Human Activity Recognition for Pervasive Interaction

Thesis by

Cuong, PHAM VAN

In Partial Fulfilment of the Requirements

for the Degree of

Doctor of Philosophy

School of Computing Science

Newcastle University

Newcastle upon Tyne, UK

2012

(Submitted 31 July 2012)

ACKNOWLEDGMENTS

I would like to thank my supervisor Professor Patrick Olivier who has highly motivated me to enter the world of academic research, inspired me to work on a very interesting research topic, and enthusiastically provided me constant guidance during my academic journey. Patrick was really a fantastic, warm, caring, and extremely supportive supervisor. He blows me away with his amazing research ideas and knowledge that I learnt a lot from him. He ensured my continuous research funding which allowed me to focus on my studies. Many thanks, professor Patrick! I am very grateful. I would like also to express my thanks to Dr. Thomas Plötz for his relentless guidance on machine learning. At Culture Lab, Newcastle, I have enjoyed the interactions with my warmest friends and colleagues: Anja Thieme, Paul Dunphy, Tom Bartindale, Rob Comber, Jonathan Hook, Nils Hammerlam, Bin Gao, Santi Phithakkitnukoon, Roisin McNaney, Stephen Lindsay, Karim Ladha, Cassim Ladha, Clare Hooper, Jettie Hoonhout (Philips Research), Selina Sutton, John Vines and many more. Some of them had contributed to some parts of this thesis: major thanks Dan Jackson, John Shearer, and Jurgen Wagner for invaluable help on the development of kitchen utensils and sensors.

I would like to say greatly thank you the Vietnamese people as this PhD work is mainly sponsored by them through the Vietnamese government's 322 project, executed by the Vietnamese Ministry of Education and Training. Although my love country Vietnam is still poor and is in developing, I was well funded, including tuition fees and living expenses for first three years of my PhD study, thanks to the funding of the 322 project. Without this financial support, my PhD work would be impossible.

I forever owe a huge debt from my parents who have made countless sacrifices for bringing up and encouraging me good education. Even when I started studying my PhD in the UK, they were working on the harsh, windy and sunny fields in a small village of the North of Vietnam, where I was born and brought up. I love you so much my Mummy and Daddy.

As I am writing these lines, my wife Nguyen Thi Quynh Hoa, and my two lovely sons Pham Kien and Pham Vuong Quoc Anh have left me for Vietnam few months ago. Their absence has made me realize how much they mean to me. I can't express how much I miss them. This thesis is dedicated to them. Finally, last but not least, thanks to Marie Curie Fellowship (funded through the EU FP7 Marie Curie Balance@Home project), I have a chance to work as a Marie Curie Research Fellow at Philips Research (High tech campus, Eindhoven, the Netherlands) for last six months, where I had a successful PhD viva and enjoyed an unforgettable summer in the "windmill" country.

ABSTRACT

This thesis addresses the challenge of computing food preparation context in the kitchen. The automatic recognition of fine-grained human activities and food ingredients is realized through pervasive sensing which we achieve by instrumenting kitchen objects such as knives, spoons, and chopping boards with sensors. Context recognition in the kitchen lies at the heart of a broad range of real-world applications. In particular, activity and food ingredient recognition in the kitchen is an essential component for situated services such as automatic prompting services for cognitively impaired kitchen users and digital situated support for healthier eating interventions. Previous works, however, have addressed the activity recognition problem by exploring high-level-human activities using wearable sensing (i.e. worn sensors on human body) or using technologies that raise privacy concerns (i.e. computer vision). Although such approaches have yielded significant results for a number of activity recognition problems, they are not applicable to our domain of investigation, for which we argue that the technology itself must be genuinely "invisible", thereby allowing users to perform their activities in a completely natural manner.

In this thesis we describe the development of pervasive sensing technologies and algorithms for fine-grained human activity and food ingredient recognition in the kitchen. After reviewing previous work on food and activity recognition we present three systems that constitute increasingly sophisticated approaches to the challenge of kitchen context recognition. Two of these systems, Slice&Dice and Class-based Threshold Dynamic Time Warping (CBT-DTW), recognize fine-grained food preparation activities. Slice&Dice is a proof-of-concept application, whereas CBT-DTW is a real-time application that also addresses the problem of recognising unknown activities. The final system, KitchenSense is a real-time context recognition framework that deals with the recognition of a more complex set of activities, and includes the recognition of food ingredients and events in the kitchen. For each system, we describe the prototyping of pervasive sensing technologies, algorithms, as well as real-world experiments and empirical evaluations that validate the proposed solutions.

TABLE OF CONTENTS

Aspects of this research have been previously presented in the following publications:

1. Pham, C., Hooper, C., Lindsay, S., Jackson, D., Shearer, J., Wagner, J., Ladha, C., Ladha, K., Plötz, T., Olivier, P. 2012; The Ambient Kitchen: A Pervasive Sensing Environment for Situated Services (demonstration paper). Accepted at the ACM Conference on Designing Interactive Systems (Newcastle, UK, 11-15 June 2012). DIS'2012.

2. Hooper, J., Preston, A., Balaam, M., Seedhouse, P., Jackson, D., Pham, C., Ladha, C., Ladha, K., Plötz, T., Olivier, P. 2012; The French Kitchen: Task-Based Learning in an Instrumented Kitchen. Accepted at the 14th ACM International Conference on Ubiquitous Computing (Pittsburgh, Pennsylvania, United States, 5-8 September 2012). Ubicomp'2012.

3. Wagner, J., van Halteren, A., HoonHout, J., Plötz, T., Pham, C., Moynihan, P., Jackson, D., Ladha, C., Ladha, K., Olivier, P. 2011; Toward a Pervasive Kitchen Infrastructure for Measuring Cooking Competence. In the Proceedings of the 5th International ICST Conference on Pervasive Computing Technologies for Healthcare (Dublin, Ireland, 23-26 May 2011). PervasiveHealth'2011.107-114

4. Visalakshmi, S., Paul, E., Watson, P., Pham, C., Jackson, D., Olivier, P. 2011; Distributed Event Processing for Activity Recognition. In the Proceedings of the 5th ACM International Conference on Distributed Event-Based Systems (New York, NY, 11-14 July 2011). DEBS'2011. 371-372

5. Plötz, T., Pham, C., Olivier, P. 2011; Who is Cooking? Sensor-Based Actor Identification in the Kitchen. In the Proceedings of Pervasive 2011 workshop on Frontiers in Activity Recognition using Pervasive Sensing (San Francisco, California, CA, 12 June 2011). IWFAR'2011. 12-17

6. Plötz, T., Moynihan, P., Pham, C., Olivier, P. 2011; Activity Recognition and Healthier Food Preparation. Book chapter In Activity Recognition in Pervasive Intelligent Environments. Chen, L.; Nugent, C.D.; Biswas, J.; Hoey, J. (Eds.). 1st Edition, 2011, Atlantis Press

7. Hoey, J., Plötz, T., Jackson, D., Monk, A., Pham, C., Olivier, P. 2010; SNAP: SyNdetic Assistance Processes. In the Proceedings of NIPS 2010 workshop on Machine Learning for Assistive Technologies, (Whistler, BC, Canada, 10 December 2010). MLAP'2010

8. Hoey, J., Plötz, T., Jackson, D., Monk, A., Pham, C., Olivier, P. 2010; Rapid Specification and Automated Generation of Prompting Systems to Assist People with Dementia. Pervasive and Mobile Computing: 7(3) 2011. 299-310. DOI: http://dx.doi.org/10.1016/j.pmcj.2010.11.007

9. Pham, C., Plötz, T., Olivier, P. 2010; A Dynamic Time Warping Approach to Real-Time Activity Recognition for Food Preparation. In the Proceedings of the 1th International Joint Conference on Ambient Intelligence. (Malaga. Spain, 10-12 November 2010). AmI'2010. 21-30

10. Pham, C., Olivier, P. 2009; Slice&Dice: Recognizing Food Preparation Activities Using Embedded Accelerometers. In the Proceedings of the European Conference on Ambient Intelligence (Salzburg, Autria, 18 – 21 November 2009). AmI '2009. 34-43

LIST OF FIGURES

LIST OF TABLES

Chapter 1: Introduction

This thesis addresses two challenges: the development of pervasive sensing technologies for the kitchen; and development of methods for context recognition in the kitchen. Pervasive interaction, a term refering to interactions with pervasive computing technologies [97], or in the other words, technologies that are pervasively embedded and subjectively hidden into object surroundings. We borrowed the definition of "pervasive interaction" from this to our context: interaction between human and pervasive sensing technologies in which the technologies themselves are invisible to the users, and systems proactively support users in their natural interactions by leveraging of automatically sensed contextual information. Whilst in the kitchen there are many sources of contextual information, our primary concern is the development of a pervasive sensing and a context recognition system aware of food preparation tasks, and we therefore concern ourselves with the problem of recognising human food preparation activities and food ingredients. This chapter will describe our motivation for concerning ourselves with context recognition in the kitchen, and in particular the problems of human activity and food ingredient recognition using pervasive sensing. We then outline goals of the research and conclude with an outline of the structure of the thesis.

1.1 The need for context recognition in the kitchen

The number of older people with cognitive impairments in the UK surpassed 800,000 in 2010 and providing for their care has been estimated to cost the UK economy £23 billion a year [1]. The demand for the development of technology that supports such people during their activities of daily living (ADL) is now an officially recognised priority in the UK [42] and many other nations. The development of such technology and its effective deployment in people's homes has real potential to provide age-related impaired people with a more autonomous lifestyle, while at the same time reducing the financial burden on the state and these people and their families. The kitchen plays an important role in people's lives, as it is an indispensable place where many activities of daily living, such as cooking and food preparation, take place. Although the pervasive computing research community has recently made significant advances in the provision of real-world applications in healthcare, research into technology driven support for cooking and food preparation activities is still in its infancy.

Context recognition in the kitchen has previously been identified as a necessary underpinning technology for a range of situated support services in the kitchen. The scope of such applications is as broad as the

range of people who cook, from teenagers to cognitively impaired older adults. Indeed, existing examples of applications based on context recognition in the kitchen include prompting people with dementia for meal preparation [26, 79], task-based language learning [71] and healthier food preparation [76]. Potential applications include nutrition and healthier eating advice systems, situated cooking support systems, and even meal planning. Cooking is a relatively complex task, involving the use of tools and ingredients and requires significant physical (e.g. skill) and cognitive capability (e.g. planning, monitoring, memory). For certain classes of cognitively impaired people, for example, for people in the early stages of dementia and with mild cognitive impairment, preparing food and drinks is particularly demanding. As Wherton and Monk [42] identified in their comprehensive study of the lives and opinions of people in the early stages of dementia (and their carers) the capability to automatically monitor a users kitchen activities and provide situated prompts was highly desired, i.e., subjects in the study considered that facility to be prompted through food and drink preparation activities would positively impact on their actual independence and prolong the period of time that they could stay in their own home (a serious concern for people in the early stages of dementia and their carers). Such situated services would require the automatic recognition of what the user is doing, what food ingredient is being processed, and when it is done. These lead to the need for the recognition of human activities and food ingredients, two key components of the kitchen context.

1.2 Thesis goals

Although activity recognition problem is a matter of on-going concern for the research community, the problems of recognizing fine grain food preparation activities and food ingredients using pervasive sensing, particularly in real-time, remains an open problem. As we have already described, context recognition in the kitchen is likely to be a fundamental element for situated services to support people's cooking, nutrition and general wellbeing. Tracking the progression of food preparation steps within a recipe, for example, will need to utilize the information of both human activities and food ingredients in real-time to guide or prompt people while they are cooking. The goals of this thesis are therefore to develop solutions to two aspects of this problem food preparation activity recognition and food ingredient recognition.

1.2.1 Activity recognition (AR) in the kitchen

The AR problem in the kitchen is the problem of recognising fine-grained food preparation activities performed by a user during a food preparation and cooking task. Given the requirement placed on pervasive interaction that the technology platforms themselves should not impinge on people's natural engagement in activities we formulate a number of research challenges to be addressed:

1. How to develop pervasive sensing technologies to support the recognition of food preparation activities.
2. How to recognise fine-grained activities from real-time data streams of pervasive sensors that are completely embedded into kitchen utensils and appliances. By *fine-grained* food preparation activities we mean activities which occur over few seconds such as *chopping*, *scooping*, *dicing* in normal recipes. These are distinct from *high-level* activities such as "*making a tomato salad*", "*making tea*" or "*cooking pasta*" which often involve more than one fine-grained activities and occur over longer time periods (minutes).
3. What is the appropriate way to measure the performance of such a recognition system both in terms of the experimental design (the data collection scenarios) and the methodologies (the annotation methods and the evaluation metrics).

The first challenge requires us to explore the design space for sensing technologies that are both technically feasible but also appropriate to the everyday kitchen. Concerns about privacy inevitably steered us away from the general deployment of computing vision technologies and we instead have explored the design space for instrumented utensils and appliances. There is a clear requirement in such a case that whatever instrumented utensils we develop they must allow users (i.e. people) to perform their regular food preparation activities in unobtruted manner.

The second challenge requires us to develop pattern recognition algorithms that can automatically segment and classify human activities from sensor data in real-time. The segmentation of sensing data must appropriate to the most common fine-grained activities and system latency must not place unnecessary constraints on the responsiveness of applications that might depend on activity recognition. Furthermore, approaches adopted should be sensitive to the challenges of collecting large scale annotated training (i.e. the smaller the amount of training data that is required the better).

The final activity recognition challenge is to evaluate whether the recognition system is feasible and reliable for real-world applications. System must be rigorously evaluated using methodologies that are appropriate to their likely deployment. This includes the requirement that the performance of the recognition system must be evaluated on real-world datasets collected in ecologically valid settings.

1.2.2 Food ingredient recognition in the kitchen

Previous approaches to food recognition have been dominated by computer vision, RFID based technology, or acoustic sensing. Each approach has significant shortcomings, not only in relation to actual performance, but also to practicality of deployment (RFID) and privacy (i.e. computer vision). We therefore formulate food ingredient recognition in term number of research challenges:

1. How to develop pervasive sensing technologies that can "sense" food ingredients completely unobtrusively and without raising privacy concerns. And, how the technologies can be made compact and to "look like" natural objects.
2. How to recognise a food ingredient while it is being prepared.
3. What is the appropriate way to measure the performance of a food recognition system both in terms of the experimental design (the data collection scenarios) and the methodologies (the annotation methods and the evaluation metrics).

The distinct challenge here is the development of a new pervasive sensing technology, as existing approaches are either impractical or considered unacceptably invasive [15]. The impracticality of using RFID identification on food ingredients primarily relates to the detection of fresh foods. However, such fresh food are typically prepared with a knife and a chopping board (i.e. for chopping or slicing foods), and as instrumented knives are to be developed to support food preparation activity recognition we sought to leverage these in the development of a chopping board which can detect food ingredients placed on it, both before being chopped as well as while it is being chopped.

1.3 Thesis outline

Chapter 2 review related works and includes an overview of context recognition, particularly human activity and food recognition. We classify activity recognition research into three approaches (wearable, pervasive and pervasive-wearable sensing approaches for AR) and food recognition research into two approaches (computer vision and audio approaches).

Chapter 3 describes our first activity recognition study, the Slice&Dice system, which includes a prototype of 4 kitchen utensils (3 knifes and one large spoon) instrumented with modified Wii Remotes, along with activity recognition algorithms from the WEKA library, and an initial evaluation. We also describes how we collected a real-world dataset captured based on 20 users preparing a mixed salad and sandwich, and how we annotated this with 11 distinct food preparation activities.

Chapter 4 focuses on the development and implementation of a class-based threshold dynamic time warping system (CBT-DTW) for real-time activity recognition. We refine our evaluation methodology adopted in Chapter 3 and also demonstrate how this new activity recognition algorithm can deal with background activities such as *idle* and *unknown* activities, and how the CBT-DTW system can deal with activities for which only a small amount of training data is available.

In Chapter 5, we describe the development of the Fiber Chopping Board (FCB), an optical an acoustic sensing system embedded in a functional chopping board. Algorithms for food recognition are developed

in two phases: (i) the recognition of food before being prepared using the optical imaging system (i.e. when it is placed on the board); and (ii) the recognition of food while it is being prepared using sounds recorded by a microphone hidden inside the FCB. Two food image classification algorithms are implemented and evaluated (a k-Nearest Neighbour and a linearly Support Vector Machine) and a Gausian Mixture models is used for food classification based on chopping and slicing sounds.

Chapter 6 describes KitchenSense, a real-time context recognition software framework, including the system's architecture, software, hardware and data infrastructure. The components of this 4 tier system communicate within the constraints of a publisher-subscribe messaging framework. Furthermore, the chapter describes the re-design of the Slice&Dice instrumented utensils, in which we utilise the WAX3 sensors of OpenMovement. In brief, this is a "put it all together" chapter which integrates the real-time activity recognition algorithm developed in Chapter 4 and the Fiber Chopping Board and food ingredient recognition methods described in Chapter 5 into a single framework.

Chapter 7 covers our most significant empirical study and the evaluation of the KitchenSense's performance. We describe the collection of a large, complex, real-world dataset collected from 12 people who each prepared a spaghetti recipe 3 times (over 30 hours of food preparation data was collected and annotated). The dataset consists of more than 83,076 frames (i.e. seconds), 59 activities including unknown activities (i.e. 41 activities excluding unknown activities), 14,229 food images, and 8,798 seconds of chopping and slicing foods. The dataset was independently annotated by two coders and an inter-rater reliability procedure was applied to assess the reliability of the annotation. Rigorous subject independent and subject dependent evaluations are carried out to measuring the performance of the recognition of 59 fine-grained human activities and 8 food ingredients. For each evaluation, both standard frame-by-frame analysis results and event-timing analysis results are reported.

Chapter 8 summarizes the main contributions of the thesis, that is, our findings in pervasive sensing technology development, activity recognition, and food ingredient recognition. We also report the main limitations of our research and potential future work.

Chapter 2: Literature Review

This chapter presents a review of prior research work on activity and food recognition. After a brief discussion of the history of pervasive computing we consider previous related work in the fields of wearable sensing, pervasive sensing, and wearable-pervasive activity recognition. We then proceed to consider previous applications of computer vision-based and audio-based approaches to the problem of food recognition.

2.1 The origins of ubiquitous computing

About 60 years ago, the first computing era emerged (i.e. the *Mainframe era*) in which multiple people shared a single computer. In the second wave (1970s), the so-called *Personal Computer era*, users had a one-to-one relationship with computers. In the last 15 years, 8 billion embedded microprocessors have been produced every year and this number is dramatically increasing [98]. This has significantly changed the way people use computers, heralding the third wave, the so-called *Ubiquitous Computing era*: one person uses many computers, and this computational power is increasingly embedded in the world around us.

Originating in Mark Weiser's vision [1], the term "ubiquitous computing" refers to a world in which computing devices 'disappear' as they are woven into the fabric of our everyday surroundings. An alternative term, "pervasive computing", was defined by Satya [2] as "the creation of environments saturated with computing and communication capability, yet gracefully integrated with human users". Given the meanings of both terms are broad, and the similarity between the two terms obvious, ubiquitous computing and pervasive computing are used interchangeably throughout this thesis, as they typically are in the relevant literature [3].

Context recognition is a key problem in pervasive and ubiquitous computing. However, the term *context* itself is rather broad and is typically used to include any information that characterizes a situation. One of the most important elements of many contexts that are relevant to pervasive computing applications and services is the activity that a user engages in, that is, the *human activity*. Human activity recognition plays a vital role in a broad range of applications such as situated prompting, preventive health care systems and proactive service provision, yet the development of robust and generally applicable approaches to the representation and automatic recognition of human activity remains a basic challenge.

2.2 Human activity recognition (HAR)

2.2.1 Wearable sensing activity recognition

A large number of works have addressed the problem of activity recognition (AR) using accelerometers and other sensors worn on different parts of a user's body. Most approaches detect body-level activities, such as running, walking, or cycling, and many have produced significant results (i.e. recognition accuracies of 80% or higher). The majority of these studies employed wearable accelerometers [5, 17, 21, 22, 23, 25] and supervised learning approaches, although there are a number of examples that employed heterogeneous arrays of sensors [7, 84], used unsupervised learning [81, 82] or transfer learning [83] for applying AR across different domains.

In an early work on human activity recognition, Ravi et al. [5] used a wireless 3-axis accelerometer (sampling at 50 Hz) which was worn by a subject on the pelvis. Two subjects performed eight activities, including *standing*, *walking*, *running*, *climbing up stairs*, *climbing down stairs*, *sit-ups*, *vacuuming*, and *brushing teeth*, multiple times for each activity. Features were computed along with sliding windows (of size 256) for training different classification algorithms. The performances of various classifiers were evaluated using 10-fold cross validation and a range of different test settings. In subject-dependent tests accuracies as high as 90% were achieved, although in subject-independent tests accuracies as low as 60% were reported (i.e. where one subject was used to train the system and the other to test the system). The results were promising, particularly subject-dependent test results. However, the dataset was relatively simple in that it only collected and evaluated data for the pelvis worn accelerometer. Furthermore, as data was only collected for two subjects, and in a non-naturalistic setting, the true scope of the variation in the conduct of the activities of interest is unlikely to have been captured.

In a more general study of human activity, Bao et al. [7] sought to recognise 20 daily activities in 20 lay subjects who were asked to wear 5 wireless accelerometers on various points of their bodies (thigh, ankle, arm, wrist, and hip). The dataset was collected under semi-naturalistic settings, with the subjects being notified of start and end times of activities, and in an innovative twist the collected dataset was then annotated by the subject themselves. Various algorithms including C4.5 Decision Tree learning, Naïve Bayesian Networks, Instance Based Learning, and Decision Tables were trained and tested on the annotated data. A recognition rate of 84% was achieved for subject-independent evaluation, consequently this study has proven to be one of the most notable works in demonstrating the technical feasibility of recognizing human activities using multiple wireless wearable accelerometers.

In contrast to studies that utilise a single modality of sensing (typically an accelerometer) for full-body activity recognition, Ward et al. [17] investigated the use of heterogeneous wearable sensors for AR in the

context of a workshop in which skilled and semi-skilled manual work was conducted. Accelerometers and microphones were attached to the dominant wrist and upper arm of each subject with the goal of recognising different workshop activities associated with physical assembly, such as *sanding*, *drilling* and *grinding*. It was demonstrated that the combination of both accelerometer and audio data could significantly enhance the classification procedure and thereby improve recognition rates. However, in a manner similar to [5], the dataset collection was not wholly naturalistic, but instead conducted under relatively controlled laboratory settings with the subjects being told (by the experimenters) both how and when to perform activities. However, the study constitutes a good example of the recognition of skilled and semi-skilled activities using tools (rather than traditional full-body activities such as walking) and is therefore a promising precursor for our own work on kitchen-based AR for which tool use is likely to be a significant component of the problem.

In [21] Huynh et al. extended approaches to recognising low-level activities to classify both low- and high-level activities. Each subject wore 3 sensors, one on the wrist, hip and thigh. A 10-hour dataset of 16 low-level activities, such as *sitting*, *sleeping* and *walking*, and 3 high-level activities, *morning* (i.e. activity associated with getting up in the morning), *housework* (i.e. doing housework) and *shopping* (i.e. shopping in a market and or store) were collected in realistic and relatively unconstrained settings in which subjects were able to make their own decisions as to which activities they performed and how they performed them. The collected data was tested using 4-fold cross validation and yielded an overall recognition rate of 79% for low-level activities, and 91.8% for high-level activities. While low-level activity recognition performance was not particularly high, compared to previous work, it was notable that the dataset was uncharacteristically large and naturalistic. Indeed, the naturalism of the data set (and the associated drop in recognition performance) illustrated the importance of collecting realistic data, as compared to Bao et al. [7] (reviewed above) or Zinnen et al. [22] who collected a dataset of 10 short and non-repeatable car-based activities such as *open hood*, *close hood*, *heating on*, *heating off, open oil, close oil* etc. using an accelerometer worn on a subject's right wrist (one single subject performed the activities for a duration of 18 minutes in total).

Maekawa et al. [81] developed a highly heterogeneous sensing device which incorporated a camera, microphone, accelerometer, illuminometer, and a digital compass. The device was worn on a subject's wrist and was designed to enable the detection of home-based activities such as *making juice*, *cooking pasta*, *listening to music*, etc. Indeed the diversity of the activities targeted in part explains the diversity of the sensing modalities used (e.g. listening to music is probably best characterised by the presence of music but the absence of physical movement, or presence of rhythmical movement). In their experiment, the HMM+AdaBoost and HMM+C4.5 classification algorithms were applied to datasets, collected in a

semi-realistic manner, comprised of 15 activities from two home-like environments. During data collection, subjects (rather unnaturally) wore a laptop backpack that logged data collected by the sensor and transmitted via a wired connection. While it is inevitable that this could lead to atypically cumbersome performance of activities the subjects themselves were otherwise free to act as they desired.

In a broad investigation of the design of a wearable activity recognition system Tapia et al. [25] sought to detect 52 activities (including the intensities of some activities and estimated energy expenditure), and experimented with a set of wearable sensors including 7 accelerometers, a bodybugg™ armband, a pedometer, 2 ActiGraph activity monitoring devices and a heart rate monitor. However, 3 accelerometers on the hip, wrist and foot, and a heart rate monitor, were finally selected. An overall activity recognition rate (including the intensities of activities) of 50.6% and 87.9% were achieved for subject independent and dependent evaluations, respectively. Posture and exercise activities could be accurately and reliably classified, while household and resistance activities were generally more problematic. In addition to activity recognition, the work demonstrated that energy expenditure estimation could be improved using both a heart rate monitor and activity dependent models rather than using accelerometer data only.

The full range of sensing modalities from which activity can be usefully inferred is wide. For example, Bulling et al. [85] analysed eye movement data for activity recognition using a wearable electrooculography (EOG) device. Patterns of eye movement are characteristic of a subject's conscious and unconscious visual attention and therefore likely to be a good discriminator of intentional actions. Bulling et al. extracted 90 features that best describe eye movement and trained a SVM classifier on data from 8 subjects for which 6 typical office activities were annotated: *copying a text*, *reading a printed paper*, *taking hand-written notes*, *watching a video*, *browsing the web* (plus an additional *unknown activity* category). The approach yielded promising results in a subject-independent evaluation, with a 76% precision rate, 70.5% recall rate, and an overall accuracy of 72.7% (56.7% true positives + 16% true negatives).

Wearable activity recognition systems have a number of open problems related to their general applicability, most notably these include: (a) their sensitivity to the placement of sensors, for which approaches to the automatic adaptation wearable sensors (for displacement on a user's body) have been developed [81]; and (b) the need for annotated datasets for supervised learning approaches, for which a number of approaches have been developed, including, unsupervised training [82], transfer learning [83], self-taught learning [85], and routine discovery using unsupervised learning [90]. Addressing the placement and training problems not only reduces the time and cost required to annotate datasets (i.e. [82,

83, 90]) and the error introduced by sensor displacement, but also improve the general performance of activity recognition systems (i.e. [84]).

A number of applications of wearable activity recognition have been proposed, and prototyped, particularly in the domain of health and wellbeing. These include dietary monitoring [24], the estimation of energy expenditure [25, 88] and behaviour observation and quantification for children with autism spectrum disorders [87]. In general, wearable computing activity recognition has been shown to be a promising direction for the development of pervasive computing applications. However, it is also well understood (and has been widely observed) that users are generally not comfortable wearing most sensor systems. Wearable sensors are often obtrusive to users' activities and have the potential to impact on the natural performance of many tasks. Interestingly, in our own application context activities in the kitchen relating to food preparation (i.e. chopping, peeling, coring, stirring etc.) are highly dependent on the motions of kitchen instruments themselves (i.e. kitchen utensils such as knives, spoons, whisks etc.) and these are rather distinct from the movements of user's body.

2.2.2 Pervasive sensing activity recognition

In contrast to wearable sensing, traditional pervasive sensing involves the embedding of sensors in the objects and the environment. Technologies such as Radio Frequency Identification (RFID) [9, 11], simple and cheap state-change sensors (such as reed or piezoelectric switches) [19] and acoustic sensors [89] have been widely exploited for high-level activity recognition at homes. Other systems have combined RFID technology and load sensing to detect eating activities and estimated calorie [29], and have integrated load sensors under a work surface in a nutrition-awareness application [30]. In general, RFID-based systems infer human activities based on the identifying collections of objects involved. Objects typically have embedded, battery-free, passive RFID tags which can be detected by RFID readers also embedded in the environments or worn on the user's body [10, 14, 18]. Some previous work [16, 28] used sensors embedded in kitchen utensils, such as knives or spoons, to classify fine-grained food preparation activities in an unobtrusive manner. Notably, [26] detected low-level activities and prompted actors posing as people with dementia in support of their food and drink preparation tasks in the Ambient Kitchen [27].

In [11] Intel researchers developed sensing devices called WISPs (Wireless Identification and Sensing Platform) that can both communicate with, and are powered by, RFID readers. One advantage of WISPs is that they can transmit a RFID tag identifier along with the most recent acceleration data of a moving object. This increases the reliability of the detection of objects which are in use. In their experiment, a dense sensing infrastructure including 25 WISPs and 3 RFID readers were deployed in a studio

apartment. 10 subjects performed 14 high-level activities such as using the phone and making cereal. Results as high as 90% (precision and recall) were reported. These results demonstrate significant promise for real-world pervasive computing applications. However, the embedded technologies used in this work were not completely invisible to the users (i.e. WISPs were visibly attached to bowls and cups).

Tapia et al. [19] deployed 77 simple ubiquitous sensors (i.e. RFID, reed and piezoelectric switches) in a home setting to detect high-level household activities such as *preparing lunch*, *toileting*, *bathing*, and *grooming*; collecting a dataset from two subjects over 14 days. A particular innovation was the fact that the annotation procedure was carried out by subjects *while* they were performing activities. Subjects were given a PDA running an experience sampling tool. Every 15 minutes, the PDA asked each subject to select a label and a duration time for the activity that was currently being performed. In this way, the dataset was continuously annotated *while* it collected. While such a data collection protocol is applicable to high-level activities (i.e. activities of duration greater than a minute), it would be difficult to argue that such a configuration of user and technology (i.e. user and data collection technology) is naturalistic, and subjects are unlikely to perform their activities in a wholly natural manner. Furthermore, such an approach is generally unsuitable for fine-grained activities (such as the elements of food preparation) that can occur within relatively short time intervals (i.e. within few seconds) and involve significant tool use.

Pervasive acoustic sensing has been deployed in various contexts to facilitate home-based activity sensing. For example, Fogarty et al. [89] deployed unobtrusive microphone-based sensors at water distribution infrastructure locations. Sensors were attached to the outside of the water pipes of sinks, toilets, showers, and appliances in the home. Activities were inferred from the sounds produced by water usage based on pairs of zero-crossing rate and root mean square features that were extracted from sound streams. Accuracies ranged from 73-100% for water-usage related activities such as *dishwasher usage*, *showering*, *clothes washer usage*, *kitchen sink activity*, *bathroom sink activity* and *toilet flushes*. Indeed, the study has shown the feasibility of pervasive sensing for activity recognition in domestic homes even with low-cost, simple sensors, although inevitably there are a lot of activities that are not water usage related (such as food preparation).

Pervasive sensing has been applied to a number of different activities related to both food preparation and the measurement of food and drink consumption. Chang et al. [29] addressed the problem of the estimation of *prepared foods* that are consumed by individuals with a meal. To do so, they instrumented the surface of a table with RFID and weight sensors (using a two layered surface) so as to track the food, and quantities of food, that was consumed by users. The surface of the table was divided into 9 cells, each of which had an embedded load cell and a RFID reader antenna (RFID tags were applied to food

containers). The user's behaviours were modelled as a sequence of event changes. The amount of consumed food was computed using a rule-based weigh-matching algorithm. To verify their approach, 4 experiments were conducted and accuracies as high as 80% were reported. However, there was a clear assumption of a priori knowledge of food ingredients, which is not practical for real world dietary monitoring. Similarly, in practice, people often placed more than one dish on a single plate. In such cases the technical configuration described would not be able to distinguish between one dish (on a plate) and any other (on the same plate).

A number of projects have also investigated the instrumentation of kitchens to support food and drink preparation [27, 30, 31, 32]. One notable early project was MIT's *CounterIntelligence* [31], an augmented kitchen that sought to provide instructive information to users while they are cooking. *CounterIntelligence* focussed on the design of situated interaction (rather than activity recognition) and incorporated several examples of how information could be displayed on kitchen surfaces to direct a user's attention appropriately. Reiko et al. [30] developed a cooking navigation system that guided novice cooks during cooking sessions. A cook could either learn how to cook or simply follow multimedia instructions to complete a task. After each preparation step, the cook had to manually click on the character icon on a screen to manually select the next step of the recipe. Although the system was intended to help a cook follow an optimized cooking plan, as with many of these early systems, context awareness, such as the recognition of preparation activities, was not incorporated. In addition to supporting food preparation skills and competence, another potential goal of situated support in the kitchen is to promote people's nutritional awareness. Chen et al. [32] provided nutrition information to users while they were cooking in their *Smart Kitchen*. Although the calorific value of foods could be semi-automatically estimated by the system's weight matching algorithm, the system was not able to detect the identity of food ingredients processed on the work surface. The users themselves had to manually update information for each ingredient before processing it. As such, [30, 32] constitute initial forays into the area of providing situated support for cooking, and go some way to demonstrating the need for a system that can automatically recognize food preparation activities and food ingredients.

Olivier et al. [27] developed the *Ambient Kitchen* (at Culture Lab, Newcastle University), a high fidelity prototype for exploring the design of pervasive computing algorithms and applications for food and drink preparation. The environment integrates data projectors, cameras, RFID tags attached to food containers and 4 RFID readers installed under work surfaces, wireless accelerometers embedded into objects, and pressure sensors integrated under the floor. The *Ambient Kitchen* is a lab-based replication of a real-world kitchen in which careful design has hidden technologies from users. The *Ambient Kitchen* was aimed to support both the evaluation of pervasive computing prototypes and the simultaneous capture of multiple

synchronised sensor data streams. Indeed, while previous work exploring the requirements for situated support for people with cognitive impairments motivated the design of the physical sensing infrastructure, the *Ambient Kitchen* is primarily proposed as a platform of research in kitchen-based situated support, acting as: (a) design tool for designers for cooking applications (i.e. cookbook); (b) an observatory to collect sensor data for activity recognition algorithm development; and (c) as an evaluation test bed. Our recognition system for the analysis of the food preparation activities [16] was integrated into the Ambient Kitchen and the real-time analysis version continuously runs as a background service simultaneously analyzing data from multiple, synchronized accelerometer sensors (see Chapter 3 and Chapter 7 for details of the integration of our results within the *Ambient Kitchen*).

The principal attraction of pervasive computing (as opposed to wearable computing) is that by embedding devices into everyday objects (i.e. kitchen utensils and appliances) the underlying technology remains invisible to people. As such, people do not need to wear new digital devices to interact with proposed services, thereby allowing people to undertake their everyday activities unencumbered. Having said this, there are still many shortcomings associated with inexpensive and widely available technologies such as RFID technology. While RFID is widely used for embedded sensing, it still has a number of limitations, including that it cannot be attached on metal objects as this typically results in dropped signals. Moreover, in our experiments, several food ingredients that underwent significant processing (e.g. were chopped) were fresh foods and as such could not have RFID tags attached to them. To overcome this, we developed a chopping board and a set of kitchen utensils and appliances that can recognise both human activities and food ingredients non-invasively (see Chapter 5).

2.2.3 Activity recognition using a combination of pervasive and wearable sensing

The final sensing configuration to consider for human activity recognition, is the combination of embedded and wearable sensors. That is, the use of a combination of sensors that are embedded in objects in the environment as well as worn on the user's body. Much previous pervasive computing research has addresses the AR problem with such a configuration [8, 10, 12, 13, 14, 15, 18, 23].

Wang et al. [8] developed a home-based multi-modal sensing system to investigate human activities performed by single and multiple users. Users were asked to wear an audio recorder, two wireless accelerometers (on their wrists), and two RFID readers (on their palms). Common household objects such as cups, teaspoons, and computer mice were augmented through the attachment of RFID tags. Two subjects performed 21 daily living activities (e.g. *watching TV*, *making coffee*, *brushing teeth*) over ten days; in their study most of the activities targeted were high-level. In [10], Wu et al. combined RFID technology and computer vision to recognise high-level kitchen activities specifically. A Bayesian

network algorithm was developed, and by incorporating common-sense knowledge this was used to learn models from video and RFID sensing. Their experiment in a kitchen was conducted with 33 RFID-tagged objects and 16 high-level activities including *making tea*, *making sandwiches* and *boiling water*. A camera was installed with a view of the kitchen counter and subjects wore a bracelet RFID reader. It is well known that many kitchen utensils and appliances (e.g. pots and pans) are made of metal and this study showed that computer vision could enhance object detection. Indeed, the addition of computer vision increased activity recognition rates from 60% (using of RFID sensing only) to 80%, although it is worth noting that users expressed clear reservations about the use of cameras and the invasion of their private space by a technology more commonly associated with surveillance. More recently, the Opportunity project [12] deployed a very rich sensor environment to collect a large-scale dataset of activities. The environment simulated a studio flat with a chair, a kitchen, doors, a coffee making machine, and a table, which in total was instrumented with 72 wired and wireless sensors. From observation of the video captured during dataset collection it is readily apparent that the users were, to a significant degree, aware of technology embedded in their surroundings, so although the goal of the project is to collect a gold standard sensor dataset for daily activities, the ecological validity of the dataset itself is rather weak.

The Quality of Life Technology Centre (QoLT) project [34] at Carnegie Mellon University, is typical of recent large-scale initiatives that aim to develop "technologies that will improve and sustain the quality of life for all people". In addition to a multitude of cross-disciplinary research activities relating to the development of the methods that enable older adults and people with disabilities to live independently, QoLT also deployed a real-world kitchen. Spriggs et al. [33] describe their development an activity segmentation and recognition system that analysed a set of fine-grained activities (such as *pour oil in cup* and *stir the mix*) in the kitchen. Like Opportunity, the QoLT study employs a mixture of worn and environmental sensors, with subjects wearing cameras and inertial measurement units on their body, and other cameras and microphones were installed in the environment.

2.3 Food recognition

The ability to automatically recognise foods in our everyday environment, and particularly the kitchen, has many potential applications including in healthier eating interventions and dietary intake estimation applications. However, food recognition is a challenging problem and approaches are significantly less developed that for activity recognition more generally. The large majority of approaches to the food recognition are based on computer vision [34, 35, 36, 37, 38, 39, 40, 41] with a smaller number employing audio classification while the foods are being processed (e.g. being chopped) or consumed (e.g. chewed) [15, 24].

2.3.1 Computer vision and food recognition

Similar to classical object recognition in computer vision, foods recognition are generally proceeds in 4 steps: (a) food image pre-processing; (b) segmentation; (c) feature computation; (d) feature classification. For example, Shroff et al. [35] developed DiaWear, a food recognition system for a mobile phone. A neural network classifier with 5-10-5 feed-forward back propagation was employed for training and classifying colour, size, and texture features calculated from food images. Food images were manually taken by a user using a mobile phone camera. Although many variations in the layout of food (and occlusions) can occur in real-world settings, in DiaWear foods were assumed to be non-touching, complete and the background was assumed to be single coloured and lighter than object's colours (such assumptions are very strong and in impractical for real-world applications). Results were however reported for an experimental study on 4 foods: *Hamburgers*, *Fries*, *Chicken Nuggets*, and *Apple Pies* which showed recognition rates between 50-75% over a relatively small dataset of 120 static food images. Kok et al. [36] utilized colour in 2 steps for recognizing natural objects, by first creating an artificial colour contrast based pre-filter to detect an object's surroundings and then extracting bounded box features (SFBB). This method was evaluated for *meat*, *chicken*, *bone*, and *grapefruit* detection and significantly outperformed a 3-layer neural networks and SVMs (99% compared to 85% and 74%, respectively).

Unsurprisingly, the detection of *fast food* appears to be a major concern of computer vision researchers in this domain [38, 40, 41]. Wu et al. [41], for example, collected a fast food database of 101 foods from 9 restaurants in the USA. To recognise foods in videos of eating, key points of SIFT (scale-invariant feature transform) descriptors were extracted from each frame. Matching key points were ranked across food items. Several matching criteria were used, and SIFT with cosine matching of 3 key points was found to perform best, giving a recognition rate of 73% for 9 restaurants. Using this fast food database, Shulin et al. [40] proposed a pair-wise local feature based method for food recognition. Pairwise local features represent the spatial relationship between pixels of different food types. Pairwise local features of two pixel p_1 and p_2 included: pairwise distance (the distance between p_1 and p_2), pairwise orientation (the orientation of the line between p_1 and p_2) and pairwise midpoint (the category of the pixel in the middle of p_1 and p_2), and between pair (the category of all pixels between two pixels p_1 and p_2). The extracted pairwise features were classified using a SVM with X^2 kernel function. The experiment on 7 food categories including *sandwich*, *salad&sides*, *bagel*, *donut*, *bread&pastry*, *chicken*, and *taco* yielded a 78% accuracy rate which was a significant improvement on 49%, using colour features only, and 55%, using SIFT features only.

2.3.2 Audio-based food recognition

There has been a relatively small number of food recognition studies based on acoustic data. For example, Amft et al. [24] developed a system using a microphone worn inside the subject's ear canal to classify different types of food from chewing sounds. The acoustic analysis was based on two steps: (a) chewing segment identification; and (b) chewing sound classification. Step (a) only used the intensity of audio signal and a simple classification procedure. In step (b) several features were computed, including: zero-crossing rate, spectrum fluctuation, band energy ratio, and band width over frames of size 11.6 ms with a SIFT of 8.7 ms. These features then were averaged over continuously segmented frames (of one chewing segment or a single chew). An experiment with a C.4.5 decision tree and 10-fold cross validation was conducted for 4 foods (*chips*, *apple*, *pasta*, and *lettuce*) and gave a 66-86% overall accuracy for a single chew and 80-100% for a chewing segment.

In contrast to the use of wearable sensors in [24], Kranz et al. [15] installed a AKG C1000S microphone in the Aware Kitchen and an augmented a knife with force and torque sensors. The microphone and a camera were positioned 30cm above the chopping board and audio and video streams were merged into one file for synchronisation. A study was conducted for 4 subjects cutting *carrots*, *bananas*, *leeks*, *kohlrabi*, *peppers*, and *apple*s over 3 sessions. The audio log was segmented into 259 episodes for which each contained a cut or peel action, and a Hamming window of 20 ms frames (50 frames per second) was applied. Several features were computed for each frame, including: contours, pitch, energy, amplitude, and bandwidth (harmonics-to-Noise Ratio), and the computed features were then classified using a SVM with polynomial kernel. The overall recognition rate of 85% accuracy for 5 foods demonstrated the potential for audio-based food classification, although again the visibility and obtrusiveness of the technologies and general instrumentation suggest that a more sensitive product design for such technology-enhanced utensils is needed for practical and near real-world deployments.

2.4 Conclusion and discussion

A significant body of prior work exists that examines human activity recognition (HAR) and in particular the recognition of food preparation, consumption and food recognition itself. The literature covers three common approaches sensing configurations for HAR, wearable sensing, pervasive sensing, and wearable-pervasive sensing combinations; and there exist two common approaches to food recognition, computer vision-based and audio-based. The principal drawback of previous work on wearable sensing and wearable-pervasive sensing for HAR is the obtrusive nature of the systems that users are required to wear, and in many studies users consistently express their scepticism as to the practicalities of wearing sensors solely for the purpose of recognising their everyday activities. One alternative, the use of computer vision based approaches (e.g. [10]), gives rise to more pointed concerns about privacy, indeed, computer vision

in particular is widely considered inappropriate in private settings such as the home. Furthermore, for applications such as the situated support of cognitively impaired people, it is unreasonable to expect such user to always remember to wear such sensors before performing activities. For these reasons, we deem that both wearable sensing and wearable-pervasive sensing combinations are not suitable technology configurations for our approach.

By contrast, a small number of purely pervasive sensing approaches have emerged as a viable approach to activity and context recognition in the kitchen. Crucially, pervasive sensing has the potential to allow people to naturally interact in their environment providing the technology itself can be rendered either effectively invisible, or, through sensitive design, embedded appropriately into objects and the environment. However, existing pervasive sensing has several limitations. Embedding RFID technology in the kitchen environment (e.g. [9, 11, 19]), for example, is acceptable for food containers such as olive oil or sugar, but is not applicable to most fresh food ingredients which often are not kept in packaging or containers. Moreover, RFID technology does not reliably work if RFID tags are embedded into objects made from metal or out of the range of sensing which is often limited to relatively small regions where readers' antennas are located.

Similarly, the relatively small body of work on food recognition mostly use computer vision techniques and cameras that are not unobtrusively embedded either into objects or the environment. Food photographs are usually manually taken by the users (e.g. [35, 39]) or are automatically taken by cameras installed in the environment (e.g. [38, 40, 41]) once again raising concerns about privacy and thus the viability of such systems for real-world applications. Moreover, previous work on food recognition [38, 40, 41] has almost exclusively concentrated on fast food rather than ingredients to be used in a recipe (as in our scenario). Although one previous study [37] embedded a camera inside a microwave oven to detect the food on the dish, this was applicable only for cooked foods or food that needs to be warmed before being eaten. To our knowledge, only one work [15] classified food ingredients in the preparation stage using a microphone embedded in the kitchen environment and not in the object. The sensor is therefore easily aware by the user. We see this approachas a most potential approach. In the chapters that follows we develop such an approach, but embed the sensor into the object, to improve its performance and real-world applicability through the development of better recognition algorithms, better design of the kitchen utensils and use of a wider range of utensils, and finally by extending the sensing modalities to include computer vision applied in the privacy sensitive capture and classification of images of food placed on a chopping board.

Chapter 3: A Preliminary Study of Activity Recognition in the Kitchen

In this chapter we describe our first HAR prototype, Slice&Dice, a proof-of-concept system, which aims to perform recognition of low-level food preparation activities in the kitchen. Slice&Dice comprises a set of specially designed kitchen utensils (three knives and one large spoon) prototypes using 3D printing technology and embedded (modified) Wii Remotes. Three classification algorithms are developed and trained using a real-world dataset collected from 20 subjects. As such, even for this prototype we have conducted a significantly more substantial and ecologically valid study than previous research, and the system is evaluated using a subject-independent protocol. The results show significant promise, and in subsequent chapters we build on these and extend the algorithms, sensing modalities and artefact design (i.e. the utensils).

3.1 Introduction and the need for low-level activity recognition

The kitchen is the household location where everyday activities such as cooking and food preparation are conducted. The overarching motivation for our inquiry into activity recognition in the kitchen is a general recognition that the provision of situated support services in the kitchen has significant potential to improve the health and wellbeing on individuals and families, for example, through the provision of nutritional advice and cooking skill support and cooking knowledge tutoring. For particular classes of users, such as cognitively impaired people, the situated support services in the kitchen have a significant role to play in helping them maintain more independent lifestyles. For example, a study by Wherton and Monk of people with dementia and their formal and informal carers [42] found that so-called failures in task completion (i.e. food and drink preparation tasks) could occur when low-level actions were unexpectedly suspended or prolonged. In this case, the caregivers typically provide fine-gained prompts that helps the person with dementia complete low-level actions such as putting a teabag into a teapot or buttering a piece of toast. While the support of people with dementia is in itself sufficient motivation for the need for low-level activity recognition of food and drink preparation activities, we should note that even situated support that is not targeted at cognitively impaired users requires the detection of activities at sub-task level, for example, for prompting the next step in a recipe based on the detection of the completion of the previous step.

Our first prototype system therefore addresses a fine grained set of food preparation activities in the kitchen context, in which we attempt to recognise 11 common activities related to the preparation of a mixed salad based on data collected from 20 participants in a realistic (although admittedly) lab-based kitchen. An additional requirement is that we must make the technology used (primarily the sensors themselves) as unobtrusive and unencumbering as possible (unlike previous studies) and to do these we

choose to design complete utensils (using 3D printing technology) and embed consumer off-the-shelf sensors into the handles of these utensils. The chapter begins with the design and development of Slice&Dice system in Section 3.2. The data collection process is presented in Section 3.2. Section 3.3 describes about data annotation and Section 3.4 is about system evaluation. Finally, conclusions and discussions are presented in the Section 3.5.

3.2 Slice&Dice prototype

3.2.1 System requirements

Following our review of previous research studies and systems we have established a number of key requirements which our prototype has sort to address:

(1) *Low-level activity recognition*: as previously argued, recognising low-level activities is necessary for the development of a range of situated services in the kitchen (i.e. [42]) our prototype must therefore be able to classify low-level activities. The range of low-level activities is primarily determined by three factors, the range of utensils that we intend to use, the range of recipes for which we will prepare using these utensils (and thus the set of actions), and the assumed skill level of the cook. By reviewing cooking videos uploaded to YouTube we identified 11 distinct low-level activities that were used in the instructions in these videos: *chopping*, *peeling*, *slicing*, *dicing*, *coring*, *spreading*, *eating*, *stirring*, *scooping*, *scraping* and *shaving*. The distinction between these activities are matters as we toward our framework for guiding or prompting people to cook. For example, it is better if some foods can be sliced (i.e. meat, ham), some can be diced (i.e. make cubes for a potato salad). Although this may not matter in all recipes, it is particularly for a mixed salad where lettuces need to be sliced, carrots need to be chopped (not only served foods, but also its good-lookings). We therefore require our prototype to be capable of classifying utensil use for each of these activities.

(2) *The sensing infrastructure should not interfere with the conduct of the activity itself.* In other words, the sensing technologies must be hidden from the users and be part of the system comfortable-to-use. Meeting this requirement will ensure that users to perform their activities in a natural manner as if they were performed in a regular non-instrumented environment (i.e. a regular home-based kitchen). Even the most unobtrusive wearable sensor systems does not fulfil this requirement due to the very fact that users are always required to remember to wear a sensor when preparing food or drinks.

3.2.2 The Design of kitchen utensils

The Wii Remote [43] is a consumer off-the-shelf wireless sensing system and games controller which supports two functionalities of relevance to our application: (i) input detection through an embedded accelerometer; and (ii) data communications through Bluetooth. The Wii Remote comprises a printed

circuit board (which is encapsulated by a white case) and uses an AXDL 330 accelerometer [3] and a Broardcom BCM2042 chip that integrates the entire profile, application, and Bluetooth protocol stack [92] (see Figure 3.1). Based on Micro Electro Mechanical System (MEMS) technology the AXDL 330 accelerometer is a small, low power, 3-axis accelerometer with signal conditioned voltage outputs. The AXDL 330 accelerometer can sense acceleration in three axes with a minimum full-scale range of ±3g. While the static acceleration of gravity can be used to implement tilt-sensing in applications, dynamic acceleration measurement can be detected through the quanitifies of motion, shock or vibration.

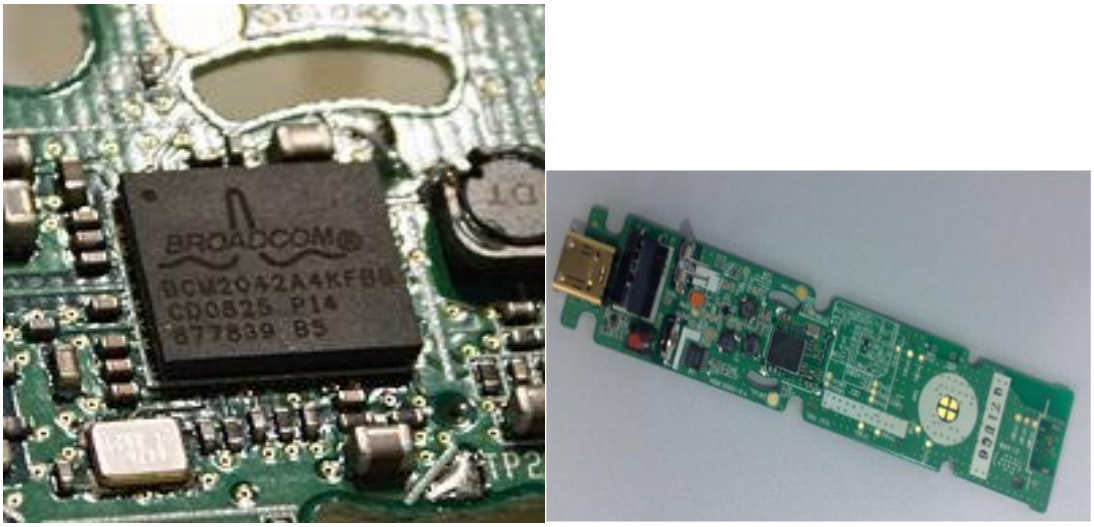


Figure 3.1: Broadcom BCM2042 (left) and Wii Remote printed circuit board (right).

The Broadcom BCM2042 board is a system-on-chip which integrates an on-board 8051 microprocessor, random access memory/read only memory, human interface device profile (HID), application, and Bluetooth protocol stack. Furthermore, multiple peripherals and an expansion port for external add-ons are embedded on the board. The integration of these components and the technology's adoption in a mass market consumer games console has significantly reduced the cost of BCM2042. The Wii Remote's input capabilities include buttons, an infrared sensor and an accelerometer. The infrared sensor is embedded in a camera which detects IR light coming from an external sensor bar. The accelerations are measured in X, Y, and Z axes (relative to the accelerometer) and the three directions of the movement (X, Y, Z) can be computed through tilt angles. Wii Remote inputs and sensor values are communicated to a Bluetooth host through the standard Bluetooth HID protocol. Values for acceleration are transmitted with a sampling frequency of 40Hz. While we envisage that future versions of Slice&Dice might use the ADXL330 as a component of a smaller wireless sensor, the Wii Remotes provides an excellent platform for developing and evaluating classification algorithms in kitchen utensils that still retain a usable form factor.

Figure 3.2: Utensils instrumented using a modified Wii Remote.

Four kitchen utensils: one big knife, one bread knife, one small knife, and one large serving spoon are instrumented with modified Wii Remotes as shown in Figure 3.2. The utensils casing themselves were designed using a 3D modelling tools and fabricated in acrylic using Fused Deposition Modelling (FDM) rapid prototyping.

3.2.3 Classification algorithms

The Slice&Dice system recognises human activities in 3 steps: *data segmentation*, *feature computation* and *classification*. In the *data segmentation stage*, acceleration data from utensils was converted into pitch and roll rotations as follows:

$$Pitch = 2\ arctan\left(\frac{y}{\sqrt{x^2+z^2}}\right) \quad (3.1)$$

$$Roll = 2\ arctan\left(\frac{x}{\sqrt{y^2+z^2}}\right) \quad (3.2)$$

where x, y, z are acceleration values of the three axis and a triplet of (x,y,z) is called a sample. The samples are grouped into windows than have a 50% sample overlap between two adjacent windows. The length of windows is an algorithm parameter that we vary in in the evaluation the performance of different classification algorithms.

In the *feature computation* step 4 different types of features: *mean*, *standard deviation*, *energy* and *entropy* are computed for the set of samples within a sliding window. Where *n* is the length of a slicing window, the features for acceleration data along with x-axis are computed as follows:

$$Mean(x) = \frac{\sum_{i=1}^{n} x_i}{n} \quad (3.3)$$

$$\textit{Standard deviation}(x) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(x_i^2\right) - [Mean(x)]^2} \quad (3.4)$$

$$\textit{Energy}(x) = \frac{\sum_{i=1}^{n} x_i^2}{n} \quad (3.5)$$

$$\textit{Entropy}(x) = -\sum_{i=1}^{n} p(x_i) log(p(x_i)) \quad (3.6)$$

where x_i is an acceleration value, $p(x_i)$, a probability distribution of x_i within the sliding window, can be estimated as the number of x_i in the window divided by *n*.

As we have 4 utensils, each has a 3-axis acceleration (i.e. *x*, *y* and *z*) and 2 rotations (i.e. *pitch* and *roll*) the input feature vector therefore comprises of 80 features that will be used for training classifiers in the next step. In the *classification* step, three classification algorithms, Naïve Bayes, Bayesian Networks, and Decision Tree C4.5 (as implemented in Weka [45]), were used to classify the food preparation activities in this first version of Slice&Dice.

3.3 Data collection

It is widely accepted that datasets collected under real-world settings are crucial to the development of robust and reliable machine learning algorithms [20]. For this reason, for our data collection activity subjects participated in a relatively open food preparation task and in doing so conducted themselves without any imposed time-constraint or detailed instructions as to what to do and how to do it.

3.3.1 Experiment settings

Figure 3.3: The Ambient Kitchen, our laboratory based instrumented kitchen environment.

The experiment was carried out in the *Ambient Kitchen* [27]. Of the environment's five IP cameras (installed inside the wall of the *Ambient Kitchen*) two were directly focused on the work surface where food was prepared by the subjects. The food ingredients provided to subjects included: *tomatoes*, *potatoes*, *lettuce*, *carrots*, *onions*, *pepper*, *grapefruit*, *kiwi fruit*, *garlic*, *bread* and *butter*. Four utensils: a *big knife*, a *bread knife*, a *small knife* and a *serving spoon* (see Figure 3.2) were placed on the work surface in preparation for each data collection session. These utensils, in which modified Wii Remotes were embedded into their handles wirelessly communicated with a computer behind one of the kitchen walls using a Bluetooth dongle device. A logging program running on the computer recorded one timestamp for each sample written into the log files to allow later synchronisation of acceleration and video data. The kitchen also included a (non-instrumented) wooden chopping board for cutting activities and a salad bowl.

Twenty subjects without professional cooking experience were recruited from Newcastle University (through personal contacts of members of Culture Lab). Subjects were only asked to use the utensils and ingredients provided to prepare a mixed salad and a sandwich. No time constraints, nor other instructions from researchers, were imposed on the subjects. Consequently, the time taken to complete the task is varied significantly between subjects. Note that ethical approval was provided for the study, and subjects all signed privacy waivers relating to the video and accelerometer data collected. The annotated video and acceleration data has been made freely available to other researchers (see Appendix A).

3.3.2 Real-world dataset challenges

In the recorded videos, and the subsequent process of annotation, we observed that the dataset collected under realistic conditions gave rise to a number of particular challenges:

(1) *Variability of activities*: as no instructions on how to perform the recipe were given to the subjects, the subjects could perform activities as they wished (i.e. on might argue "in a more natural way"). For example, some subjects chopped a carrot quite fast while others chopped it slowly, actions such as peeling were conducted in very different ways by different subjects, and utensil selection (for different ingredients) varied between subjects.

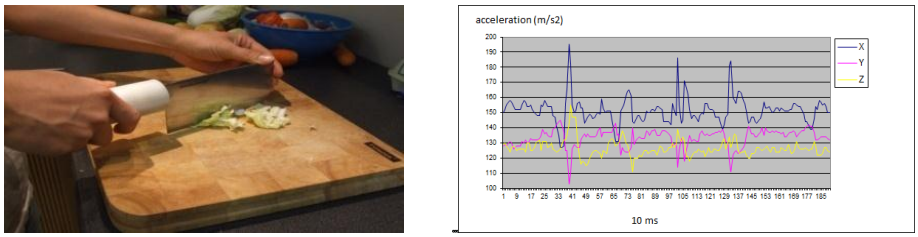
(2) *Ambiguity of annotation*: preparation activities in the kitchen do not always fall into neat categories that can be readily classified by a human observer. For example, the distinction between chopping and dicing may be clear in some cases, in other cases subjects appear to move between the two, in an almost continuous manner. Establishing clear annotation protocols that ensure the consistent observer-based classification of activities requires us to address this problem

through the preparation of appropriate training materials and quality assurance (i.e. checking the annotators work systematically).

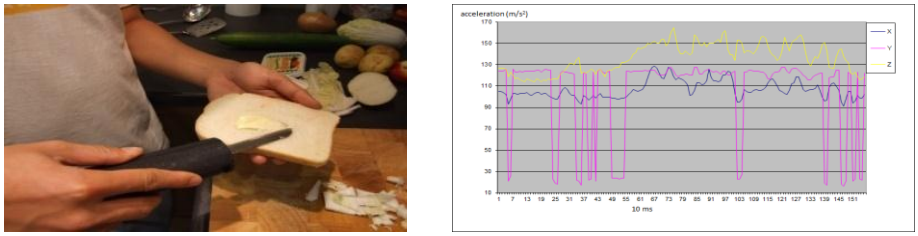
(3) *Inherent imbalance of the dataset*: while all subjects performed a significant number of *chopping*, *scooping*, and *peeling* activities, only a small number of subjects performed *eating* (i.e. using the serving spoon to eat ingredients), *dicing* (i.e. rapid fine-grained chopping), and *scraping* (i.e. rather than *peeling*) activities. As a result distribution of activities in the captured dataset is not even. For example, in our collected dataset, for a window length of 64, there were 3116 instances of chopping but only 78 instances of dicing.

3.4 Data annotation

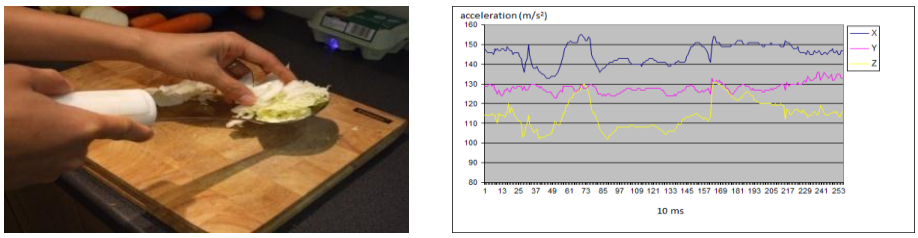
A set of 11 activity labels was decided upon based on an informal survey of language used in several hours of English language cooking videos found on YouTube. The activity labels were *chopping*, *slicing*, *peeling*, *stirring*, *scooping*, *dicing*, *shaving*, *scraping*, *eating*, *spreading*, and *coring*. Some illustrations of activity, and the acceleration patterns for these activities, are shown in Figure 3.4.



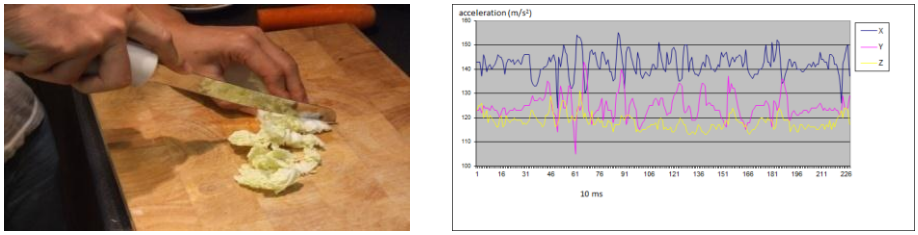
(a) Dicing



(b) *Spreading*



(c) *Scooping*



(d) *Slicing*

Figure 3.4: Examples of food preparation activities.

The collected videos were independently annotated by 3 annotators using the Anvil multimodal annotation tool [46]. Each annotator was provided with an informal description of the 11 activities previously identified, and was asked to annotate time period in the video when subjects were undertaking these activities (identifying each specific one). Post-annotation, two datasets were created for training and testing the classification algorithms. Dataset A was the intersection of 3 annotated datasets where only labelled data for which all three annotators agreed was extracted. This corresponds to the data where there is complete agreement between to the annotators as to the activity being performed. As expected, it was noticed that the precise timing of the boundaries of the 11 activities was often unclear. While dataset A was the subset of the data for which all annotators agreed, we repeated our experiments on a second dataset (B) which was the complete dataset annotated by one single annotator. Dataset B was therefore a larger dataset corresponding to a single annotator's interpretation of the labels and video data.

3.5 Evaluation

This section presents the first evaluation of the Slice&Dice system on the dataset collected from 20 subjects, for which we applied a subject-independent protocol.

3.5.1 Procedures

Although previous research on human activity recognition has generally used sliding window sizes ranging from 4.2 to 6.7 seconds [7], our problem domain is quite different in that our sensors are attached into the utensils themselves and we are well aware of relatively short time window within which some

activities might be best characterized. However, as this was the first version of Slice&Dice, we needed to determine window size with which the best results could be achieved. Therefore, the data was processed using a range of different window sizes: 0.8 seconds (i.e. 32 samples), 1.6 seconds (i.e. 64 samples), 3.2 seconds (i.e. 128 samples), 6.4.seconds (i.e. 256 samples), and 12.8 seconds (i.e. 512 seconds) using the timestamp of the sample. As already described a 50% overlap between two consecutive windows was used. We deemed it not necessary to consider a window size smaller than 0.8 seconds as none of the activities we are concerned with take place at this temporal scale, likewise, the upper bound for window was set to 12.8 seconds (i.e. too slow activity).

A vector of 80 features was computed from each window, and was used the input to the classifiers. In this experiment, Slice&Dice was evaluated under the subject independent ("leave-one-subject-out") protocol. Under this protocol, we trained 19 subjects and tested the one remaining subject. The process was repeated until all subjects were tested. Finally, the results were aggregated. We report the evaluation results in terms of the accuracy of each activity, *Acc(a)*, and the overall accuracy, which are calculated as:

Let n_i be the total number of instances of activity a_i

$$Acc(a_i) = \frac{the\ number\ of\ a_i\ is\ detected}{n_i} \tag{3.7}$$

$$Overall\ accuracy = \frac{\sum_{i=1}^{11} Acc(a_i) * n_i}{N} \tag{3.8}$$

where N is the total number of instances in the dataset.

3.5.2 Results

We evaluated Slice&Dice on both datasets A and B. The preliminary results are presented as follows.

(a) Performance results for dataset A

Dataset A was constructed in an attempt of ours to reduce inconsistency between activity labels. The results of the performance of Slice&Dice on dataset A are shown in the Table 3.1. In this table, the highest accuracy (82.9%) was achieved by the Decision Tree C4.5 algorithm with a window size of 256. It is also apparent that the accuracy slightly increased when the sizes of the window used was larger (up to, but not including 512, due to the fact that the data significantly decreases to 252 instances for a window size of 512).

Algorithm	Window size				
	32	**64**	**128**	**256**	**512**
Decision Tree C4.5	77.7	78.7	80.3	82.9	80.2
Bayesian Networks	70.8	72.5	79.7	78.9	69.1
Naïve Bayes	47.5	45.6	50.5	52.4	51.3

Table 3.1: Overall accuracies (%) on dataset A.

The next most accurate was the Bayesian Network which demonstrated accuracy of ~79% for window sizes of both 128 and 256. Naïve Bayes yielded the worst accuracy of just around 50% for all window sizes. The precise (action-by-action) details of accuracies for Decision Tree C4.5 are present in the Table 3.2.

	Window size									
	32		64		128		256		512	
Activity	instances	%	instances	%	instances	%	instances	%	instances	%
chopping	2040	87.3	986	87.5	476	86.1	220	87.7	102	86.5
peeling	712	96.9	342	95.9	170	97.1	80	97.5	36	94.4
slicing	160	19.4	64	26.6	36	30.3	10	80.0	4	0
dicing	184	19.6	86	18.6	38	15.8	12	4.2	4	0
coring	354	73.4	170	77.7	80	76.3	36	80.6	10	60.0
spreading	224	46.0	126	44.4	56	57.1	26	53.9	10	40.0
eating	94	10.6	44	31.8	18	27.2	8	50.0	0	n/a
stirring	392	78.6	192	85.9	86	90.7	36	91.7	14	100.0
scooping	906	89.5	460	86.3	222	89.2	98	86.7	42	92.9
scraping	98	50.5	48	56.3	22	18.2	8	75	2	0
shaving	388	60.3	176	59.7	80	72.3	30	69.9	14	60.3

Table 3.2: Detailed classification results (%) for Decision Tree C4.5 on dataset A.

For dataset A, the best overall accuracy was achieved with a window size of 256 for which high levels of accuracy was also achieved for a number of the individual activities, *chopping*, *peeling*, *slicing*, *coring*, *stirring* and *scooping* (i.e. more than 80%). Notably poor accuracy was achieved for *dicing*. Accuracies of around 50% were achieved for *spreading* and *eating*, while *scraping* and *shaving* stood at around 70%.

(b) Performance results on the dataset B

Dataset B was significantly larger than dataset A, and the overall accuracy of the C4.5 was slightly lower (see Table 3.3). On reflection this is to be expected as in dataset A all annotators agreed, and thus we expect the actions that have been annotated to unambiguously reflect the guidance given as to what use of a utensil corresponds to each activity. Note that C4.5 still achieved an overall accuracy of nearly 80% for all window sizes.

	Window size				
Algorithm	32	64	128	256	512
Decision Tree C4.5	77.0	76.8	80.2	77.5	80.1
Bayesian Networks	67.5	70.2	73.6	71.3	74.5
Naïve Bayes	61.3	61.8	62.2	73.5	72.7

Table 3.3: Overall accuracies (%) on dataset B.

The performance of Naïve Bayes on dataset B is moderately better than on dataset A as the dataset B is bigger than dataset A and Naïve Bayes requires significant amounts of data for training.

	Window size									
	32		64		128		256		512	
Activity	instances	%	instances	%	instances	%	instances	%	instances	%
chopping	6184	88.3	3116	86.7	1510	88.1	726	86.6	328	90.9
slicing	472	31.6	230	75.7	86	67.4	50	70.0	16	75.0
peeling	788	72.5	378	27.5	150	35.3	82	24.4	36	19.4
dicing	162	3.7	78	14.1	38	15.3	16	12.5	4	25
coring	1246	94.5	612	94.8	300	92.7	130	86.6	64	82.8
spreading	2177	88.0	1034	90.5	498	90.4	260	86.9	116	82.8
eating	326	21.5	164	10.4	82	31.7	28	35.7	18	5.6
stirring	1122	74.7	558	70.8	190	77.4	134	70.9	64	81.3
scooping	796	37.3	390	39.5	186	51.1	76	47.2	32	81.3
scraping	716	62.2	364	64.3	172	65.2	82	65.8	32	56.3
shaving	848	75.9	416	79.6	200	90.5	88	86.4	42	92.9

Table 3.4: Detail classification results (accuracy in %) for Decision Tree C4.5 on dataset B.

The details (activity-by-activity) of the performance of C4.5 on dataset B are presented in Table 3.4 and it can be seen that these are consistent with that of dataset A. The activities of *chopping*, *coring*, *spreading* and *shaving* were accurately classified (i.e. >85%). Accuracies for the classification of *slicing*, *stirring*, and *scraping* were around 70%. Again the lowest performance was for *dicing* (12.5%).

3.6 Conclusion and discussion

The aggregated results revealed the best performance for the C4.5 classifier at different window sizes across different activities. As one might expect, all classifiers performed worst for activities that are intuitively less well defined and in themselves were difficult to label for our human annotators. Classifier performance for relatively unambiguous activities such as *chopping*, *peeling*, *coring*, *stirring*, and *scooping* was above 80%, while classification of less distinct activities was significantly less accurate. This may in part be due to the low number of training instances for activities such as *dicing*, *slicing*, and *scraping*. We also saw significant improvements in accuracy for these activities on dataset B, where the number of training samples was higher (this is particularly true for *scraping*). Notably, as one might expect, since the activities themselves have different temporal scales (e.g. *dicing* vs. *eating*) the window size for which a classifier is most accurate varies across the activities.

Our initial experiments showed that there are several areas in which Slice &Dice will need to improve in order to develop a real-time AR system for kitchen activities that can provide the sort of situated support for food preparation that we envisage. First, although a generally good level of overall accuracy was achieved (i.e. 82.9%), the accuracy for each activity varied widely, from over 90% for peeling, to near 10% for dicing. Therefore, further work is required to find a better classifier, in particular one for which the accuracies between activities are more balanced. Secondly, in addition to accuracies (or recall), evaluation metrics such as false positive rates, precision, and confusion matrices should be used to provide more insight into the results. Finally, practical real-time AR must incorporate a rejection option for abnormal activities (i.e. unknown activities). We have addressed all this improvements in the next chapter (Chapter 4).

In conclusion, Slice&Dice should be considered as a proof-of-concept activity recognition prototype that can recognise 11 mixed-salad preparation activities. With the reliable recognition rate of more than 80%, the system achieved our aim of demonstrating the feasibility of the use of pervasive sensing in the recognition of fine-grained human activities (in the kitchen). To the best of our knowledge, Slice&Dice was the a first work that systematically addressed the AR problem with embedded sensors in multiple kitchen utensils. Slice&Dice is the first step in the development of a low-level activity framework for fine-grained food preparation activities.

Chapter 4: Real-time Activity Recognition

This chapter builds on our initial Slice&Dice prototype described in Chapter 3 and presents a real-time recognition framework for low-level food preparation activities. The recognition method is based upon dynamic time warping (DTW) which is a simple, fast, yet effective technique for real-time classification. As in our initial Slice&Dice application the recognition framework analyzes frames of contiguous sensor readings, but in this case the classification is in real-time and with low latency. Our DTW approach adapts to the idiosyncrasies of utensil use by automatically maintaining a template database. We demonstrate the effectiveness of the classification approach through a number of real-world practical experiments on our publically available dataset. Notably, our adaptive system shows superior performance when compared to a static recognizer. Furthermore, we demonstrate the generalization capabilities of the system by gradually reducing the number of training samples used, and demonstrate that the final system achieves appropriately accurate classification results even if only a small number of training samples is available. This is particularly relevant to our envisaged scenario of real-time situated support of food preparation in the kitchen. In section 4.2 we present the DTW-based method for real-time AR including an overview of DTW and the development of a DTW-based recognizer that incorporates the rejection option (i.e. detection of unknown activities as identified as necessary in Chapter 3), and how to improve the recognizer with adaptation. Our experimental evaluation is presented in section 4.3, and in section 4.4 we discuss our results and report our conclusions.

4.1 Introduction

In Chapter 3 we explored the nature of various low-level food preparation activities in detail, and it became clear that even the most fundamental activities exhibit significant variance. As already identified, the perceived variance is in part due to inherent ambiguity in our informal definitions of different food preparation activities (e.g. there is no widely accepted commonsense definition of *dicing*). But this variance also stems from a number of additional factors, including personal preferences of how to handle food ingredients and utensils, the level of formal or informal training a person has, the nature of the food ingredient and its intended post-preparation form, and even the biomechanical profile of the user (e.g. the size and strength of their hands). For example, consider the process of *shaving a carrot* using a *knife*. Some people perform long, slow movements of the utensil along the carrot towards themselves, which is comparable to “carving” the vegetable. Others tend to perform short, fast cuts of the carrot’s surface thereby using the knife in more of a “chopping” action. Although both kinds of movements differ substantially, they represent the same kind of activity and an automatic recognition system needs to cope with this. People with experience of cooking generally use utensils in a more coordinated manner, and

depending on the size of the utensils handle, the utensil itself better suits people with hands of different sizes.

In order to address this variation, we developed a fully automatic, real-time activity recognition system (the processing pipeline for which is outlined in Figure 4.1). As before, accelerometer data is recorded while a person is working in the kitchen and we use data from the modified Wii remotes integrated into standard kitchen utensils. The continuous sensor data streams are (x, y, z) acceleration triplets at a sampling frequency of 40Hz, and frames of 64 contiguous samples are extracted in a sliding window procedure. The use of sliding window size of 64 (i.e. 1.5 seconds) is necessitated by the need for real-time processing and thus a practical recognition rate (an ultimately response in an interactive system). Following some basic pre-processing and trivial movement detection (using a simple threshold based procedure), the actual classification of the extracted frame regarding the activities of interest is performed (central component of Figure 4.1). This recognition procedure is performed as a DTW-based template comparison with an automatically maintained template database. This database contains representative templates for the activities of interest together with activity specific thresholds for acceptance and rejection. By analyzing the DTW scores the template database can be continuously adapted to represent the idiosyncrasies of the particular activities performed by different users. The output of the system consists of classification hypotheses for every extracted frame, including possible rejection, which in effect means segmentation of continuous sensor data streams

4.2 Dynamic Time Warping based activity recognition for food preparation

In this section we first briefly summarize the theoretical foundations of Dynamic Time Warping before presenting a detailed description of the key components of our real-time AR system for food preparation.

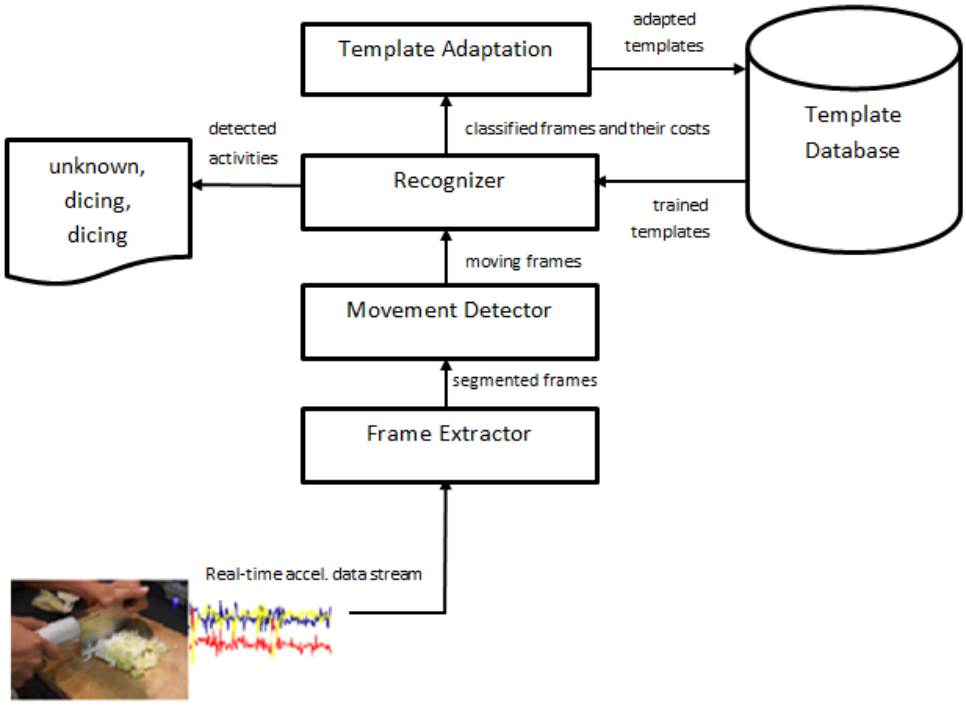


Fig. 4.1: Activity recognition for food preparation tasks (system overview).

4.2.1 Dynamic Time Warping: a brief overview

DTW [48] is a constrained, non-linear pattern matching method based on dynamic programming for measuring the dissimilarity between two time series. Let $O = o_1,o_2,..,o_m$ and $Y = y_1,y_2,..,y_n$ be two time series. DTW finds an optimal mapping from O to Y by reconstructing a warp path W that optimizes the mapping of the two sequences: $W = w_1,w_2,\ldots,w_K$ where $max\{m,n\} \leq K < m+n$, where K denotes the length of the warp path. The warp path is constrained in the sense that it is anchored by the start and end points of both sequences. To map the *in-between* elements of both time series a step-wise distance minimization is performed for every position:

$$\delta(W) = min\ \{ \sum_{k=1}^{K} \delta(w_{ki}, w_{kj}) \} \qquad (4.1)$$

The actual distance calculation is usually (but not necessarily) based on the Euclidean distance $d(o_i,y_j)$ and dynamic programming [48]:

$$\delta(i,j) = d(o_i,y_j) + min\ \{ \delta\ (i\text{-}1,j),\ \delta\ (i\text{-}1,j\text{-}1),\ \delta(i,j\text{-}1) \} \qquad (4.2)$$

where i and j represent monotonically increasing indices of the time series O and Y. The resulting matching cost, $\delta(W)$, is then usually normalized to the range [0,1] to ensure comparability.

4.2.2 DTW-based recognition

As before, our classification problem relates to the context in which accelerometers integrated into kitchen utensils continuously stream three-dimensional (x,y,z) acceleration data. A sliding window procedure aggregates 64 contiguous samples as frames, and adjacent frames overlap by 50%. The actual parameterization of the frame extraction process has been optimized in previous experiments (i.e. [16]). For every observation frame *O[u]*, which is recorded for a particular utensil *u*, activity recognition is performed using the DTW-based algorithm as described in Figure 4.2. Note that by means of a trivial threshold comparison only those frames are considered where the utensil was actually moving. After computing and sorting the DTW scores for all templates (lines 3:8 in Figure 4.2), the set of the smallest *min(K,n)* scores is compared to the activity-specific thresholds (lines 9:15 in Figure 4.2), so-called class-based threshold DTW (each activity has its own threshold) or CBT-DTW. If none of the DTW scores contained in the sorted list cost is smaller than the particular threshold the observation frame *O* is rejected, i.e., assigned to the unknown class. Heuristically we chose *K=10*, which provided reasonable results for acceptance and rejection on a cross validation set. The threshold function (*Thresh*, line 10 in Figure 4.2) retrieves the class-based threshold of the template *Y[index[i]]*.

```
Input:      Observation frame O; Utensil u;
            Number K of sorted match scores to analyze for final result
Output:     Activity hypothesis

      //Extract templates
1:    CurrentTemplates= ExtractTemplateDB(u);
2:    n = the number of templates in CurrentTemplates;
3:    For i from 1 to n do
4:          Y[i] = CurrentTemplate.template[i];
5:          cost[i] = DTW(O,Y[i]);
6:          index[i] = Y[i].Id;
7:    End for
      // sort and maintain indices of the templates
8:    Sort(cost, index);
9:    For i from 1 to min{K,n} do
10:         If cost[i] < Thresh(Y[index[i]]) then
                  //Acceptance
11:               activity(Y[index[i]]) -> activity_list;
12:               break;
13:         Else
                  //Rejection
14:               unknown activity -> activity_list;
15:         End if
16:   End for
17:   Return (activity_list);
```

Figure 4.2: DTW-based activity recognition.

The recognition procedure utilizes activity specific thresholds T_a. Applying Chow's rule for reject option with multiple class-based thresholds [47] to our domain (i.e. unknown activities need to be rejected), an observation frame *O* is classified as being a particular activity, *a*, if:

$$\delta(O,t) = \{ \delta(O,t_j)\} < T_a \quad (4.3)$$

where *t* represents a template from the database, which represents the activity *a*. *N* is the overall number of templates in the database. A frame is classified as unknown (i.e., to be rejected) if:

$$\delta(O,t) = \{ \delta(O,t_j)\} \geq \{T_i\} \quad (4.4)$$

where n denotes the number of classes of interest. The class-based thresholds (CBT) were manually selected through a 5-fold cross validation procedure.

4.2.3 Template adaptation

In order to adapt the overall recognition system to account for the idiosyncrasies users (e.g. personal preferences, biomechanics, etc) and use (e.g. the ingredient being acted upon), the template database is continuously updated, that is, templates are removed and added when necessary. The adaptation scheme used can be described as follows. Let f_k define the weighted histogram of recognition hypotheses for a particular time step *k* consisting of activity specific entries *fk(a)*, where *a* specifies the particular activity, and w denotes an (heuristically chosen) adaptation weight:

$$f_k(a) = \begin{cases} f_{k-1}(a).w & \text{if } \delta(a,t) < \alpha \\ f_{k-1}(a).(1-w) & \text{otherwise} \end{cases} \quad (4.5)$$

α denotes an acceptance/rejection threshold, which is derived from the activity specific template thresholds:

$$\alpha = T_a / (1 + T_a) \quad (4.5a)$$

All thresholds were optimized in a separate cross-validation procedure. Let $\gamma_k(a)$ be the cumulative number of recognitions of activity *a* at time *k*. The positive probability of activity *a* at time *k* is computed as:

$$\rho_k(a) = \frac{f_k(a)}{\gamma_k(a)} / \sum_k \frac{f_k(a)}{\gamma_k(a)} \quad (4.6)$$

The negative probability of activity *a* at time *k* is therefore defined as:

$$\phi_k(a) = \frac{1-\rho_k(a)}{\sum_k 1-\rho_k(a)} \quad (4.7)$$

At time k, if an observation frame makes $\varphi_k(a) = \min \{\varphi_i(a)\}$ for i =1..k, then this frame will be updated as a template in the *positive list* of the template database for later use (i.e. adaptation). The template which makes $\varphi_k(a) = \max\{\varphi_i(a)\}$ for i=1..k is moved to the *negative list* in the template database at the same time. The negative list is used only when recognizer has recognised an unknown activity on the positive list, then recognizer one more classification attempt on the negative list before returning the activity list.

4.3 Experimental evaluation

In order to evaluate the applicability of the proposed class-based threshold dynamic time warping approach (CBT-DTW) to activity recognition, we performed a number of experiments. We used the (publically available) dataset from Chapter 3, in which 20 persons pursued typical food preparation tasks (salad and sandwich making) using our sensor-equipped utensils. Ten typical low-level activities were subject to recognition, namely *chopping*, *peeling*, *slicing*, *dicing*, *scraping*, *shaving*, *scooping*, *stirring*, *coring*, and *spreading*. Additionally, a considerable amount of sensor data, the activities associated to which does not belong to one of these ten known identified activities, nor to "idle" (i.e. utensils not active) was included in the dataset. In total more than 6 hours of sensor data was collected from the four sensor-equipped kitchen utensils (i.e the knives and the spoon).

Extending our proof-of-concept Slice&Dice system, presented in Chapter 3, we aimed to conduct realistic experiments using the recognition system in a real-world scenario. This implies that the recognition system was applied online, i.e., continuous data streams had to be segmented and classified (open lexicon with rejection) in real-time, i.e., with negligible latency (the results are given in section 4.1). Furthermore, as previously explained, we were interested in the dependency of the recognition procedure on the number of annotated samples available for training. Since manual annotation is tedious and costly, it is somewhat unrealistic to rely on extensive training sets in order to apply recognition systems of this class. Consequently, we performed a second set of experiments in which the number of training samples was systematically decreased step-by-step. For this systematic exploration of the impact of training set size we report the classification results are reported in section 4.3.2

Recognition results are reported as frame-wise precision and recall values. The *precision* for an activity was calculated by dividing the number of correctly classified frames by the total number of frames classified as being a particular activity (i.e. *true positives*/(*true positives* + *false positives*)). *Recall* was calculated accordingly as the ratio of the number of correctly classified frames to the total number of frames of an activity (i.e. *true positives*/*total number of frames of an activity*). For comparison, baseline results for the evaluation of the decision tree-based system described have already been provided in chapter 3.

4.3.1 Results for full training set

For the first set of experiments we used all the available training data (see below) to evaluate the recognition system. The dataset was manually annotated by 3 independent annotators, and the consensus of the three annotators served as ground truth. Since we recorded complete cooking sessions the dataset, for obvious reasons, is dominated by idle "activities" (i.e., frames recorded while the particular utensil of interest is not moved at all). A quick subject-independent test (including all idle frames) led to 96.36% overall accuracy as the recognition of *idle* is almost perfect (based on a simple threshold comparison). To avoid this over-optimistic (and not so informative) evaluation, we limited the set of "idle" frames to four per utensil, which were randomly selected per subject. This effectively truncates the dataset to 12,265 frames (of 64 samples each).

Activity	**CBT-DTW**			**Decision Tree C4.5**		
	Precision	**Recall**	**False Positive**	**Precision**	**Recall**	**False Positive**
chopping	82.61	88.54	2.22	82.21	87.5	7.37
coring	77.02	81.94	0.21	74.02	77.7	4.12
dicing	51.16	54.63	0.25	24.87	18.7	4.25
peeling	72.76	80.63	0.53	88.7	95.9	3.91
scraping	80.09	81.1	0.12	56.8	56.3	3.37
shaving	72.79	82.73	0.28	55.11	59.7	2.91
slicing	70.31	70.73	1.21	33.47	26.6	4.95
spreading	71.06	86.57	0.77	54.33	44.4	2.32
scooping	97.92	94.55	0.78	91.2	86.3	2.6
stirring	84.77	86.98	0.08	81.63	85.92	1.26
idle	100.00	100.00	0.00	100.00	100.00	0.00
unknown	91.00	80.92	4.96	85.30	83.20	9.82
Overall	**83.02±4.8**	**82.78±5.5**	**2.61±1.03**	**77.9±8.7**	**76.7±6.5**	**6.29±2.1**

Table 4.1: One-subject-leave-out evaluation (all figures are percentages).

The evaluation was performed in a "leave-one-subject-out" manner, that is, we trained the recognizer using the data from 19 subjects, and tested on the data recorded for the remaining subject. This process was repeated for all 20 subjects and results were averaged. The overall performance of the proposed DTW-based approach is presented in Table 4.1. Additionally, the results are compared to those of the baseline system (Decision Tree C4.5). It can be seen that the new approach clearly outperforms the baseline with an overall precision rate of approximately 83% (CBT-DTW) as compared to 77.9% (Decision Tree C4.5), and an overall recall rate (averaged all classes) of 82.8% (CBT-DTW) compared to 76.7% (Decision Tree C4.5). All differences are statistically significant.

Additionally Table 4.2 illustrates the aggregated confusion matrix for the subject-independent evaluation of CBT-DTW. More details of CBT-DTW performance metrics are present in this confusion matrix, which has been aggregated from the recognition results of the "leave-one-subject-out" evaluation procedure.

In addition, as seen in Table 4.1, areas of low recognition rate performance by C.4.5 are significantly improved upon by CBT-DTW, for example, dicing (from 18.7% to 54.6%), slicing (from 26.6% to 70.7%). As shown in Chapter 3, the number instances of dicing and slicing was much smaller than the number instances of other activities, and consequently the initial low level of performance was due to the fact that these machine learning-based recognition algorithms evaluated in Chapter 3 (including C4.5) are highly dependent on the number of samples available to train the models. Therefore, we conducted another evaluation to explore the relationship between the number of trained samples and the recognition performances of both C4.5 and CBT-DTW, which are presented in Section 4.3.2.

	chop	*coring*	*dicing*	*peeling*	*scraping*	*shaving*	*slicing*	*spreading*	*scooping*	*stirring*	*idle*	*unknown*
chop	88.4	0.1	5.5	0	0.3	0	4.6	0	0	0	0	1.1
coring	0	81.9	0	6.4	3.1	0	0	38.9	0	0	0	4.7
dicing	36.6	0	54.6	0	1.2	0	5.2	0	0	0	0	2.5
peeling	0	4.0	0	80.6	5.8	3.7	0	0	0	0	0	5.9
scraping	2.3	1.1	0.1	4.1	81.1	2.6	1.0	0	0	0	0	7.8
shaving	2.0	0	0	8.4	1.2	82.7	0	0	0	0	0	5.6
slicing	15.0	1.7	7.63	0	1.4	0	70.7	0	0	0	0	3.5
spreading	0	0	0	0	5.8	1.7	0	86.6	0	0	0	6.0
scooping	0	0	0	0	0	0	0	0	94.6	2.1	0	3.4
stirring	0	0	0	0	0	0	0	0	8.1	87.0	0	5.0
idle	0	0	0	0	0	0	0	0	0	0	100	0
unknown	3.28	0.92	0.03	2.3	2.96	0.58	1.4	3.51	3.18	0.95	0	80.9

Table 4.2: Aggregated confusion matrix for "leave-one-subject-out" evaluation of CBT-DTW (%).

4.3.2 Results for reduced training sets

In the second set of experiments we analysed the dependency of classification accuracy on the number of samples available for training the recognizer. We did this by gradually reducing the amount of annotated training samples through a random selection of frame to be excluded from training, and evaluated the resulting recognizers on the remaining data. Figure 3 summarizes the classification accuracies of both the proposed CBT-DTW approach and the baseline system C4.5 classification algorithm. The number of frames used for model training is display on the *x*-axis, whereas the *y*-axis represents the classification accuracies. For the sake of clarity in the illustration the (discrete) accuracy and sample number values are displayed as (continuous) curves by interpolation through the sample values. It can be seen that the proposed DTW-based recognition produces significantly generalizable models even when only small amounts of training data are available. For example, with only 5 labeled frames for training (per activity) the accuracy of CBT-DTW is still approximately 78% (see figure 4.3). The performance of the baseline C4.5 system here drops to approximately 55%.

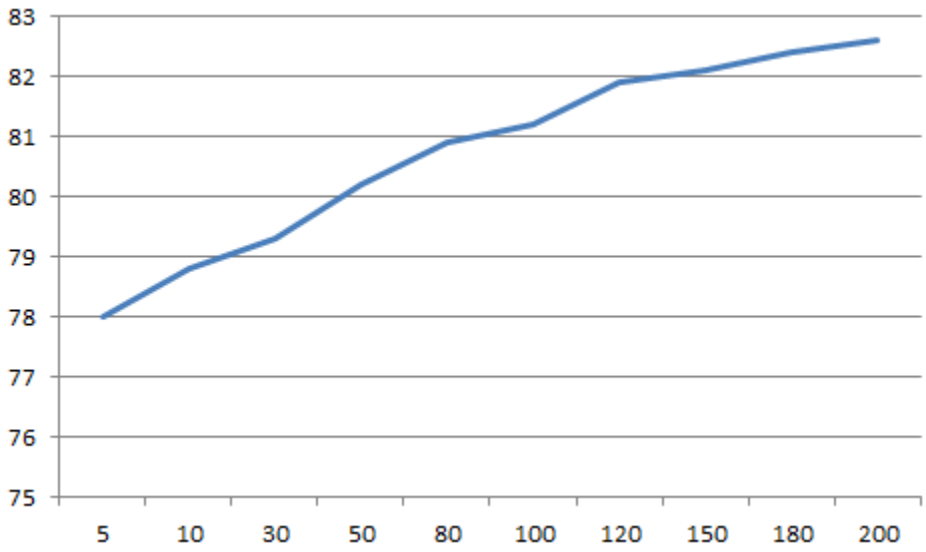


Fig. 4.3: Classification accuracies of CBT-DTW in dependence of the amount of training data.

4.4 Conclusion and discussion

Our improved Slice&Dice prototype activity recognition system has the potential to automatically adapt towards the idiosyncrasies of people using kitchen utensils. Based on a Dynamic Time Warping procedure a template matching system has been developed, which successfully segments and recognizes ten low-level kitchen activities by analysing contiguous frames of sensor values. The adaptation of the

activity recognition systems to variation in certain activities is pursued by an automatic maintenance procedure, which effectively updates the template database when necessary. By means of an experimental evaluation on a large, realistic datasets that cover unconstrained food preparation we demonstrated the capabilities of the proposed approach. In a second set of experiments we gradually reduced the number of training samples. The proposed DTW-based recognition system shows superior generalization even if only a few samples are available for training.

As we have argued, the automatically monitoring of food preparation activities is a key element of a situated support system, but such systems must be non-intrusive. Intrusiveness can occur in a variety of forms. In Chapter 3, we showed how by careful physical design we can develop utensils that have the sensors for the activity recognition into the actual objects (thereby obviating the need to wear a sensor or have invasive computer vision systems in the kitchen). However, ease of deployment is another significant source of intrusiveness and with our CBT-DTW based system we have demonstrated that sufficiently accurate recognition results can be achieved using just a small number of examples. While we have not undertaken the interaction design that would be associated with the deployment of such a set of instrumented utensils, one might reasonably anticipate that if such systems are to adapt to the particular users then this adaptation must be effective with minimal additional training data (as we have demonstrated).

Finally, although we have demonstrated the feasibility of fine-grained activity recognition in the kitchen using pervasive sensing, the practical application of such a technology faces two additional challenges. Firstly, the food ingredients themselves provide the most valuable context information in the kitchen, and as yet we have not proposed a technical approach to their recognition. Moreover, while Wii Remotes have been embedded into specially modified handles of the knives and large spoon of our prototype, they are clearly inappropriate for many other kitchen utensils or appliances, for example, a *peeler*, *sauce pan* or *frying pan.* Based on these observations, in the subsequent chapters we demonstrate our efforts to develop a full kitchen deployment based on custom-design wireless accelerometers (whose size is much smaller than the Wii-remote) and embedded sensing objects such as a chopping board by which we can recognise fresh food ingredients.

Chapter 5: Fiber Chopping Board

This chapter presents the development of the Fiber Chopping Board (FCB) for fresh food ingredient recognition in the kitchen. We begin by describing the properties of materials and the hardware chosen for building the FCB, and then describe the process of its design and construction in more detail. The process of sensing image calibration and processing is presented in Section 5.4. Two food recognition algorithms are developed and implemented. One is based on optical fiber imaging (Section 5.5), and the other utilizes the acoustic data stream while food ingredients are being acted up (Section 5.6). Our conclusions and discussion is presented in Section 5.7.

5.1 Introduction

Previous research [26, 51] has demonstrated that situated services, such as prompting, can be significantly improved if they can derive contextual information (i.e. 56% accuracy for context-unaware compared to 74% for context-aware in [51]). As food is an essential element of food preparation then its automatic recognition of is a key functionality of applications such as nutrition advices [30], dietary maintenance and monitoring [29], or assisted cooking, particularly for novice cooks [32]. However, existing pervasive kitchen environments [27, 29, 30, 32, 34] are either not able to detect (or have very limited detection capabilities for [15]) food ingredients while people perform cooking tasks. This is particularly true where fresh food ingredients are components of a recipe [30, 32].

Whereas in previous chapters we have demonstrated the feasibility of the recognition of human activities using accelerometers embedded in knives and spoon,in this chapter, we present the development of a compact, natural-looking chopping board using fiber optics and audio sensing technology. It is noticed that before developing the fiber chopping board we did several pilot studies about using acclerometer data from chopping/slicing activities to classify food ingredients while they are being chopped. The classification results are not so good (i.e. ~40% for 5 foods). Therefore we need to develop the fiber chopping board for food recognition. In contrast to previous work such as [15] where a readily observable microphone was installed in general kitchen environment, or [24] which used a wearable microphone, we used an optical sensing technology and a microphones that was completely embedded inside the chopping board (thereby satisfying our requirement for unobtrusive pervasive technology in the kitchen).

5.2 Materials and hardware devices for the FCB

5.2.1 Optical fibers

The basis of the FCB is a flat, low resolution imaging system for which we will use an array of fiber optic cables. An optical fiber [53] is made of a pure glass which is flexible and immune to electromagnetic

interference. Light can be transmitted between two tips (ends) of a length of fiber (see Figure 5.1). Critical for our application is the fact that fiber only propagates light that enters within the acceptance cone of the fiber, which is related to its numerical aperture (NA). The half-angle of the acceptance cone (the acceptance angle) is ($\sin^{-1}$ NA).

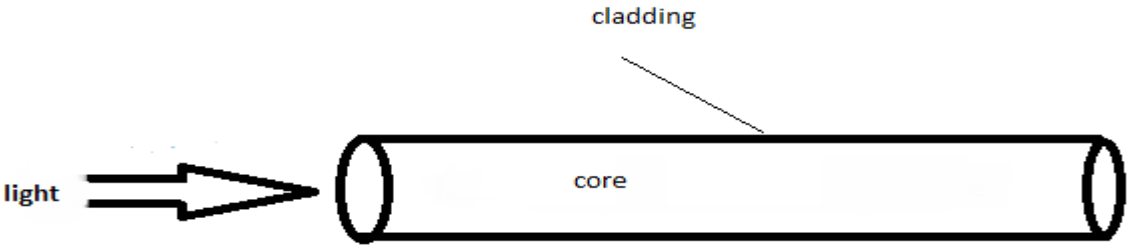


Figure 5.1: An optical fiber.

Both the core and the cladding of the fiber are transparent, and the core has a greater index of refraction than the cladding, which facilitates light being transmitted along the length through a process of total internal reflection. As in the FCB we want to make the chopping compact as a regular chopping board, we used 0.5mm unsheathed polymetylmethacrylate (PMMA) cored optical fibers with the following physical, optical and mechanical properties [56]:

- Resin core: 486μm diameter
- Refractive index: 1.49
- Cladding: fluorinated polymer (0.5 mm diameter)
- Strength: 14 N
- Bend radius: 10 mm
- Weight:0.3 g/m
- Optical mode: Multiple
- Numerical Aperture (NA): 0.5
- Acceptance angle: 30◦
- Temperature for permanent usage: from -40◦C to 70◦C

The first compact fiber optic sensing board was FiberBoard, developed by Jackson et al. [54] using a matrix of optical fibers to a channel light to a camera. FiberBoard is a surface that images infrared light scattered from touch points on a Frustrated Total Internal Reflection (FTIR) layer. Similarly, the FCB also uses a two-dimensional matrix of optical fibers to sense each discrete sensing point and terminate in a bundle that is placed at the aperture of a digital camera (in our case a commodity webcam). As the light can carry the colour information, when bundled, fibers can transmit imaged pattern of light which then is

decoded to produce an image map of the light transmitted through the fibers and can be further processed to determine images of the food placed on the surface.

5.2.2 Acrylic glass

Acrylic glass is a synthetic plastic material containing one or more derivatives of acrylic acid or poly methyl metacrylate (PMMA). An example of clear acrylic plastic sheet is shown in Figure 5.2. We chose acrylic glass to frame the FCB. The top sheet of the FCB is a transparent acrylic glass sheet (see figure 5.1 (left)), and the bottom sheet and surrounded frames are made of black acrylic glass. A white acrylic sheet was also use for the optical fiber mounting with which the fibers are securely held. All sheets used for the construction of the FCB were 5mm thick. The acrylic sheet has a number of attractive qualities including its lightweight, but also its strength and stiffness, in that it can endure food preparation activities (e.g. chopping) without being flexing significantly or fracturing.

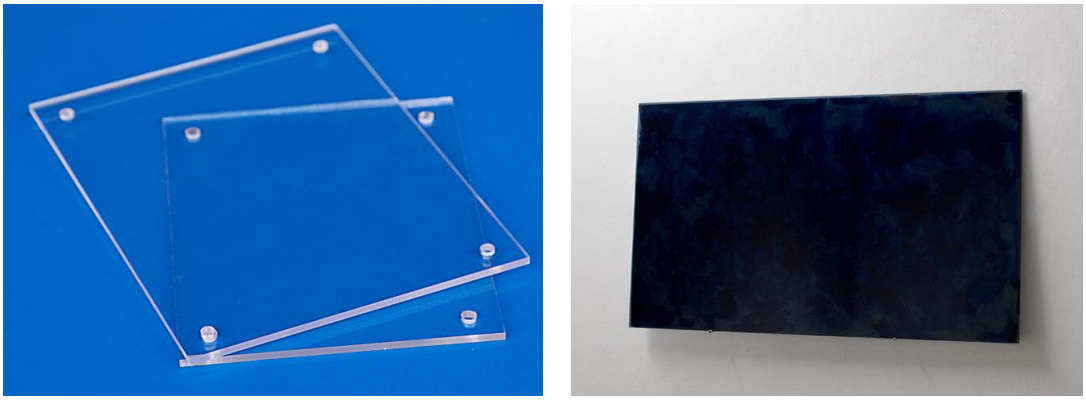


Figure 5.2: A transparent acrylic glass sheet (left) and a black acrylic glass sheet (right).

5.2.3 Embedded camera and microphone

We used a Logitech Quickcam Pro 5000 webcam board – after removing the external case this was small enough to embed within FCB. As a consumer webcam, the Logitech Quickcam Pro 5000 is a cheap imaging solution, but also captures video at resolutions up to 640x480 at a rate of 30 frames per second. The camera was particularly good for video capture in low-light conditions compared to other models we tested (i.e. the low intensity light that emerges from our bundle of fibers).

Figure 5.3: Logitech Quickcam Pro 5000 webcam (left) and its uncased version (right).

We utilised the webcam's built-in microphone, which itself has embedded (*Rightsound*™) acoustic processing algorithms that analyses the entire spectrum of sound to better isolate the ambient noise. To integrate the camera and microphone into the FCB, the camera is "hidden" in a box inside but the microphone is positioned in contact with the top sheet of the FCB to better capture the sound of utensils interacting with the food during preparation on the surface of the FCB (see 5.3.1).

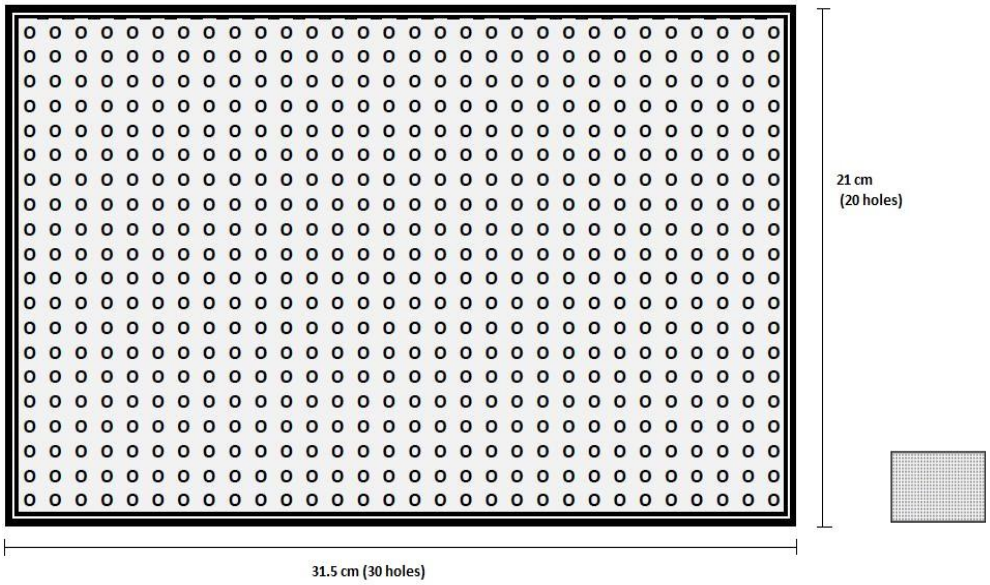


Figure 5.4: Fiber support sheet 31.5×21.0×0.5 cm (left) and 5.0×3.0cm fiber bundle sheet (right).

5.3 The design and construction

The *fiber support sheet* (see Figure 5.4 (left)) contains 600 holes for the optical fibers, hence 600 fibers arranged in a 30×20 matrix with dimensions of 31.5×21.0×0.5 cm. The grid covers an area of approximately 660 cm^2, which is typical of the surface area of the chopping board in everyday use. An additional *fiber bundle sheet* (5.0×3.0cm) is used to gather the receiving end of the fiber and hold these in place in front of the camera. The tip of the collecting end of the fiber is inserted into a hole from the underside of the support sheet and fixed in place, with the tip abutting the top face of the support sheet, using adhesive that is applied to both the *fiber support sheet* (underside hole opening) and the fiber itself.

The tip of the receiving end of the fiber is similarly inserted into the *fiber bundled sheet* (Figure 5.5) and fixed with adhesive.

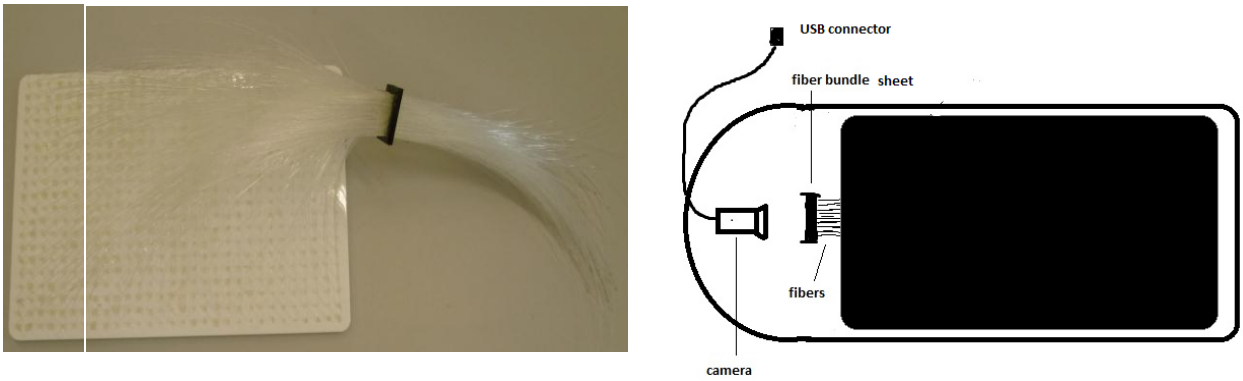


Figure 5.5: (a) support sheet and bundle sheet (left); plan view of the FCB configuration (right).

The surface of the FCB is a transparent acrylic sheet, which sits about the tips of the fibers that abut the top of the support sheet, this protects the fibers from damage or physical "clogging" by food stuffs. The microphone is fixed in contact with this surface layer (hidden from view) using adhesive tape. The frames of the side and bottom of the FCB are shown in Figure 5.5 (right) this physical support for the support, bundle, and surface sheets are made from black coloured acrylic glass. The dark colours ensures that ambient light from the sides and underside of the FCB does not enter the camera and thus the majority of the light transmitted by the optical fibers enters from the top-side of the FCB (as intended). Note that various spatial configurations and optical properties for materials were considered in the process of designing the FCB.

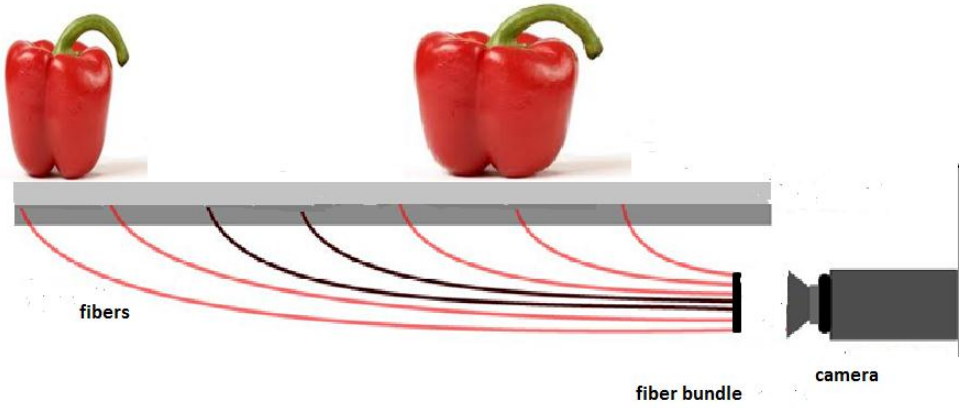


Figure 5.6: Sensing configuration: red fibers transmit colour information from the surface of the FCB.

Figure 5.6 shows the configuration of the fiber sensing system. As described there are two layers to the FCB's, the lower layer is a 600-hole grid, adhered to 600 tips of fibers, and the higher layer is actual surface of the FCB (on which food preparation occurs). In essence, the fibers allow us flexibility of camera placement, and allow us to produce a low-resolution imaging device whilst maintaining a reasonable overall thickness for the chopping board (in practice this is now limited only by the minimum bend radius of the fiber) – the resulting board is only slightly greater than 6cm in thickness. The fibers terminate in an incoherent bundle that is interfaced to a camera (see figure 5.6). Details of how the images of food on the surface of the FCB are reconstructed from the image emmerging at the fiber bundle sheet are presented in Section 5.4 (image calibration & processing).

5.4 Image calibration & processing

The optical fiber bundle is incoherent as there is no linear spatial mapping between the fibers in the support sheet and the fiber bundle, thus image from the incoherent bundle of fibers must be calibrated in order to compute the in-out correspondence, and thereby allow reconstruction of the actual input image. The raw output image is then filtered with a nearest neighbour algorithm and processed; this produces an image that is sufficient for feature extraction and classification.

5.4.1 Calibration

Developing an effective calibration mechanism would significantly enhance the utility of low cost fiber bundles and allow the development of the sort of thin form factor image processing applications that we are proposing. In practice, the quality of the image is mainly affected by non-uniform fiber distributions and the presence of regions with a relative large interstitial spacing (between fibers). For FCB imaging, a two-pass calibration process is used for creating the mapping that transforms the camera image of the incoherent fiber bundle to an image of the surface of the FCB.

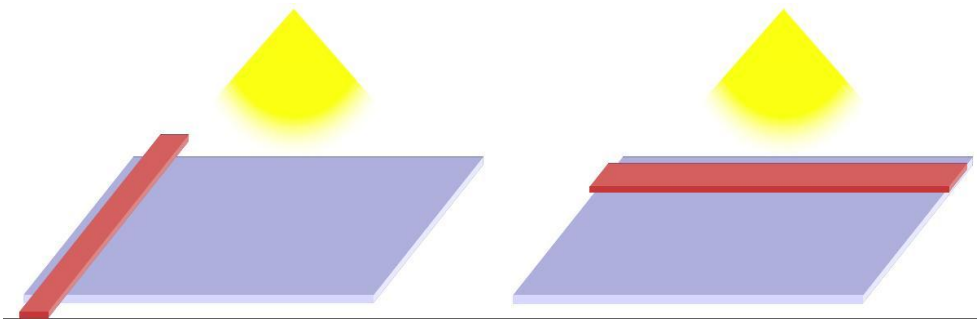


Figure 5.7: Two-pass calibration sweeping the tool over surface to occlude light (left and right).

We based out calibration software for the FCB on the approach used in [54]. First, a physical (visually opaque) rectangular tool is moved at a roughly constant velocity perpendicular to each of two orthogonal axes of the FCB's surface whilst the camera's images of the optical fiber bundle are recorded. The relative timing of the resulting fluctuating light, from each fiber, is used by the calibration software to determine the mapping between the camera image of the fiber bundle and the fiber tip locations in the support sheet. The tool is a simple strip of opaque material having a width of a similar order to the fiber separation distance, and a length no less than that of the FCB's surface. When placed on the FCB in a well-lit environment, the tool creates a clear shadow that can be observed at the camera as light attenuation from the occluded optical fibers. During calibration the tool is first moved along the full length of the FCB (see Figure 5.7 (left), the "horizontal" sweep), aligned so that its length is perpendicular to the direction of motion, while the sequence observed at the camera is recorded. This process is then repeated for the orthogonal axis (see Figure 5.7 (right), the "vertical" sweep). The two resulting video streams form the input to the calibration software.

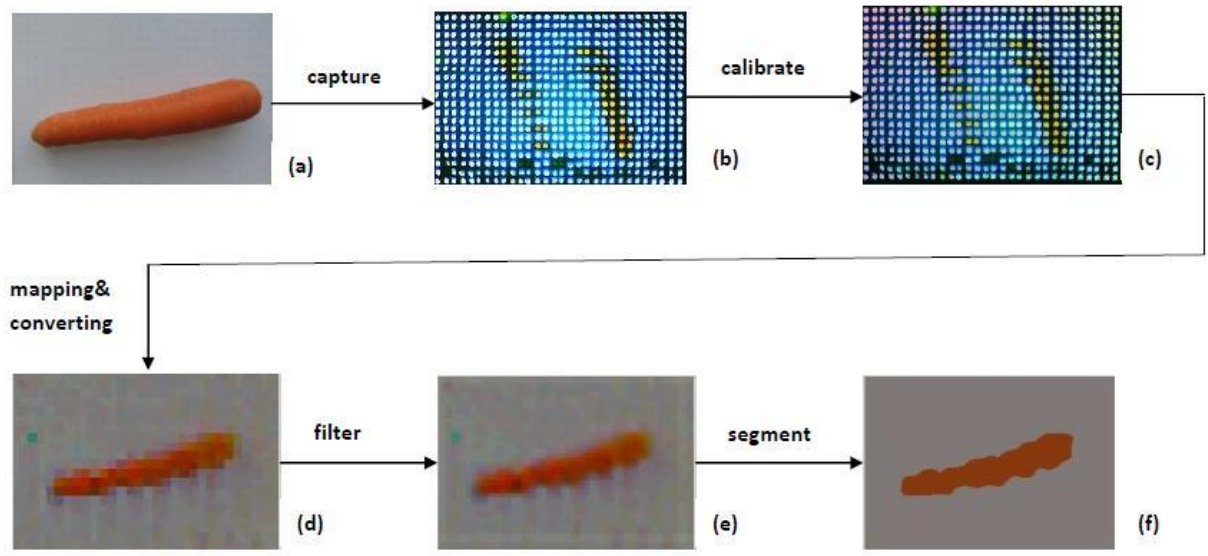


Figure 5.8: Image calibration & processing pipeline: (a) input food ingredient; (b) raw input image from the fiber bundle; (c) the mapping, which assigns each fiber to a specific polygon in the virtual representation; (d) raw output image; (e) image after nearest neighbour filtering; (f) segmented image.

The first stage of calibration analyses the video frames from the two sweeps, evaluating the luminance of each pixel to find its minimum. Pixels that vary sufficiently over time are assumed to represent a part of the fiber bundle, and the time of minimum illumination is proportional to the distance along the sweep axis that the optical fiber was occluded. For a given pixel, combining the values from both sweeps

provides a relative coordinate that corresponds to the position of the sensing end of the optical fiber in the plane of the FCB.

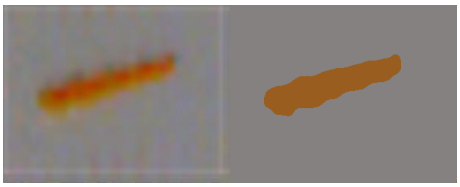
The output from the first stage produces a mapping (see Figure 5.8c) that does not identify individual fibers – the mapping must therefore be normalized to combine pixels that represent the same fiber. This cannot easily be done in the plane of the camera image as fiber tips with similar mapping results can be immediately adjacent making them difficult to distinguish. Instead, the normalization can be performed by first using the destination mapping values to produce a reverse-projected image in the plane of the FCB. An image is produced as a result of taking each mapping coordinate and including a point with a low opacity and a small radius at that location. The accumulated image is thresholded and a "blob detection" pass identifies and labels connected regions. Each region is grown up to its neighbouring regions and, finally, this data is used to normalize the coordinates in the original output from the first stage. In this final input mapping the pixels are grouped together where they correspond to the same sensing fiber tip and, in addition, the amount each pixel changes is used to determine a weighting value for that pixel's contribution.

At run-time, the input mapping is applied to the camera image, giving the average colour and brightness for each fiber. Further image processing requires an image of the FCB surface, so the individual fiber point values must be interpolated between their centres. In order to perform this calculation efficiently at run-time, three "output maps" are pre-computed, each specifying a fiber point and a weighting to apply at that location. To produce these output maps, the fiber points are tessellated into a triangular mesh and a bilinear interpolation is used to calculate the contribution from each vertex. After interpolation, the raw image is output (i.e. figure 5.8d). It can be noticed that the pixels of the raw image sometimes have a low quality. To address this a nearest-neighbour filtering algorithm is applied to the output image to smooth the noise. The nearest-neighbour filter approximates the colour of a point based on the colours of neighbouring points (using minimum distance). The filtered image is then used as the input to the final processing stages (i.e. figure 5.8e) segmentation and feature extraction.

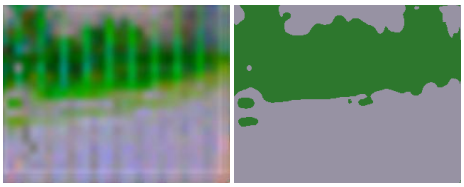
5.4.2 Image processing

The filtered image from the FCB is processed in two steps: *segmentation* and *smoothing*. In image processing, an image is a set of pixels with their colour values. Segmentation seeks to add real world meaning to an image by assigning each pixel with a corresponding label. Once segmented, an image may still contain noise clusters that are often small artefacts of the capture technology used, and not associated with the region of interest. Smoothing will remove such regions.

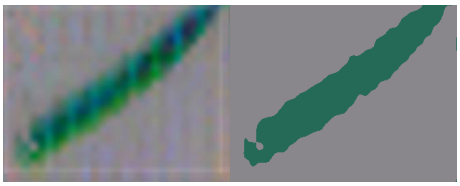
(a) Carrot



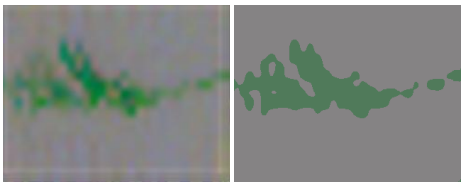
(b) Celery



(c) Cucumber



(d) Dill

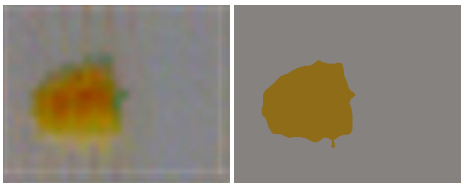


(e) Green pepper

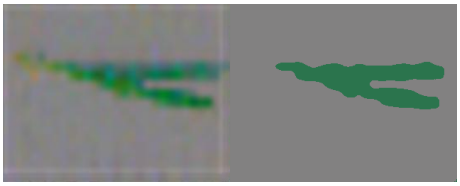
(f) Red pepper

(g) Peeled Onion

(h) Yellow pepper



(i) Spring onion



(j) Bacon

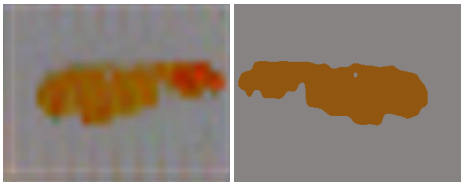


Figure 5.9: Examples of the FCB's filtered and segmented images.

5.4.2.1 Segmentation

Image segmentation is the process of assigning a "meaning" to every pixel in the image through the assignment of a label. As the number of colours of a natural food image is unknown, an unsupervised method is necessary to perform food image segmentation. Therefore, an unsupervised K-Means Clustering algorithm [58] was implemented to perform real-time colour segmentation. In brief, K-Means is an unsupervised statistical method used for partitioning *n* instances into *k* clusters in which each instance belonging to the cluster with the nearest mean.

The clustering procedure follows 3 steps:

***Step 1*:** Assign each pixel in the image to the cluster that minimizes the distance between pixel and the cluster centroid: $\text{argmin}_S \sum_{i=1}^{k} \sum_{x_j \in S_i} ||x_j - \mu_i||^2$ (5.1)

Where $\{x_i\}$ is a set of pixels of the image, μ_i is the centroid (mean) of pixel values in S_i

***Step 2*:** Re-estimate the centroid of each cluster by averaging all of the pixel values in the cluster.

***Step 3*:** Repeat steps 1 & 2 until no pixels change clusters (i.e. no pixel moves to another cluster).

We start by setting the number of clusters, *k=2*, and then gradually increase *k* until the stop condition holds. In this way, all pixels of the image after segmentation are labelled (i.e. with the label of the cluster to which they belong).

5.4.2.2 Smoothing

When captured under realistic settings the image will contain noise, which gives rise to some (small) clusters not connected to the region of interest (the region corresponding to the ingredient on the FCB). Therefore, we apply a morphological closing-opening operation [59] to smooth the segmented image. The main region of interest that contains food will be extracted using colour and SURF features for food recognition (in the next step).

5.5 FCB's Imaging based food recognition

The final step to be performed is the actual food recognition step based on the processed image generated by the FCB fiber imaging system. The recognition algorithm utilises SURF and colour features to classify the food images and was implemented to allow for real-time food recognition. We demonstrate the performance of the system with a pilot study.

5.5.1 Feature extraction

Speeded Up Robust Features (SURF) [60] is widely known as one of the most robust feature detectors and is used in numerous object tracking [61], object recognition [62] applications. Like SIFT [63], SURF features are descriptor-based features; unlike SIFT which is based on *Difference of Gaussians* (an approximation of the Laplacian of Gaussian), SURF is based on a *Fast-Hessian Detector* and has been demonstrated to be faster than SIFT and thus a more likely candidate for our real-time processing system. SURF is also well known to handle serious blurring (i.e. much better than SIFT) as it can still extract features from the blurred images. Moreover, SURF features are invariant to rotation and scale. These characteristics are very important for classification of food ingredients processed on the chopping board as the position of food will vary and they will have different sizes (i.e. a "baby" carrot is essentially a smaller version of a "large" carrot). While a SURF descriptor include the discriminating features one

might expect, such as *angle*, *edges* and *points*, the standard SURF ignores *colour*. However, colour information is very important for discriminating between food ingredients so in addition to SURF features, we also extract colour features and use these in our recognition pipeline.

To classify food, a feature extractor (FE) was implemented that comprised two main procedures. One is an implementation of Fast-Hessian detector, and the other is an RBG colour histogram. The input of FE is an image segmented by our implementation of the K-Mean Clustering algorithm, the output of which is two lists: one contains a 64-element list of SURF interest points (SURF features) $S=(s_1, s_2,.., s_{64})$, and the other is a 64-element colour histogram $C=(c_1, c_2 ,\ldots,c_{64})$. After normalization, these lists are combined into a 128-element feature vector:

$$V = [\alpha*s_1, \alpha*s_2,..,\alpha*s_{64}, (1-\alpha)*c_{65}, (1-\alpha)*c_{66} ,\ldots,(1-\alpha)*c_{128}] \quad (5.2)$$

where α is a weight by which SURF and colour matching features are proportionately ranked. In our experiment, a value of $\alpha=0.4$ was heuristically chosen (by evaluating different values of α in a pilot study). So colour features are slightly more weighted than SURF features.

5.5.2 Matching & Rejection

Two algorithms were compared for food image classification: k-*Nearest Neighbour* (k-NN) and *Support Vector Machines* (SVM). The former was implemented from scratch for real-time food recognition and the latter was based on the SVM library developed by Chang et al. [93]. In brief, k-NN classifies a food image based on the closest distance in feature space between test and training images. In the algorithm, k was set to 1 and the Euclidean distance in feature space was used. As irrelevant objects unintentionally placed on the chopping board (i.e. a knife or a cook's hand) must be rejected, a threshold t_i was assigned for each training food class f_i. The threshold was manually defined as the result of a 4-fold cross-validation procedure on the Euclidean distance between test and training food images. Such a food recognition algorithm is simple, but fast enough for real-time image classification.

A food is matched if the distanceof at least one of k closest images is greater than a threshold for the food class, otherwise, f_i is rejected (classified as an unknown food).

$$argmin_i\{d_i\} > t_i \quad (5.3)$$

Where d_i denotes the Euclidean distance of the test food image and the training food image of class i.Support Vector Machines [96] are a method commonly used for classification and regression machine

learning problems, and deal with the classification problem by finding the optimal separation hyperplane between classes based on the detection of representative training samples, so-called support vectors, of the boundaries of the class. One of the reasons for the selection of SVM for our food image classification problem is that SVM can find (and therefore classify) support vectors of the class even if only a small number of training samples are available (a reasonable assumption for real-world datasets which is often unbalanced).

More formally, where $x = (x_1, x_2, \ldots, x_{128})$ denotes a 128-element feature vector for the food image $I(x)$, SVM maps x into a high dimensional feature space H by I and constructs an optimal separating hyperplane in this space using a kernel function. Different kernel functions will construct different SVMs but as we need a real-time classifier we selected a linear SVM for our food image classification problem, and the remaining SVM parameters were set as follows:

- *Kernel function*: $f(x) = w^T x + b$ (5.4)
  in which $w = (w_1, w_2, \ldots, w_{128})$ is the weight vector, and b is the bias. $w^T x$ denotes the scalar product between the weight vector w and feature vector x:
  $w^T x = \sum_{i=1}^{128} w_i * x_i$ (5.5)
- *Cost parameter C and γ*, the maximal distance in H space between the hyperplane and the closest image $I(x)$, are estimated using a 4-fold cross-validation through the dataset.

The reason for the choice of k-NN and SVM is both these algorithms can deal well with high dimentional data (i.e. 128-d feature vectors) and are quite fast for real-time implementation. Furthermore, SVM is able to deal effectively with unbalanced dataset (real-world datasets often unbalanced) .

5.5.3 Pilot study #1

A pilot study was conducted to evaluate the food recognition algorithms performance (using the FCB's imaging technology). The collected dataset is comprised of 1800 images of 12 food ingredients listed in the below table. The number of each type of food image was randomly selected between 50 and 250. The collected food images varied in position, rotation, and included hand "distractions". The evaluation results of k-Nearest Neighbors algorithm are presented in Table 5.1 and the FCB sensed images, segmented images, computed colour and SURF features are available at:

http://di.ncl.ac.uk/publicweb/AmbientKitchen/ChoppingBoard/Choppingboard2/Dataset1/

In Table 5.1 it can be seen that *bacon* and *carrot* have high recognition rates (over 90%). While the colour of bacon and carrot are similar, their SURF features are quite distinct consequently only very few of the images of carrots and bacon were misclassified. Images yielding large numbers of false positives (i.e. over 20%) were of *peeled onions* and *yellow peppers*, thus the misclassification of (and confusion between) *peeled onions* and *yellow peppers* was considerable, as was the case for *tomatoes* and *red peppers.* In a few instances, sticks of celery and leaves of *lettuce* were misclassified each other where both colour and SURF features were very similar. However, the overall recognition precision and recall rate was approximately 80% on 1800 images of 12 food ingredients demonstrating that the optical fiber imaging method in FCB has promised as a practical embedded food recognition technology.

Food ingredient	**Precision (%)**	**Recall (%)**	**False Positive (%)**
Bacon	85.23	96.15	8
Carrot	90.08	92.18	4
Celery	59.09	87.64	16
Cucumber	87.30	88.00	4
Dill	88.27	78.61	8
Green pepper	85.29	88.46	15
Lettuce	93.20	67.71	18
Onion	75.00	60.78	27
Red pepper	73.30	81.64	26
Spring onion	83.33	87.12	2
Tomato	89.20	69.27	31
Yellow pepper	73.50	72.08	24
Total	**82.76**	**78.77**	**15**

Table 5.1: Preliminary evaluation results of food recognition using FCB imaging.

5.6 Food recognition using acoustic data

Acoustic event detection (EAD) approaches are widely used for activity recognition [65, 66] or speaker identification [64]. Applying EAD methods for food recognition is relevant because relatively distinctive sounds are generated when different food is being chopped. In this subsection, a food recognition algorithm using sound from a microphone embedded in the FCB is described. The algorithm recognizes the food being chopped in 3 steps: (i) audio segmentation; (ii) feature extraction and (iii) classification.

5.6.1 Audio segmentation

The audio stream (while food is being chopped) is segmented into one-second frames (each frame contains 44,100 samples). Each frame is then segmented into 50 blocks of 20ms each (each block contains 882 samples). It can be observed that the sounds made when chopping food occurs before a knife

make contact with the chopping board, so, to address such issues, an adaptive threshold based on energy is used – the segmentation algorithm works as follows.

```
1:    For each block in the frame do
            Compute energy;
2:    Compute Mean of energies of all 50 blocks in frame;
3:    Find the peak block i (highest energy);
4:    For blocks nearest to block i do
5:          If energy > 2*mean then remove block;
6:             remove block I;
   //possibly these are noises made from contact between knife and FCB
7:    For each block in the frame do
8:          If energy < 0.5 the remove block //silent block
```

Figure 5.10: Audio data segmentation.

The threshold is adaptive to the mean of energy as each food has a different audio energy mean. In the pseudo-code (figure 5.10), line 1 and 2 compute energy and mean of energy. Since the highest energy sound is most likely made by the contact between the knife and the chopping board this needs to be removed along with the nearest blocks to the peak if their energies are greater than two times of the mean, (actually an adaptive threshold). Line 6 removes silent blocks (i.e. when the knife is "in the air" before contacting to the food) as the blocks with energy smaller than 0.5*mean are likely silent blocks.

5.6.2 Audio feature extraction

After removing noise and silent blocks with the segmentation algorithm, the following audio features on each block of the remaining blocks are computed: (i) the first 13 Mel-frequency cepstrum coefficients (MFCCs); (ii) the spectral centroid; (iii) the spectral roll-off; (iv) energy; (v) entropy; and the zero crossing rate; all of which are widely used in AED classification problems [64, 65, 66]. Thus an 18-element feature vector computed for each block, and feature vectors are averaged for each frame. This results in one feature vector per frame (after averaging).

5.6.3 Audio classification

We use the widely deployed Gaussian Mixture Model (GMM) for audio classification (i.e. [64]). A GMM is the weighted sum of a number of Gaussians where the weights are determined by a Normal distribution.

Parameters $\theta_A = (\mu_i, C_i, w_i | i = 1. . . K_A)$ are extracted from training data to model the likelihood of feature vectors for every food class of interest *A*:

$$p(\vec{f}, \theta_A) = \sum_{i=1}^{K_A} w_i \mathcal{N}(\vec{f} \mid \vec{\mu}_i, \boldsymbol{C_i}) \quad (5.4)$$

where $\sum_{i=1}^{K_A} w_i = 1$ and w_i is the prior probabilities for the i^{th} mixture components.

$\mathcal{N}$ denote a Normal probability distribution with mean vector μ and covariance matrix C_i:

$$\mathcal{N}(\vec{f}|\vec{\mu},\mathbf{C}) = \frac{1}{|\sqrt{2\pi\mathbf{C}}|} e^{-\frac{1}{2}(\vec{f}-\vec{\mu})^T \mathbf{C}^{-1}(\vec{f}-\vec{\mu})} \quad (5.5)$$

Training mixture models for known food classes is a straightforward process. In brief, the parameters, w_i, μ_i and C_i, are estimated on class-specific training data using an Expectation-Maximization (EM) algorithm. EM is an iterative algorithm that converges to a local optimum by assigning posterior probabilities to each component density with respect to each observation. Obviously, the posterior probabilities for each point imply that each data point has some probability of belonging to each cluster. Note that K_A Gaussians are used to represent a food class. A must to be determined via 4-fold cross-validation. Each Gaussian component is defined by its mean and covariance, and the mixture is defined by a vector of mixing proportions. As a rule of thumb, the classification performance is proportional to the number of Gaussians that can be robustly estimated.

5.6.4 Pilot study #2

To test our audio-based food recognition algorithm, a pilot study was conducted on 4 food ingredients in the Ambient kitchen. One subject was asked to chop 10 carrots, 10 cucumbers, 10 tomatoes, and 10 lettuces. The recording session for each food started when the researcher notified the subject to start chopping allowing each food to be manually assigned a label by the researcher (e.g. *chopping a carrot*). After collection, the audio data was segmented, and the features extracted. Feature vectors were organized into 10 folders, 9 were used to train the GMM. The remaining folder was used for testing, and as before the procedure is repeated for all folders.

Food	**Frames**	**Accuracy (%)**
Carrot	169	89.94
Cucumber	218	77.06
Lettuce	123	91.06
Tomato	91	75.82
Total	**601**	**83.36**

Table 5.2: Food recognition results using acoustic data in Pilot Study 2.

With an overall recognition rate of 83% on 4 foods, the results demonstrate the food recognition using audio stream of the microphone embedded in the chopping board has some potential. Our approach differs in configuration with previous attempts (e.g. Kranz et al. [15]) in that we use a cheap, built-in microphone on a Logitech quickcam camera that was completely embedded inside the chopping board (and the food is classified only when it is being chopped).

5.7 Conclusion

Although our first FCB prototype is comparable to a regular chopping board in terms of size, dimensionality, weight, and functionality, there are a number of obvious deficiencies in its design. Firstly, the board itself took several weeks to construct by hand, particularly the adhesive fixing of the fiber tips in the holes in the bundle and support sheets. Moreover, its frames and components are made of acrylic glass, a material that is relatively expensive. Furthermore, the camera integrated into this first version of the FCB was a wired (with a USB connector) which is still some way from the form factor one might expect from an everyday (essentially non-digital) appliance. Despite these shortcomings, we have presented the design and development of a truly novel everyday appliance with embedded context-recognition, including our justification for the selection of materials and hardware, its design and construction, image calibration, processing and food recognition algorithms (based on the images and audio data the FCB produces).

Through two pilot studies our preliminary evaluations of the performance of both the underlying capture technology and the recognition algorithms demonstrate that the FCB is a viable candidate for a "disappearing technology" approach to context-recognition in the kitchen, in particular, for the challenging and previously unaddressed problem of robustly and unobtrusively recognising fresh food ingredients. Moreover, the pilot study results serve to motivate the design, development and evaluation of technologies and algorithms for a larger-scale, real-world experiment of context recognition in the kitchen. To this end, in chapters 6 and 7 we will present the design, conduct and evaluation of a study that involved a much larger number of redesigned instrumented utensils and the collection of naturalistic data from 12 subjects who cooked a typical spaghetti-based recipe with 59 activities and 8 food ingredients involved.

Chapter 6: KitchenSense: Real-time Context Recognition in the Kitchen

This chapter presents KitchenSense, a real-time context recognition framework in the kitchen. First, the design of an entirely new set of instrumented kitchen utensils is described, and then the implementation of KitchenSense for recognizing human activities and food ingredients in the naturalistic kitchen settings is presented. The framework can run as a background for context-aware based situated services and applications in a real kitchen. One such service is the tracking of progression steps in a recipe. Tracking progression steps within recipes can help novice user to cook more effectively, supporting their development of both skills and cooking knowledge, but it might also serve as the basis for a prompting system for cognitively impaired people in the kitchen. The utensil design is presented in the Section 6.2. The implementation of the recognition framework is described in Section 6.3 and we conclude with a discussion of KitchenSense in Section 6.4.

6.1 Introduction

In contrast to majority of related activity recognition research, this work utilizes three different types of sensors: wireless accelerometers, optical imaging, and acoustic data, all of which are embedded into either the chopping board (i.e. FCB) or other every day kitchen utensils. The main contribution of this chapter is two-fold. Firstly, we contribute the design of a new utensil set and the implementation of a real-time recognition framework in the kitchen. The key requirement for the former is that the kitchen utensils must be unobtrusive to users and the full functionality and other properties of the kitchen utensils are maintained. Secondly, one of the most important requirements for a recognition framework is that it must be scalable. That means, that the environment must be capable of supporting the full range and number of instrumented utensils and that one sensor-embedded utensil can easily be replaced with another. Moreover, the framework must provide a platform for the realization of future applications or situated services which utilize activity and food recognition.

To realize these requirements, the OpenMovement sensor platform was developed by the Digital Interaction Group's embedded engineering team (Dan Jackson, Cassim Ladha & Karim Ladha) at CultureLab, Newcastle University. OpenMovement includes a miniaturized wireless accelerometer, the WAX (formally WAX3), which can be easily integrated into modified utensils (see: [67]). Also, the physical handles and other elements of the kitchen utensils (adapted from commercially available utensils) have been completely re-modelled to provide both a more pleasing aesthetic and facilitate easier charging and replacement. The resulting sensors each have a *latch* that can be opened and closed to allow ready servicing of the embedded WAX devices.

For the implementation of the real-time recognition framework, a publishing-subscribe messaging paradigm (pub-sub for short) was proposed. One of the distinct advantages of pub-sub is that the components within the framework and applications can communicate with each other via a messaging mechanism, but can independently be developed. The architecture of the framework comprises four tiers: seven components, two event detectors, three training datasets, and three GUI applications. As an everyday kitchen is not used for large parts of the day, the event detectors utilize the cooking and food processing events to activate relevant components of the framework. In summary, the sensing technologies integrated into the redesigned utensils are invisible and unobtrusive to users in the kitchen and the scalability of the real-time recognition framework is maintained. Details of the kitchen utensil design and the implementation of the recognition framework are presented in the sections that follow.

6.2 Kitchen Utensils

Key requirements for the sensor platform and the instrumented utensils include that they are unobtrusive to users and maintain the full functionality (and other properties) of traditional kitchen utensils. Therefore, the sensors must be wireless and miniature such that they can easily be integrated into the bodies of utensils. As for Slice&Dice, the parts (mostly handles) of the utensils have been re-designed and modified using Fused Deposition Modelling (FDM) rapid prototyping technology.

6.2.1 The OpenMovement sensor platform

The OpenMovement WAX3 wireless triaxial accelerometer was developed by the Culture Lab Digital Interaction group embedded engineering team (Newcastle University) [67]. A WAX is miniature (32×13×9mm), lightweight (5g) (see Figure 6.1 (right)) and combines a MEMS accelerometer with an ultra low power IEEE802.15.4 radio. In form and performance they have been specifically designed to be easy to be embed into everyday objects (in our case kitchen utensils and appliances). In addition, WAXs are optimized for low-power sensing and transmission (i.e. several hours of continuous telemetry, and months in a ‘wake on activity’ state) and are encapsulated in a hygienic and robust housing [18]. As the default setting, WAX sensors are configured to a sampling frequency of 50Hz (although this can be configured). For activity recognition, the sampling rate of 50Hz would be reasonable while this would effectively save battery. We also did another study using 100 Hz logging accelerometers [28], and the quality of activity patterns is not significantly improved (i.e. 76% accuracy). By design the sensors are an ideal low cost mechanism for collecting real-time movement data that is wirelessly transmitted to a receiver (at a configurable rate and channel). Note that each device can be used as a transmitter or receiver, where the receiving WAX3 device uses a USB connection to pass received data to a PC, for device configuration and for battery recharge.

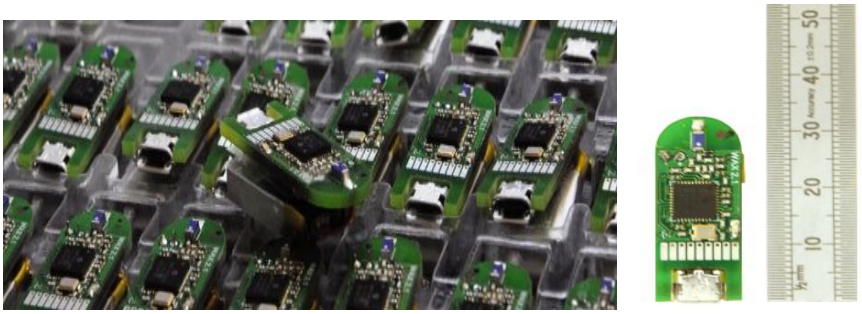


Figure 6.1: Culture Lab's OpenMovement wireless triaxial accelerometers (WAX3).

6.2.2 Knives

In our previous study (chapters 3 and 4) users used different knives while cooking, and their choice was highly dependent on their preferences and cooking skills. Therefore, a set of knives with re-designed and modified handles were developed. The knives used in KitchenSense constitute a significant improvement on those in Slice&Dice in that their form and balance are much closer to those one would expect to find in a regular kitchen (see figure 6.2).

Figure 6.2: The set of WAX embedded knifes (left); opened knife (right).

The set includes a *small knife*, a *bread knife*, a *chef knife*, and a *slicing knife*. While *slicing knife* and *chef knife* are likely to be used for similar tasks by users, the *bread knife* has a specialist use (for all but the most naïve cooks). The blades of the knives were fixed within the handle using strong epoxy resin glue. Inside the handle, pockets were designed, in both parts of the handle, to keep the sensor stable and correctly oriented and such that the antenna of the sensor is directed away from the steel blades of the knife.

6.2.3 Spoons & whisk

We group the spoons and the whisk, as these are utensils that can be used for *stirring*, *scooping*, or *whisking*. In contrast to Slice&Dice, which had only one large spoon, a set of 4 different cooking spoons and a whisk were developed for KitchenSense (see Figure 6.3 (left)). Similar to the knives, the handles of the spoons were re-designed and modified to integrate WAX sensors. The set of modified spoons include a *slotted spoon*, a *whisk*, a *spatula*, a *spoon* and a *ladle* (respectively left-to-right in Figure 6.3 (left)). In pilot studies we found that the *slotted spoon*, *spoon* and *spatula* were used interchangeably in some situations, although the *whisk* was by far the most likely utensils to be used for whisking and beating (i.e. an egg) and the *ladle* for scooping liquid ingredients.

Figure 6.3: The set of WAX embedded spoons (left); opened spoon (right).

Figure 6.4: Wax embedded sieve and colander (left); opened sieve (right).

6.2.4 Specialist utensils

Two more specialist utensils were included because of their importance to the recipe selected for the large-scale user study: a modified *sieve* and a *colander*. The sieve's handle was re-designed to place the sensor (see figure 6.4 right), the upper part can be stripped for opening or closing the handle. For the colander, we simply attached the WAX sensor at the side using epoxy resin. The *sieve* is a specialist utensil for *sieving* flour, although both the *sieve* and *colander* are often used interchangeably for draining pasta or rice.

6.2.5 Peeler

Due to the relative complexity of a peeling mechanism (compared to a handle) the *peeler* was the most complex utensils to redesign. Its inclusion is important since the peeling of fruit and vegetables is an activity that takes place intermittently in food preparation, and often (but not always) punctuates the preparation of different ingredients on the chopping board. The complete support frame of the *peeler* was re-designed (figure 6.5 (right)), again as two parts, and the blade from the original utensil re-inserted.

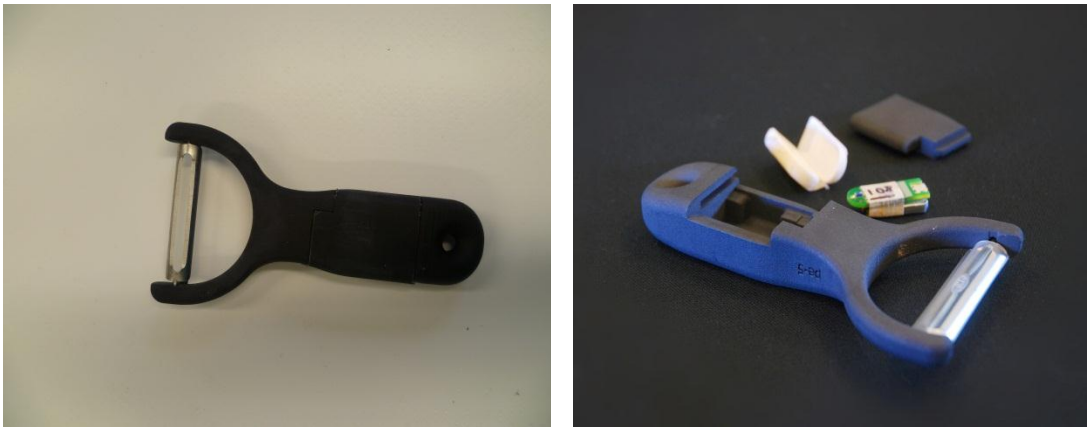


Figure 6.5: A WAX embedded peeler (left); opened peeler (right).

Figure 6.6: WAX embedded saucepans and frying pan (left); opened (right).

The set of pots and pans included two saucepans and three frying pans. For the saucepans, both the lid's handle and the pan's handle were modified to embed WAX (so two sensors per pan, see Figure 6.6 (top-right)). For the frying pans, only the handles were modified and one sensor was embedded per frying pan handle (figure 6.6, (bottom-right)).

6.2.6 Other sensors

In addition to the above kitchen utensils, KitchenSense includes the Fiber Chopping Board (Chapter 5) as a part the recognition framework, and several other WAX sensors were attached to ingredient containers including the bottle of vegetable oil, the salt container and the bottle of olive oil. Finally a WAX was attached to the water facet to allow us to detect when the water in the kitchen sink was being used. In total 22 sensors were registered to the framework.

To sum up, the pervasive sensing technologies used in the next study (see chapter 7) is developed in co-operation with the Culture Lab members. For example, the utensil set was designed by the Culture Lab designers (Isaac Teece, Juergen Wagner), wireless accelerometers are developed by the electrical engineering team (Dan Jackson, Cassim Ladha & Karim Ladha), and the hardwares of fiber chopping board was designed and constructed with the help of John Shearer.

6.3 The Implementation of KitchenSense

In order to realise the complex event-based heterogeneous sensing system, which includes 22 accelerometers, an imaging stream and an audio stream, a publishing-subscribe messaging based architecture was proposed for KitchenSense.

6.3.1 Architecture

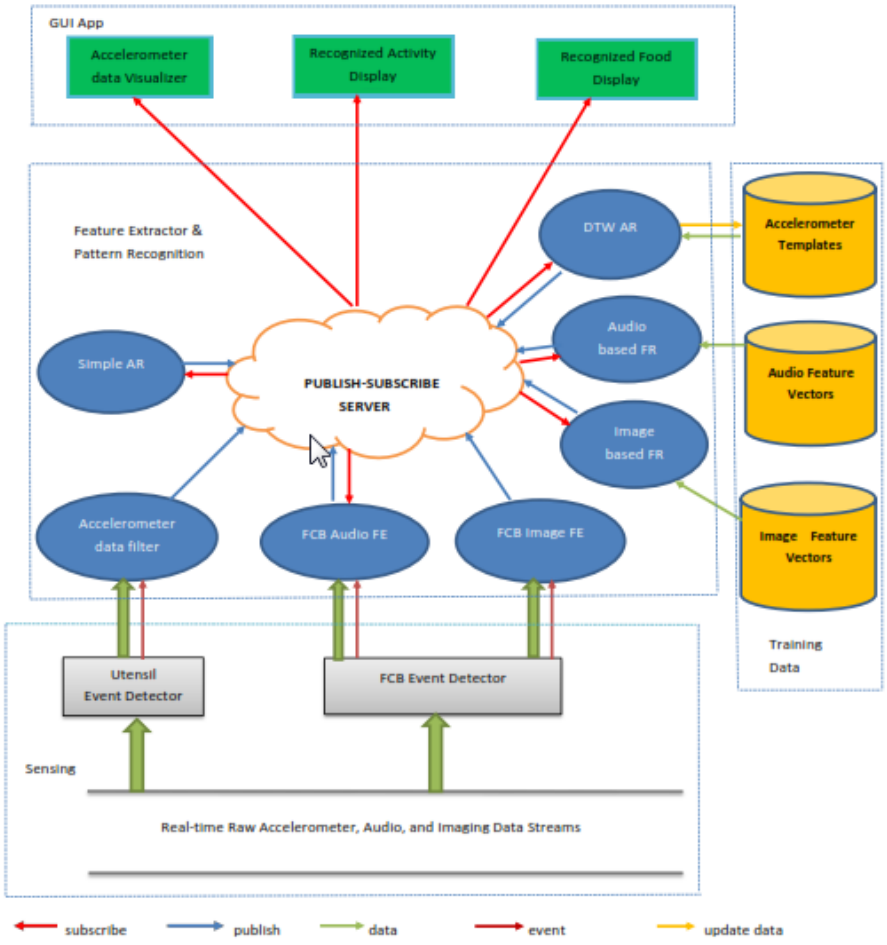


Figure 6.7: The KitchenSense architecture.

The architecture of KitchenSense is illustrated in Figure 6.7. A topic-based publishing-subscribe messaging schema allows message-based communications among components of the framework in Tier 3. In the pub-sub paradigm, components belong to one of three categories: (i) publisher; (ii) subscriber; and (iii) pub-sub server. All messages have the same format; the first part is the command, either "publish" or "subscribe", the second part is a topic name, and the rest of the message is event data. The publisher broadcasts its message to the server without any specific-knowledge of the subscriber, and the subscriber acts upon the message (or not) based on its topic.

Overall, the architecture has 4 modules. The sensing module comprises real-time fusion data streamsand Event Detectors which can detect the events such as utensils and food placement on the chopping board. The Feature Extractor and Pattern Recognition module is the core of the framework and includes the *Feature Extractor* (FE) and the *Recognizer* components (components coloured blued in the Figure 6.7) and the pre-trained data. The GUI App module contains the graphical user interface (GUI) applications such as the sensor data visualizer and the activity and food recognition displays. Our intention is that future situated services, which utilize context recognition, will be developed in this module; the components themselves are described in Section 6.3.2.

6.3.2 Components

There are seven relevant components (blued in the Feature Extractor and Pattern Recognition module on the architecture) in KitchenSense.

Accelerometer Data Filter: this component performs the pre-processing of real-time acceleration data. The filter is activated when at least one utensil is in use, that is, the global variable *Utensil_InUse = true* passed by the Utensil Event Detector in the sensing module. The acceleration data streams are pre-processed in two steps. The first step segments acceleration data into frames of one-second duration, associating a timestamp with each frame. The choice of frame length is based on our previous results (chapters 3-5) and the observation that this would cover most common of fine-grained accitivties which typically occur in one second (i.e. one chop), and this also allows an appropriate recognition rate without unnecessary delays. Ideally, at a sampling frequency of 50Hz each frame contains 50 samples of X, Y, Z acceleration triplets. In practice, real-world factors means that some samples are lost or dropped (e.g. metallic items placed between the sensors and the receiver). Furthermore, the sensors themselves can yield noisy readings (e.g. too large or small). In such cases, a filter is applied to remove noise and fill out lost samples. Therefore, in a second step, the data filter performs both a low-pass filtering (removing abnormally low sample values) and a high-pass filtering (removing abnormally high sample values).

Finally the frame is assessed and an appropriate action taken based on the following cases:

Case 1: If the frame contains 50 samples, it is published to the pub-sub server.

Case 2: If the frame contains less than 35 samples (i.e. less that 70% of its full complement) it is discarded on the grounds that there is insufficient information to classify activities.

Case 3: If the frame contains more than or equal to 35 samples, but less than 50 samples, it is resampled using a linear interpolation method [68] to fill out the lost samples; the re-sampled filtered frame is them published to the pub-sub server.

All messages (frames) published by the *Accelerometer Data Filter* are assigned the topic "*Accelerometers*" and the data fields contain the utensil identification and time stamp information, followed by the accelerometer data.

Simple AR: is an activity recognition component that can detect movement of a utensil. Once a utensil is in use, *Simple AR* subscribes to the accelerometer data frame which has the topic *"Accelerometers"* and calculates the energy of the frames acceleration data. The calculated energy is then compared to a pre-defined threshold to determine whether the utensil has intentionally been moved. The threshold is estimated through a 4-fold cross validation procedure on the training dataset. After the frame has been classified, the activity, along with its utensil name (e.g. *ChefKnife_moving*) is published to the pub-sub server under the topic "*SimpleAR"*.

DTW AR*:* is the dynamic time warping activity recognition component as described in Chapter 4 (i.e. Real-time Activity Recognition). DTW is lightweight (i.e. O(n2) for standard DTW or O(n) for its improved version [99]) and works well with small number of training examples. The former give us a good chance for real-time implementation as multiple acceleration data streams can be classifed (one DTW classifies one data stream) concurrently, and the later allows DTW to deal with unbalanced real-world dataset as the number of data highly dependant on the users behaviours and preferences (greatly different behaviours between users and the preferences of using utensils to perform their activities in our dataset). For example, chopping patterns with the small knife are significantly different from chopping patterns with the chef-knife. Furthermore, in the future (beyond these studies and this thesis) plans included running, the classifier will directly on the sensor where computational resources much more constrained. Therefore DTW will be a good candidate. In the KitchenSense framework, the DTWAR component works as follows. Once the variable *Utensil_InUse* is set to *true*, the *DTW AR* component subscribes accelerometer data frames whose topic is *"Accelerometers"* as an observation frame of the DTW algorithm. The DTW classifier also needs training data (i.e. the *Accelerometer Templates* dataset in the Figure 6.7) to classify the activities. As real-time DTW-based activity recognition can detect both known and unknown activities, it publishes the recognized activity or unknown activity along with the utensil (e.g. *ChefKnife_chopping*, *Spatula_unknown*) to the pub-sub server under the topic "*DTWAR"*. In addition, as template adaptation is implemented in the component, adaptive templates are also updated in the *Accelerometer Templates* dataset (i.e. yellow arrow).

FCB Audio FE: is the feature extractor which extracts the features from the audio data stream of the microphone embedded inside the Fiber Chopping Board. The FCB Audio FE component is activated if there is a food placed on the FCB (i.e. the global event variable *Food_OnFCB =true*). First, the

component subscribes to the recognized activities that have the topic "*DTWAR",* and checks if the activity is *chopping* or *slicing.* If it is the case, the component segments audio data stream into one-second audio frames. Next the audio frame is segmented and then extracted to a feature vector (see session 5.6 in chapter 5). As uncompressed and compressed audio data can be extremely big and unsuitable for messaging between the publisher, subscriber and the pub-sub server, only the computed feature vector is published to the pub-sub server. The audio feature vector along with its timestamp is published by the FCB Audio FE component with the assigned topic named *AudioFeature*.

FCB Imaging FE: is the feature extractor component that extracts a feature vector from an optical fiber sensing image of the Fiber Chopping Board. Once an ingredient is placed on the FCB (i.e. *Food_OnFCB=true*), the component gets an observation image (output from the image calibration and processing procedure as described in Section 5.4). The camera runs at a frequency of 3 Hz and each image is labelled with a time stamp. The image is then segmented to extract SURF and colour features (see section 5.5). The computed features of one image are stored in a feature vector that, along with its timestamp, is then published to the pub-sub server under the topic "*ImageFeature"*.

Audio based FR: is the food recognizer based on audio features. The Audio based FR component subscribes to messages with the topic "*AudioFeature"* as audio observation data. The component also needs the pre-trained audio feature dataset for training the Gaussian Mixture Models before classifying observed food. The recognized food (possibly unknown) is then published to the pub-sub server under the topic "*AudioFood"*.

Image based FR: is the food recognizer component that uses the fiber images from the FCB. The component consists of real-time k-Nearest Neighbour and a Support Vector Machine [94] based classifiers (see Section 5.6). The component subscribes to feature vector messages with the topic "*ImageFeature"* as an observation and the *Image Feature Vectors* dataset for training models. The output of the component is a food recognition result which is published to the pub-sub server with the topic "*ImageFood*".

6.3.3 Data & Events

There are two type of data used in the recognition framework: training data which incorporated in the Training Data module (i.e. yellowed in the Figure 6.7) and real-time observation data.

Training data: Training data for the recognition components of the framework consists of three datasets: (i) accelerometer templates; (ii) audio feature vectors, and (iii) image feature vectors. The accelerometer templates dataset consists of a set of accelerometer data frames that are filtered (i.e. band-passed and re-sampled). Each frame is stored in a text file whose name includes an absolute timestamp in seconds (at collection this is synchronized with other sensors) and the utensil and activity label.
The pre-trained audio feature vectors dataset is a flat file that consists of a set of audio feature vectors. The file name identifies the utensil and the activity (e.g. *ChefKnife_chop.csv*). Each row is a feature vector which comprises of the first 13 *MFCC features*, followed by *pitch*, *energy entropy*, *zero crossing rate*, *spectral roll off*, *short time energy*, and lastly the name of the food ingredient. A feature vector in the pre-trained image feature vectors dataset is similar to the audio feature vector but the size is much larger (128 features). Examples of feature vectors can be found in the Appendix B.

Observation data: For the accelerometer, audio and image data, an observation frame has the same form as their templates for training data, but for the accelerometer data the activity label is omitted, and for the audio and image data the food is omitted (as the activity and food are what need to be recognized).

Events: Two types of events, utensil events and food events, are detected by two event detectors in the Sensing module. All utensils of interest to the context recognition framework are instrumented with WAX sensors which go into a sleep mode when idle (to save energy). In use mode (i.e. when moved) the WAX (embedded in a utensil) sends acceleration data along with a sensor identifier to a queue. Therefore, the *Utensil Event Detector* detects whether a utensil is in *use mode* or *idle mode* by simply checking (i.e. polling) whether the queue is empty. If there is at least one utensil in *use mode*, the event global variable *Utensil_InUse* is set to be *true* and the *Accelerometer data filter* is activated to fetch accelerometer data from the queue of the *Detector*.

To detect a food event, the *FCB Event Detector* performs a simple colour-histogram based procedure to detect whether there is any food placed on the FCB. The *FCB Event Detector* maintains two queues: one for audio and another for images. The detector uses a background image as the trained template and maintains its background colour histogram h_1 (sized 8x8). As the recognition framework does not know when a user has placed a food on the chopping board, the *FCB Event Detector* computes the colour histogram h_2 of each observation image, and then produces a cost using the Bhattacharyya distance between the observation image and the background image:

$$\text{cost} = \sum_{i=1}^{64} \sqrt{h_1(i) * h_2(i)} \qquad (6.1)$$

If the cost is smaller than a pre-defined threshold, then it is judged that food has been placed on the chopping board and the event global variable *Food_OnFCB* is set to *true* (otherwise *Food_OnFCB* is set to false). Simultaneously, the *FCB Imaging FE* is activated to compute features from the observation image. A pre-defined threshold is estimated from cross-validation procedure on various background images under ambient light conditions. This mechanism also improves the energy efficiency of KitchenSense. As in practice, kitchens are mostly not in use (i.e. people rarely spend more than 4 hours a day preparing food kitchen), therefore event detectors significantly reduce the amount of time that components of the recognition framework are active.

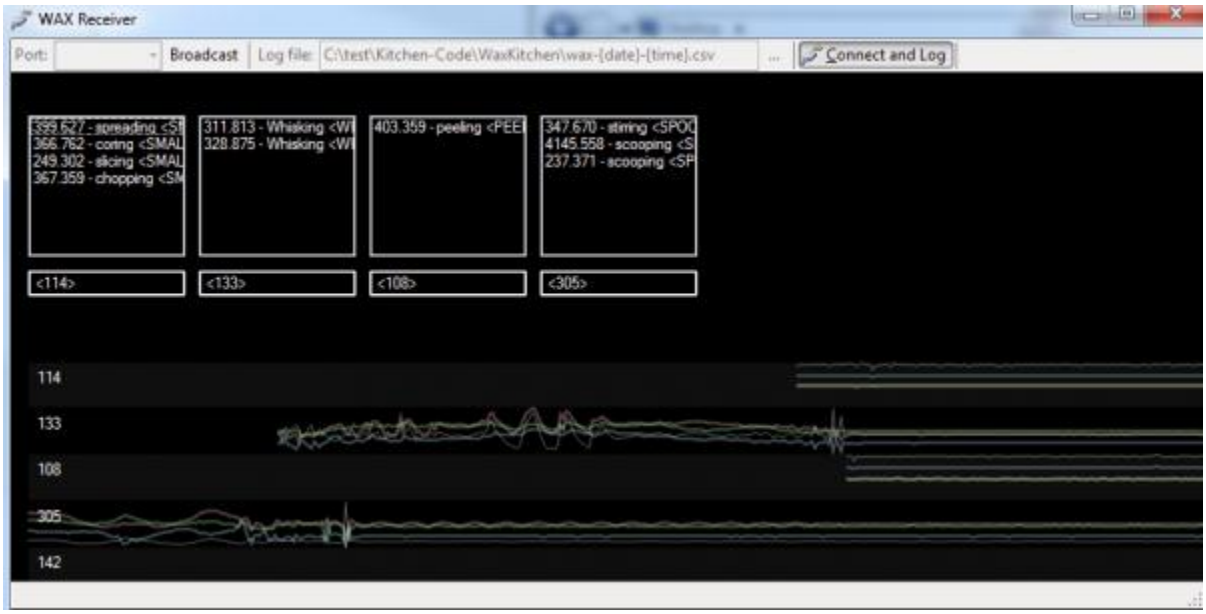


Figure 6.8: Real-time accelerometer visualizer and recognized activity display.

6.3.4 Graphical User Interfaces

A number of GUI applications have been developed to support the situated display and visualization of the outputs of the recognition components. These applications reside in the GUI App module of the architecture and all are subscribers. The *Accelerometer Data Visualizer* subscribes to acceleration data, i.e. messages with a topic "*Accelerometers",* and visualizes the data. The *Recognized Activity Display* simply subscribes the messages with the topic "*DTWAR"* or "*SimpleAR"* and displays these on the GUI. The GUI *Recognized Food Application* simply subscribes the messages which have topic "*ImageFood"* or "*AudioFood"* and displays this. For convenience, both the accelerometer visualization and the recognized activities are displayed on a single screen (see Figure 6.8) on the wall of the Ambient Kitchen.

6.4 Conclusion and discussion

KitchenSense was implemented in the Ambient Kitchen and represents a significant improvement over the initial implementation [27] replacing the web service-base software infrastructure of the previous version with a pub-sub messaging server. Ambient Kitchen 2.0 is thus a scalable and naturalistic kitchen environment designed for improving cooking skills, promoting healthier eating, and helping cognitively impaired people to live more independently in their own homes. In total the kitchen is instrumented with an embedded sensing infrastructure including RFID (not used in our study), WAX embedded utensils and kitchen objects (22 in total) and the Fiber Chopping Board. The Ambient Kitchen 2.0 also uses 4 large LCD screens hidden behind a glass façade on the wall as a situated display (replacing the original blended projector displays). We do not demonstrate any actual situated services, but in Chapter 7 only evaluate the real-time recognition framework, the environment is "application ready" as has since been demonstrated in the task-based language learning application, the French Kitchen [71].

We have presented the design of the new kitchen utensil set and our design and implementation of KitchenSense, a real-time context recognition framework in the kitchen. The integration of the miniature OpenMovement WAX sensors into modified kitchen utensils, and the design and implementation of Fiber Chopping Board, has fully realised our vision of invisible and unobtrusive context-recognition framework and environment. Scalability issues have been appropriately addressed allowing developers to readily to add or replace sensors or whole utensils, and develop new applications or situated services within the GUI App module of the application framework.

The main goal of the development of the KitchenSense was to provide the infrastructure capability for situated services and assistive applications in the kitchen. Such situated services, prompting, improving cooking competence, or providing nutrition intake advice, are very likely to rely on the automatic detection of human activity and foods in their inference mechanisms. Tracking the progression steps in a recipe, for example, requires the reliable detection of food preparation activities and food ingredients. The next step in supporting such activities is however far from trivial. Recipe tracking is likely to utilise statistical graphical models which can be trained from both daily cooking activities as well as common sense knowledge of recipes. In such a graphical model for a recipe, the steps would be nodes and the edges (conditional) transition probabilities between nodes. Search (i.e. finding a sequence of steps) on the graph could be heuristically guided by the recipe knowledge. However, the full realisation of recipe tracking is beyond the scope of our immediate concerns and in Chapter 7 we conduct a full evaluation of KitchenSense for multiple subjects preparing a meal that is considerably more complex than has been undertaken in previous research.

Chapter 7: Experiment and Evaluations

This chapter describes an experiment that is the technical culmination of our work, the performance evaluation of KitchenSense, the real-time context recognition framework in the Ambient Kitchen 2.0. A dataset was collected from 12 people cooking a *Spaghetti Röstie* using the utensils and chopping board described in the chapters 5 and 6. The large dataset collected was independently annotated by two coder and cross-checked by an inter-rater procedure for reliability. The KitchenSense framework was then rigorously evaluated under both the subject independent protocol and subject dependent protocols at both frame-by-frame and continuous event levels.

7.1 Introduction

A dataset collected under naturalistic conditions is more likely to be of value to the development and evaluation of reliable, robust machine learning algorithms for activity recognition, as it would capture much of the variety that actually arises as a result of the naturalistic performance of food preparation activities. Such an evaluation would be further enhanced if subjects that take part in such studies have different levels of skills (i.e. professionals, amateurs, and novices). The contributions of this chapter are twofold. First we describe the design and collection of a dataset involving 12 subjects, in which each subject cooked the same meal in three separate sessions in the realistic setting of the French Kitchen, our Ambient Kitchen 2.0's sibling. These subjects were given a spaghetti recipe (*Spaghetti Röstie),* the ingredients for the recipe, and the instrumented utensils and chopping board (as previously described). During the cooking session subjects performed the activities at their own pace, and in their own style, without any instructions or guidance from the experimenters. The total length of all the recorded videos was approximately 30 hours. The video was annotated by two independent coders. The annotated results were then checked using an inter-rater reliability procedure. Our second contribution is the exhaustive performance evaluation of KitchenSense, on this annotated dataset, both a frame-by-frame and an event-timing evaluation was conducted under the subject independent and subject-dependent protocols.

7.2 Dataset collection

This section describes the data collection exercise which was conducted in the French Kitchen [71], a sibling of the Ambient Kitchen 2.0 [69], but which is also equipped with a fully functional sink and cooker (both hobs and stove). As for the Ambient Kitchen 2.0 the environment had 22 WAX embedded utensils, the Fiber Chopping Board (FCB) plus a number of cameras placed in the environment (for observation only).

7.2.1 Realistic settings for the kitchen environment

The French Kitchen (see figure 7.1) is organized much like any IKEA kitchen installed in a regular house. Olive oil, salt, and soy sauce are located in an overhead cabinet. The set of knives, frying pans and the peeler are located on a shelf, while the saucepans, sieve, colander and a mixing bowl are located on a lower shelf. The cooking spoon, whisk, slotted spoon, and spatula lie in a cutlery container (a flat box) on one of the work surfaces. The Fiber Chopping Board also sits on the work surface located between the stove and sink. To facilitate capture of cooking activities (for the annotation) a number of digital cameras were installed; a map of these is presented in Figure 7.2. One digital camera (Camera 3) and a Kinect camera (the data for which was not used in our experiments) were positioned in the upper cabinet so as to capture a "bird's eye" view of food preparation activities, in particular, activities on the FCB. Camera 1 and Camera 2 were positioned above the height of a user to afford a clear view of both the work surfaces and the whole kitchen space (for example, so it can be observed how and when a subject gets and adds salt, or drains the spaghetti in the sink). Camera 4 had a dedicated view of the stove and thus preparation activities that involve the hob and the oven.

Figure 7.1: The French Kitchen, a sibling of the Ambient Kitchen 2.0.

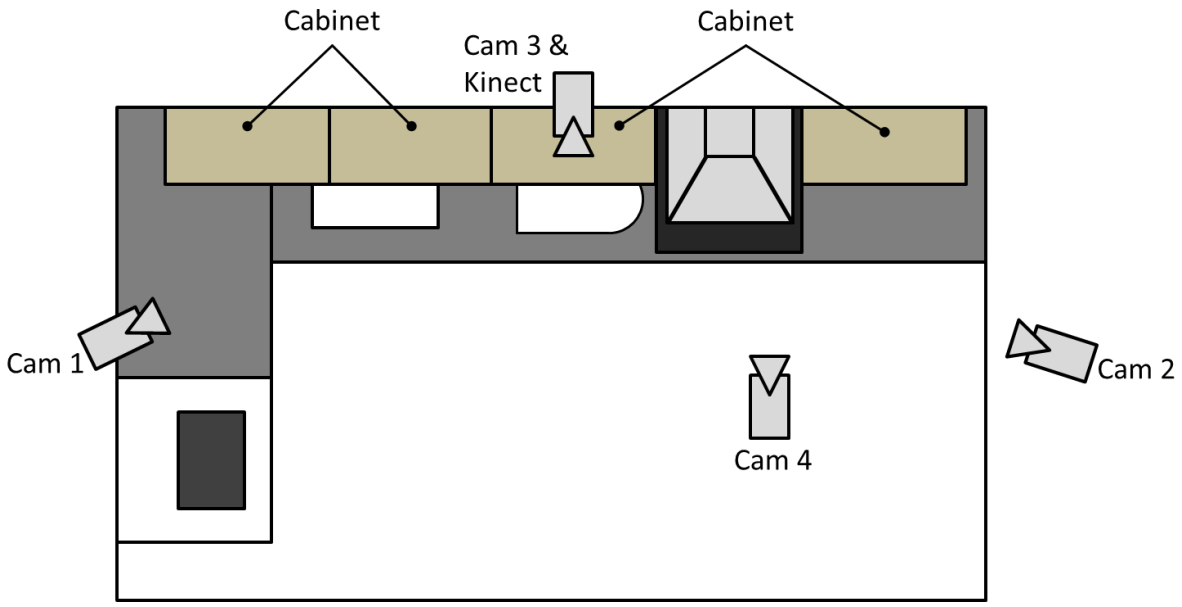


Figure 7.2: 4 Digital cameras installed in the kitchen for recording cooking videos.

The full set of utensils and the FCB used in the experiment are shown in the Figure 7.3. The set of utensils is divided into five groups. The knife group consisted of one *chef's knife*, one *slicing knife*, one *small knife* and one *bread knife*. The spoon group included one *spoon*, one *slotted spoon*, one *spatula* and one *whisk*. The pan group included 2 *saucepans*, including instrument lids, and 3 *frying pans*. The group of food containers included a *mixing bowl* (i.e. big bowl), a *colander*, and a *sieve*. A final group covers the *water faucet* (i.e. the tap) and non-fresh food ingredient containers including *salt*, *olive oil* and *soy sauce*. Examples of activities that can apply to one or more members of group are given in Table 7.1.

Figure 7.3: The utensil set and the FCB used for the experiment; closed (left) and opened (right).

The rationale and design of the utensils has been described in Chapter 6. These designs allow WAX sensors to be embedded into the handle, side handles, lid, or attached to a jar, bottle or bowl.

Group	**Utensil**	**Example activities**
Knife	*Chef's Knife*, *Slicing Knife*, *Small Knife*, *Bread Knife*	*chop*, *slice*, *scrape*
Spoon	*Spoon*, *Slotted Spoon*, *Spatula*, *Whisk*	*stir*, *scoop*, *whisk*, *tap*, *press*
Pan	2 *Saucepans*, 3 *Frying Pans*	*shake*, *move*
Container	*Big Bowl*, *Sieve*, *Colander*	*drain*, *pour*, *wash*
Other	*Tap*, *Salt*, *Olive Oil*	*in use*, *add*, *pour*

Table 7.1: Groups of utensils and example activities.

Ingredients	**Instruction steps**
Onion	1. Cook spaghetti until al dente.
Mushroom	2. Drain spaghetti in a colander and rinse it with cold water.
Courgette	3. Cut the spaghetti into 3 cm long pieces.
Spaghetti	4. Clean mushrooms wipe them dry.
Chives	5. Cut mushrooms into slices.
Basil	6. Peel courgette.
Parsley	7. Cut courgette into slices.
Ham	8. Wash and dry the chives.
Egg	9. Cut chives into small rings.
Cream	10. Peel onion and cut it into halves.
Olive oil	11. Dice one onion half.
Salt	12. Heat oil in a frying pan.
Soy sauce	13. Fry onions in frying pan.
	14. Add mushroom and courgette slices into pan.
	15. Sauté mushrooms and courgette until the liquid has evaporated.
	16. Deglaze pan content with soy sauce add cream and boil it.
	17. Add salt and pepper into the sauce and keep it warm.
	18. Before serving add chives into the sauce.
	19. Cut the second onion half into thin slices.
	20. Wash and dry basil and parsley and pluck of their leaves.
	21. Cut basil into thin strips.
	22. Chop parsley finely.
	23. Cut ham into thin strips.
	24. Beat eggs in a bowl.
	25. Add spaghetti pieces, onion slices, parsley, ham and eggs into a bowl and mix them together.
	26. Heat oil in frying pan and fry small portions of the spaghetti mix until golden brown (Rösties).
	27. Serve Rösties with mushroom sauce.

Table 7.2: The Spaghetti Röstie recipe.

7.2.2 Recipe selection

The *Spaghetti Röstie* recipe was chosen on the grounds that it involves 13 different food ingredients, of which 8 ingredients can be prepared on the chopping board if the subject so chooses (i.e. as a recipe it served as a good candidate for the evaluation of the performance of the food recognition algorithms with the FCB). The ingredients cover a wide range of common recipe ingredients such as *onions*, *mushrooms*, *ham*, *eggs*, *soy sauce* and *olive oil* etc. Moreover, the recipe has 27 instruction steps, most of which could be performed with one or more utensils of our utensil set (Table 7.2) and cover most common kitchen activities, including chopping, slicing, scooping, adding ingredients, and peeling etc.

7.2.3 Data collection procedure

12 subjects were recruited through an email advertisement to staff and students at Newcastle University. Overall each subject prepared the Spaghetti Röstie recipe three separate times (3 sessions). To start, a subject was introduced to the methods and procedure of the experiment, and signed both ethics and consent forms. The subject was then given the recipe. An important element of our experimental design was that we wanted subjects to have a good understanding of all the steps of the recipe prior to them preparing the dish. This was with a view to avoiding the situation whereby they slavishly return to the written recipe again and again while cooking (i.e. we aimed to create a more cooking context). The subjects learnt about the recipe through a specially designed board game played with food and activity cards, in which players used the cards and the board to perform simulated cooking activities and manipulate the locations of the ingredients, utensils, and food containers. Thus the game allowed subjects to practice the meal preparation (with the recipe) before embarking on any actual cooking. Only when a subject was confident as to the steps required in the preparation of the meal were they introduced to the kitchen, that is, where to find the ingredients and utensils and how the hob and oven work.

A cooking session started with a synchronization procedure. This was carried out by the experimenter who in plain view of a camera distinctively hit (i.e. for acceleration data synchronization) the small knife on a kitchen surface 5 times, thereby making distinct noises (i.e. for audio synchronization). After the synchronization step, the subject was left alone to prepare the meal. During the meal preparation, no time-constraints and no instructions were provided to the subject. After finishing the cooking session, the synchronization procedure was performed again. In between cooking sessions, a questionnaire was administered to the subject. The questionnaire addressed issues of meal preparation routines and whether, in general, a subject was aware the technologies inside the kitchen and within the utensils. On completion of all three cooking sessions subjects were paid £20 to compensate them for their time.

All data were recorded on a server located behind the wall of the kitchen. The accelerometer data for the WAX sensors were written into one logging file. Each sample was written with its timestamp. For the Fiber Chopping Board, the embedded camera recorded three images per second and the image files were named with timestamps (i.e. capture times). Audio was recorded when the session started at the original frequency of 44,100 Hz which was then down-sampled to 8000 samples per second using the Audacity software tool, as for feature extraction previous studies [15, 24, 64, 65, 66] have shown that an audio sampling rate between 2K and 12K sufficient for food classification (and event detection).

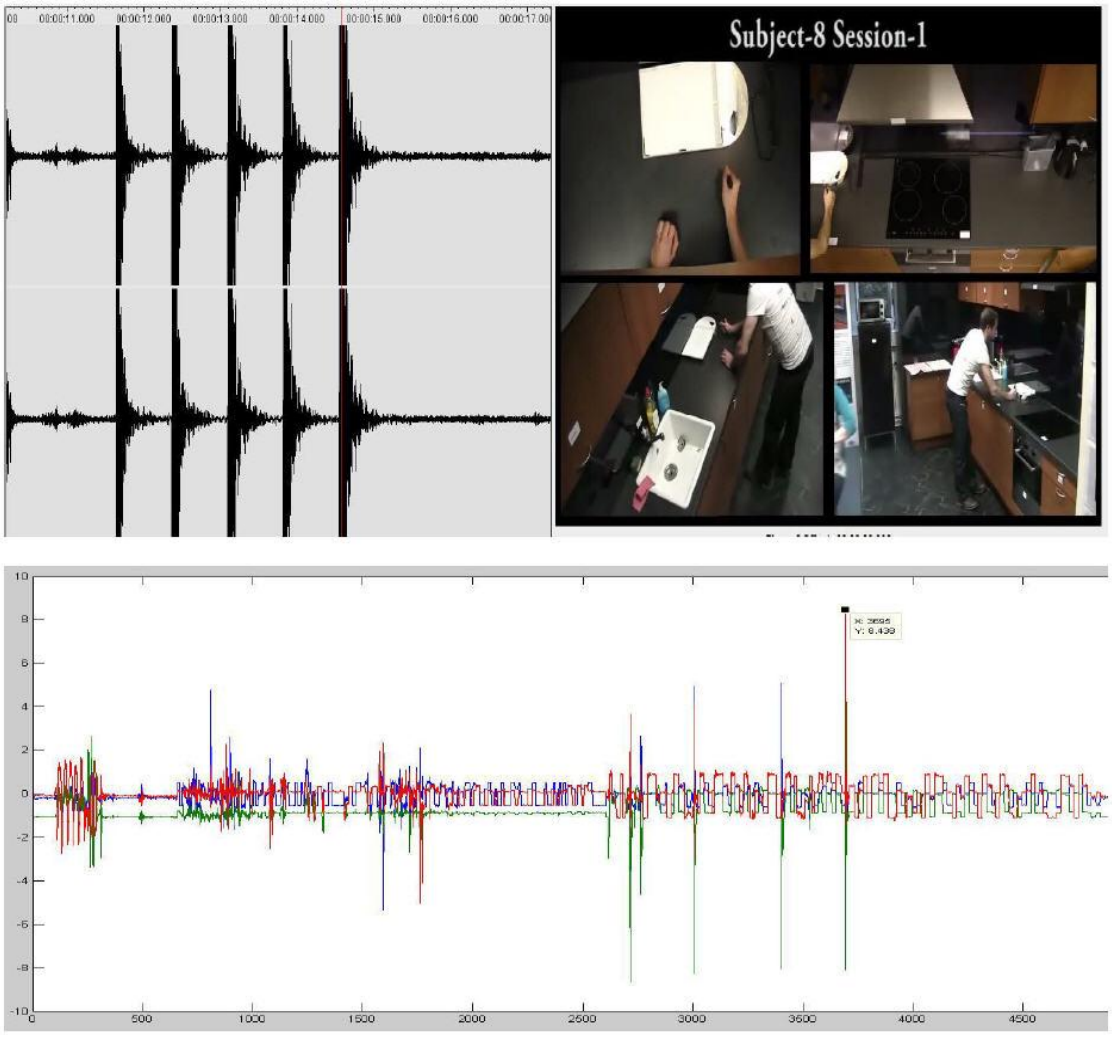


Figure 7.4: An example of data synchronization.

7.2.4 Data synchronization

Once collected (in log files) the sensor data needed to be synchronized with the observational videos. To synchronize the audio with the videos, we used the media synchronization function of the ELAN (EUDICO Linguistic Annotator) toolkit [73]. Here the audio data is first visualised in ELAN such that the 5 audio peaks produced by the synchronization procedure at the beginning of the session are clearly visible. The fifth peak is marked as the *offset time*, which we took to be the absolute start time. Then, audio data from the beginning of the log file to the *offset time* was removed. Similarly, the acceleration data was visualized and the fifth peak of 5 continuous peaks was marked as the offset and the corresponding row number in the acceleration data (and its timestamp) was identified. Data, from the beginning to the offset time, was then removed (see figure 7.4). The synchronization of collected food images of the FCB to the videos was a more straightforward process as each image file had its own name that includes the timestamp of the recording.

7.3 Annotation & inter-rater reliability

Annotating the data is the first step in the data analysis, that is, the generation of a ground truth for evaluating the performance of the recognition algorithms. Also, the annotated data can be segmented for training supervised machine learning algorithms. In our previous study we demonstrated that the consistency of the data was significantly improved where the annotation was repeated by more than one coder. Hence, two coders were asked to independently annotate the dataset. The annotated data was then crosschecked using an agreement inter-rater reliability scheme.

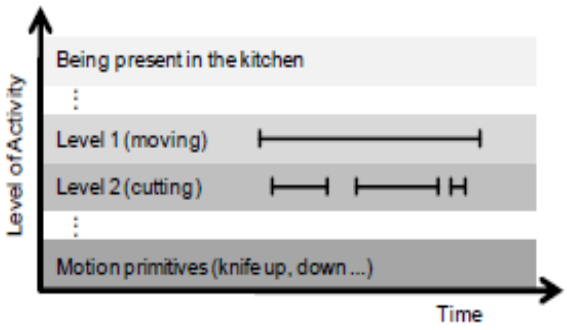


Figure 7.5: Hierarchical annotations.

7.3.1 Annotation protocol

The ELAN annotation tool [73] was used to annotate the dataset. ELAN is particularly useful in that it supports the sensor data visualization and hierarchical annotations (see Figure 7.5). Visualization is valuable for visually identifying sensor errors and the alignment between sensor data and videos data (as already described). Hierarchical annotation allows better support for annotations allowing the association

of specific activities to specific utensils (ultimately sensors). In our annotation schema each utensil has two associated annotation levels (referred to as *tiers*). One tier relates to the overall movement of the utensil. The second corresponds to activity annotations made by the coder (annotator) such as *chopping*, *slicing* etc. As 22 sensors are used in this study, approximately 40 tiers are displayed in an annotation window. With two-tier annotations for each utensil, common errors such as the use of the incorrect tier (i.e. a coder annotating the wrong utensil with an activity) can be minimized.

An annotation document was prepared for the coders. The document contained instructions as to how to use ELAN, definitions of activity labels, and detailed examples describing how to make an annotation for each activity. In addition, sample videos were made for some activities that were potentially ambiguous to coders such as the difference between *chopping* and *slicing*. The procedure of annotating a food ingredient processed on the FCB was rather different. Cases where food were placed on the FCB varied from the simple (and most common) case of one single item, to multiple food items being placed on the FCB. The case where a ingredient was placed on the FCB, but had not been acted upon was straightforward, but when it was being chopped, knife activities had to be annotated with annotation labels that combined the knife activities and ingredient labels (e.g. *chop,basil*). This schema significantly reduced the redundancies of repeated annotations that we required in pilot annotation exercises that we conducted.

Coders were recruited from Newcastle University via a job advertisement on the University's vacancies website. The requirements for coders were that they had a good understanding of English, basic computer-skills, and were not one of our research team members or a person associated with the research project in any way. The coders were paid £10.00 per hour. All coders were trained and practiced ELAN using the annotation scheme instruction document. The coders were asked to make sure they were confident that they understood all label definitions in the document, and were asked to pay careful attention to the sample videos. All of the collected videos were independently annotated by two coders.

7.3.2 Activity labels

16 activity labels are defined for coders to use when annotating the video. In order to avoid unnecessary ambiguity, a set of utensils to which these activity annotations generally applied were also specified (see Table B.1 in Appendix B). For example, *scooping* generally only applied to *spoon*, *slotted spoon*, and *spatula*, while *draining* generally only applies to *sieve* or *colander*. Some activities, such as *add* and *pour* are distinct to a set of ingredient. For example, *pour* generally only applies to liquid ingredients such as *soy sauce* or *olive oil* and containers, while *add* applies to "solid" ingredients such as *salt*. In practice

these definition of activity-object relations form part of our working of our definition for an activity. Table 7.3 lists the activity labels along with utensils to which they apply.

As described, annotation document provided to coders includes: (i) an activity label has a definition that elaborates how the activity is performed with a utensil; (ii) a how-to-annotate note using ELAN; and (iii) an example of the label. The *scooping* label, for example, is defined as:

> "*scooping is a move a food or mix from a container/pan/plate to another container/pan/plate using the spoon, the slotted spoon, or the spatula. Annotation begins with the spoon enters the container and ends with the food portion got out of the spoon".*

Definitions of most activities are straightforward, although a small number of activities such as *chopping* and *slicing* need more refined definitions as we found that in pilot annotation sessions they were often confused with each other. For example, the definition of *chopping* provided was as follows:

> "*chopping is a distinct cutting activity where the knife moves up and down causing it to make contact with the chopping board and then break contact with the chopping board. Some minor forward and backward movements may be seen but these do not significantly cut the ingredient*."

In contrast, *slicing* is defined as:

> "*slicing is any cutting activity* [excludes peeling] *with the knife that is not chopping. If the knife remains in contact with the board or if the predominant movement is forwards and backwards then this cutting activity is not chopping*."

In such cases, in addition to definitions, we also made example videos for coders. The labels for food ingredients were taken from the recipe and are well defined. All are elaborated in the annotation document. There is one special activity, *unknown*, which was referred as a *baggage* or in other words a *null* activity [17, 72] and this label was applied to any other movements of the utensil which are not defined in the activity list. Note that coders did not need to annotate unknown activities that had been automatically subtracted from the moving data.

7.3.3 Inter-rater reliability

Our previous studies showed the consistency of the annotated data could be significantly improved if an activity label assigned to a data segment was agreed upon by more than one coder [16]. Table 7.3 shows the agreement rate for each label assigned by two coders. The agreement rate procedure first takes two sequences annotated by two coders and segments these into one-second frames. Each frame is associated with its label and an absolute time. A longest common subsequence (LCS) based algorithm [74] was used

to compute agreement rates of the frames agreed by both coders. LCS is a dynamic programming algorithm which solves the problem of finding a longest common subsequence from two sequences by dividing the sequence into subsequences until they become as simple as possible (i.e. a subsequence contains one single frame only), and then the solution is constructed starting with a comparison between the two simplest subsequences. However, unlike the standard LCS, which only considers the element values (often one simplest subsequence contains one element), the LCS used in our agreement rate procedure used both the label and the absolute time associated with each frame.

Activity & food labels	**Utensils**	**Agreement (%)**
chop,cour	chef knife, slicing knife, small knife	49.3
chop,mush		62.1
chop,chiv		37.59
chop,onio		68.51
chop,basil		32.69
chop,spag		70.65
chop,ham		49.82
slice,cour		67.11
slice,chiv		49.23
slice,pars		25.72
slice,onio		62.65
slice,spag		73.22
slice,ham		72.4
slice,basil		31.74
slice,mush		62.35
scrap		93.27
pour	soy sauce, olive oil, bowl	91.2
peel	Peeler	93.18
add	Salt	89.63
drain	sieve, colander	82.14
wash	sieve, colander, bowl	79.25
stir	spoon, slotted spoon, spatula	84.67
scoop		80.61
turn		81.38
press		73.72
tap		69.82
shake	frying pans	79.65
in use	tap water	96.58
cour	fiber chopping board	92.46
mush		82.62
chiv		74.36
onio		89.95
basi		62.38
pars		61.41
spag		95.63
ham		90.24
Overall		**73.69**

Table 7.3: Agreement inter-rater reliability (%).

Figure 7.6 is an example of agreement between two sequences labeled by two coders in 8 seconds. The upper sequence labeled by coder 1 contains 6 frames with one slice followed by 5 chops and the lower sequence labeled by coder 2 contains 7 frames with one slice followed by 5 chops, then the other slice.

Although both sequences have 5 common chops, only 4 of them are overlapped (i.e. each pair has the same label and absolute time, blued in the figure 7.6). Therefore the agreement rate is 4/8 = 50%.

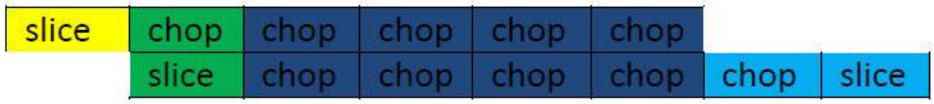


Figure 7.6: Agreement example of two coders.

In Table 7.3 the overall agreement rate is 73.69%. Despite our best efforts to improve the training material for coders, low agreement was still apparent for *chopping* and *slicing* activities, particularly for *chopping*/*slicing parsley* and *basil*. This was not only because of the apparently similar appearance of the *chop* and *slice* action, but because *parsley* and *basil* look similar when being *chopped* and *sliced* in the videos. The *scraping* activity was found to be very distinct from other knife activities; hence agreement was 93% between both coders. In general, agreement rates for food labels were high (except for parsley and basil which was 60%). Annotations for activities performed with other utensils also exhibited agreement rates between 70-90%. In fact, the source of disagreements in the use of these labels was generally due to insertion, deletion, substitution, overfill or underfill issues within subsequences (which is described in more detail in our event-timing analysis section).

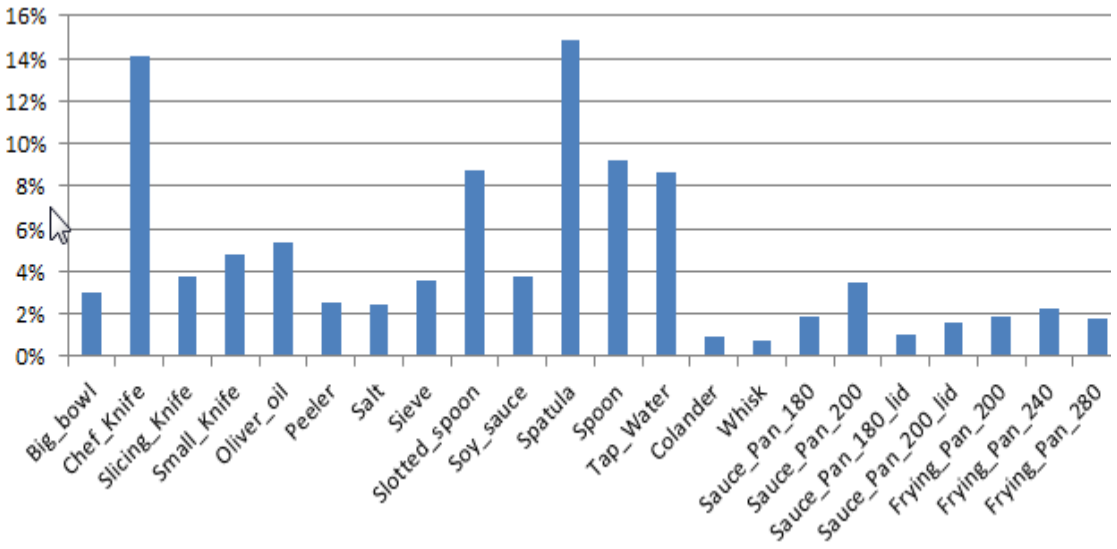


Figure 7.7: The distribution of the utensil usage.

7.3.4 Distribution of data and prior-probabilities

7.3.4.1 Distribution of the utensil usage

The distribution of utensil usage is presented in the Figure 7.7. In the collected dataset, *chef knife* and *spatula* dominate at around 14% of usage time, while the *whisk* and *colander* are below 1%. From our observations of the collected videos, the *sieve* was preferred over the colander for draining the cooked *spaghetti*. Also, the subjects had a tendency to use the *spoon* rather than *whisking* with the *whisk*.

7.3.4.2 Distribution of the chop and slice food ingredients

Table 7.4 presents results for the use of *chef knife*, *slicing knife* and *small knife* for *chopping* and *slicing*. As frame sizes are of one-second duration, the figures correspond to both the number of frames and seconds.

Food	**Duration for food on the FCB (s)**	***Chef knife***		***Slicing knife***		***Small knife***	
		chop	*slice*	*chop*	*slice*	*chop*	*slice*
Basil	637	279	207	0	97	0	29
Chive	1125	267	237	41	0	88	94
courgette	260	876	47	145	0	73	358
Ham	527	167	197	0	25	0	144
Mush	821	1363	208	77	0	89	341
Onion	409	1491	566	169	28	0	462
Parsley	744	0	149	0	15	0	67
spaghetti	220	74	124	9	0	0	28
Total (frames)	**4743**	**4517**	**1527**	**400**	**174**	**250**	**1429**

Table 7.4: The distribution of knife use for chopping and slicing (in frames).

The *chef knife* was used most at 6044 seconds (72.85%) of which 4517 seconds was spent *chopping* and 1527 seconds was spent *slicing*; followed by the *small knife* (20.24%) of which 250 seconds was spent *chopping* and 1429 seconds was spent *slicing*. It is observed that subjects tended to *chop* with *chef-knife* or *slicing knife*, and *slice* with the *small knife*. For the food ingredients, *mushrooms*, *onion* and *courgette* were mostly *chopped* while subjects preferred to *slice* parsley and spaghetti.

7.3.4.3 Distribution of the known vs. unknown activities

Known activities are the activities with pre-defined labels in the annotation description document. Any other activities are *unknown*. It was interesting to observe the proportions of each in our dataset. In Figure 7.8 (left) we can see known activity events correspond to 54% of annotated activities, but in terms of time (in frames) make up just 45%. In general, unknown activities made up around half the cooking time. Reducing this time by improving cooking skills based on kitchen context recognition would be an open

challenge for further application development, for example, developing situated advice applications that improve cooking competence [28] including cooking plan optimization.

Figure 7.8: The distribution of known and unknown activities, events (left) & frames (right) distribution.

7.3.4.4 The prior-probabilities of activity events

The prior-probabilities of activity events given utensil in use events in the dataset are present in the table 7.4a. Utensil in use events are detected when the utensils are moving (i.e. trigged by the sensor signals). Therefore, moving activity events are 100% detection for Sauce pans and their lids. Prior-probabilities of Tap water in use and unknown activity events are generally higher than thoese of other activity events. The overall of prior-probability distribution of known and unknown activity events is shown in figure 7.8 (left).

7.4 Evaluation methods and performance metrics

This section describes our evaluation of KitchenSense, the activity and food recognition framework introduced in Chapter 6. The frame-by-frame based analysis adopted in chapters 3 and 4 demonstrated the feasibility of pervasive sensing technologies for kitchen context recognition from a real-time data stream. However, because of variability in the duration of food preparation activities, for example, that chopping might take a few seconds while washing a food might take 10 minutes, we need to add to our frame-by-frame approach to the context recognition problem in the kitchen with an analysis of event duration estimation (which we refer to as event-timing). This we believe is useful not only for situated service provision, but also for time and resource optimisation. By event, we mean a set of continuous frames of variable duration (with a start and end time).

In addition, in order to correctly make comparisons between ground truth data and predicted events, we adopt the notation of a segment from [33, 72]. A segment is a variable-duration sequence of continuous frames during which the label of both the recognition system and the ground truth does not change. A segment is partitioned from a continuous data stream and can be a part of, or the whole of, an event. The

boundaries of a segment are determined by a change of either the ground truth or the predicted event (produced by a recogniser). For example, there are 4 segments differently coloured in Figure 7.6, the first segment contains the first frame labelled slice by a recogniser. The second segment contains one frame marked slice in ground truth annotation and chop by a recogniser. The third segment is the next 5 consecutive frames labelled as chop in both the predicted sequence and ground truth sequence, and the fourth segment contains last two frames of the ground truth data.

activity event | utenstil in use event	**Prior-probability**	**activity event | utenstil in use event**	**Prior-probability**
Pr(pour|Big_Bowl)	0.05	*Pr(move|SaucePan 180_lid)*	1
Pr(wash|Big_Bowl)	0.1	*Pr(drain|Sieve)*	0.06
Pr(unknown|Big_Bowl)	0.88	*Pr(wash|Sieve)*	0.01
Pr(chop|Chef_knife)	0.29	*Pr(unknown|Sieve)*	0.92
Pr(slice|Chef_knife)	0.16	*Pr(chop|Slicing_knife)*	0.21
Pr(scrap|Chef_knife)	0.04	*Pr(slice| Slicing _knife)*	0.09
Pr(unknown|Chef_knife)	0.51	*Pr(scrap| Slicing _knife)*	0.04
Pr(basil|Chopping board)	0.04	*Pr(unknown| Slicing _knife)*	0.66
Pr(chive|Chopping board)	0.06	*Pr(stir | Slotted Spoon)*	0.28
Pr(courgette|Chop. board)	0.06	*Pr(scoop| Slotted Spoon)*	0.09
Pr(ham|Chopping board)	0.06	*Pr(tap| Slotted Spoon)*	0.03
Pr(mushroom|Chop. board)	0.15	*Pr(press | Slotted Spoon)*	0.06
Pr(onion|Chopping board)	0.08	*Pr(unknown| Slotted Spoon)*	0.53
Pr(parsley|Chop. board)	0.04	*Pr(chop|Small_knife)*	0.11
Pr(spaghetti|Chop. board)	0.06	*Pr(slice| Small _knife)*	0.46
Pr(unknown|Chop. board)	0.45	*Pr(scrap| Small _knife)*	0.14
Pr(drain| Colander)	0.13	*Pr(unknown| Small _knife)*	0.29
Pr(wash| Colander)	0.02	*Pr(pour|Soy sauce)*	0.14
Pr(unknown| Colander)	0.89	*Pr(unknown|Soy sauce)*	0.86
Pr(shake| Fry. Pan 200)	0.24	*Pr(stir| Spatula)*	0.24
Pr(unknown| Fry. Pan 200)	0.85	*Pr(scoop| Spatula)*	0.04
Pr(shake| Fry. Pan 240)	0.37	*Pr(turn| Spatula)*	0.12
Pr(unknown|Fry. Pan 240)	0.63	*Pr(tap| Spatula)*	0.02
Pr(shake| Fry. Pan 280)	0.4	*Pr(press| Spatula)*	0.03
Pr(unknown|Fry. Pan 280)	0.6	*Pr(unknown| Spatula)*	0.55
Pr(pour| Olive Oil)	0.18	*Pr(stir| Spoon)*	0.37
Pr(unknown| Olive Oil)	0.82	*Pr(scoop| Spoon)*	0.15
Pr(peel| Peeler)	0.14	*Pr(tap| Spoon)*	0.06
Pr(unknown| Peeler)	0.86	*Pr(unknown| Spoon)*	0.42
Pr(add| Salt)	0.15	*Pr(in_use | Tap Water)*	0.79
Pr(unknown| Salt)	0.85	*Pr(unknown |Tap Water)*	0.21
Pr(move|Sauce Pan 180)	1	*Pr(whisk| Whisk)*	0.15
Pr(move|SaucePan 180_lid)	1	*Pr(tap| Whisk)*	0.02
Pr(move|Sauce Pan 180)	1	*Pr(unknown| Whisk)*	0.84

Table 7.4a: Prior-probabilities of activity events given utensil in use events

7.4.1 Evaluation methods

We adopt both *subject independent* and *dependent* protocols for evaluating KitchenSense. In the subject independent protocol, the subject to test is not included into the training data. Although subject independent evaluation is generally more punitive, it should lead the development of AR systems that are more feasible to deploy. In the dependent evaluation, a 10 fold cross-validation method is used, that is, the data are partitioned into 10 parts in which 9 parts are used for training and one is used for testing, then the process is repeated. Thus, the subject dependent protocol does not partition the training and test data by subject (unlike in the independent case). In practice, such an evaluation is meaningful for applications where individual differences between subjects are unlikely to have an impact on performance.

Frame-by-frame analysis is a fundamental approach to the evaluation of AR from real-time data streams, but frame-by-frame performance is not wholly sufficient as it does not quantify event errors shown in the figure 7.9 which may have serious implications for applications that incorporate functionality such as recipe step tracking. Therefore, in this chapter, in addition to standard frame-by-frame analysis, we included a performance analysis at event level (based on frames, i.e. time), which we refer to as event-timing analysis.

As we wanted to evaluate food recognition performance of the FCB with images and audio, two separate evaluations for the FCB were conducted: (i) food recognition using fiber sensing images when food is placed on the FCB, but before it is *chopped* or *sliced*, and (ii) food recognition based on sounds produced when the food is being *chopped* or *sliced*. The ground truth for the former was drawn from the set of images between the food being placed on the FCB and the start time of a *chopping* or *slicing* event, for which there was considerable agreement between coders. The ground truth for the latter constructed from *chop* and *slice* activities of knives.

7.4.2 Performance metrics

Events in the kitchen are important for both context recognition and situated services, and these are detected by our recognition framework. We also want to distinguish between a long (and important) event, such as chopping a food, which might often occur over many seconds or even minutes, and a minor event such as putting a knife aside which can occur within 1 second. Therefore we needed to report not only the number of event errors, but also the significance (i.e. in term of frames or seconds) of such event errors. Consequently, we measure the errors of the events and their lengths (hence our use of the term event-timing analysis). Standard performance metrics such as *precision* (i.e. true positives/(true positives + false positives)), *recall* (i.e. true positives/(true positives + false negatives)), and *false positives* are used for measuring the recognition performance for frame-by-frame analysis. Newer metrics including

insertion, *deletion*, *overfill*, *underfill* and *substitution* [33, 72] are be used for measuring the combination of event-timing matches and errors, and they are defined as follows.

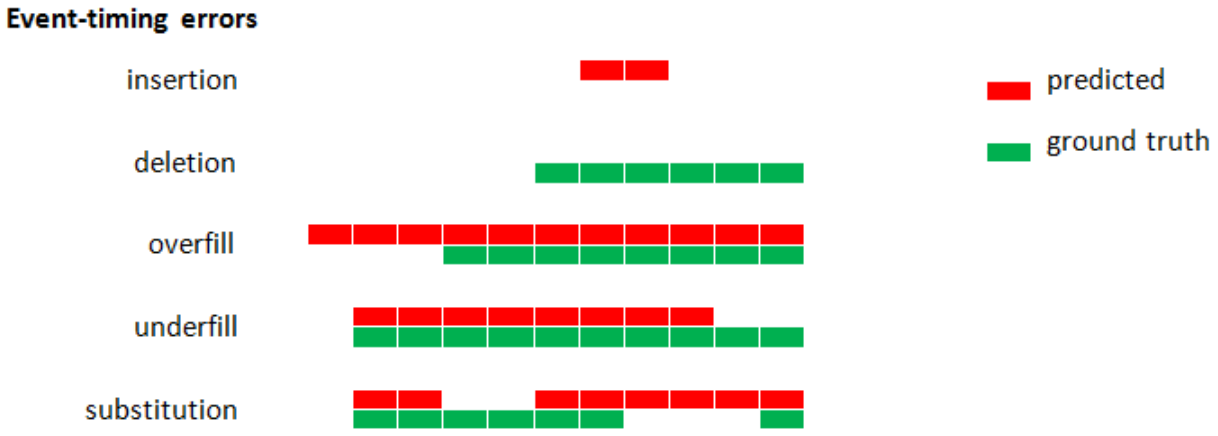


Figure 7.9: Error examples for our event-timing analysis.

- *Insertion*: a predicted event has no matched frame with any ground truth event.
- *Deletion:* a ground truth event has no matched frame with any predicted event.
- *Underfill:* at least one frame in the ground truth event is not covered by its predicted event.
- *Overfill:* at least one frame in the predicted event is not covered by its ground truth event.
- *Substitution:* incorrectly predicted (i.e. *chopping* incorrectly classified as *unknown*).

The illustration of each of the five metrics is shown in Figure 7.9.

7.5 Frame-by-frame based analysis

A *frame* is a fixed-length, one second, of data (i.e. 50 samples). The selection of this frame length can be justified on the basis that: (i) it covers most of activities in the kitchen (i.e. most fine-grained activities that we are interested in have a duration of more than one second); (ii) a latency of much more than one second would have serious implications for the timeliness of feedback in real-time applications. Moreover, our previous experiments demonstrated that an appropriate level of accuracy could be achieved with a frame length of around one second. Under the subject independent protocol, 11 subjects were trained, and one subject was left out, to test, and the process was repeated for all other subjects and the results aggregated. The subject dependent protocol uses all 12 subjects for training data, which are partitioned into 10 subsets. Each subset is tested using the other 9 subsets as training data. Again the process was repeated for other subjects and the results are aggregated.

7.5.1 Accelerometer-based Activity Recognition results

Figure 7.10 shows overall performance results of subject independent and dependent evaluations. These results include unknown activities. As anticipated, the protocol results are higher than subject independent results, with 92% vs. 87% for precision, and 84% vs. 79% recall, respectively. The number of false positives, however, was lower for the subject independent protocol.

Precision and recall rates were 87.6% and 79.3% respectively for the subject independent protocol which can generally be considered high for a recognition system which is evaluated on a relatively large-scaled dataset comprising 59 activities, and where the dataset is significantly imbalanced (see the number of frames for each activity in the third column on table 7.5).

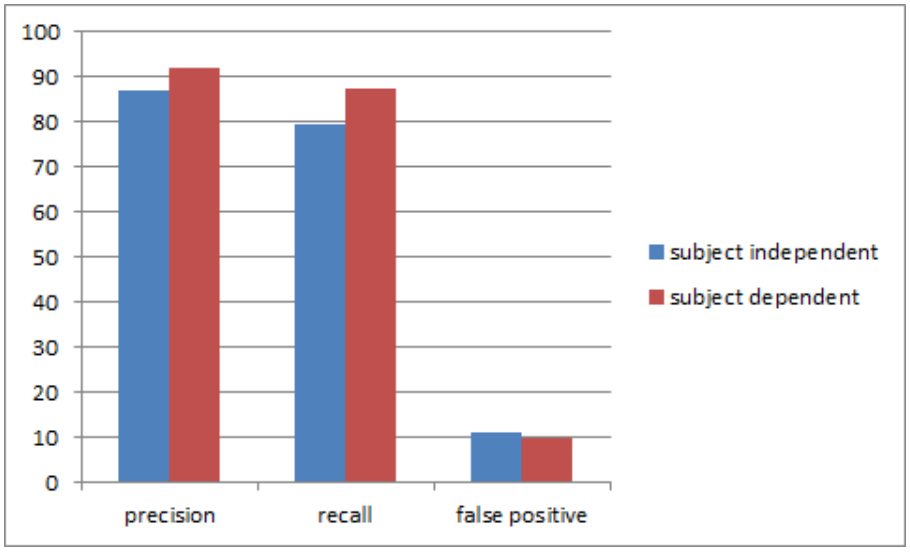


Figure 7.10: Frame-by-frame activity recognition results.

The false positive rate, which reports the percentage of incorrect recognition events for the recognition framework, was 11.22% for the subject independent protocol and approximately 10% for the subject dependent protocol. These are acceptable and can be explained by the fact that there is generally 3 or more classes of activity that needs to be classified per utensil. Table 7.5 shows the aggregated subject independent protocol results in detail, these indicate that there are a significant number of misclassifications between *known* and *unknown* activities. The activities *chop* and *slice*, have a high misclassification rate (i.e. 20%) and are often confused. As already described this is most likely due the fact that their movement patterns are quite similar. Unsurprisingly, the *saucepans* have low false positive rates (less than 1%) and high correct recognition rates (more than 90%) since only one class of activity applies to them. In practice, the activities associated with *saucepans* are detected by the simple activity recognition (SimpleAR) component which detects their movement using a pre-defined threshold.

Utensil	Activity	Ground truth (frames)	Precision (%)	Recall (%)	False Pos (%)
Bowl	Pour	56	61.82	60.71	37.50
	Wash	67	55.77	43.28	34.33
	Unknown	2,378	94.25	83.43	5.09
Chef_Knife	Chop	4,159	85.77	83.89	13.92
	Slice	1,340	69.15	66.57	29.7
	Scrap	302	94.18	91.06	5.63
	Unknown	5,894	83.58	83.07	16.32
Slicing_Knife	Chop	879	69.63	59.73	26.05
	Slice	499	76.17	87.78	27.45
	Scrap	40	92.86	97.50	7.50
	Unknown	1,708	84.60	75.23	13.7
Small_Knife	Chop	258	67.44	78.68	37.98
	Slice	1,916	87.56	59.13	8.40
	Scrap	37	65.38	45.95	24.32
	Unknown	1,751	82.11	79.44	17.30
Oliver_oil	Pour	346	92.28	89.88	7.51
	Unknown	4,074	96.56	83.33	2.97
Peeler	Peel	1,014	95.90	90.04	3.85
	Unknown	1,109	87.93	86.74	11.90
Salt	Add	343	91.45	81.05	7.58
	Unknown	1,673	85.90	88.11	14.47
Sieve	Drain	1,453	52.93	31.04	27.60
	Wash	87	71.84	85.06	33.33
	Unknown	1,386	92.89	83.84	6.42
Slotted_spoon	Stir	3,547	90.63	75.56	7.81
	Scoop	148	94.56	93.92	5.41
	Tap	83	88.46	83.13	10.84
	Press	145	77.17	48.97	14.48
	Unknown	3,325	75.93	87.55	27.76
Soy_sauce	Pour	318	92.46	88.68	7.23
	Unknown	2,812	92.72	91.47	7.18
Spatula	Stir	4,369	93.49	91.74	6.39
	Scoop	293	96.50	84.64	3.07
	Turn	686	84.97	66.76	11.81
	Tap	81	91.55	80.25	7.41
	Press	246	69.11	57.72	25.2
	Unknown	6,638	87.66	89.00	12.53
Spoon	Stir	3,059	98.90	91.11	1.01
	Scoop	352	89.46	84.38	9.94
	Tap	193	89.82	77.72	8.81
	Unknown	4,060	83.83	89.63	17.29
Tap_Water	in_use	3,276	81.52	44.29	10.04
	Unknown	3,872	80.69	33.68	8.06
Colander	Drain	275	42.50	24.73	33.45
	Wash	5	0	0	0
	Unknown	466	80.43	72.32	17.60
Whisk	Whisk	295	97.37	75.25	2.03
	Tap	9	87.50	77.78	11.11
	Unknown	306	92.93	90.20	6.86
Sauce_Pan_180	Move	859	99.10	91.92	0.83
Sauce_Pan_200	Move	1,743	95.75	91.14	4.05
Sauce_Pan_180_lid	Move	211	99.86	89.77	0.12
Sauce_Pan_200_lid	Move	505	99.83	91.95	0.15
Frying_Pan_200	Shake	296	89.79	75.89	8.63
	Unknown	1,251	86.85	82.33	12.47
Frying_Pan_240	Shake	208	82.16	72.04	15.64
	Unknown	1,652	91.88	86.99	7.69
Frying_Pan_280	Shake	188	77.40	72.87	21.28
	Unknown	1,292	96.40	84.91	3.17
Total		**83,076**	**87.63**	**79.40**	**11.22**

Table 7.5: Frame-by-frame performance for the subject independent protocol.

In Table 7.5, one activity, *washing* with the colander, has only 5 frames. This is the only clear case where our frame length is insufficient, that is, the duration of the activity too short and thus the recognition system fails to classify it. High recognition rates (80-90%) are achieved for *scooping* and *stirring* of the *spatula*, *spoon* and *slotted spoon*. These results are consistent with our previous studies (Chapter 3 and 4). Other activities such *pour* (*olive oil*), *add* (*salt*), *peel* (*peeler*) similarly have the expected high accuracy (85-90%) since just two classes of activity need to be recognised per utensil. *Drain* and *wash* activities of the *sieve*, *bowl* and *colander* have low recognition rates and high false positive rates probably based on a lack of distinctiveness between their movement patterns.

In addition to an evaluation on the open dataset, in which unknown activities are included, we also carried out a simple evaluation on the closed dataset, which excludes the unknown activities – this gives us an indication of how unknown activities impact on performance. The closed dataset contains 41 activities and is 50% smaller than the open dataset. Overall, the precision, recall and false positives for the closed dataset are 88.52%, 76.3%, and 9.93%, respectively, showing that the performance of the recognition framework is not significantly effected by the inclusion or exclusion of unknown events.

7.5.2 Image-based food recognition results

The image dataset consists of 14,229 images. As a result of the fact that the collection was under more realistic settings (than our pilot study in Chapter 5), a considerable number of noisy images are included into the dataset. Indeed compared to the collected images used for our pilot experiment in the Chapter 5, for which we had more control over the level of ambient light and the behaviour of the subjects, this dataset is significantly more challenging and includes more instance of occlusion, more variability in image quality and generally more imbalanced. The k-NN's results for food recognition based on image classification are shown in Figure 7.11. Precision, recall and false positive rates are 81.5%, 70%, 18% for the subject independent protocol, and 89%, 74%, 11% for the subject dependent protocol, respectively. Although the false positive rate is relatively high, the overall results are promising, especially given the relative complexity of the dataset (8 food ingredients) and that a number of them, such as parsley and basil, have a very similar visual appearance (and were often confused with each other by our coders).

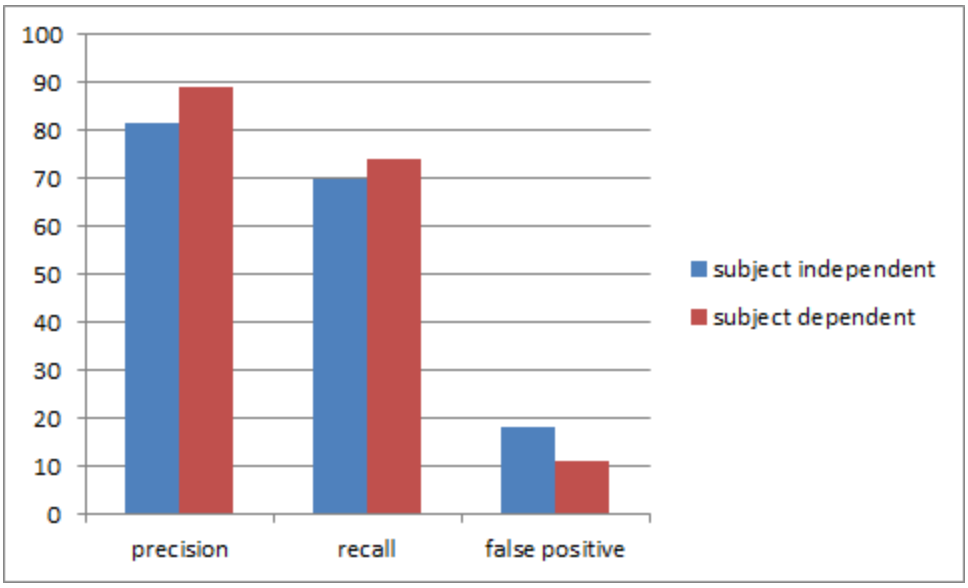


Figure 7.11: Summary results for k-NN frame-by-frame image-based food recognition.

Food	Ground truth	kNN			SVM		
		Precision (%)	**Recall (%)**	**FP (%)**	**Precision (%)**	**Recall (%)**	**FP (%)**
basil	1911	81.96	59.36	18.04	76.77	69.18	6.33
chives	3375	81.05	68.16	18.95	73.60	55.47	10.13
courgette	780	80.60	92.15	19.40	91.41	92.18	1.54
ham	1581	78.43	91.26	21.57	94.5	88.87	5.44
mushroom	2463	78.68	57.89	21.32	87.66	78.2	13.28
onion	1227	73.63	77.59	26.37	86.55	74.9	11.25
parsley	2232	86.03	62.69	13.97	72.31	70.21	17.74
spaghetti	660	96.61	91.5	3.39	95.91	92.73	3.33
Total	**14229**	**81.50**	**70.12**	**18.50**	**81.71**	**72.68**	**10.15**

Table 7.6: Frame-by-frame image-based food recognition results for subject independent protocol.

Table 7.6 presents the results for each ingredient. We see that *courgette*, *ham*, and *spaghetti* have precision and recall rates as high as 80-90%. Lower recognition rates are achieved for *basil*, *chives*, *onion*, *mushrooms* and *parsley*, around 60% for recall. *Here basil* and *parsley*, and *mushroom* and *onion* have similar colours and forms. Moreover, subjects placed more than one mushroom on the FCB to chop or slice and as a result the images are considerably noisier than for other foods. In addition, the image processing pipeline of the FCB generally does not perform well for white coloured objects such as a peeled onion or a mushroom. The false positive rates for all foods are relatively high (i.e. more than 10%) except for *spaghetti* for the k-NN classifier, and for *basil*, *courgette*, *ham* and *spaghetti* for the SVM. As

explained this was because the k-NN made numerous mutual misclassifications for *onion* and *mushroom*, *parsley* and *basil*, *courgette* and *chive*, and *ham* and *spaghetti*.

Figure 7.12 shows the results for food image classification using the Support Vector Machine (SVM) classifier with linear kernel function. The detailed results are also shown in the Table 7.6 for the subject independent protocol. Although the improvement (comared to kNN) is not particularly significant for precision and recall, the false positive rate is considerably reduced (18.5% vs. 10.15%). For the subject dependent protocol, the results for the SVM were 90.25% (precision), 78.82% (recall), and 6.49% (false positives). Overall, the SVM's performance distinctly better than the performance of the k-NN.

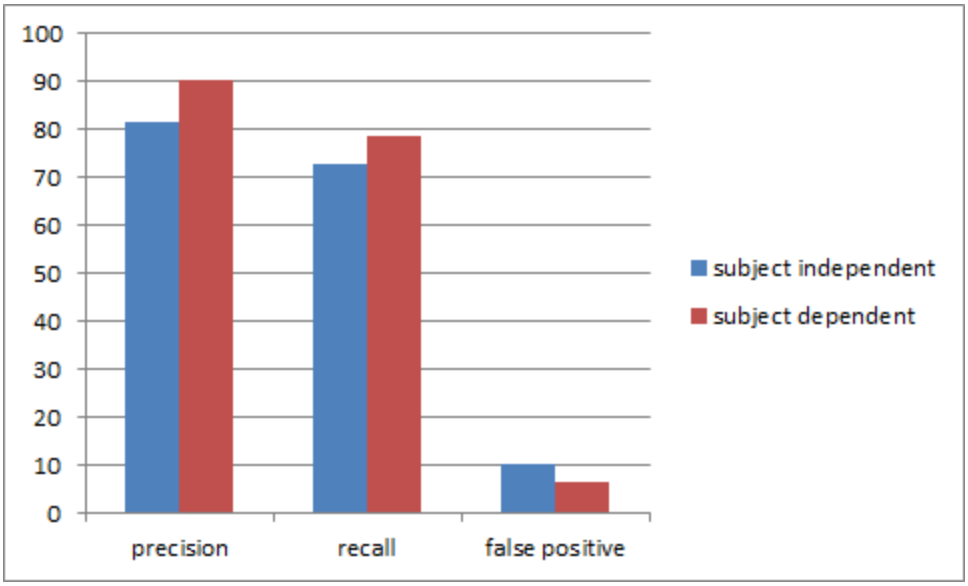


Figure 7.12: Results for SVM frame-by-frame image-based food recognition.

7.5.3 Audio-based food recognition results

Figure 7.13 presents the overall performance of food recognition based on audio classification for both the subject independent and dependent protocols. In general, recall rates are higher than 70% while precision is maintained at around 80% for both *chop* and *slice*. As expected, subject dependent results are (only moderately) higher than subject independent results. For example, the precision rate is 87% (vs. 81%) and the recall rate is 73% (vs. 72%) for *chopping* activities. These results demonstrate slightly higher classification rates than for the food images.

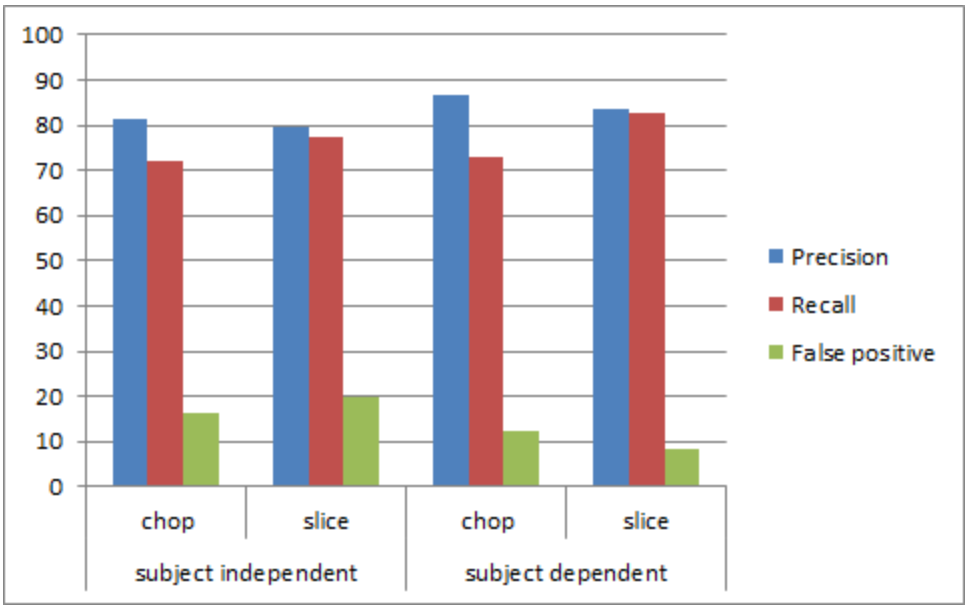


Figure 7.13: frame-by-frame audio-based food recognition results.

	Chop				Slice			
Food	**Ground**	**Precision (%)**	**Recall (%)**	**FP (%)**	**Ground**	**Precision (%)**	**Recall (%)**	**FP (%)**
basil	279	68.56	64.87	29.75	333	70.19	75.68	32.13
chive	308	67.92	52.92	25.00	331	73.54	79.76	28.70
courgette	1094	85.39	79.07	13.53	405	86.75	85.68	13.09
ham	167	86.00	77.25	12.57	366	75.14	73.50	24.32
mush	1529	80.67	72.07	17.27	549	78.25	66.85	18.58
onion	1748	85.24	72.37	12.53	1056	88.61	84.00	10.80
parsley	83	62.16	55.42	33.73	231	66.50	56.71	28.57
spaghetti	167	71.43	71.86	28.74	152	72.25	82.24	31.58

Table 7.7: Detailed frame-by-frame audio-based food recognition results (subject independent).

As detailed in Table 7.7 the highest recall and precision rates (i.e. approx. 80%) are achieved for *chopping courgette*, which is also consistent with *slicing courgette*, and a low false positive rate was maintained. High rates are also achieved for *slicing onion*, which has 84% and 88% for recall and precision respectively, and the lowest false positive rate (i.e ~10%). Parsley, however, has the lowest recognition rate using audio alone. *Chopping parsley* had 55% and 62% for recall and precision and a false positive rate of over 30% (the highest of the ingredients). *Slicing parley* was moderately better with precision and recall rates of 66% and 56% respectively. In conclusion, *slicing* food ingredients had slightly higher recognition rates than *chopping* food ingredients despite a slightly higher false positive

rate. With the overall precision and recall of over 70% for chopping and slicing food ingredients, audio-based classification still demonstrates potential for food recognition in this domain.

7.6 Event-timing analysis

As already explained, a standard frame-by-frame evaluation procedure simply counts the number of frames which belong to four statistical performance metrics categories: *true positive*, *false positive*, *true negative* and *false negative*. However, these metrics do not adequately described what we term *event recognition errors*. For some domains, the inaccurate recognition of events (i.e. too many or too few events) could be significant, even where the frame-by-frame performance is high. As our framework is designed to underpin situated services, recognizing contextual information must include the correct characterisation (and segmentation) of events. Therefore, in addition to a frame-by-frame analysis, we have evaluated the performance of KitchenSense in relation to event-timing.

We use the term *predicted sequence* to describe the sequence of activities (including unknown activities) that are recognized by KitchenSense (described in Chapter 6). Each predicted sequence is compared with an activity sequence in the ground truth, which we called the *ground truth sequence*, in computing our performance metric. For simplicity, each evaluation is treated as a binary classification problem with one activity against all other activities (i.e. chopping activity vs. not-chopping activity). In this way, at each point in time, only two classes are considered. The results of comparisons are then aggregated for the overall performance.

The event-timing evaluation procedure takes as its inputs a *predicted sequence*, which includes detected events and frames, and the ground truth sequence. Both sequences are partitioned into the sequences of segments ordered by time. A dynamic time warping algorithm is implemented to align and compare two sequences of segments. If a segment contains a whole predicted event that is not matched with any ground truth event, then this is classed an *insertion* error. If a segment contains a whole ground truth event and it is not matched with any predicted event, then it is considered a *deletion* error. A segment containing a part of a ground truth event which does not match with the predicted event, we term this an *underfill* error. A segment for which a part of a predicted event does not match with the ground truth event is termed an *overfill* error. If a segment contains a part of an event, but all frames in this part have incorrect labels, this is terms a *substitution* error. Under this evaluation scheme, some false positive frames of the frame-by-frame performance will evaluated as *overfill* errors, and false negative frames as *underfill* errors.

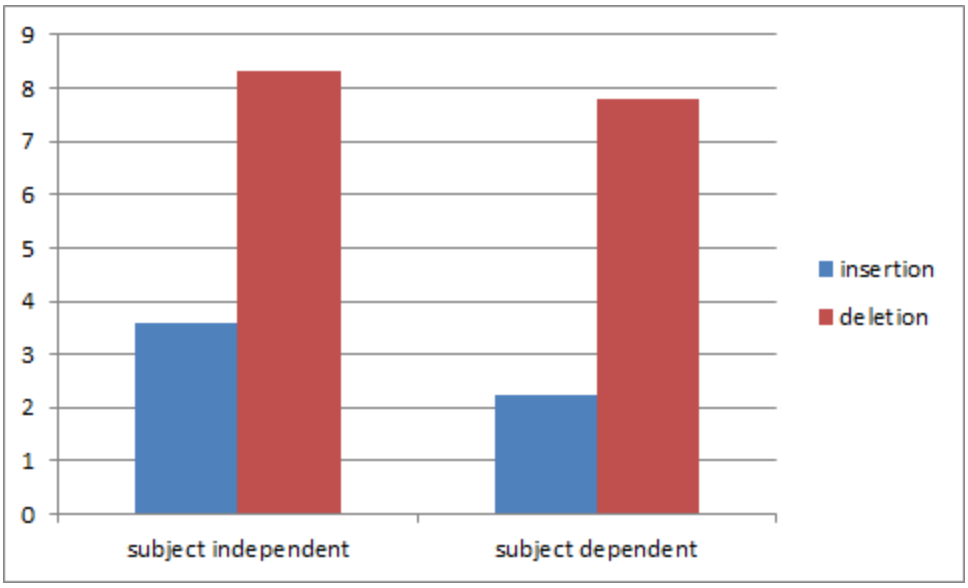


Figure 7.14: Event error detection (insertion and deletion).

Figure 7.14 presents a summary of the event errors. Here, two kinds of event errors apply, insertion and deletion. An insertion error occurs when there is no ground truth event for a whole corresponding predicted event, while a deletion error occurs when no predicted event is found for a ground truth event. Overall, as shown in Figure 7.14, the number of insertion errors for the subject dependent protocol is significantly less than the number of errors generated by the subject independent protocol. However, deletion errors are only slightly different, which indicates that they are a more serious problem.

7.6.1 Event-timing activity recognition results

Figure 7.15 presents the overall results for the event-timing performance of the recognition framework.

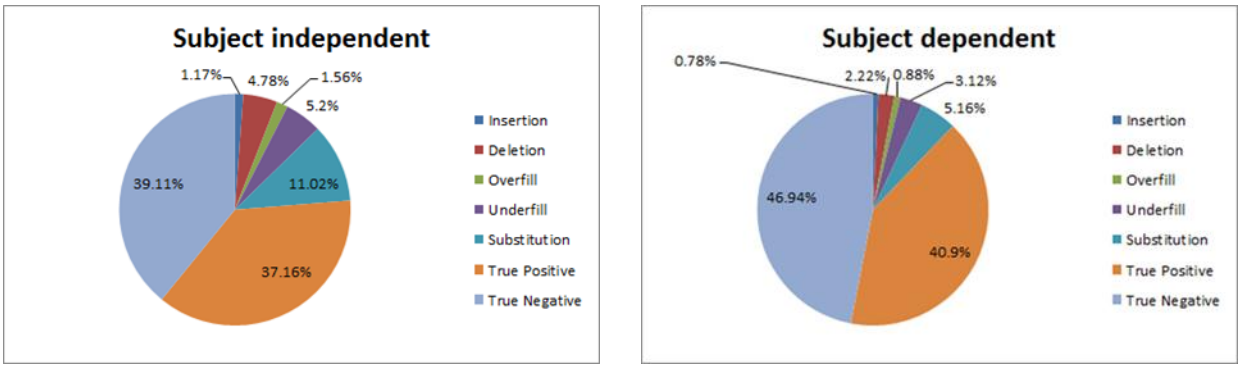


Figure 7.15: Overall event-timing activity recognition results.

Utensil	Activity	Ground	Insertion		Deletion		Overfill	Underfill
			Event (%)	*fr (%)*	*Event*	*fr. (%)*	*fr.(%)*	*fr.(%)*
Big_Bowl	*pour*	21	4.76	3.57	9.52	7.14	1.79	16.07
	wash	42	2.38	4.48	28.57	32.84	0	0
	unknown	375	2.93	1.30	12.00	5.55	5.55	6.77
Chef_Knife	*chop*	473	3.17	0.34	15.22	4.06	0.50	5.89
	slice	263	2.66	1.72	5.32	5.07	2.46	2.31
	scrap	65	1.54	0.99	0	0	0.33	10.93
	unknown	94	12.77	0.15	18.09	2.12	1.93	7.28
Colander	*drain*	63	6.35	1.82	7.94	10.54	1.82	8.73
	wash	9	0	0	66.67	100.00	0	0
	unknown	422	3.79	5.79	7.58	23.82	5.58	28.33
Frying_Pan_200	*shake*	118	9.32	3.87	5.08	10.42	0	2.08
	unknown	413	3.39	2.96	6.30	21.50	3.12	8.55
Frying_Pan_240	*shake*	397	1.26	11.37	1.26	18.48	5.21	4.27
	unknown	397	7.81	2.97	5.29	9.20	0.54	12.23
Frying_Pan_280	*shake*	601	1.50	18.62	2.33	11.70	9.04	7.98
	unknown	444	0.90	4.88	14.64	10.37	1.70	5.80
Olive_Oil	*pour*	118	7.63	4.34	4.24	0	0.87	3.47
	unknown	368	3.26	1.15	1.63	1.84	0.29	2.43
Peeler	*peel*	60	0	0	5.00	1.38	0	0.20
	unknown	368	4.89	5.50	8.97	8.30	1.71	6.13
Salt	*add*	97	0	0	1.03	1.46	0	3.50
	unknown	495	15.56	2.51	4.24	5.50	1.26	7.95
Sauce_Pan_180	*move*	463	0.22	0.83	3.24	4.23	0.51	2.05
Sauce_Pan_180_lid	*move*	251	0.80	0.28	4.38	2.00	0.49	8.69
Sauce_Pan_200	*move*	510	1.37	2.34	9.22	12.58	3.21	3.82
Sauce_Pan_200_lid	*move*	172	1.74	0.92	5.23	3.91	0.46	7.28
Sieve	*drain*	33	3.03	0.21	36.36	18.3	0.21	1.03
	wash	3	0	0	33.33	6.90	0	0
	unknown	530	6.04	2.67	11.70	8.59	3.25	6.71
Slicing_Knife	*chop*	98	3.06	1.59	9.18	8.53	0.46	0.68
	slice	43	2.33	1.60	4.65	3.41	0.40	10.82
	scrap	21	0	0	0	0	7.50	2.50
	unknown	316	4.75	2.99	18.04	12.47	7.44	8.26
Slotted_Spoon	*stir*	291	2.75	0.82	4.12	1.72	0.42	0.59
	scoop	97	1.03	3.38	1.03	1.35	2.70	15.54
	tap	32	0	0	6.25	6.02	10.84	25.30
	press	63	1.59	2.07	4.76	4.83	4.83	3.45
	unknown	67	7.46	0.39	5.97	0.66	4.90	3.70
Small_Knife	*chop*	56	5.36	2.33	3.57	1.94	0.78	5.04
	slice	244	0.41	0.10	2.87	1.98	1.62	0.47
	scrap	71	5.63	29.73	0	0	5.41	8.11
	unknown	178	10.67	2.74	10.67	6.28	1.60	13.42
Soy sauce	*pour*	50	0	0	4.00	2.20	0	0.63
	unknown	271	1.85	0.53	4.80	3.02	0.46	7.54
Spatula	*stir*	427	0.70	0.34	2.58	1.35	0.21	0.50
	scoop	72	0	0	8.33	10.58	0.68	1.02
	turn	211	5.69	5.10	11.37	9.77	1.02	5.10
	tap	35	2.86	3.70	0	0	1.23	0
	press	61	0	0	1.64	6.50	0.41	8.54
	unknown	322	2.48	0.44	16.15	2.32	0.44	1.79
Spoon	*stir*	192	4.17	0.59	0.52	0.62	0.07	1.11
	scoop	78	0	0	1.28	1.99	0	0.28
	tap	34	0	0	11.76	7.77	0	0
	unknown	202	7.43	1.08	10.40	1.55	0.07	5.27
Tap_Water	*in use*	486	0.21	0.21	27.16	11.97	0.49	10.87
	unknown	5	0	0	20	5.53	2.89	6.92
Whisk	*whisk*	35	5.71	1.69	8.57	5.08	1.02	1.69
	tap	4	0	0	0	0	0	0
	unknown	202	0.99	3.92	1.98	4.58	4.25	18.95
Total (subj. independent)		**11929**	**3.26**	**1.17**	**8.06**	**4.78**	**1.56**	**5.20**
Total (subj. dependent)			**2.25**	**0.78**	**7.75**	**2.22**	**0.88**	**3.12**

Table 7.8: Detailed event-timing errors, sums are shown at the bottom of the table.

Two metrics, true negative and true positive, are included in the recognition rate. True positive measures the proportion of frames correctly classified as they are labelled in the ground truth sequence at a certain time. For example, a chop at time t is recognized as a chop (i.e. both chop frames appeared in both predicted and ground truth sequence at time t). True negative mean correctly classified as not being the activity of concern, for example, if *not chopping* at a time is correctly classified as *not chopping*. Therefore, from our charts we can see that the overall accuracy was 76.28% (i.e. 37.16+39.11%) for the subject independent protocol and 87.84% (i.e. 40.9+46.94%) for the subject dependent protocol. These are lower than we calculated using the frame-by-frame evaluation. Five other metrics measure the incorrect event and timing classification. The incorrect recognition rate (error rate) for timing and event errors is 23.91% for the subject independent protocol and 12.2% for the subject dependent protocol. Overall, the recognition rates are reasonable, given that our study was conducted in a real-world setting (unlike comparable studies which are usually conducted in much more controlled environments).

Deletion and substitution errors for the subject dependent protocol are significantly less than those for the subject independent protocol, 5% compared to 2.2% for deletion, and 11% compared to 5% for substitution. Using Table 7.8 to consider this in more detail we see that *washing* with the *colander* has 100% deletion, which led to a 0% recognition rate. This problem was not apparent in the confusion matrix of the frame-by-frame evaluation as it was the result of an event deletion. Other activities have high deletion errors (10%-30%) including *washing* with the *big bowl*, *draining spaghetti* using *colander*, and *draining spaghetti* using the *sieve*. However, we should accept the possibility that in these cases the sensors may have got wet leading to the signal being dropped.

As shown in Table 7.8, *scoop*, *stir*, *pour*, *add*, *chop* and *slice* have between 0-3% event-timing error rates, which are extremely low. In particular, *scraping* and *pouring oil* have a 0% deletion time error (i.e. no frame or event is missing when classified). In contrast, the *colander*, *big bowl* and *sieve* have high deletion error rates. The *frying* pan and *saucepan* activities have moderate deletion error rates (5-10%) although these are relatively high compared to other utensils. It is unclear whether the heat from hob might be causing signal drops. High insertion errors rates are apparent for *scraping* with the *small knife* (29%) and *tap water* (27%) even though there are on a small number of insertion events. This can be explained by the fact that the events inserted are relatively long. In practice, sensors attached to a tap faucet will move in a distinct manner, but only for relatively short durations when the user turns the tap on or off. Similarly, *scraping* with the *small knife* is similar to a number of *unknown* movements. Therefore, a number of *unknown* events were readily classified as *scraping* and led to insertion and

substitution errors. By contrast, *scraping* with the *chef knife* and *slicing knife* are quite distinct and very few unknown events were inserted. Subjects often scraped with a "push" action, pushing chopped ingredients across the chopping board to make space to chop other ingredients with the chef knife or the slicing knife.

Unknown activities had high overfill and underfill error rates with considerably more underfills than overfills (i.e. 5.2% vs 1.56% for subject independent and 3.12% vs. 0.88% for subject dependent). Overfill and underfill corresponds to misalignment between predicted and ground truth sequences. These can in part be explained by the occurrence of noisy movements (actually *unknown* activities) at the beginning and end of segments that were not annotated by the coders.

7.6.2 Event-timing results for image-based food recognition

The events of placing a food ingredient on the FCB (before it is chopped) are listed in Table 7.9. These events are detected by the food image classification component of the recognition framework. The overall results are shown in the figure 7.16.

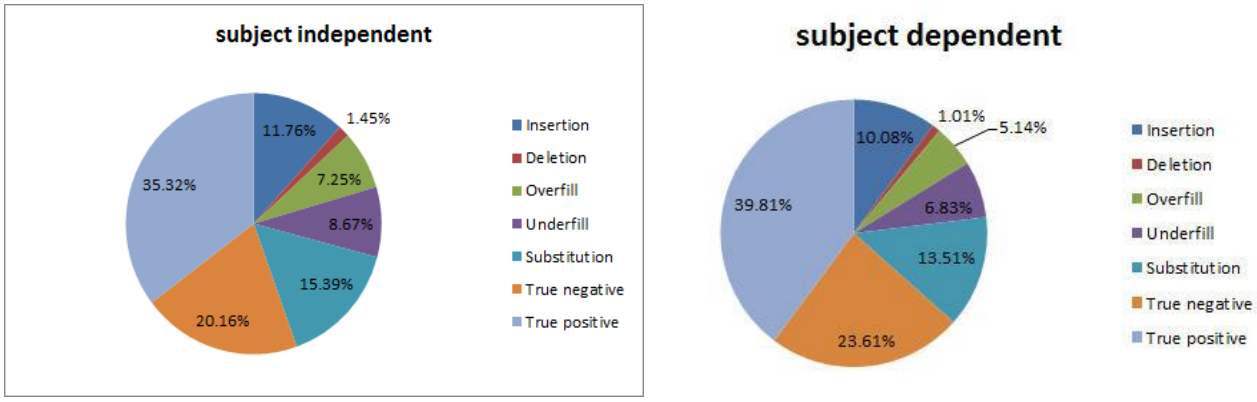


Figure 7.16: Event-timing k-NN results for image food recognition.

As shown in Figure 7.16, the overall recognition accuracy rate (true positive + true negative) was 55.48% for the subject independent protocol and 63.42% for the subject dependent protocol. These are considerably lower than the frame-by-frame analysis results. As seen, substitution (incorrectly classified) and insertion errors mainly contribute to the event-timing errors. This can be explained from our observations of collected videos, that users sometimes placed non-food items such as a knife or a spoon on the FCB. These items were not annotated by coders (not part of our annotation protocol) and therefore were omitted from the ground truth – although they would have been likely to be identified by the recognition framework. Such cases would have led to insertion errors. As seen the chart, deletion error rates are low (~1%) primarily because the FCB reliably recognises such events in general.

	Ground Truth		Insertion(%)		Deletion(%)		Overfill(%)	Underfill(%)	Substitution(%)
Food	event	time	event	time	event	time	Time	Time	Time
basil	27	637	48.15	14.60	3.70	0.94	2.98	5.65	20.09
chive	39	1125	10.26	13.07	2.50	0.80	5.24	7.91	24.00
courgette	35	260	14.29	5.77	2.86	4.23	6.54	10.38	2.31
ham	35	527	5.71	3.98	8.57	0.57	17.27	5.88	5.31
mushroom	93	821	12.90	13.15	2.15	0.73	3.05	11.45	11.45
onion	52	409	11.54	10.27	1.92	2.20	3.18	7.09	7.09
parsley	24	744	37.50	13.58	25.00	1.48	10.89	9.01	22.45
spaghetti	35	220	25.71	14.09	5.71	6.36	17.73	17.27	3.64

Table 7.9: Detailed event-timing errors for kNN subject independent image-based food recognition

From Table 7.9 we can see that more than 20% of *basil*, *chive* and *parsley* is substituted by other foods or each other. As already described, from the collected images it can readily be seen that the appearance of basil and parley are quite similar in both shape and colour. 11% of *mushroom* and 7% of *onion* are also substituted. The lowest substitution rate is achieved by *courgette* (2.3%) and *spaghetti* (3.6%) which both have distinct shapes and colours. Most foods have relatively high insertion error rates (10%-14%) except for *courgette* and *ham*. This is also because one single *courgette* was *chopped* or *sliced* at a time (actually there is just one per recipe), and the *ham* often entirely covered FCB, considerably reducing the possibility of noise. Deletion time, however, was high for both *courgette* and *spaghetti* and *spaghetti* also had high overfill and underfill errors (approximately 17%).

Food	**Ground truth**		**Insertion(%)**		**Deletion(%)**		**Overfill(%)**	**Underfill(%)**	**Subst.(%)**
	event	**Time**	**event**	**time**	**event**	**time**	**Time**	**Time**	**time**
basil	27	637	18.52	9.89	3.70	2.04	5.18	5.65	17.9
chive	39	1125	12.82	10.76	10.26	0.98	7.91	8.44	14.49
courgette	35	260	5.71	3.46	8.57	0.77	2.31	1.92	2.69
ham	35	527	2.86	4.36	5.71	0.57	1.33	1.52	1.33
mushroom	93	821	4.30	9.62	2.15	0.85	4.38	5.85	9.26
onion	52	409	11.54	7.82	3.85	1.71	3.67	4.65	6.36
parsley	24	744	12.5	7.66	4.17	1.48	5.78	6.32	16.94
spaghetti	35	220	8.57	8.18	2.86	0.91	2.73	2.27	6.82

Table 7.10: Detailed event-timing errors for SVM subject independent image-based food recognition.

SVM performance for event-time food recognition based on images is presented in Figure 7.17, the detailed results for which are presented in Table 7.10. The results show significant improvement over the k-NN's performance. For the subject independent protocol, the SVM's overall accuracy was 68.62%,

which is a 13% improvement on the kNN. Subject dependent performance of SVM was 76.79% for overall accuracy which was a 12% improvement in the k-NN.

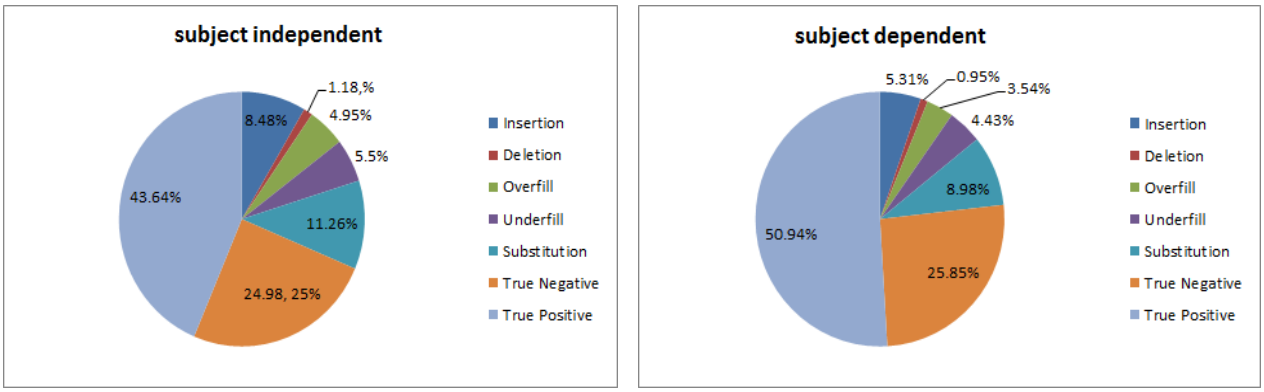


Figure 7.17: Event-timing SVM results for food recognition based on images.

In summary, the overall subject dependent accuracy for food recognition based on images using the kNN was 55.5% and these results were significantly improved using the linear SVM to 68.6% and 76.8% for the subject independent and the subject dependent protocols respectively. These results are promising for optical fiber imaging for food recognition in the kitchen under ambient lighting conditions.

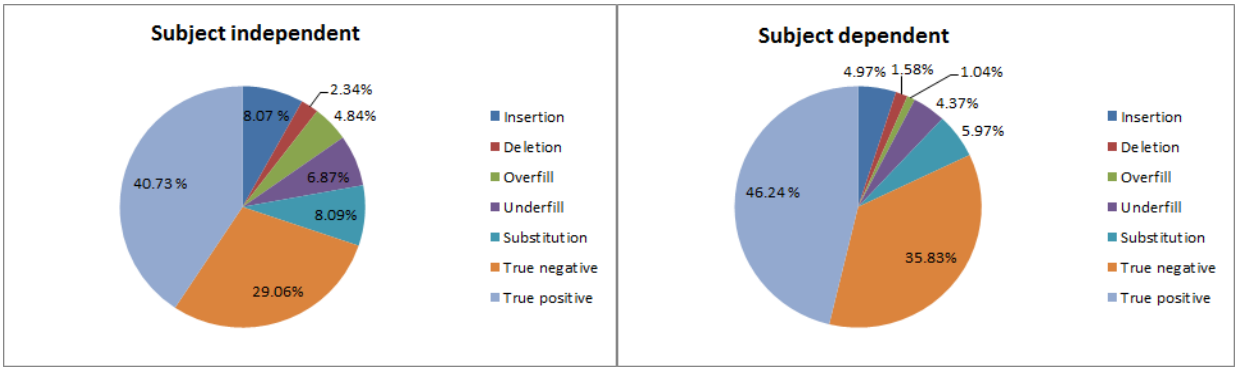


Figure 7.18: Audio event-timing food recognition results for chooping.

7.6.3 Event-timing results for food recognition based on audio

7.6.3.1 Chopping food event-timing results

Overall, the recognition rates were approximately 70% for the subject independent protocol and 82.07% for the subject dependent protocol. Timing errors were approximately 30% and 18% for both evaluations, and were dominated by substitution and insertion. The high substitution time indicates that a considerable number of audio frames were misclassified. Using Table 7.11 to consider these results in more detail it is apparent that *mushroom*, *parsley*, and *chive* have a high substitution time of more than 10%, while *ham*

and *courgette* both have low substitution and deletion time errors. The high insertion and underfill time errors for onion is probably due to the users who chopped the onion into smaller parts and then made chopped each part individually (this can produce very short chopping events). Less timing errors, hence higher accuracy, was achieved for *courgette* and *ham*, which is consistent with frame-by-frame results.

	insertion		**deletion**		**overfill**	**underfill**	**substitution**
Food	**event**	**time**	**event**	**time**	**Time**	**time**	**time**
Basil	5.71	8.24	2.86	1.08	2.15	0.72	11.47
Chive	7.32	9.42	2.44	0.65	2.92	3.57	12.66
Courgette	13.89	0.46	0.10	0.27	1.37	0.46	0.73
Ham	8.33	1.80	0.33	2.99	2.99	1.80	2.99
Mushroom	8.87	10.73	4.81	4.38	8.44	7.13	14.98
Onion	5.13	10.47	3.85	2.23	4.52	13.22	5.66
Parsley	9.68	9.64	3.23	3.61	4.82	2.41	14.46
Spaghetti	3.70	11.38	7.01	2.40	7.78	3.59	6.59

Table 7.11: Event-timing errors for chopping food recognition based on audio (all figures in percentages).

Figure 7.19: Audio event-timing results for food recognition with the slicing activity.

7.6.3.2 Event-timing results for slicing food

The overall event-timing results for audio-based recognition of food with the slicing activity are shown in Figure 7.19. In general, the results are consistent with the result for *chopping* (but slightly better). As shown in Figure 7.18, overall accuracy for the subject independent protocol was 70.6% and for the subject dependent protocol was 81.4%. This can be explained that sounds produced by slicing are more distinct and less noisy. In fact, most noise during chopping or slicing was made by the contact between the knife's blade and the surface chopping board. The fact that this effect is slightly less for slicing than for chopping probably explains the lower overall event and timing errors. As shown in the chart Figure

7.16, substitution and insertion time, dominate event-timing errors at 15% for the subject independent protocol and 10% for the subject dependent protocol (again these are a slight improvement on the chopping food activity). The results are also better than for image based food recognition.

	Insertion		**deletion**		**Overfill**	**underfill**	**substitution**
Food	**event**	**time**	**event**	**time**	**Time**	**time**	**time**
basil	3.23	9.61	3.23	2.70	3.30	1.20	9.31
chive	7.14	6.34	2.38	0.91	2.72	0.91	11.48
courgette	4.26	2.22	0.1	0.74	1.98	0.25	0.49
ham	4.08	1.64	2.04	1.09	1.09	0.82	1.37
mushroom	13.04	15.66	7.61	5.83	11.48	11.11	11.84
onion	4.82	7.86	2.41	2.65	6.06	9.19	9.56
parsley	8.57	0.87	8.47	4.76	5.63	9.09	8.23
spaghetti	14.29	18.42	10.2	1.97	7.89	1.32	2.63

Table 7.12: Event-timing errors for food recognition based on audio for slicing (all figures in percentages).

As detailed in Table 7.12, high insertion and substitution event timing errors are apparent for *chive*, *mushroom* and *onion*. In contrast, *ham* and *courgette* have low insertion and substitution event timing errors. These results are consistent with *chopping* food activity results. Also *slicing spaghetti* has less substitution timing errors than *chopping spaghetti* (i.e. 2.6% vs. 6.5%). However, *slicing spaghetti* has more insertion time than *chopping spaghetti* (i.e. 18.4% vs. 11.3%).

7.7 Conclusion and discussion

This chapter constitutes the culmination of our research in that we presented the results of a real-world evaluation of KitchenSense's performance in a full-blown and naturalistic study. A dataset was collected from 12 subjects who each prepared a Spaghetti Röstie three times in the French Kitchen. The kitchen setting for the experiment was almost equivalent to a home-based kitchen and the subjects performed their cooking and food preparation activities in a natural manner. No time constraints were imposed on the cooking time and no instructions were given to the subject when cooking. We collected videos for 30 hours food preparation and these videos were independently annotated by two (trained) coders. Annotated data was then subjected to an inter-rater reliability procedure.

We evaluated the accuracy of both activity recognition and food recognition. Both aspects rigorously analysed using both frame-by-frame and event-timing approaches. Accelerometer data were used for evaluating activity recognition. Image and audio data are for food recognition. Each evaluation was carried out under the subject independent protocol and subject dependent protocols. For the frame-by

frame analysis of activity recognition, subject independent protocol performance results of 79% for recall and 87% for precision were achieved. For the subject dependent protocol (on the open dataset of 59 activities including unknown activities) result of 84% for recall and 92% for precision were achieved. An evaluation on the closed dataset of 41 activities (excluding unknown activities) produced consistent results that were just 2% lower than those for the open dataset. The frame-by-frame analysis for food recognition has also demonstrated the potential for optical and audio sensing within a chopping board. Generally, food image classification results were also very promising. The k-NN classifier produced results of 70% for recall and 81.5% for precision (subject independent), and 74% for recall and 89% for precision (subject dependent). The SVM's results were 73% for recall and 82% for precision (subject independent), and 79% for recall and 90% for precision (subject dependent). Moreover, audio food classification results were around 70% for recall and 80% for precision for both chopping and slicing food. These results are very promissing for practical AR-based applications.

In addition to the frame-by-frame analysis, an evaluation of event-timing was carried out as the recognition of events and their timings is very important to a number of the situated support services that we envisaged in our motivation. The results for event-timing activity recognition were slightly lower than those achieved with the frame-by-frame approach, as five more errors relating to event and timing were included. Activity recognition performance maintained overall accuracies of around 76% (subject independent) and 88% (subject dependent). Note that for image-based recognition the SVM's results achieved 68% accuracy (subject independent) and 77% (subject dependent) notably higher than the k-NN, but also notably lower than those of the frame-by-frame analysis. However, the audio-based recognition results were considerably better at 70% (subject independent) and 80% (subject dependent) for both chopping and slicing food ingredients. In short, the performance achieved by KitchenSense has gone some way towards demonstrating the feasibility of activity and food recognition using pervasive sensing in the kitchen contexts.

The results also point to a number of areas where improvement is required. The event-timing deletion errors are probably the most serious error category, and the level of these is still high for our accelerometer-based activity recognition. Likewise, as with most vision systems, image-based food recognition performance is significantly affected by the ambient light-conditions. Thus the recognition algorithm should be improved to be more adaptive as ambient lighting changes. Moreover, one phenomenon that had a direct impact on the performance results was the noise generated by non-food items being placed on the FCB. To this end the rejection of unknown items could be improved (i.e. with adaptive-thresholds). Finally, one of the most effective means of improving event-timing recognition rates

for both food and activities would be to use smoothing. As seen in the evaluation section, event-timing errors could be significantly reduced by smoothing over detected events.

Chapter 8: Conclusion

8.1 Key contributions

8.1.1 Human activity recognition

On addressing the problem of human activity recognition in the kitchen, our main contributions relate to the development of pervasive sensing technologies, algorithms, and empirical evaluation of our activity recognition framework.

Rapid prototyping of Wii-Remote instrumented utensils: Our preliminary study used modified versions of 4 Wii Remotes to instrument 3 knives and one large serving spoon. This involved retaining the Wii-Remote's BCM2042printed circuit board, on-board 8051 microprocessor, and ADXL330 accelerometer and Bluetooth HID communication, but removing the cases, buttons and IR camera. This results was in effect a wireless sensor unit that was small enough to enable us to embed them into purpose made utensil handles. The handles were designed (using 3D modeling software) and then printed out using Fused Deposition Modelling rapid prototyping technology. This was the first step, and our first successful attempt, to fulfill the requirement of producing a sensing platform that maintained the full functionality of the utensil but that was invisible to users (i.e. the subjects of our studies).

The collection and annotation of a real-world dataset with Wii-Remote instrumented utensils: using the first version of our modified utensils were collected our first dataset. This dataset was collected from 20 subjects whilst preparing a mixed salad and sandwich under semi-realistic settings in the Ambient Kitchen. The subjects were given food ingredients and the instrumented utensils and were asked to perform activities in a natural manner without any (apparent) observation or instruction from members of the research team. The collected videos were independently annotated by three coders using 12 food preparation activity labels.

Activity classification of accelerometer data for 4 Wii-Remote instrumented utensils: In our the first attempt at activity recognition in the kitchen, several classification algorithms, including a Decision Tree C4.5, Naïve Bayes, and a Bayesian network, were used. The feature vectors were extracted from different sized sliding windows for training and testing those classifiers. Through an initial frame-by-frame and subject independent evaluation we established that the Decision Tree C4.5's performance was the highest with an activity recognition rate of 82.9% (without unknown activities).

Real-time detection of 12 activities including unknown and idle activities: Using the dataset collected in our first study (Wii Remote instrumented utensils) and with the inclusion of unknown and idle activities (i.e. open dataset), we explored the feasibility of developing a recognition algorithm for food preparation

activities in real-time. A class-based threshold dynamic time warping recognition system (CBT-DTW) was developed and implemented. Our findings shows that a CBT-DTW activity recognition system performed well on this real-world dataset even when a number of activities had a small number of training examples (such imbalance is typical of real-world dataset. On a frame-by-frame evaluation, the system could detect 12 activities with 82% recall and 83% precision (subject independent) in real-time.

The development of a set of 18 utensils and appliances instrumented with 22 WAX3 accelerometers: Our improved set of kitchen artefacts contained most common utensil used in everyday kitchen. The use of the OpenMovement WAX3 accelerometers made it extremely easy to embed (and thereby hide) these sensors inside utensils that retained the look and feel of a consumer product. The handles and various parts of utensils (e.g. lids of the saucepans) were re-designed and again printed out using the 3D FDM prototyping technology.

The development of the KitchenSense activity framework: the development of a scalable four tier publishing-subscribe architecture by which the basic activity recognition. Tier 1 comprises real-time fusion data streams. Tier 2 is the Event Detector layer which can detect the events such as utensils and food placement on the chopping board. Tier 3 is the core of the framework and includes the *Feature Extractor* (FE) and the *Recognizer* components (and the pre-trained data. Tier 4 contains allows the implementation of situated services lie behind the motivation for this research. The system was implemented and deployed in both the Ambient Kitchen 2.0 and its sibling installation the French Kitchen (where we conducted our main study).

The collection and annotation of a realistic, complex dataset using KitchenSense: given a recipe, a set of 18 instrumented utensils, and 4 further instrumented objects, 12 subjects were asked to prepare the *Spaghetti Röstie* recipe 3 times each (i.e. 3 cooking sessions) without observation or instruction members of the research team. Approximate 30 hours of food preparation time was video recorded, and the footage was annotated by two independent coders. The annotation was then evaluated using an agreement inter-rater reliability scheme. The dataset consisted of 83,076 frames (one second per frame) of 59 activities including unknown activities, 14,229 images of 8 food ingredients (from the FCB), and 8,798 audio frames for the chopping and slicing foods. This constitutes a genuinely unique and complex dataset collected under naturalistic conditions in a kitchen.

Real-time recognition of 59 fine-grained food preparation activities including unknown activities: Through a rigorous frame-by-frame analysis, we established an activity recognition performance of 79.4% (recall) and 87.6% (precision) for the subject independent protocol, and 84% (recall) and 92% (precision) for the subject dependent protocol. These results demonstrated the feasibility of pervasive

sensing technologies and the recognition of larger numbers of fine-grained food preparation activities in the kitchen. The results have the potential to underpin practical applications that utilize context recognition such as situated support and prompting services.

Continuous (event-timing) real-time recognition of 59 fine-grained food preparation activities: We argued that event information is important for many of our envisage applications and thus recognised that some classes of error are likely to be missed if we only evaluated at the frame-by-frame level. We therefore designed and conducted a rigorous event-timing analysis for continuous activity recognition. This resulted in recognition accuracies of 76.3% (subject independent) and 87.8% (subject dependent) and we characterised the nature and likely source of our event and timing errors, that is, the occurrence and likely source of insertion, deletion, substitution, overfill, underfill, and substitution errors.

As a result, we envisage that many of the above contributions will form the basis of next generation applications that utilize real-time activity recognition, including prompting people with dementia, tasks-based language learning, user identification, and healthier cooking in the kitchen. Indeed, although not reported in this thesis, we were involved in the development of prototypes for many of these applications and these were developed during the course of this research [16], [26], [28], [69], [70], [71], [76], [77], [78] and [79].

8.1.2 Food recognition

The development of the fiber chopping board (FCB): the FCB is an instrumented chopping board comprising 600 embedded optical fibers, a webcam camera for food image sensing and a microphone for discriminating foods being chopped or sliced. With the integration of optical fibers into the FCB, we were successful in developing the first embedded sensing device for fresh foods (as they are prepared). Two algorithms, k-Nearest Neighbour and Support Vector Machines, were developed for classifying the food images captured by the FCB and a Gaussian Mixture Model algorithm was used for audio-based food recognition based on the FCB's audio capture capability. In two pilot studies we established the reliability of the FCB as a food recognition technology.

The recognition of food ingredients using optical fiber sensing images: Our frame-by-frame analysis of 14,229 images of 8 food ingredients in our main study yielded recognition rated of the k-Nearest Neighbor of 70% (recall) and (81.5%) precision under the subject independent protocol, and 74% (recall) and 89% (precision) under the subject dependent protocol. The Support Vector Machine slightly outperformed the k-NN with subject independent results of 82.7% (precision) and 72.7% (recall), and subject dependent results of 90.2% (precision) and 78.8% (recall). These results demonstrate considerable and promising for embedded optical sensing in this domain.

Audio-based classification of food on the chopping board: Using the FCB's audio capture capability we demonstrated the feasibility of using the sounds produced when a food is *chopped* and *sliced* to recognise the food. These sounds were first pre-processed (i.e. noise reduction) and then a large feature vector was extracted. A Gaussian Mixture Model algorithm was trained and tested using these features and a frame-by-frame analysis on 5,375 images. For *chopping* we achieved subject independent recognition rates of 72% (recall) and 81.3% (precision) and subject dependent rates of 73% (recall) and 86.7% (precision). For slicing we achieved subject independent recognition rates of 77% (recall) and 79.6% (precision) and subject dependent rates of 82.7% (recall) and 83.4% (precision).

Continuous recognition of food ingredients from embedded imaging: An evaluation of event-timing recognition for 8 different food ingredients using the optical fiber sensed images of the FCB demonstrated a subject independent accuracy of 55.5% for k-NN and 68.6% for SVM, and a subject dependent accuracy is 63.4% for k-NN and 76.8% for SVM.

Continuous recognition of food ingredients using sounds from chopping activities. Features extracted from food *chopping* sounds captured by the microphone embedded inside the FCB were classified using a pre-trained GMM classifier and an event-timing analysis yielded a subject independent accuracy of 70% and subject dependent accuracy of 82%.

Continuous recognition of food ingredients using sounds from slicing activities: Features extracted from food *slicing* sounds captured by the microphone embedded inside the FCB were classified using a pre-trained GMM classifier and an event-timing analysis yielded a subject independent accuracy of 70.6% and subject dependent accuracy of 81.4%.

8.2 Limitations and discussions

8.2.1 Limitations on activity recognition

The results of our design, implementation and experimental studies of activity recognition are bound by a number of limitations, some of which are feasible topics for future development and research:

- Although our instrumented utensils prototypes had most of the functional and aesthetic qualities of regular cooking utensils, a key deficiency was their lack of full water resistance. Reflection on some of the evaluation results led us to believe that some of the embedded sensors, particularly those in the handles of the *sieve* and *colander* utensils, might have got wet when being washed or being used to drain ingredients. Such cases are likely to have led to dropped signals and will have impacted negatively on the evaluation of the recognition framework (i.e. event deletion).

- Although a most activities have good recognition rates (i.e. more than 70% accuracy), a few activities had notably low accuracies. For instance, *draining* with the *sieve* (53%), *draining* with the *colander* (43%), *washing* the *colander* (0%), *chopping* with the small *knife* (67%), and *washing* the *big bowl* (56%). While many of the low accuracy activities are related to draining or washing activities, and therefore might be due to signal losses when sensors got wet, further detailed exploration is required as it may be that the lexicon of activity labels for utensils such as the small knife is inappropriate (i.e. the fine grained set of activities used for the small knife is unrealistically large and we should simply have one activity for the *small knife*, for example, *cut*).
- A small number of activities have high false positive rates (i.e. more than 30%). For example, draining with the colander (33.4%), washing the sieve (33.3%), chopping with the small knife (37.9%), pouring with the big bowl (37.5), this might explained in the same manner and the low activity recognition rates.
- In general, substitution and underfill errors caused by event and timing were relatively high for continuous activity recognition (i.e. 11% substitution and 5.2% underfill for the subject independent protocol, and 5.2% substitution and 3.12% underfill for the subject dependent protocol).

8.2.2 Limitations on food recognition

The fiber chopping board (FCB) is the first prototype of its kind for performing embedded food recognition. Although it fulfilled many of the functionalities of a normal chopping board (in that the sensing infrastructure was largely "invisible" to users), several limitations were apparent:

- As all the components of the board were made of acrylic glass, the weight of the first prototype was rather heavier than a normal (e.g. wooden) chopping board.
- The camera and microphone embedded inside the FCB was wired. This wired quality of the board, and its unusually high weight, is potentially incongruous to a regular kitchen user. Indeed chopping boards are usually frequently washed between the preparation of ingredients and that users did not attempt to do this is indicative of the fact that they were not using it completely naturalistically. Replacement by a wireless webcam is a readily available solution.
- The recognition rates of some foods such as basil and parsley are still not high (i.e. between 50-60% accuracy), without higher resolution imaging (i.e. an alternative imaging technology to fiber optics) this is unlikely to be achievable. However, the rise in popularity of multi-touch interaction more generally means that surface-based imaging technology is a matter of considerable commercial interest.

- In general, the overall continuous food recognition accuracies of 68.6% for image classification and 70% for audio classification (subject independent) are still quite low and they need to be further improved. Specific cases such as the high false positives rates for *mushrooms* and *parsley* (10-20%) indicate that the sensitivity of the colour imaging (and the impact of ambient light on this) and the resolution of the imaging are the primary factors here.
- In general, the two most common event-timing errors for food recognition were substitution and insertion; 5-9% for insertions and 9-15% for substitutions. This significantly impacted on the overall accuracy of the continuous food recognition.

8.3 Future work

8.3.1 Deployable pervasive sensing technology

Some of the simplest next steps for this research relate to practical issues of deployment, and changes to the design of the sensors and utensils that impact on this. Firstly, as already described, utensils need to be fully water-proof. Furthermore, while the power-saving functionality of the WAX3 sensors means that they can be deployed for weeks without recharging, a solution for charging the sensor batteries that does not require the utensils to be physically dismantled is required. Similarly, there are a number of obvious improvements that could be made to the FCB; the wired camera and microphone should be replaced with wireless one, and the FCB's frames, currently made from acrylic, should be replaced with wood (or a lighter weight plastic). As we ultimately intend that KitchenSense will support applications such as dietary monitoring, calorie intake estimation and healthier nutrition advice systems, it is likely that food ingredients will need to be weighed and KitchenSense should be capable of estimating their weight (post-preparation) and make this information available to situated support service applications. The integration of load sensors in food bearing surfaces is already a feature of a number of other research prototypes and this is a relatively simple enhancement for the FCB.

8.3.2 Recognition algorithms

Improving the recognition rates for activities and foods as well as reducing the false positive and event-timing errors are the key issues for the development of the recognition algorithms of the KitchenSense. For example, future versions of the activity recognition algorithm should also utilize the concurrent movement patterns of utensils as these often directly rate to the activities being performed. For instance, washing a food container can be detected with the combination of the sensors embedded in the food container and the sensor attached to the tap. Moreover, common sense knowledge of the recipe being prepared, and the sub-activities of the recipe, could have a significant impact on recognition rates (i.e. people nearly always *peel* a *carrot* before *chopping* it).

In the initial experiments, k-Nearest Neighbor and Support Vector Machine algorithms performed reasonably well on the food image classification task (i.e. 70%-78% overall accuracies). In order to reduce false positives (frame-by-frame analysis) and substitution and insertion errors (event-timing analysis) a better rejection schema could be used which is based on an adaptive-threshold and will perform better rejection of non-food items placed on, or sensed by, the Fiber Chopping Board. The adaptive-thresholds should also be automatically adjusted in relation to the ambient lighting conditions.

Although the Gaussian Mixture Model algorithm performed well on the food recognition task, using the sounds associated with the *chopping* and *slicing* food activities, it is highly desirable that both the recognition rate and real-time performance is improved. One candidate enhancement would be to address the high dimensionality of the audio feature vector used in KitchenSense. Solutions for reducing the dimensionality of the feature vector include Principal Component Analysis [94] or the selection of an optimised feature set based on feature learning [80]. Furthermore, smoothing sliding windows might be considered as a means of improving the accuracy while reducing the incorrect classification rate.

8.3.3 Recipe tracking

While KitchenSense has clear potential as a deployable context aware sensing framework, we have not actually demonstrated its utility to any of the situated support services that we envisage. The most likely candidate would be a recipe tracking service as the monitoring the progression of food preparation steps while a user is making a meal in the kitchen is itself likely to be an underpinning service for many other applications, such as prompting people during meal preparation, guiding/teaching the novice cook for improving cooking skills, automatic cooking video segmentation for creating cooking media, etc.

To realise recipe tracking our first problem is how to represent a recipe. One possibility is that the recipe is simply represented as a sequence of food preparation steps. For example, the *Spaghetti Röstie* recipe can be split into 10 steps: *preparing spaghetti*, *preparing courgette*, *preparing parsley*, *preparing ham*, *preparing mushroom, preparing chives*, *preparing onion, preparing basil*, *cooking mix*, and *serving*. Each step might be a combination of several utensils, activities and food ingredients. For example, the step "*preparing courgette*" would be collection of *chef knife*, *peeler*, *peeling*, *chopping* or/and *slicing*, *scooping* and *courgette*. In this presentation, a step is treated as a high-level activity and its components are (in-use) utensils, foods, and low-level activities.

In order to track the on-going step within a recipe approaches such as hidden Markov models (HMM) are required to model the recipe. Food preparation steps that need to be tracked are hidden states. An observation can be an in-use utensil, a human activity or a food ingredient. Observations are actually the outputs from the KitchenSense framework. The model parameters would be the initial state matrix, state

transition matrix, and observation probability matrix. These parameters might be trained either by mining recipe knowledge from the web collections of recipes, or collected datasets, using Expectation Maximization training algorithms [96] or similar approaches.

Appendix A: Data for Slice&Dice and CBT-DTW Experiments

In the chapter 3 and 4 we described the experiments for Slice&Dice and CBT-DTW systems. We now describe the process of creating the experimental data. The collected dataset for Slice&Dice is stored in: http://di.ncl.ac.uk/publicweb/AmbientKitchen/KitchenData/Slice&Dice_dataset/

```
<Capture Time="2009-02-06T17:13:33">
<Directory>
      D:/AmbientKitchen-Logger/App/Capture/6-2-2009/Subject1
</Directory>
<Configuration>
      <Mote1 type="wiimote">
            <index>1</index>
            <samplerate>40</samplerate>
      </Mote1>
      <Mote2 type="wiimote">
            <index>2</index>
            <samplerate>40</samplerate>
      </Mote2>
      <Mote3 type="wiimote">
            <index>3</index>
            <samplerate>40</samplerate>
      </Mote3>
      <Mote4 type="wiimote">
            <index>4</index>
            <samplerate>40</samplerate>
      </Mote4>
      <Camera1 type="ipcamera">
            <address>192.168.168.21</address>
            <password>admin</password>
            <port>8200</port>
            <username>admin</username>
      </Camera1>
      <Camera2 type="ipcamera">
            <address>192.168.168.22</address>
            <password>admin</password>
```

Figure A.1: Example of configuration file

In the *\data* folder, there are data of 20 subjects each is stored in a separate folder which contains 5 video files (from 5 IP cameras), Wii Remote's acceleration data log files: bigknife.dat, knife.dat, smallknife.dat, spoon.data, and a configuration file named index.xml. Example content of a configuration file is shown in the figure A.1.

```
<el index="1" start="6.84" end="9.2">
      <attribute name="action">dicing</attribute>
      <comment>Andrew is using the bigknife to slice carrot</comment>
</el>
<el index="2" start="11.12" end="18.2">
      <attribute name="action">dicing</attribute>
      <comment>Andrew is using the bigknife to slice garlic</comment>
</el>
<el index="3" start="31.52" end="49.6">
      <attribute name="action">chopping</attribute>
      <comment>Andrew is using the knife to chop potato</comment>
</el>
<el index="4" start="56.96" end="68.64">
      <attribute name="action">spreading</attribute>
      <comment>Andrew is using the smallknife to spread butter</comment>
</el>
<el index="5" start="73.64" end="77.88">
      <attribute name="action">spreading</attribute>
      <comment>Andrew is using the smallknife to spread butter</comment>
</el>
<el index="6" start="79.44" end="83.24">
      <attribute name="action">spreading</attribute>
      <comment>Andrew is using the smallknife to spread butter</comment>
</el>
<el index="7" start="105.8" end="114.6">
      <attribute name="action">coring</attribute>
      <comment>Andrew is using the smallknife to core pepper</comment>
</el>
<el index="8" start="119.56" end="120.76">
      <attribute name="action">coring</attribute>
      <comment>Andrew is using the smallknife to core pepper</comment>
</el>
```

Figure A.2: example of an annotation file

Annotation files are stored in the *\annotationfiles* folder. Each file is for each subject and is in Anvil format. Each entry in a file contains an entry number, start, end times of the activity and an activity label. Example content of an annotation file is shown in the figure A.2.

The folder *\features* contains features computed from the acceleration data. As Slice&Dice uses WEKA Toolkit for activity classification, all entries in the feature files are organized into WEKA file format (i.e. extension with .arff)

```
@relation kitchen
@attribute MeanX real
@attribute MeanY real
@attribute MeanZ real
@attribute MeanPitch real
@attribute MeanRoll real
@attribute StandardDeviationX real
@attribute StandardDeviationY real
@attribute StandardDeviationZ real
```

```
@attribute StandardDeviationPitch real
@attribute StandardDeviationRoll real
@attribute EnergyX real
@attribute EnergyY real
@attribute EnergyZ real
@attribute EnergyPitch real
@attribute EnergyRoll real
@attribute EntropyX real
@attribute EntropyY real
@attribute EntropyZ real
@attribute EntropyPitch real
@attribute EntropyRoll real

@attribute utensil {smallknife, knife, bigknife, spoon}
@attribute  class  {chopping,  scraping,  peeling,  slicing,  dicing,  coring,
spreading, stirring, scooping, shaving, eating}

@data
152.625,127.8125,125.28125,0.574439968995822,0.706056833735001,4.463392767839
28,2.40361056537868,7.15775093430192,0.0155684224098495,0.0230641931313793,23
314.3125,16341.8125,15746.625,0.330223653756252,0.499048209468696,13.18680351
24267,16.6259575367031,10.5299181861608,4.59210007120964,4.59210007120964,-
0.439114858512793,0.31548440720205,-0
0260434161368887,bigknife,chopping

152.4375,125.703125,122.84375,0.570647638619613,0.714483991030677,4.365614933
77508,4.30726017723274,6.99155461521256,0.0180018682042745,0.0228066239037854
,23256.25,15819.828125,15139.46875,0.325962794720984,0.511007515533012,14.215
3522974959,12.0381004737584,8.98957808647481,4.33216987849966,4.3321698784996
6,0.0554137996989744,0.333639333470558,0.070324807251755,bigknife,chopping

152.609375,125.734375,125.40625,0.56673234667041,0.710025860666836,3.30538002
495553,4.06679460501449,4.24528690873774,0.0123665921286412,0.016301583826574
3,23300.546875,15825.671875,15744.75,0.321338485363426,0.504402464450937,16.5
82294924312,13.42918282878,13.6770499023078,4.91414640475065,5.00078980232065
,0.16779977808857,0.720315327063313,0.299706422881269,bigknife,chopping
```

Figure A.3: example of a feature file

The feature files are stored into window sized folders. For example, *\Features\subject1\64* contains the files of features computed from the slicing window length of 64 samples. The code implemented in C# for feature computation is present in the figure A.4.

```
public  Feature  computeFeatureWindow(AccData[]  accdata,  int  head,  int  tail,
int N)
        {

            // compute the sums of X,Y,Z, pitch, roll, and angle

            sumX = 0;
            sumY = 0;
            sumZ = 0;
            sumX2 = 0;
            sumY2 = 0;
```

```
            sumZ2 = 0;
            sumPitch = 0;
            sumRoll = 0;
            sumPitch2 = 0;
            sumRoll2 = 0;

            for (i = head; i < tail; i++)
            {
                sumX = sumX + accdata[i].X;
                sumY = sumY + accdata[i].Y;
                sumZ = sumZ + accdata[i].Z;
                sumX2 = sumX2 + accdata[i].X * accdata[i].X;
                sumY2 = sumY2 + accdata[i].Y * accdata[i].Y;
                sumZ2 = sumZ2 + accdata[i].Z * accdata[i].Z;
                sumPitch = sumPitch + accdata[i].pitch;
                sumRoll = sumRoll + accdata[i].roll;
                sumPitch2 = sumPitch2 + accdata[i].pitch * accdata[i].pitch;
                sumRoll2 = sumRoll2 + accdata[i].roll * accdata[i].roll;
            }

            // Mean features
            meanX = sumX / N;
            meanY = sumY / N;
            meanZ = sumZ / N;
            meanPitch = sumPitch / N;
            meanRoll = sumRoll / N;


            // Energy features
            EnergyX = sumX2 / N;
            EnergyY = sumY2 / N;
            EnergyZ = sumZ2 / N;
            EnergyPitch = sumPitch2 / N;
            EnergyRoll = sumRoll2 / N;

            // standard deviation
            sumX1 = 0;
            sumY1 = 0;
            sumZ1 = 0;
            sumPitch1 = 0;
            sumRoll1 = 0;

            for (i = head; i < tail; i++)
            {
                sumX1 = sumX1 + (accdata[i].X - meanX) * (accdata[i].X -
meanX);
                sumY1 = sumY1 +(accdata[i].Y - meanY) * (accdata[i].Y -
meanY);
                sumZ1 = sumZ1 + (accdata[i].Z - meanZ) * (accdata[i].Z -
meanZ);
                sumPitch1 = sumPitch1 + (accdata[i].pitch - meanPitch) *
(accdata[i].pitch - meanPitch);
                sumRoll1 = sumRoll1 + (accdata[i].roll - meanRoll) *
(accdata[i].roll - meanRoll);
            }
            standarddeviationX = Math.Sqrt(sumX1 / N);
            standarddeviationY = Math.Sqrt(sumY1 / N);
```

```
            standarddeviationZ = Math.Sqrt(sumZ1 / N);
            standarddeviationPitch = Math.Sqrt(sumPitch1 / N);
            standarddeviationRoll = Math.Sqrt(sumRoll1 / N);

            // Entropy features
            double[] PX = new double[N];
            double[] PY = new double[N];
            double[] PZ = new double[N];
            double[] Ppitch = new double[N];
            double[] Proll = new double[N];
            for (i = head; i < tail; i++)
             {
                 PX[i] = 0;
                 PY[i] = 0;
                 PZ[i] = 0;
                 Ppitch[i] = 0;
                 Proll[i] = 0;
            }
            for (i = head; i < tail; i++)
             {
                 for (j = head; j < tail; j++)
                 {
                     if (accdata[i].X == accdata[j].X) PX[i]++;
                     if (accdata[i].Y == accdata[j].Y) PY[i]++;
                     if (accdata[i].Z == accdata[j].Z) PZ[i]++;
                     if (accdata[i].pitch == accdata[j].pitch) Ppitch[i]++;
                     if (accdata[i].roll == accdata[j].roll) Proll[i]++;
                 }

             }
            EntropyX = 0;
            EntropyY = 0;
            EntropyZ = 0;
            EntropyPitch = 0;
            EntropyRoll = 0;
            for (i = head; i < tail; i++)
            {
                PX[i] = PX[i]/N;
                PY[i] = PY[i] / N;
                PZ[i] = PZ[i] / N;
                Ppitch[i] = Ppitch[i] / N;
                Proll[i] = Proll[i] / N;
            }

            for (i = head; i < tail; i++)
            {
                if (PX[i] >0)
                    EntropyX = EntropyX - (PX[i] * Math.Log(PX[i]));
                if (PY[i] > 0)
                    EntropyY = EntropyY - (PY[i] * Math.Log(PY[i]));
                if (PZ[i] > 0)
                    EntropyZ = EntropyZ - (PZ[i] * Math.Log(PZ[i]));
                if (Ppitch[i] > 0)
                    EntropyPitch    =    EntropyPitch    -    (Ppitch[i]    *
Math.Log(Ppitch[i]));
                if (Proll[i] > 0)
                    EntropyRoll     =     EntropyRoll     -     (Proll[i]     *
```

```
Math.Log(Proll[i]));
            }

            // Correlations
            double sumXY, sumYZ, sumZX;
            sumXY = 0;
            sumX2 = 0;
            sumY2 = 0;
            sumYZ = 0;
            sumZ2 = 0;
            sumZX = 0;
            for (i = head; i < tail; i++)
            {
                sumXY = sumXY + (accdata[i].X * accdata[i].Y);
                sumYZ = sumYZ + (accdata[i].Y * accdata[i].Z);
                sumZX = sumZX + (accdata[i].Z * accdata[i].X);
                sumX2 = sumX2 + (accdata[i].X * accdata[i].X);
                sumY2 = sumY2 + (accdata[i].Y * accdata[i].Y);
                sumZ2 = sumZ2 + (accdata[i].Z * accdata[i].Z);
            }

            correlationXY = (N * sumXY - sumX * sumY);
            if ((sumX2 - sumX * sumX != 0) && (sumY2 - sumY * sumY != 0))
            {
                correlationXY = correlationXY / (Math.Sqrt(N * sumX2 - sumX *
sumX) * (Math.Sqrt(N * sumY2 - sumY * sumY)));
            }

            correlationYZ = (N * sumYZ - sumZ * sumY);
            if ((sumZ2 - sumZ * sumZ != 0) && (sumY2 - sumY * sumY != 0))
            {
                correlationYZ = correlationYZ / (Math.Sqrt(N * sumZ2 - sumZ *
sumZ) * (Math.Sqrt(N * sumY2 - sumY * sumY)));
            }

            correlationZX = (N * sumZX - sumZ * sumX);
            if ((sumZ2 - sumZ * sumZ != 0) && (sumX2 - sumX * sumX != 0))
            {
                correlationZX = correlationZX / (Math.Sqrt(N * sumZ2 - sumZ *
sumZ) * (Math.Sqrt(N * sumX2 - sumX * sumX)));
            }
        }
```

Figure A.4: Feature Computation code (implemented in C#)

The CBT-DTW system used the Slice&Dice's dataset with only acceleration value features extracted from acceleration signals. Dataset for CBT-DTW's experiments with pre-processing from Slice&Dice's dataset is stored in: http://di.ncl.ac.uk/publicweb/AmbientKitchen/KitchenData/CBTDTW_dataset/

In which, the segmented 64-sample frames which are stored in \features folder, and processed data (nicer format with fully annotated and re-organized) are stored in \processed_data. Notice that as CBT-DTW can deal with idle and unknown activities, there are a significant numbers of unknown and idle frames included in \features folder.

Appendix B: Data for KitchenSense's experiments

The root stores the data for KitchenSense's experiments is: http://di.ncl.ac.uk/publicweb/AmbientKitchen/KitchenData/KitchenSense_dataset/

In which, it has 5 sub-folders:

Docs contains guide documents used for annotation process, including annotation protocols, guides on how to annotate activities and food ingrdients on chopping board, and activity descriptions. The table A.1 presents some examples of activity descriptions, and the table A.2 shows the food descriptions for annotating the fiber chopping board:

Activity	**Utensils**	**Description**
Chopping (chop)	Most knives (Bread knife, chef knife, slicing knife, small knife)	*Chopping* is a distinct cutting activity where the knife moves up and down causing it to make contact with the chopping board and then break contact with the chopping board. Some very small forward and backward movements may be seen but these do not actually cut the ingredient. **Annotation:** Begins: The knife is positioned above the ingredient in preparation. Then the knife moves downwards and makes contact with the ingredient for the first time. Ends: The knife makes contact with chopping board it is lifted up again, breaking contact with the chopping board and returning to an approximate of the starting position for the final time. There are no pauses longer than 1 second during this period. **If the knife does not move in a clear up and down movement or does not break contact with the board this is *slicing* (see overleaf).** **See Figure 1 and videos chop_cour.avi, chop_mushroom.avi and chop_onion.avi for illustrative examples.** **Label example**: chop,onio

Slicing (slice)	Most knives (Bread knife, chef knife, slicing knife, small knife)	In this study, *slicing* is any cutting activity with the knife that *is not chopping*. If the knife remains in contact with the board or if the predominant movement is forwards and backwards then this cutting activity *is not chopping,* so we call it *slicing***.** This means a great variety of movements are all categorised as *slicing* in this study. Therefore, in essence; **Annotation**: Begins: The knife is placed in a preparatory position and makes contact with the ingredient for the first time. Ends: The knife loses contact with the ingredient for the final time. There are no pauses longer than 1 second during this period. Therefore, it is not until you have watched the movement that you can determine; • What the preparatory position is, • In what direction contact with the ingredient is made (downwards or forwards), • How contact with the ingredient is lost, Some video clips have been selected to illustrate the different knife movements we would label as *slicing* because the movement *is not chopping.* *slice_basil1 and slice_ham3*: The knife is moving forwards and backwards more than up and down. When moving forwards and backwards the knife remains in contact with the board whilst cutting the ingredient. *slice_cour1 and slice_mushroom1:* The knife is moving up and down through the ingredient, however the tip of the knife remains in contact with the board. **Label example**: slice,onio
Scraping (scrap)	Most knives (Bread knife, chef knife, slicing knife, small	When *scraping* the blade of a knife pushes ingredients towards a location on the chopping board, or to push ingredients off the chopping board into a container. **Annotation:** Begins: The knife is in contact with the chopping board and pushes the ingredients for the first time. Ends: The knife has been removed from being in contact with the chopping board after it has pushed the ingredients, for the final time. With no pauses longer than 1 second during this period. **Label example**: scrap

	knife)	
Peeling (peel)	Knives (Bread knife, chef knife, slicing knife, small knife, peeler)	*Peeling* describes an action that removes a layer of an ingredient. **Annotation:** Begins: The peeler or knife makes contact with the ingredient for the first time and begins to move along it, removing the outer layer. Ends: The peeler or knife no longer has contact with the ingredient once it has reached the end of the ingredient and removed a piece of the outer layer. With no pauses longer than 1 second during this period. **Label example**: peel,cour
Stirring (stir)	Spoons (Spoon, slotted spoon, ladle, whisk, spatula)	*Stirring* is defined by using one of the spoons to stir ingredients inside a container. A s*tirring* movement can occur in circles, in "eights", backward and forward, sideways or in any other form that mixes/moves ingredients. **Annotation:** Begins: Spoon enters container and begins to stir for the first time. Ends: Spoon leaves container, or it is left in the container and the participant lets go of the utensil, for the final time. With no pauses longer than 1 second during this period. **Label example**: stir
Scooping (scoop)	Spoons (Spoon, slotted spoon, ladle)	*Scooping* is a movement in which ingredients are moved from one container into another. A spoon scoops out a portion of ingredients from container A. The spoon is used to move this portion of ingredients and add it to container B. Both containers may be on a flat surface and so are not angled or one may be angled to aid the scooping of ingredients into the spoon. **Annotation:** Begins: Spoon enters container A for the first time. Ends: The spoon's angle is changed (turned upside down) so the ingredients fall from the spoon into the new container B, for the final time. With no pauses longer than 1 second during this period. **Label example**: scoop
Whisking (whisk)	Whisk	Whisking is a quick light sweeping motions performed with a Whisk. **Annotation:** Begins: Whisk enters container for the first time and begins to whisk. Ends: Whisk leaves container, or it is left in the container and the participant lets go of the whisk, for the final time. With no pauses longer than 1

		second during this period. **Please note: This description of one whisk and how to annotate whisking is very similar to stirring. The difference is in the UTENSIL and the MOVEMENT.** **Label example**: whisk
Adding (add)	Containers (Colander, sieve, big bowl) Pans (Sauce pans, frying pans) Ingredient containers (salt)	Adding describes the movement of ingredients from container, ingredient container or pan A to container or pan B. Container A's angle increases until the ingredient falls out. In some cases the participants used a spoon to support the ingredients movement. **Annotation:** Begins: Container A's angle begins to increase for the first time. Ends: Container A's angle reduces until it returns to the original angle or the participant moves the container away or begins to put it down for the final time. With no pauses longer than 1 second during this period. **Please note: Container B receiving the ingredient is not annotated. If the ingredient is liquid it is likely the pouring label is most appropriate.** **Label example**: add or add,salt
Pouring (pour)	Containers (Colander, sieve, big bowl) Pans (Sauce pans, frying pans) Ingredient containers (Olive oil, soy sauce)	Pouring describes the movement of ingredients from container, ingredient container or pan A to container or pan B. The difference between pouring and adding is that when pouring the ingredient moving is either purely, or the majority of it, is liquid. Container A's angle increases until the ingredient slides out. In some cases the participants used a spoon to support this action. **Annotation:** Begins: Container A's angle begins to increase for the first time. Ends: Container A's angle reduces until it returns to the original angle or the participant moves the container away or begins to put it down for the final time. **Please note: Container B receiving the ingredient is not annotated. If the ingredient is not liquid based it is likely the adding label is most appropriate.** **Label example**: pour,mix
Draining	Containers	The contents of a sauce pan is moved to a sieve or colander to separate the

(drain)	(Colander, sieve)	ingredient from the water. This action is done over or in the sink. **Annotation:** Begins: All the ingredient has been transferred from the sauce pan into the sieve or colander. Ends: All the ingredient has been removed from the sieve or colander into another container. **Label example**: drain
Shaking (shake)	Pans (Sauce pan, frying pan)	*Shaking*, also described as throwing, is only performed with a frying pan or sauce pan without using another utensil. The user slightly lifts the pan and performs a backward-forward movement. This movement causes the ingredients to be shaken/mixed in the pan. With this movement the ingredients may be thrown out of the pan and have to be caught again. **Label example**: shake

Table B.1: Activity description

Utensil	**Description**
Chopping board	When annotating the *chopping board* focus is on what ingredients have been placed on it. The annotation begins when the ingredient is placed onto the board. The annotation ends when the ingredient is A) removed from the board in the same state as when it was first placed on the board, or B) when it begins to be cut or sliced. Once the ingredient has begun to be chopped or sliced it is no longer a whole ingredient, therefore it is ignored as far as the annotations for the chopping board are concerned even though it may stay on the board for a significant period of time. The best way to annotate the chopping board is by watching the video and identify when the ingredient is placed on the board. Annotate a small annotation, only 1 or 2 seconds, to highlight when the ingredient is placed on the board. Next, continue to watch the video and create an annotation to highlight either; A) the ingredient is removed, or B) has begun to be chopped or sliced. If the ingredient has been removed then merge these two annotations by using the 'Merge with annotation before' function as discussed earlier. If the ingredient has begun to be chopped or sliced, create a second annotation just before this event to highlight when the ingredient was still whole. Then merge this annotation with the annotation highlighting when the ingredient was placed on the chopping board (both these processes are illustrated below). A) Ingredient is placed onto chopping board and then removed:

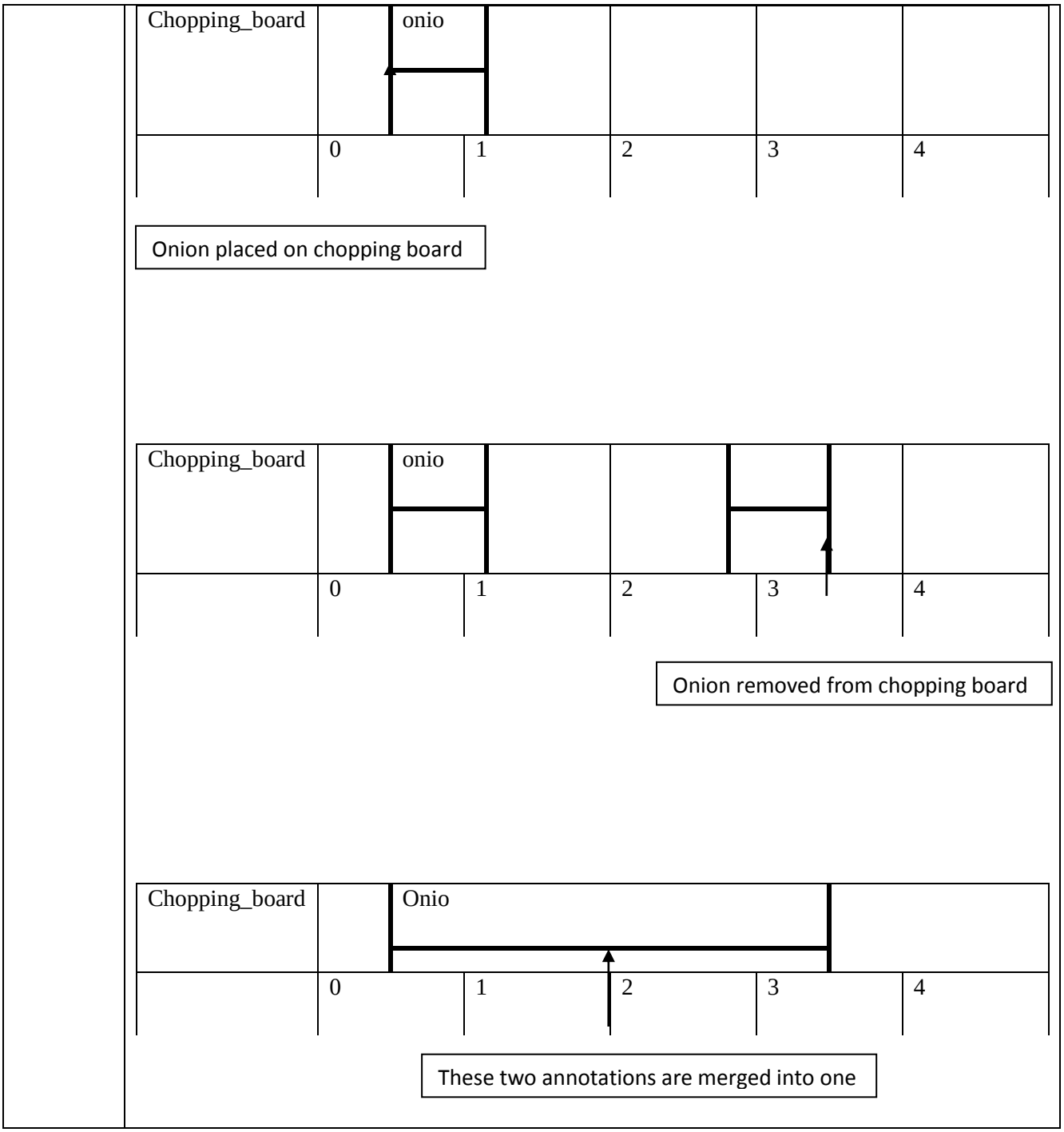


Table B.2: Food ingredient descriptions for annotating food processed on the FCB

Data-processing Codes contains the codes used for processing the experimental data, for example, codes for generating ELAN files, pre-processing data, inter-rater reliability, data extraction and segmentation, feature computation etc. The codes might have been written in C# or Matlab.

FCB-SampleImages contains sample images from fiber chopping board for KitchenSense's experiments. These images were randomly setected from 12 subjects. They are also used for training and evaluating the KitchenSense system.

data contains all data collected from 12 subjects. Each subject has 3 sessions. In each session, 5 folders contains *video*, *accelerometers*, *annotation*, *audio*, and *images*. In order to reduce sizes of data, the folders are stored in compressed files (i.e. accelerometers.zip, annotation.zip, audio.zip, and images.zip) except for the video file. Below are detail descriptions for each folder.

- *video:* contains a video file which is combined from 4 videos from digital cameras. This video file is the one that was synchronized with accelerometers, embedded camera, and microphone. The start-time of the video is also the absolute time for all other sensors.
- *annotation:* contains 2 ELAN-output files which have an extension of "EAF" and are in XML format (see Figure B.1), independently coded by two annotators and two .txt files which are converted from these two EAF files for event ground truth. For example, s1_s3_a1.eaf and s1_s3_a1.txt are files annotated by the annotator 1 for subject 1, session 1 (some entries were shown in the Figure B.1 and B.2).

```
<ALIGNABLE_ANNOTATION      ANNOTATION_ID="a600"      TIME_SLOT_REF1="ts300"
TIME_SLOT_REF2="ts333">
                <ANNOTATION_VALUE>chop,cour </ANNOTATION_VALUE>
            </ALIGNABLE_ANNOTATION>
        </ANNOTATION>
        <ANNOTATION>
            <ALIGNABLE_ANNOTATION                      ANNOTATION_ID="a601"
TIME_SLOT_REF1="ts339" TIME_SLOT_REF2="ts340">
                <ANNOTATION_VALUE>chop,cour </ANNOTATION_VALUE>
            </ALIGNABLE_ANNOTATION>
        </ANNOTATION>
        <ANNOTATION>
            <ALIGNABLE_ANNOTATION                      ANNOTATION_ID="a602"
TIME_SLOT_REF1="ts342" TIME_SLOT_REF2="ts344">
                <ANNOTATION_VALUE>chop,cour </ANNOTATION_VALUE>
            </ALIGNABLE_ANNOTATION>
        </ANNOTATION>
        <ANNOTATION>
            <ALIGNABLE_ANNOTATION                      ANNOTATION_ID="a603"
TIME_SLOT_REF1="ts351" TIME_SLOT_REF2="ts362">
                <ANNOTATION_VALUE>chop,cour </ANNOTATION_VALUE>
            </ALIGNABLE_ANNOTATION>
        </ANNOTATION>
        <ANNOTATION>
            <ALIGNABLE_ANNOTATION                      ANNOTATION_ID="a604"
TIME_SLOT_REF1="ts363" TIME_SLOT_REF2="ts368">
                <ANNOTATION_VALUE>chop,cour </ANNOTATION_VALUE>
            </ALIGNABLE_ANNOTATION>
        </ANNOTATION>
```

```
        <ANNOTATION>
            <ALIGNABLE_ANNOTATION                       ANNOTATION_ID="a605"
TIME_SLOT_REF1="ts384" TIME_SLOT_REF2="ts385">
                <ANNOTATION_VALUE>chop,chiv </ANNOTATION_VALUE>
            </ALIGNABLE_ANNOTATION>
        </ANNOTATION>
        <ANNOTATION>
            <ALIGNABLE_ANNOTATION                       ANNOTATION_ID="a606"
TIME_SLOT_REF1="ts396" TIME_SLOT_REF2="ts399">
                <ANNOTATION_VALUE>chop,onio </ANNOTATION_VALUE>
            </ALIGNABLE_ANNOTATION>
        </ANNOTATION>
        <ANNOTATION>
            <ALIGNABLE_ANNOTATION                       ANNOTATION_ID="a607"
TIME_SLOT_REF1="ts404" TIME_SLOT_REF2="ts405">
                <ANNOTATION_VALUE>chop,onio </ANNOTATION_VALUE>
            </ALIGNABLE_ANNOTATION>
        </ANNOTATION>
        <ANNOTATION>
            <ALIGNABLE_ANNOTATION                       ANNOTATION_ID="a608"
TIME_SLOT_REF1="ts406" TIME_SLOT_REF2="ts414">
                <ANNOTATION_VALUE>chop,onio </ANNOTATION_VALUE>
            </ALIGNABLE_ANNOTATION>
        </ANNOTATION>
```

Figure B.1: Example of Annotation file (ELAN format).

```
Chef_Knife_activity          331851      350679      chop,mush
Chef_Knife_activity          353749      400913      chop,mush
Chef_Knife_activity          510767      522525      chop,cour
Chef_Knife_activity          526099      554158      chop,cour
Chef_Knife_activity          580156      588341      chop,cour
Chef_Knife_activity          600821      612324      chop,cour
Chef_Knife_activity          639833      655126      chop,cour
Chef_Knife_activity          662451      670032      chop,cour
Chef_Knife_activity          721036      733470      chop,chiv
Chef_Knife_activity          805896      810795      chop,onio
Chef_Knife_activity          818368      823763      chop,onio
Chef_Knife_activity          959410      971800      slice,onio
Chef_Knife_activity          974790      977690       chop,onio
Chef_Knife_activity          1620997     1703080     slice,basi
Chef_Knife_activity          1704288     1705172     silce,ham
Chef_Knife_activity          1731562     1737400     chop,spag
Chef_Knife_activity          1740554     1743128     chop,spag
ChoppingBoard           141011      331860      mush
ChoppingBoard           509037      510758      cour
ChoppingBoard           712913      721028      chiv
ChoppingBoard           804354      805865      onio
ChoppingBoard           1618467     1621006     basi
```

Figure B.2: Example of Annotation file (text format).

In figure B.2, the first column is the tier of annotation which defines the activities of a specific utensil. The second column is the start time and the third is end time (in milliseconds) of the activity, and the fourth column is the activity label.

- *images* contains images of food ingredients captured from optical fiber sensing of FCB.
- *accelerometers* contains an original acceleration data log file (i.e. see file wax-2011-09-22-11-19-07.csv and sample content in the Figure B.3), accelerometer data file after synchronized (i.e. s11_s2.csv in the Figure B.4), the synchonisation information in startTime.txt, the map.txt file contains mapping information between sensor Ids and Utensils, the folder Utensil contains utensil's acceleration data files which were extracted from the synchonized acceleration data file and map.txt, and the Frame folder, which contains segmented frames for inputs and training data for the CBT-DTW activity recogtnition algorithm (see Figure B.4).

```
ACCEL,2011-09-22 10:19:46.473,305,113,-0.03125,0.96875,-0.1875
ACCEL,2011-09-22 10:19:46.493,305,114,0,0.9375,-0.125
ACCEL,2011-09-22 10:19:46.513,305,115,-0.03125,0.9375,-0.15625
ACCEL,2011-09-22 10:19:46.533,305,116,-0.03125,0.9375,-0.125
ACCEL,2011-09-22 10:19:46.553,305,117,0,0.9375,-0.15625
ACCEL,2011-09-22 10:19:46.573,305,118,-0.03125,0.9375,-0.125
ACCEL,2011-09-22 10:19:46.593,305,119,-0.03125,0.9375,-0.15625
ACCEL,2011-09-22 10:19:46.613,305,120,-0.03125,0.9375,-0.125
ACCEL,2011-09-22 10:19:46.633,305,121,-0.03125,0.9375,-0.125
ACCEL,2011-09-22 10:19:46.653,305,122,-0.0625,0.9375,-0.15625
ACCEL,2011-09-22 10:19:46.673,305,123,-0.03125,0.9375,-0.15625
ACCEL,2011-09-22 10:19:46.693,305,124,-0.03125,0.9375,-0.15625
ACCEL,2011-09-22 10:19:46.713,305,125,-0.03125,0.9375,-0.15625
ACCEL,2011-09-22 10:19:46.733,305,126,-0.03125,0.9375,-0.15625
ACCEL,2011-09-22 10:19:46.753,305,127,-0.03125,0.96875,-0.15625
ACCEL,2011-09-22 10:19:46.773,305,128,-0.03125,0.9375,-0.15625
ACCEL,2011-09-22 10:19:46.793,305,129,-0.03125,0.96875,-0.15625
ACCEL,2011-09-22 10:19:46.813,305,130,-0.03125,0.96875,-0.15625
ACCEL,2011-09-22 10:19:46.833,305,131,-0.0625,0.96875,-0.125
ACCEL,2011-09-22 10:19:46.853,305,132,-0.03125,0.96875,-0.15625
ACCEL,2011-09-22 10:19:46.582,103,118,0.03125,0.25,-1.0625
ACCEL,2011-09-22 10:19:46.602,103,119,0.03125,0.25,-1.03125
ACCEL,2011-09-22 10:19:46.622,103,120,0.03125,0.25,-1.03125
ACCEL,2011-09-22 10:19:46.642,103,121,0.03125,0.25,-1.03125
ACCEL,2011-09-22 10:19:46.662,103,122,0.0625,0.21875,-1.03125
ACCEL,2011-09-22 10:19:46.682,103,123,0.03125,0.25,-1.03125
ACCEL,2011-09-22 10:19:46.702,103,124,0.03125,0.25,-1.03125
ACCEL,2011-09-22 10:19:46.722,103,125,0.03125,0.25,-1.03125
ACCEL,2011-09-22 10:19:46.742,103,126,0.03125,0.25,-1.03125
ACCEL,2011-09-22 10:19:46.762,103,127,0.03125,0.25,-1.03125
```

Figure B.3: the content of accelerometer log file.

The first column of accelerometer log file is marked “ACCEL”, the second column is the timestamp which is in *yyyy-mm-dd hh:mm:ss.ms* format, the third column is the sensor Id, the fourth column is the row number for each sensor, and the rest 3 columns are X,Y, and Z acceleration values.

Figure B.4: The content of accelerometers folder.

It is noticed that the data in the Utensil and Frame folders were processed and ready for training and testing the activity recognition algorithm. Example of the data is shown in the Figure B.5:

```
43.8499999999985,-1,-0.09375,-0.375,1
43.8700000000026,-0.96875,-0.09375,-0.34375,1
43.8899999999994,-0.96875,-0.09375,-0.3125,1
43.9099999999962,-0.96875,-0.0625,-0.3125,1
43.9300000000003,-0.96875,-0.09375,-0.34375,1
43.9490000000005,-0.96875,-0.09375,-0.3125,1
43.9700000000012,-0.71875,0.09375,-0.375,1
43.989999999998,-1,0,-0.3125,1
44.010000000002,-1,0.03125,-0.28125,1
44.010000000002,-1,-0.03125,-0.40625,1
44.0299999999988,-0.96875,0.03125,-0.3125,1
44.0500000000029,-0.96875,0,-0.28125,1
44.0699999999997,-1,0,-0.28125,1
44.0899999999965,-0.96875,0.03125,-0.25,1
44.1100000000006,-1,0,-0.34375,1
44.1299999999974,-1.0625,-0.0625,-0.34375,1
44.1500000000015,-1,-0.03125,-0.28125,1
44.1979999999967,-0.9375,0.0625,-0.3125,1
```

Figure B.5: Example of utensil and frame data.

The first column is the absolute time stamp, followed by X, Y, Z acceleration values, then sampling bit. If sampling bit = 1, the sample is from sensor. otherwise, the sample is resampled.

- *audio* contains an audio log file, frames segmented from chopping food audio data, and frames segmented from slicing food audio data, and their feature files (see Figure B.6).

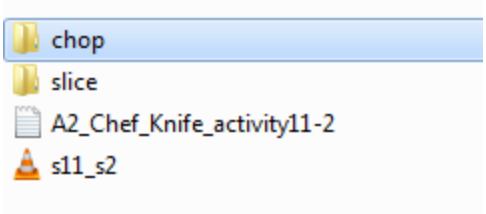


Figure B.6: audio folder

In chop and slice folders, there is a file named features_session_removesilence.txt which contains features computed from chopping and slicing food audio data after segmented and removed silence (see Figure B.7). The values from left to right order are MFCC1,…, MFCC13, pitch, Energy_Entropy, Zero-crossing rate, SpectralRollOff, and ShortTimeEnergy, respectively.

```
-16.257397,1.056199,0.696328,0.672959,0.201935,0.538859,0.161169,-
0.023742,0.014536,0.115432,0.124885,0.139197,0.089206,136.868162,0.303315,0.0
16382,0.000845,0.130052,cour
-
16.368237,1.340869,0.830765,0.811322,0.151580,0.505590,0.191896,0.000198,0.05
1181,0.142693,0.091469,0.204422,0.161538,143.039448,0.241015,0.009009,0.00121
3,0.065860,cour
-
16.011404,1.281156,0.873534,0.698709,0.202831,0.410811,0.306292,0.028607,0.01
0376,0.134441,0.148607,0.260930,0.084682,137.485793,0.149459,0.047463,0.00094
1,0.024518,cour
-16.297820,1.235793,0.600426,0.609774,0.080476,0.457046,0.287025,-0.103231,-
0.020666,0.164749,0.175675,0.204826,0.088676,148.185773,0.102286,0.022942,0.0
01124,0.014382,cour
-16.311679,1.569056,0.685693,0.674889,0.218303,0.374531,0.189603,-
0.115052,0.000028,0.150939,0.145680,0.269995,0.092848,141.712543,0.219090,0.0
36296,0.000827,0.051404,cour
-16.039122,1.615481,0.795496,0.784691,0.218628,0.325293,0.123262,-0.098830,-
0.027929,0.112514,0.115311,0.189310,0.149675,149.967465,0.251346,0.012411,0.0
01408,0.072512,cour
-16.031185,1.613834,0.958423,0.804143,0.121936,0.309499,0.317905,-0.077206,-
0.169446,0.102259,0.105967,0.202647,0.053469,145.110115,0.161256,0.008933,0.0
00670,0.029687,cour
```

Figure B.7: Audio feature file of chopping activity

References

[1] Mark Weiser. 1995. The Computer for the 21st Century. *Human-computer interaction*, Ronald M. Baecker, Jonathan Grudin, William A. S. Buxton, and Saul Greenberg (Eds.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA 933-940.

[2] Jiehan Zhou, Ekaterina Gilman, Mika Ylianttila, and Jukka Riekki. 2010. Pervasive Service Computing: Visions and Challenges. In *Proceedings of the 2010 10th IEEE International Conference on Computer and Information Technology* (CIT '10). IEEE Computer Society, Washington, DC, USA, 1335-1339.

[3] Vince Stanford. 2002. Using Pervasive Computing to Deliver Elder Care. *IEEE Pervasive Computing* 1, 1 (January 2002), 10-13.

[4] Stephen S. Intille, Kent Larson, Emmanuel Munguia Tapia, Jennifer S. Beaudin, Pallavi Kaushik, Jason Nawyn, and Randy Rockinson. 2006. Using a live-in laboratory for ubiquitous computing research. In *Proceedings of the 4th International Conference on Pervasive Computing* (Pervasive'06), Kenneth P. Fishkin, Bernt Schiele, Paddy Nixon, and Aaron Quigley (Eds.). Springer-Verlag, Berlin, Heidelberg, 349-365.

[5] Nishkam Ravi, Nikhil Dandekar, Preetham Mysore, and Michael L. Littman. 2005. Activity recognition from accelerometer data. In *Proceedings of the 17th Conference on Innovative Applications of Artificial Intelligence - Volume 3* (IAAI'05), Bruce Porter (Ed.), Vol. 3. AAAI Press 1541-1546.

[6] Tam Huynh, Ulf Blanke, and Bernt Schiele. 2007. Scalable recognition of daily activities with wearable sensors. In *Proceedings of the 3rd International Conference on Location-and Context-Awareness* (LoCA'07), Jeffrey Hightower, Bernt Schiele, and Thomas Strang (Eds.). Springer-Verlag, Berlin, Heidelberg, 50-67.

[7] Ling Bao, Stephen S. Intille. 2004. Activity recognition from user-annotated acceleration data. In *Proceedings of the 2nd International Conference on Pervasive Computing* (Pervasive'2004), Alois Ferscha, Friedemann Mattern (Eds.). Springer-Verlag, Berlin, Heidelberg, 1-17.

[8] Liang Wang, Tao Gu, Xianping Tao, Hanhua Chen, and Jian Lu. 2011. Recognizing multi-user activities using wearable sensors in a smart home. *Pervasive Mob. Comput.* 7, 3 (June 2011), 287-298.

[9] Kenneth P. Fishkin, Bing Jiang, Matthai Philipose, Sumit Roy. 2004. I Sense a Disturbance in the Force: Unobtrusive Detection of Interactions with RFID-tagged Objects. In *Proceedings of the 6th International Conference on Ubiquitous Computing* (Ubicomp'2004), Nigel Davies, Elizabeth D. Mynatt, Itiro Siio (Eds.). Springer-Verlag, Berlin, Heidelberg, 268-282.

[10] Jianxin Wu, Adebola Osuntogun, Tanzeem Choudhury, Matthai Philipose, James M. Rehg. 2007. A Scalable Approach to Activity Recognition based on Object Use. In *Proceedings of the 11th International Conference on Computer vision* (ICCV'2007). IEEE Press, Rio de Janeiro, Brazil, 1-8.

[11] Michael Buettner, Richa Prasad, Matthai Philipose, and David Wetherall. 2009. Recognizing daily activities with RFID-based sensors. In Proceedings of the *11th International Conference on Ubiquitous Computing* (Ubicomp '09). ACM, New York, NY, USA, 51-60.

[12] Paul Lukowicz, Gerald Pirkl, David Bannach, Florian Wagner, Alberto Calatroni, Kilian Förster, Thomas Holleczek, Mirco Rossi, Daniel Roggen, Gerhard Tröster, Jakob Doppler, Clemens Holzmann, Andreas Riener, Alois Ferscha, Ricardo Chavarriaga. 2010. Recording a Complex, Multi Modal Activity Data Set for Context Recognition. In *Proceedings of the 1st Workshop on Context systems Design, Evaluation and Optimisation (*ARCS Workshop'2010). Hannover, Germany, 161-166.

[13] Ekaterina H. Spriggs, Fernando De la Torre Frade, and Martial Hebert. 2009. Temporal Segmentation and Activity Classification from First-person Sensing. In *Proceedings of the IEEE Workshop on Egocentric Vision* (CVPR'2009). IEEE Computer Society, Miami Beach, FL, USA.

[14] Donald J. Patterson, Dieter Fox, Henry Kautz, and Matthai Philipose. 2005. Fine-Grained Activity Recognition by Aggregating Abstract Object Usage. In *Proceedings of the 9th IEEE International Symposium on Wearable Computers* (ISWC'05). IEEE Computer Society, Washington, DC, USA, 44-51.

[15] Kranz, M., Schmidt, A., Rusu, B., R., Maldonado, A., Beetz, M., Hornler, B., Rigoll, G.: Sensing Technologies and the Player-Middleware for Context-Awareness in Kitchen Environments. In *Proceedings of 4th International Conference on Networked Sensing Systems* (INSS'2007). IEEE Computer Society, Braunschweig, Germany, 179-186.

[16] Cuong Pham and Patrick Olivier. 2009. Slice&Dice: Recognizing Food Preparation Activities Using Embedded Accelerometers. In *Proceedings of the 3rd European Conference on Ambient Intelligence* (AmI'09), Manfred Tscheligi, Boris Ruyter, Panos Markopoulus, Reiner Wichert, Thomas Mirlacher, Alexander Meschterjakov, and Wolfgang Reitberger (Eds.). Springer-Verlag, Berlin, Heidelberg, 34-43.

[17] Jamie A. Ward, Paul Lukowicz, Gerhard Troster, and Thad E. Starner. 2006. Activity Recognition of Assembly Tasks Using Body-Worn Microphones and Accelerometers. *IEEE Trans. Pattern Anal. Mach. Intell.* 28, 10 (October 2006), 1553-1567.

[18] Matthai Philipose, Kenneth P. Fishkin, Mike Perkowitz, Donald J. Patterson, Dieter Fox, Henry Kautz, and Dirk Hahnel. 2004. Inferring Activities from Interactions with Objects. *IEEE Pervasive Computing* 3, 4 (October 2004), 50-57.

[19] Emmanuel Munguia Tapia, Stephen S. Intille, Kent Larson. 2004. Activity Recognition in the Home Using Simple and Ubiquitous Sensors. In *Proceedings of the 2nd International Conference on Pervasive Computing* (Pervasive'2004), Alois Ferscha, Friedemann Mattern (Eds.). Springer-Verlag, Berlin, Heidelberg, 158-175.

[20] Stephen S. Intille, Kent Larson, Emmanuel Munguia Tapia, Jennifer S. Beaudin, Pallavi Kaushik, Jason Nawyn, and Randy Rockinson. 2006. Using a live-in laboratory for ubiquitous computing research. In *Proceedings of the 4th International Conference on Pervasive Computing* (Pervasive'06), Kenneth P. Fishkin, Bernt Schiele, Paddy Nixon, and Aaron Quigley (Eds.). Springer-Verlag, Berlin, Heidelberg, 349-365.

[21] Tam Huynh, Ulf Blanke, and Bernt Schiele. 2007. Scalable recognition of daily activities with wearable sensors. In *Proceedings of the 3rd international conference on Location-and context-awareness* (LoCA'07), Jeffrey Hightower, Bernt Schiele, and Thomas Strang (Eds.). Springer-Verlag, Berlin, Heidelberg, 50-67.

[22] Andreas Zinnen, Kristof Van Laerhoven, and Bernt Schiele. 2007. Toward recognition of short and non-repetitive activities from wearable sensors. In *Proceedings of the 2007 European Conference on Ambient Intelligence* (AmI'07), Bernt Schiele, Alejandro Buchmann, Anind K. Dey, Hans Gellersen, and Boris De Ruyter (Eds.). Springer-Verlag, Berlin, Heidelberg, 142-158.

[23] Jonathan Lester, Tanzeem Choudhury, and Gaetano Borriello. 2006. A practical approach to recognizing physical activities. In *Proceedings of the 4th International Conference on Pervasive Computing* (Pervasive'06), Kenneth P. Fishkin, Bernt Schiele, Paddy Nixon, and Aaron Quigley (Eds.). Springer-Verlag, Berlin, Heidelberg, 1-16.

[24] Oliver Amft, Mathias Stäger, Paul Lukowicz, and Gerhard Tröster. 2005. Analysis of chewing sounds for dietary monitoring. In *Proceedings of the 7th International Conference on Ubiquitous*

Computing (UbiComp'05), Michael Beigl, Stephen Intille, Jun Rekimoto, and Hideyuki Tokuda (Eds.). Springer-Verlag, Berlin, Heidelberg, 56-72.

[25] Tapia E. M.:Using Machine Learning for Real-time Activity Recognition and Estimation of Energy Expenditure. Ph.D. Thesis, Massachusetts Institute of Technology, 2008.

[26] Jesse Hoey, Thomas Plötz, Dan Jackson, Andrew Monk, Cuong Pham, and Patrick Olivier. 2011. Rapid specification and automated generation of prompting systems to assist people with dementia. *Pervasive Mob. Comput.* 7, 3 (June 2011), 299-318.

[27] Patrick Olivier, Guangyou Xu, Andrew Monk, and Jesse Hoey. 2009. Ambient kitchen: designing situated services using a high fidelity prototyping environment. In *Proceedings of the 2nd International Conference on PErvasive Technologies Related to Assistive Environments* (PETRA '09). ACM, New York, NY, USA, , Article 47 , 7 pages.

[28] Juergen Wagner, Aart van Halteren, Jettie Hoonhout, Thomas Plötz, Cuong Pham, Paula Moynihan, Daniel Jackson, Cassim Ladha, Karim Ladha, Patrick Olivier. 2011. Towards a Pervasive Kitchen Infrastructure for Measuring Cooking Competence. In *Proceeding of the 5th International ICST Conference on Pervasive Computing Technologies for Healthcare* (PervasiveHealth'2011). IEEE Computer Society, Dublin, Ireland, 107 – 114.

[29] Keng-hao Chang, Shih-yen Liu, Hao-hua Chu, Jane Yung-jen Hsu, Cheryl Chen, Tung-yun Lin, Chieh-yu Chen, and Polly Huang. 2006. The diet-aware dining table: observing dietary behaviors over a tabletop surface. In *Proceedings of the 4th International Conference on Pervasive Computing* (Pervasive'06), Kenneth P. Fishkin, Bernt Schiele, Paddy Nixon, and Aaron Quigley (Eds.). Springer-Verlag, Berlin, Heidelberg, 366-382.

[30] Jen-Hao Chen, Peggy Pei-Yu Chi, Hao-Hua Chu, Cheryl Chia-Hui Chen, and Polly Huang. 2010. A Smart Kitchen for Nutrition-Aware Cooking. *IEEE Pervasive Computing* 9, 4 (October 2010), 58-65.

[31] Chia-Hsun Lee, Leonardo Bonnani, and Ted Selker. 2005. Augmented reality kitchen: enhancing human sensibility in domestic life. In *ACM SIGGRAPH 2005 Posters* (SIGGRAPH '05), Juan Buhler (Ed.). ACM, New York, NY, USA, Article 60.

[32] Reiko Hamada, Jun Okabe, Ichiro Ide, Shin'ichi Satoh, Shuichi Sakai, and Hidehiko Tanaka. 2005. Cooking navi: assistant for daily cooking in kitchen. In *Proceedings of the 13th Anual ACM International Conference on Multimedia* (MULTIMEDIA '05). ACM, New York, NY, USA, 371-374.

[33] Jamie A. Ward. 2006. Activity monitoring: continuous recognition and performance evaluation. Ph.D thesis, ETH Zurich, Switzerland, 2006.

[34] The Quality of Life Technology Centre (QoLT) project, http://kitchen.cs.cmu.edu.

[35] Geeta Shroff, Asim Smailagic, and Daniel P. Siewiorek. 2008. Wearable context-aware food recognition for calorie monitoring. In *Proceedings of the 2008 12th IEEE International Symposium on Wearable Computers* (ISWC '08). IEEE Computer Society, Washington, DC, USA, 119-120.

[36] Lee Kok-Meng., Li Q Qiang, and Daley Wayne. 2007. Effects of Classification Methods on Color-Based Feature Detection With Food Processing Applications. *IEEE Transactions on Automation Science and Engineering* 4(1) (2007) 40-51

[37] Aono, T., Kimura, H., Yamauchi, Y.: A Food Recognition Algorithm based on Dish Recognition. 2002. In *Proceedings of the 28th IEEE Conference on Industrial Electronics Society* (IECON'2002). IEEE Press, Meliá Lebreros Hotel, Sevilla, Spain, 1145-1150.

[38] Qing Wang and Jie Yang. 2009. Drinking activity analysis from fast food eating video using generative models. In *Proceedings of the ACM Multimedia 2009 Workshop on Multimedia for Cooking and Eating Activities* (CEA'09). ACM, New York, NY, USA, 31-38.

[39] Keigo Kitamura, Toshihiko Yamasaki, and Kiyoharu Aizawa. 2009. FoodLog: capture, analysis and retrieval of personal food images via web. In *Proceedings of the ACM Multimedia 2009 Workshop on Multimedia for Cooking and Eating Activities* (CEA'09). ACM, New York, NY, USA, 23-30.

[40] Shulin Yang, Mei Chen, Dean Pomerleau, Rahul Sukthankar. 2010. Food recognition using statistics of pairwise local features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (CVPR'2010). IEEE Computer Society, San Francisco, CA, USA, 2249-2256.

[41] Wen Wu and Jie Yang. 2009. Fast food recognition from videos of eating for calorie estimation. In Proceedings of the 2009 *IEEE International Conference on Multimedia and Expo* (ICME'09). IEEE Computer Society, Piscataway, NJ, USA, 1210-1213.

[42] Joseph P. Wherton and Andrew F. Monk. 2008. Technological opportunities for supporting people with dementia who are living at home. *Int. J. Hum.-Comput. Stud.* 66, 8 (August 2008), 571-586.

[43] Wii Remote: http://en.wikipedia.org/wiki/Wii_Remote

[44] ADXL 330 accelerometer:
http://www.analog.com/en/mems-sensors/inertial-sensors/adxl330/products/product.html
[45] Weka toolkit: http://www.cs.waikato.ac.nz/ml/weka/

[46] Michael Kipp. 2001. Anvil- a generic annotation tool for multimodal dialogue. In *Proceedings of 7th European Conference on Speech Communication and Technology* (EUROSPEECH'01). Paul Dalsgaard, Børge Lindberg, Henrik Benner, Zheng-Hua Tan (Eds.). Aalborg, Denmark, 1367-1370.

[47] Giorgio Fumera, Fabio Roli, Giorgio Giacinto. 2000. Reject Option with Multiple Thresholds. Pattern Recognition 33(12) 2000 2099-2101.

[48] Xiaopeng Xi, Eamonn Keogh, Christian Shelton, Li Wei, and Chotirat Ann Ratanamahatana. 2006. Fast time series classification using numerosity reduction. In *Proceedings of the 23rd international conference on Machine learning* (ICML '06). ACM, New York, NY, USA, 1033-1040.

[49] C. S. Myers and L. R. Rabiner. 1981. A comparative study of several dynamic time-warping algorithms for connected word recognition. *The Bell System Technical Journal* 60(7) 1981 1389-1409

[50] Muzaffar Bashir and Jürgen Kempf. 2011. DTW Based Classification of Diverse Pre-Processed Time Series Obtained from Handwritten PIN Words and Signatures. *J. Signal Process. Syst.* 64, 3 (September 2011), 401-411.

[51] Sengul Vurgun, Matthai Philipose, and Misha Pavel. 2007. A statistical reasoning system for medication prompting. In *Proceedings of the 9th International Conference on Ubiquitous Computing* (UbiComp'07), John Krumm, Gregory D. Abowd, Aruna Seneviratne, and Thomas Strang (Eds.). Springer-Verlag, Berlin, Heidelberg, 1-18.

[52] Beth Logan, Jennifer Healey, Matthai Philipose, Emmanuel Munguia Tapia, and Stephen Intille. 2007. A long-term evaluation of sensing modalities for activity recognition. In *Proceedings of the 9th International Conference on Ubiquitous Computing* (UbiComp'07), John Krumm, Gregory D. Abowd, Aruna Seneviratne, and Thomas Strang (Eds.). Springer-Verlag, Berlin, Heidelberg, 483-500.

[53] Optical Fiber: http://en.wikipedia.org/wiki/Optical_fiber

[54] Daniel Jackson, Tom Bartindale, and Patrick Olivier. 2009. FiberBoard: compact multi-touch display using channeled light. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces* (ITS '09). ACM, New York, NY, USA, 25-28.

[55] Acrylic Plastics: http://www.enotes.com/acrylic-plastic-reference/acrylic-plastic

[56] Optical Fiber Manufacturer: http://www.moritexusa.com/files/POF.pdf

[57] Acrylic glass properties: http://www.kaysons.in/acrylic/physicalproperties.pdf

[58] Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. 2002. An Efficient k-Means Clustering Algorithm: Analysis and Implementation. *IEEE Trans. Pattern Anal. Mach. Intell.* 24, 7 (July 2002), 881-892.

[59] Mathematical morphology: http://en.wikipedia.org/wiki/Mathematical_morphology

[60] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. 2008. Speeded-Up Robust Features (SURF*). Comput. Vis. Image Underst.* 110, 3 (June 2008), 346-359.

[61] Duy-Nguyen Ta, Wei-Chao Chen, Natasha Gelfand, Kari Pulli. 2009. SURFTrac: Efficient Tracking and Continuous Object recognition using Local Feature Descriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (CVPR'2009). IEEE Computer Society, Miami, Florida, USA, 2937-2944.

[62] Ricardo Chincha and Yingli Tian. 2011. Finding objects for blind people on SURF features. In *Proceedings of the IEEE International Conference on Bioinformatics and Biomedicine Workshops* (BIBMW'2011). IEEE Computer Society, Atlanta, GA, USA, 526-257.

[63] David G. Lowe. 1999. Object Recognition from Local Scale-Invariant Features. In *Proceedings of the International Conference on Computer Vision-Volume 2* (ICCV'99), Vol. 2. IEEE Computer Society, Washington, DC, USA, 1150-1157.

[64] Hong Lu, A. J. Bernheim Brush, Bodhi Priyantha, Amy K. Karlson, and Jie Liu. 2011. SpeakerSense: energy efficient unobtrusive speaker identification on mobile phones. In *Proceedings of the 9th International Conference on Pervasive Computing* (Pervasive'11), Kent Lyons, Jeffrey Hightower, and Elaine M. Huang (Eds.). Springer-Verlag, Berlin, Heidelberg, 188-205.

[65 Jianfeng Chen, Alvin Harvey Kam, Jianmin Zhang, Ning Liu, and Louis Shue. 2005. Bathroom activity monitoring based on sound. *In Proceedings of the 3rd International Conference on Pervasive Computing* (Pervasive'05), Hans-W. Gellersen, Roy Want, and Albrecht Schmidt (Eds.). Springer-Verlag, Berlin, Heidelberg, 47-61.

[66] Xiaodan Zhuang, Xi Zhou, Mark A. Hasegawa-Johnson, and Thomas S. Huang. 2010. Real-world acoustic event detection. *Pattern Recogn. Lett.* 31, 12 (September 2010), 1543-1551.

[67] Open Movement WAX3: http://code.google.com/p/openmovement/

[68] Linear interpolation: http://en.wikipedia.org/wiki/Linear_interpolation

[69] Pham, C., Hooper, C., Lindsay, S., Jackson, D., Shearer, J., Wagner, J., Ladha, C., Ladha, K.,Plötz, T., Olivier,P.: The Ambient Kitchen: A Pervasive Sensing Environment for Situated Services. *Demonstration in the Designing Interactive Systems Conference* (DIS'2012). ACM, Newcastle Upon Tyne, United Kingdom.

[70] Cuong Pham, Thomas Plotz, and Patrick Olivier. 2010. A dynamic time warping approach to real-time activity recognition for food preparation. In *Proceedings of the First International Joint Conference on Ambient Intelligence* (AmI'10), Boris de Ruyter, Reiner Wichert, David V. Keyson, Panos Markopoulos, Norbert Streitz, Monica Divitini, Nikolaos Georgantas, and Antonia Mana Gomez (Eds.). Springer-Verlag, Berlin, Heidelberg, 21-30.

[71] Hooper Clare J., Preston Anne, Balaam Madeline, Seedhouse Paul, Jackson Daniel, Pham Cuong, Ladha Cassim, Ladha Karim, Ploetz Thomas, Olivier Patrick. 2012. The French Kitchen: Task-Based Learning in an Instrumented Kitchen. In *Proceedings of the 14th International Conference on Ubiquitous Computing* (Ubicomp'2012). Pittsburgh, Pennsylvania, United States.

[72] Jamie A. Ward, Paul Lukowicz, and Hans W. Gellersen. 2011. Performance metrics for activity recognition. *ACM Trans. Intell. Syst. Technol.* 2, 1, Article 6 (January 2011), 23 pages.

[73] Language Archiving Technology, "Elan – linguistic annotator," 2010: http://www.lat-mpi.eu/tools/elan/

[74] L. Bergroth, H. Hakonen, and T. Raita. 2000. A Survey of Longest Common Subsequence Algorithms. In *Proceedings of the 7th International Symposium on String Processing Information Retrieval* (SPIRE'00). IEEE Computer Society, Washington, DC, USA, 39-48.

[75] Dementia 2010 report: http://www.dementia2010.org/

[76] Plötz Thomas, Moynihan Paul, Pham Cuong and Olivier Patrick. 2011. Activity Recognition and Healthier Food Preparation. *Activity Recognition in Pervasive Intelligent Environments*. Chen, L.; Nugent, C.D.; Biswas, J.; Hoey, J. (Eds.). 1st Edition, 2011, Atlantis Press

[77] Visalakshmi Suresh, Paul Ezhilchelvan, Paul Watson, Cuong Pham, Dan Jackson, and Patrick Olivier. 2011. Distributed event processing for activity recognition. In *Proceedings of the 5th ACM*

International Conference on Distributed Event-Based System (DEBS '11). ACM, New York, NY, USA, 371-372.

[78] Plötz Thomas, Pham Cuong, and Olivier Patrick.2011. Who is Cooking? Sensor-Based Actor Identification in the Kitchen. In *Proceedings of International Workshop on Frontiers in Activity Recognition using Pervasive Sensing* (IWFAR'2011). ACM, San Francisco, CA, USA, 12-17.

[79] Jesse Hoey, Thomas Plötz, Dan Jackson, Andrew Monk, Cuong Pham, and Patrick Olivier. 2011. SNAP: SyNdetic Assistance Processes. In *Proceedings of Annual Conference on Neural Information Processing Systems Workshop on Machine Learning for Assistive Technologies* (MLAT'2010). Whistler, British Columbia, Canada.

[80] Thomas Plötz, Nils Y. Hammerla, and Patrick Olivier. 2011. Feature learning for activity recognition in ubiquitous computing. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence - Volume 2* (IJCAI'11), Toby Walsh (Ed.), Vol. Two. AAAI Press 1729-1734.

[81] Hamidreza Bayati, Jose del R. Millán, and Ricardo Chavarriaga. 2011. Unsupervised Adaptation to On-body Sensor Displacement in Acceleration-Based Activity Recognition. In *Proceedings of the 2011 15th Annual International Symposium on Wearable Computers* (ISWC '11). IEEE Computer Society, Washington, DC, USA, 71-78.

[82] Takuya Maekawa and Shinji Watanabe. 2011. Unsupervised Activity Recognition with User's Physical Characteristics Data. In *Proceedings of the 15th Annual International Symposium on Wearable Computers* (ISWC'11). IEEE Computer Society, Washington, DC, USA, 89-96.

[83] T. L. M. van Kasteren, G. Englebienne, and B. J. A. Kröse. 2010. Transferring knowledge of activity recognition across sensor networks. In *Proceedings of the 8th International Conference on Pervasive Computing* (Pervasive'10), Patrik Floréen, Antonio Krüger, and Mirjana Spasojevic (Eds.). Springer-Verlag, Berlin, Heidelberg, 283-300.

[84] Takuya Maekawa, Yutaka Yanagisawa, Yasue Kishino, Katsuhiko Ishiguro, Koji Kamei, Yasushi Sakurai, and Takeshi Okadome. 2010. Object-based activity recognition with heterogeneous sensors on wrist. In *Proceedings of the 8th International Conference on Pervasive Computing* (Pervasive'10), Patrik Floréen, Antonio Krüger, and Mirjana Spasojevic (Eds.). Springer-Verlag, Berlin, Heidelberg, 246-264.

[85] Oliver Amft. 2011. Self-Taught Learning for Activity Spotting in On-body Motion Sensor Data. In *Proceedings of the 15th Annual International Symposium on Wearable Computers* (ISWC '11). IEEE Computer Society, Washington, DC, USA, 83-86.

[86] Andreas Bulling, Jamie A. Ward, Hans Gellersen, and Gerhard Troster. 2009. Eye movement analysis for activity recognition. In *Proceedings of the 11th International Conference on Ubiquitous Computing* (Ubicomp'09). ACM, New York, NY, USA, 41-50.

[87] Fahd Albinali, Matthew S. Goodwin, and Stephen S. Intille. 2009. Recognizing stereotypical motor movements in the laboratory and classroom: a case study with children on the autism spectrum. In *Proceedings of the 11th International Conference on Ubiquitous Computing* (Ubicomp'09). ACM, New York, NY, USA, 71-80.

[88] Fahd Albinali, Stephen Intille, William Haskell, and Mary Rosenberger. 2010. Using wearable activity type detection to improve physical activity energy expenditure estimation. In *Proceedings of the 12th ACM International Conference on Ubiquitous Computing* (Ubicomp'10). ACM, New York, NY, USA, 311-320.

[89] James Fogarty, Carolyn Au, and Scott E. Hudson. 2006. Sensing from the basement: a feasibility study of unobtrusive and low-cost home activity recognition. In *Proceedings of the 19th annual ACM Symposium on User Interface Software and Technology* (UIST '06). ACM, New York, NY, USA, 91-100.

[90] Tam Huynh, Mario Fritz, and Bernt Schiele. 2008. Discovery of activity patterns using topic models. In *Proceedings of the 10th International Conference on Ubiquitous Computing* (UbiComp'08). ACM, New York, NY, USA, 10-19.

[91] Andreas Bulling, Jamie A. Ward, Hans Gellersen, and Gerhard Troster. 2009. Robust Recognition of Reading Activity in Transit Using Wearable Electrooculography. In *Proceedings of the 6th International Conference on Pervasive Computing* (Pervasive'08), Jadwiga Indulska, Donald J. Patterson, Tom Rodden, and Max Ott (Eds.). Springer-Verlag, Berlin, Heidelberg, 19-37.

[92] BCM2042: http://www.broadcom.com/products/Bluetooth/Bluetooth-RF-Silicon-and-Software-Solutions/BCM2042

[93] Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.* 2, 3, Article 27 (May 2011), 27 pages.

[94] Wenming Zheng, Cairong Zou, and Li Zhao. 2005. An Improved Algorithm for Kernel Principal Component Analysis. *Neural Process. Lett.* 22, 1 (August 2005), 49-56.

[95] Matsuyama, Y. 2011. Hidden Markov model estimation based on alpha-EM algorithm: Discrete and continuous alpha-HMMs. In *Proceddings of International Joint Conference on Neural Networks* (IJCNN'2011). IEEE Computer Society, San Jose, California 808-816.

[96] Corinna Cortes and Vladimir Vapnik. 1995. Support-Vector Networks. *Mach. Learn.* 20, 3 (September 1995), 273-297.

[97] Norbert A. Streitz. 2006. From human-computer interaction to human-environment interaction: ambient intelligence and the disappearing computer. In Proceedings of the 9th conference on User interfaces for all (ERCIM'06), Constantine Stephanidis and Michael Pieper (Eds.). Springer-Verlag, Berlin, Heidelberg, 3-13.

[98] Yuchun Tang, Yan-Qing Zhang, Nitesh V. Chawla, and Sven Krasser. 2009. SVMs modeling for highly imbalanced classification. Trans. Sys. Man Cyber. Part B 39, 1 (February 2009), 281-288.

[99] Stan Salvador and Philip Chan. 2007. Toward accurate dynamic time warping in linear time and space. Intell. Data Anal. 11, 5 (October 2007), 561-580.