61] H. Wu, M. Siegel, R. Stiefelhagen, and J. Yang. Sensor fusion using dempster-shafer theory. In *Proceedings of IEEE Instrumentation and Measurement Technology Conference, Anchorage, AK, USA, May 21-23, 2002*, May 2002.
- [62] H. Wu, M. Siegel, and S. Ablay. Sensor fusion for context understanding. In *Proceedings of IEEE Instrumentation and Measurement Technology Conference, Anchorage, AK, USA, May 21-23, 2002*, May 2002.
- [63] K. Sentz and S. Ferson. Combination of evidence in dempster-shafer theory. Technical Report 2002-0835, SAND Epistemic Uncertainty Project, 2002.
- [64] T.K. Ho. Multiple classifier combination: Lessons and next steps. In *Hybrid Methods in Pattern Recognition, World Scientific*, 2002.
- [65] T.K. Ho, J.J. Hull, and S.N. Srihari. Decision combination in multiple classifier systems. *IEEE Trans. PAMI*, 16(1):66–75, Jan 1994.
- [66] A. Hoover, G. Jean-Baptiste, X. Jiang, P. Flynn, H. Bunke, D. Goldof, K. Bowyer, D. Eggert, A. Fitzgibbon, and R. Fisher. An experimental comparison of range image segmentation algorithms. *IEEE Trans. PAMI*, 18(7):673–689, 1996.
- [67] H. Müller, W. Müller, D. McG. Squire, S. Marchand-Maillet, and T. Pun. Performance evaluation in content-based image retrieval: Overview and proposals. Technical report, Uni. Geneve, Switzerland, 1999.
- [68] Y.J. Zhang. A review of recent evaluation methods for image segmentation. In *Intl. Symp. on Signal Processing and its Applications*, Aug 2001.
- [69] J. R. Pierce. Wither speech recognition. *J.Acoust. Soc. Amer.*, 46:1049–1051, 1969.
- [70] Roger K. Moore. Evaluating speech recognizers. *IEEE Trans. Acoust. Speech and Sig.Proc.*, 25(2):178–183, 1977.
- [71] S. Greenberg. Whither speech technology? - a twenty-first century perspective. In *7th Intl. Conf. on Speech Com. and Tech. (Eurospeech)*, pages 3–6, 2001.

- [72] E. Keogh and S. Kasetty. On the need for time series data mining benchmarks: a survey and empirical demonstration. In *8th int'l. conf. on knowledge discovery and data mining*, pages 102–111, NY, USA, 2002. ACM.
- [73] N. Kern, B. Schiele, H. Junker, P. Lukowicz, and G. Tröster. Wearable sensing to annotate meeting recordings. In *Proc. IEEE Int'l Symp. on Wearable Comp.*, pages 186–193, October 2002.
- [74] P. Lukowicz, J.A. Ward, H. Junker, G. Tröster, A. Atrash, and T. Starner. Recognizing workshop activity using body worn microphones and accelerometers. In *Pervasive, LNCS*, 2004.
- [75] M. Stäger, P. Lukowicz, N. Perera, T.v. Büren, G. Tröster, and T. Starner. Soundbutton: Design of a low power wearable audio classification system. In *Proc. IEEE Int'l Symp. on Wearable Comp.*, 2003.
- [76] R. Duda, P. Hart, and D. Stork. *Pattern Classification, 2nd Edition*. Wiley, 2001.
- [77] Jeff Bilmes. What hmms can do. Technical Report 2002-0835, Dept or EE, University of Washington, 2002.
- [78] T. Starner, J. Makhoul, R. Schwartz, and G. Chou. On-line cursive handwriting recognition using speech recognition methods. In *ICASSP*, pages 125–128, 1994.
- [79] Ira Cohen, Ashutosh Garg, and Thomas S. Huang. Emotion recognition from facial expressions using multilevel hmm. Technical report, Institute for Advanced Science and Technology, Univ. of Illinois, 2000.
- [80] K.P. Murphy. The hmm toolbox for MATLAB, <http://www.ai.mit.edu/~murphyk/software/hmm/hmm.html>, 1998.
- [81] J.A. Ward, P. Lukowicz, and G. Tröster. Roc analysis of partitioning method for activity recognition using two microphones. In *Adjunct Proc. of the 3rd Int. Conf. on Pervasive Comp.*, volume 191, May. 8-13 2005.
- [82] T. Fawcett. *ROC Graphs: Notes and Practical Considerations for Researchers*. Kluwer, 2004.
- [83] J.A. Ward, P. Lukowicz, G. Tröster, and T. Starner. Activity recognition of assembly tasks using body-worn microphones and accelerometers. *To appear in IEEE Trans. Pattern Analysis and Machine Intelligence*, 2006.
- [84] D. Black. *The Theory of Committees and Elections, 2nd Ed.* Cambridge University Press, 1963.

- [85] E.M. Tapia, S.S. Intille, and K. Larson. Activity recognition in the home using simple and ubiquitous sensors. In *Pervasive, LNCS 3001*, pages 158–175, 2004.
- [86] F. Provost, T. Fawcett, and R. Kohavi. The case against accuracy estimation for comparing induction algorithms. In *IMLC, 15th Int'l Conf.*, 1998.
- [87] I.T. Phillips and A.K. Chhabra. Empirical performance evaluation of graphics recognition systems. *IEEE Trans. PAMI*, 21:9:849–870, 1999.
- [88] C.J. van Rijsbergen. *Information Retrieval, 2nd edition*. Dept. of Computer Science, University of Glasgow, 1979.
- [89] M. Stäger, P. Lukowicz, and G. Tröster. Implementation and evaluation of a low-power sound-based user activity recognition system. In *Proc. IEEE Int'l Symp. on Wearable Comp.*, 2004.
- [90] G. Ogris, T. Stiefmeier, H. Junker, P. Lukowicz, and G. Tröster. Using ultrasonic hand tracking to augment motion analysis based recognition of manipulative gestures. In *Proc. IEEE Int'l Symp. Wearable Comp.*, 2005.
- [91] J.A. Ward, P. Lukowicz, and G. Tröster. Gesture spotting using wrist worn microphone and 3-axis accelerometer. In *Soc-Eusai '05, Proceedings*, Oct. 12-14 2005.
- [92] J.A. Ward, P. Lukowicz, and G. Tröster. Evaluating performance in continuous context recognition using event-driven error characterisation. In Mike Hazas, John Krumm, and Thomas Strang, editors, *LoCA*, volume 3987 of *Lecture Notes in Computer Science*, pages 239–255. Springer, 2006.
- [93] A.F. Clark and C.Clark. Performance characterization in computer vision a tutorial. In <http://www.peipa.essex.ac.uk/benchmark/>, Essex, UK, 1999.
- [94] T. Kanungo, G. A. Marton, and O. Bulbul. Paired model evaluation of ocr algorithms. Technical report, Center for Automation Research, Uni.Maryland, 1998.
- [95] J. Lester, T. Choudhury, N. Kern, G. Borriello, and B. Hannaford. A hybrid discriminative/generative approach for modeling human activities. In *IJCAI*, 2005.

- [96] V. Pavlovic and J. Rehg. Impact of dynamic model learning on classification of human motion. In *Comp. Vision and Pattern Rec. (CVPR)*, pages 788–795, 2000.
- [97] M. Lampe, M. Strassner, and E. Fleisch. A ubiquitous computing environment for aircraft maintenance. In *ACM symp. on Applied comp.*, pages 1586–1592, 2004.
- [98] S. Eickeler and G. Rigoll. A novel error measure for the evaluation of video indexing systems. In *Int'l. Conf. on Acous., Speech & Sig. Proc.*, Jun 2000.
- [99] W. Hsu, L. Kennedy, C.-W. Huang, S.-F. Chang, C.-Y. Lin, and G. Iyengar. News video story segmentation using fusion of multi-level multi-modal features in trecvid 2003. In *ICASSP*, May 2004.
- [100] NIST. *Proc. of TREC Video Retrieval Evaluation Conference (TRECVID)*. 2003.
- [101] D. Berndt and J. Clifford. Using dynamic time warping to find patterns in time series. In *Proc. KDD Workshop*, pages 359–370, Seattle, WA, 1994.
- [102] R. Agrawal, K.-I. Lin, H. S. Sawhney, and K. Shim. Fast similarity search in the presence of noise, scaling, and translation in time-series databases. In *21st Int'l Conf. on Very Large Data Bases*, pages 490–501, Zurich, CH, 1995. MKaufmann.
- [103] C-S.Perng, H.Wang, S.R.Zhang, and D.S.Parker. Landmarks: a new model for similarity-based pattern querying in time series databases. In *ICDE*, 2000.
- [104] H. Brashear, T. Starner, P. Lukowicz, and H. Junker. Using multiple sensors for mobile sign language recognition. In *Proc. IEEE Int'l Symp. on Wearable Comp.*, pages 45–53, White Plains, NY, 2003.
- [105] O. Amft, H. Junker, and G. Tröster. Detection of eating and drinking arm gestures using inertial body-worn sensors. In *Proc. IEEE Int'l Symp. on Wearable Comp.*, Oct. 2005.
- [106] H. Junker, P. Lukowicz, and G. Tröster. Continuous recognition of arm activities with body-worn inertial sensors. In *Proc. IEEE Int'l Symp. on Wearable Comp.*, pages 188–189, 2004.
- [107] D. Minnen, I. Essa, and T. Starner. Expectation grammars: Leveraging high-level expectations for activity recognition. In *IEEE Proc. Comp. Vision and Pattern Rec.*, June 2003.

- [108] J.A. Ward, P. Lukowicz, and G. Tröster. A categorisation of performance errors in continuous context recognition. In *Proc. IEEE Int'l Symp. on Wearable Comp, Workshop for On-Body Sensing*, 2005.

# Curriculum Vitae

## Personal Information

Jamie A. Ward  
Born 24 March 1979, Glasgow, Scotland  
Citizen of Great Britain

## Education

- Nov. 2001–2006 PhD studies in information technology and electrical engineering (Dr. sc. ETH.) at ETH Zurich, Switzerland
- 1996–2000 BEng(Hons) in Computer Science and Electronics at the University of Edinburgh, Scotland
- 1991–1996 SCE Highers, St. Ninian's High School, Kirkintilloch, Scotland

## Work Experience

- Nov. 2001– Research and teaching assistant at Electronics Laboratory, ETH Zurich, Switzerland
- 2000 – 2001 Analog Circuit Designer, Newlogic Technologies, Lustenau, Austria (1 year)
- Summer 1999 Hardware Engineer (intern.), Edinburgh Designs (small company specialising in the design and construction of precision wave machines), Scotland
- Summer 1998 Technical Support Engineer (intern.), Spectrum Energy and Information Technology (medium sized company involved with the digital logging of well heads for the oil industry, Scotland)
- Summer 1997 Documentation and Web Development (intern.), Xilinx Development Corp. (reconfigurable processor team), Scotland

