Multi-Modal Recognition of Manipulation Activities through Visual Accelerometer Tracking, Relational Histograms, and User-Adaptation

Sebastian Stein

Doctor of Philosophy

University of Dundee

Scotland

September 2014

Contents

List of Figures

List of Tables

List of Symbols

$\mathbf{a}$	Proper 3D translational acceleration (relative to free fall).
$\mathbf{a}'$	Coordinate 3D translational acceleration (relative to the stationary camera).
α	Temporal decay.
γ_{RBF}	Scaling parameter for the Gaussian RBF kernel.
γ_{χ^2}	Scaling parameter for the exponential χ^2 kernel.
$\mathcal{C}$	Codebook for bag-of-words representation.
c_x, c_y	Principal point (camera intrinsic parameter).
d_G	Distance between points on a regular grid.
d_I^{min}	Minimum distance between locations of two trajectories in the most recent frame.
f	Focal length (camera intrinsic parameter).
$\mathbf{f}$	Feature vector.
f_{acc}	Accelerometer sampling rate.
f_{vid}	Video frame rate.
$\hat{\mathbf{g}}$	Estimated 3D vector of gravitational acceleration.
k_1, k_2, k_3	Radial distortion coefficients.
$\mathcal{M}$	Similarity map consisting of pairs of location hypotheses and similarity scores.

N_T^{max}	Maximum number of active trajectories.
P	Point trajectory as a sequence of 2D image locations.
P'	Point trajectory as a sequence of 3D world coordinates.
p_1, p_2	Tangential distortion coefficients (camera intrinsic parameters).
S	Similarity measure for pairs acceleration sequences.
s	Scale parameter in 3D-to-2D projection.
s_x, s_y	Dimensions of the image elements (camera intrinsic parameters).
T	Tracklet, a fixed length 2D point trajectory as l_1-normalized sequence of displacements.
t	Index of most recent video frame.
τ_a	Noise threshold for binary quantization of acceleration magnitudes.
τ_d	Distance threshold for trajectory termination.
τ_t	Temporal threshold for detecting accelerometer stationarity.
$\mathbf{v}'$	Velocity in 3D world coordinates.
$\mathbf{x}$	2D image location.
$\mathbf{x}'$	Point location in 3D world coordinates.
z	Estimated distance between the camera and an object in orthogonal direction to the view plane.

List of Publications

1. Sebastian Stein and Stephen J. McKenna.
   Accelerometer Localization and Relational Histograms for Multi-modal Activity Recognition (Under revision).
   IEEE Transactions on Pattern Analysis and Machine Intelligence,
   Submitted on February 10, 2014.

2. Sebastian Stein and Stephen J. McKenna.
   User-adaptive Models for Recognizing Food Preparation Activities
   ACM International Conference on Multimedia (ACMMM 2013) ,
   5th Workshop on Multimedia in Cooking and Eating Activities (CEA 2013),
   Barcelona, Spain, October 2013.

3. Sebastian Stein and Stephen J. McKenna.
   Combining Embedded Accelerometers with Computer Vision for Recognizing Food Preparation Activities.
   The 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp 2013), **Acceptance rate: 18%** excluding resubmissions,
   Zürich, Switzerland, September 2013.

4. Sebastian Stein and Stephen J. McKenna.
   Towards Recognizing Food Preparation Activities in Situational Support Systems, **Best Student Paper**.
   Digital Futures 2012 (3rd Annual Digital Economy All Hands Conference),
   Aberdeen, UK, October 2012.

5. Sebastian Stein and Stephen J. McKenna.
   Accelerometer Localization in the View of a Stationary Camera.
   The 9th Conference on Computer and Robot Vision (CRV 2012),
   Toronto, Ontario, Canada, May 2012.

Acknowledgements

I am particularly grateful to Stephen J. McKenna. Through his supervision and guidance, I have learned to think independently and critically, about my work and the work of others, while being confident about my research and appreciating the contributions of others.

Our collaborators at CultureLab, Newcastle University, particularly Patrick Olivier, Thomas Plötz, Dan Jackson, Nils Hammerla, Cuong Pham and Thomas Wagner, have been extremely helpful, supportive and inspiring. They introduced me to the area of pervasive computing, provided accelerometers and the know-how to analyze their data, and helped with pilot data collection. Jesse Hoey at Waterloo University was so kind to invite me to come to Waterloo, which was a fantastic visit and a great opportunity to discuss my work with other researchers in assistive technology. I would like to thank my examiners Ioannis Patras and Jianguo Zhang for the interesting discussion and their valuable comments, and my mock-examiners Emanuele Trucco and Ruixuan Wang for their tough questions and useful feedback. It was a pleasure to be part of the growing and diverse Computer Vision and Image Processing group (CVIP) at the University of Dundee. I am also grateful to Vicki Hanson and the members of the SiDE Accessibility group for the opportunity to get involved in additional SiDE activities. Finally, I shall thank my friends and family for their moral support and welcome distraction.

The research presented in this thesis is funded by RCUK Digital Economy Research Hub EP/G066019/1 SIDE: Social Inclusion through the Digital Economy, from December 2010 until May 2014.

Declarations

Candidate's Declaration

I, Sebastian Stein, hereby declare that I am the author of this thesis; that I have consulted all references cited; that I have done all the work recorded by this thesis; and that it has not been previously accepted for a degree.

Supervisor's Declaration

I, Stephen J. McKenna, hereby declare that I am the supervisor of the candidate, and that the conditions of the relevant Ordinance and Regulations have been fulfilled.

Abstract

Activity recognition research in computer vision and pervasive computing has made a remarkable trajectory from distinguishing full-body motion patterns to recognizing complex activities. Manipulation activities as occurring in food preparation are particularly challenging to recognize, as they involve many different objects, non-unique task orders and are subject to personal idiosyncrasies. Video data and data from embedded accelerometers provide complementary information, which motivates an investigation of effective methods for fusing these sensor modalities.

This thesis proposes a method for multi-modal recognition of manipulation activities that combines accelerometer data and video at multiple stages of the recognition pipeline. A method for accelerometer tracking is introduced that provides for each accelerometer-equipped object a location estimate in the camera view by identifying a point trajectory that matches well the accelerometer data. It is argued that associating accelerometer data with locations in the video provides a key link for modelling interactions between accelerometer-equipped objects and other visual entities in the scene. Estimates of accelerometer locations and their visual displacements are used to extract two new types of features: (i) *Reference Tracklet Statistics* characterizes statistical properties of an accelerometer's visual trajectory, and (ii) *RETLETS*, a feature representation that encodes relative motion, uses an accelerometer's visual trajectory as a reference frame for dense tracklets. In comparison to a traditional sensor fusion approach where features are extracted from each sensor-type independently and concatenated for classification, it is shown that

combining *RETLETS* and *Reference Tracklet Statistics* with those sensor-specific features performs considerably better. Specifically addressing scenarios in which a recognition system would be primarily used by a single person (e.g., cognitive situational support), this thesis investigates three methods for adapting activity models to a target user based on user-specific training data. Via randomized control trials it is shown that these methods indeed learn user idiosyncrasies.

All proposed methods are evaluated on two new challenging datasets of food preparation activities that have been made publicly available. Both datasets feature a novel combination of video and accelerometers attached to objects. The *Accelerometer Localization* dataset is the first publicly available dataset that enables quantitative evaluation of accelerometer tracking algorithms. The *50 Salads* dataset contains 50 sequences of people preparing mixed salads with detailed activity annotations.

Chapter 1

Introduction

1.1 Complex Manipulation Activities

Activity recognition research in computer vision has made a remarkable trajectory from distinguishing full-body motion patterns like *running, boxing* and *waving* [88] through detecting actions of interest in movies [55, 56, 59] to reasoning about complex human-human [85] and human-object interactions [5, 32, 84], and tracking through multi-step processes [39]. In recent years these challenging problems have gained comparable interest in the ubiquitous computing community [38, 75, 77, 82] but the literature shows few examples of creative cross-fertilization and of methods for integrated activity recognition from video and embedded sensors [5, 18, 118]. This thesis argues that a multi-modal approach - integrating data from multiple sensor types - to recognizing complex activities can be highly beneficial if the complementary information provided by different sensor types is encoded effectively.

The terms *action*, *activity*, *manipulation activity* and *complex manipulation activity* are used extensively throughout this thesis. Therefore, it makes sense to define these terms informally here and use them consistently later on. The term *action* refers to the process of executing a bodily movement within a short (sub-second) period of time, e.g. running, waving or grabbing. Although there might be objects involved in executing these actions, this term exclusively focuses on the performed motion pattern. *Activity*, on the other hand, describes a short sequence of actions. For example, the activity *placing a tea-bag into a cup* involves grabbing a tea-bag, moving the tea-bag over a cup, and releasing the tea-bag into a cup. An activity that involves a person's interaction with one or more objects with the intent of manipulating them is referred to as a *manipulation activity*. Cutting a tomato into pieces and placing a tea-bag into a cup are examples of manipulation activities, whereas sitting down on a chair is not. A sequence of manipulation activities is called *complex manipulation activity*. The order of individual activities in a complex activity is not necessarily deterministic. One example of a complex manipulation activity would be *preparing a toasted sandwich*, as it involves activities such as toasting bread, spreading butter on a slice of bread, placing a slice of ham on top, and so on.

In a wide range of application areas the benefit of accurate activity recognition outweighs the cost of creating a sensor-rich environment. This includes, for example, augmented reality [37], cognitive situational support [38, 39], supervision of assembly tasks [5], and skill assessment [81]. In these contexts, manipulation activities involve a potentially large number of objects, complex interactions between hands, tools and manipulated objects, and constrained but non-unique orderings in which interactions may be performed.

The challenges of recognizing complex manipulation activities are well illustrated by food preparation tasks. Kitchen utensils are hard to recognize and track visually as objects are often partially occluded and object categories are defined in terms of affordances. While ingredients may initially be discriminated based on visual appearance, manipulation activities such as *peeling, cutting, mixing* and *frying* severely change their colour, shape and texture, even forming new materials from multiple components. Food preparation activities usually involve transforming one or more ingredients into a target state without specifying a particular technique or utensil that has to be used. As a potentially wide range of techniques and utensils may be employed for each activity, achieving good generalization is particularly challenging. Additionally, many individuals maintain unique sets of recipes and personal variations of common recipes. These characteristics, among others, make it hard to automatically recognize and track through food preparation activities. Despite some characteristics that are unique to processing food, many of the challenges faced in this context re-occur in the context of tracking and guiding a person through other complex processes such as manufacturing and assembly tasks. The study of recognizing food preparation activities has therefore huge potential to push the boundaries of automatic activity recognition in general.

1.2 Sensor Selection and Sensor Fusion

Recognition and tracking of objects from video under illumination changes, occlusion, intra-class appearance variations and object deformations is challenging. In contrast,

embedded sensors such as accelerometers attached to objects provide explicit information about object identity and object motion by design. Accelerometers capture subtleties in object motion and continuous miniaturisation allows them to be inconspicuously integrated into a wide variety of objects. However, reasoning about interactions between objects solely based on accelerometers requires that each participating object has a sensor attached to it, considerably limiting the applicability of such an approach. It is not practical to equip objects such as unpackaged food items with sensors or tags. On the other hand, visual data effectively capture spatial relations and interactions between visual entities, assuming that they can be identified and localized. The complementarity of these sensing modalities suggests that methods for effectively combining visual data with data from embedded accelerometers have the potential to significantly improve recognition of manipulation activities and, importantly, to increase the range of activities that recognition systems can address.

While computer vision may be very helpful to recognize visibly pronounced gestures and untaggable objects, Pham et al. [76] have shown that accelerometers may successfully be used to recognize more precise and localized hand movements (e.g. cutting actions). Depending on the application, feature extraction algorithms in computer vision tend to be noisy and error prone in realistic scenarios, and extracted features do not represent semantic concepts (e.g. interest points). Localizing accelerometers in the camera field of view could provide means by which essential information about an action may be filtered, focusing attention and facilitating suppression of background clutter. Additionally, the location of an accelerometer provides a valid *reference point* for modeling geometric constellations in actions, facilitating *translation invariant* recognition of interactions.

Previous work on activity recognition also incorporated radio-frequency identification (RFID) readings alongside vision systems or accelerometers. Hoey et al. [38] used RFID readers placed at strategic locations in order to recognize when an object is placed at or removed from a particular location (e.g. a tray). Wu et al. [118] attached an RFID reader to a bracelet in order to detect hand-object interactions. Object detection based on RFID

does not require object appearance models or methods for handling visual occlusion, but RFID readings are subject to other kinds of noise. Hoey et al. [38] noted that the range of an RFID reader was too short to reliably detect whether an object is placed on a tray, whereas Wu et al. [118] reported that an RFID reader often detected false hand-object interactions when a hand passed close to an object while grabbing something else.

In the ubiquitous computing research community, video is mainly recorded to enable manual annotation of the *actual* sensor data from, e.g., RFID readers, accelerometers and gyroscopes [61, 98]. Often video is not recognized as a valid source of information due to the widespread belief that cameras are too intrusive and would not be acceptable. Over the past few years the number of cameras per household has grown exponentially through the emergence of smart-phones, tablets, and video-based input devices such as Kinect. While users may object to the idea of another person watching them via CCTV-like installations in their homes they are often accepting of visual sensors as part of a closed system that enables desirable services.

In the computer vision community some multi-modal activity recognition datasets have been released including synchronized video, RFID, audio and IMU data (e.g., [5, 18]). Interestingly, in all of these datasets, accelerometers and gyroscopes have been attached to the subject's body, which is often practically inconvenient. In pervasive intelligent environments it is therefore common practice to embed sensors in objects involved in an interaction instead [13].

1.3 A Motivating Application: Cognitive Situational Support

A particularly promising application for recognizing manipulation activities with a strong potential for social impact is cognitive situational support. As demographic projections suggest, the aging society we live in today imposes new challenges on medical and social care [78]. The ratio of people needing support to those who can provide it continues to

increase. Therefore, the role of technology to develop new assistive solutions gains in importance.

Approximately 35.6 million people worldwide live with dementia and the total social costs were estimated at $604 billion in 2010 [117]. This disease mainly affects older people and the likelihood of developing dementia roughly doubles every five years after age 65. Causing problems in memory, planning and decision making, people with severe dementia may often perform activities of daily living (ADL) such as dressing, washing, toileting, eating and kitchen tasks only with assistance from a professional or family carer. Neuropsychologists have shown a strong correlation between dementia severity and the ability to perform ADL tasks [30]. Regarding the person's well-being, the constant dependency on a caregiver in everyday activities is associated with a diminished quality of life, poor self-esteem, anxiety and social isolation [11].

One way in which technology can help people with cognitive impairments is through *situational support systems*. A situational support system recognizes or tracks a person's activities, identifies situations where help is needed and provides support by issuing guiding prompts to the user. Such a system could guide people with, e.g., dementia through activities of daily living and thereby enable them to live more independently of carers. Researchers particularly focusing on the personal consequences of dementia have been able to identify specific problems arising with ADL tasks [115, 116] and have successfully designed first prototypes of situational support systems for cognitively impaired people. Hoey et al. [40], for example, developed a system guiding a person through the task of hand washing and proposed a general specification method for support systems based on this approach, illustrated on the preparation of a cup of tea [39]. Their method successfully integrates many desirable aspects for a situational support system: it incorporates multiple types of sensors, multiple types of prompting actions, models cognitive abilities and the user's attitude, and provides *to the best of its knowledge* in each situation the optimal guiding prompt. Nevertheless, there are some shortcomings to be pointed out. Only a small portion of the specification process is performed automatically and the system needs

to be tailored to each complex activity individually. Additionally, solving for the optimal prompting strategy is very time-consuming. It is unclear whether this approach would be feasible for more complex activities such as *make lasagne*, and whether it would scale well to recognizing a wide range of tasks. A higher degree of automation and flexibility of the prompting system would be desirable. It seems sensible to decouple the user attitude model and prompting actions from the core activity recognition model, such that the recognition model has a high flexibility and may be *plugged into* the prompting system. The resulting model should be capable of recognizing multiple complex activities without the need for time consuming system specifications for each new complex activity and should be easily adaptable to new environments (e.g. the kitchen in a particular person's home).

Although first prototypical support systems exist, there are still many challenges to overcome. For example, these systems need to be easily adaptable to a particular person's home and cognitive capabilities for easy deployment. While prototypes of situational support systems are able to track and guide a cognitively impaired person through some activities of daily living (ADL) using embedded sensors, food preparation activities appear to be particularly challenging. Such systems would be deployed in a person's home and continuously gather data from the same person, so adaptation to the user's idiosyncrasies is desirable. Finally, they would need to be able to robustly track and guide through a wide range of activities to be a reasonable alternative to a personal caregiver.

In a situational support system, actions must be recognized online, imposing strong constraints on computational cost and requiring temporal segmentation of activities, a hard problem in itself. Modelling and tracking activities at a detailed level and issuing sensible prompts to the user are further open research problems.

1.4 Goals and Contributions

This thesis investigates the problem of recognizing manipulation activities during food preparation from video and accelerometers attached to objects. Embedded accelerometers

and video provide complementary information that is useful for activity recognition.

Traditionally, features from different sensor modalities are either combined for classification by concatenating feature vectors (*early fusion*), by combining semantic concept classifiers (*mid-level fusion*), or by merging classification results obtained separately from each modality (*late fusion*). This thesis investigates the hypothesis that extracting features from each sensor modality independently discards important *cross-modal* relational properties. Features modelling cross-modal relations, which relate data captured by accelerometers to information in the video, provide key information for modelling complex interactions between accelerometer-equipped objects and other visual entities in the scene. It will be shown that relational features can be used to build activity recognition systems that are more robust to variations in activity execution.

In order to reason about complex interactions from video and accelerometer data it is useful to relate the motion captured by an accelerometer to locations in the image space. This thesis proposes an accelerometer localization and tracking algorithm that enables tracking of accelerometer-equipped objects in the visual field of a camera without relying on their visual appearance. For recognizing manipulation activities, a family of feature descriptors is proposed that encodes visual interactions by describing properties of local visual features with respect to the location and motion of tracked objects. This approach is applicable to any scenario in which only a subset of all objects of interest can be tracked reliably, regardless of the sensor modality used to acquire object tracks. One example of this approach is discussed in detail in which dense tracklets are described relative to reference tracklets in histograms of RElative TrackLETS (RETLETS). Each histogram captures visual motion relative to a reference object. The accelerometer localization algorithm is applied to acquire trajectories of accelerometer-equipped objects that serve as reference tracklets for RETLETS. The effectiveness of this method for multi-modal activity recognition is evaluated on the *50 Salads* dataset, a novel dataset of food preparation activities that includes synchronized data from RGB-D video and accelerometers attached to objects. The dataset captures people preparing mixed salads where activities correspond

to individual tasks of the recipe and accelerometers are attached to kitchen objects.

This thesis focuses on motion features for three reasons. Firstly, the performance of accelerometer and visual motion descriptors for activity recognition are compared quantitatively. This experiment can inform future decisions on sensor selection, how these sensors are used, and where they are placed. Secondly, as accelerometer localization and dense tracklets are both based on dense optical flow, the proposed multi-modal features can be extracted with little additional computational cost. Thirdly, recognition performance based on appearance would be strongly biased. As the manipulation of ingredients severely changes their appearance, appearance-based activity models likely capture comparably stable appearance properties of kitchen utensils. Unless a dataset with a wide variety of bowls, pots, pans, dispensers, knifes, spoons etc. is recorded, which is hard to do for practical reasons, appearance-based activity models are likely to learn the appearance of particular object instances, and their generalization performance would be difficult to assess reliably.

As food preparation activities are subject to large inter-person variability, this thesis also investigates methods for adapting an activity recognition system to a particular user. Methods for user-adaptation are viable in scenarios where a recognition system is primarily used by a single person (e.g., in cognitive situational support), and could eventually enable tracking and assessment of a person's behavior change over time.

In summary, the contributions of this thesis are:

- An algorithm for accelerometer localization and tracking.
- A family of feature descriptors encoding relational properties between tracked objects and local visual features.
- A method for online activity recognition based on multi-modal features from video and accelerometer data.
- A study of methods for user-adaptive activity recognition.

- A comparative evaluation of motion features from accelerometers and video for activity recognition.

Additionally, the following resources have been made available to the research community as part of this PhD project:

- *50 Salads*: a carefully designed and fully annotated dataset of food preparation activities that includes RGB-D video and accelerometers attached to objects.
- A dataset for quantitative evaluation of accelerometer localization algorithms that includes annotations of ground truth accelerometer locations in every frame of a video.

1.5 Thesis Structure

Chapter 2 reviews relevant related work on visual and accelerometer-based activity recognition, and methods for fusing vision with inertial sensors and RFID.

Chapter 3 proposes an algorithm for localizing and tracking accelerometers in the view of a stationary camera. The localization algorithm matches acceleration estimates along visual point trajectories to accelerometer data, and this chapter discusses alternative methods for acquiring point trajectories and for measuring their similarity to accelerometer data.

Chapter 4 details the proposed method for multi-modal activity recognition. The family of object-generic relational histograms is formalized, and one instance from this family - RETLETS - is introduced. This type of feature is combined and compared with other feature types prior to classification. This chapter also describes these other feature types and discusses two methods for classification.

Chapter 5 investigates three methods for supervised adaptation of an activity recognition model to a target user: classifier combination, joint SVM training and weighted K-Nearest-Neighbour.

Chapter 6 introduces the *50 Salads* dataset and a dataset for quantitative evaluation of accelerometer tracking algorithms.

Chapter 7 reports evaluation results on the proposed method for accelerometer tracking, a comparison of visual and accelerometer-derived features, multi-modal activity recognition and user-adaptive classification.

Chapter 8 summarizes the contributions and findings of this thesis, and offers recommendations for future research.

Chapter 2

Related Work

This chapter reviews related work on visual and accelerometer-derived features for activity recognition, and methods for fusing vision with accelerometers and RFID. Related work that is only relevant to individual aspects of this thesis is reviewed in the respective later chapters. The literature reviewed in this chapter is organized into sections according to sensor-modality used. Methods specifically addressing recognition of manipulation activities or food preparation are reviewed in all sections.

2.1 Scope

There is a large body of work on recognition of human behavior applied in a wide range of contexts including surveillance [74], human-computer interaction [92], domestic care [39] and sports [28, 68, 119]. In these contexts the focus lies on different aspects of human behavior such as the body motion of a single person [88], human-human interaction [68], crowd behavior [127], and human-object interaction [20]. A broad review of methods across this wide range of aspects and application areas is outside the scope of this thesis. As this thesis addresses the problem of recognizing manipulation activities, this section focuses specifically on relevant methods for human-object interaction. For a more general introduction and wider coverage of methods the review papers [1, 47, 79, 106, 109] are recommended. There is also a large body of work on modelling the sequential or hierarchical structure of complex activities, activities, actions and events including methods using variants of hidden Markov models (HMMs) [8, 90, 119], stochastic context-sensitive grammars [45, 74, 93] and statistical relational learning [68, 69, 104]. While temporal structure modelling is certainly beneficial for recognizing complex manipulation actions, this thesis and the related work reviewed here concentrates on feature representations and sensor-fusion.

2.2 Visual Features for Activity Recognition

Features for visual activity recognition can be broadly categorized into *object-based* [2, 5, 38] and *generic* [54, 66, 67, 110] descriptors.

Object-based Methods for Activity Recognition Object-based methods identify and track objects in the scene and recognize activities by reasoning about spatiotemporal relationships between them (*high-level features*). This approach usually assumes that *all* objects of interest can be detected and tracked reliably. The necessity of training reliable object detectors for all relevant objects is a major practical limitation; issues include

dealing with detector uncertainty, modelling hard-to-detect deformable objects, and scaling to large numbers of different objects.

In the context of egocentric activities, Fathi et al. [21] devised a method for learning object detectors from weak (image-level) annotations in a multiple instance learning (MIL) framework. The scene is segmented into super-pixel regions which are tracked over time, and background regions are discarded prior to object category learning using image-stitching and change-detection techniques. Using these weakly learned object detectors, Fathi et al. [20] proposed a method for activity recognition in a probabilistic graphical model where nodes represent super-pixel regions, object labels, activities and a complex activity. Inference is performed cyclically: after an inference pass from super-pixel features to the complex activity node, distributions over activities are updated given the distribution over the ongoing complex activity, and object labels are updated based on the distribution over ongoing activities. Bansal et al. [3] propose to learn object models using active contours for segmentation and features describing each contour's shape. In every video frame one binary feature per object is extracted that indicates whether an object is in use based on the detection of a hand approaching the object, the object is moving, or the object is changing its appearance. Each frame is classified based on in-use features and features extracted from tracked hands in a HMM with emission probabilities found using support vector machines. Lei et al. [57] proposed a similar method that models objects via local colour, texture, and depth descriptors of foreground regions in RGB-D video. Hands are tracked using skin colour. Activities are recognized using hand-object interaction events and features of hand trajectories. While the results reported by Fathi et al. [21], Bansal et al. [3], and Lei et al. [57] are promising, their object detectors are trained on the specific object instances that are used at test time. Therefore, it is questionable how well these methods generalize; most likely, methods for adapting object models to new instances from the same class are required if a system is to be pre-trained and deployed into, e.g., a person's home. Non-probabilistic event detections at early stages of the recognition pipeline (such as in [3, 57]) are problematic as it is usually hard to recover from detection

errors at later stages.

Generic Features for Activity Recognition: Generic descriptors represent video as sets of local *low-level features* [54, 110] or higher-order statistics over those (*mid-level features*) [66], without making strong assumptions about the presence of specific objects. These methods have in common that local features are described relative to the image's frame of reference. Features are commonly extracted at spatio-temporal interest points [54, 55], at locations on a regular grid [112], or along fixed length point trajectories (tracklets) [110], and aggregated into histograms over spatio-temporal windows (bags-of-visual-words). In comparison to features extracted at spatio-temporal interest points [54], dense tracklets have shown superior performance on several standard action recognition datasets [110, 112], highlighting their discriminative power. Additional local appearance and motion features extracted along dense tracklets also outperformed the same descriptors extracted densely on a spatio-temporal grid [110] suggesting that they are more stable under motion and appearance variations. One challenge for action recognition based on tracklets is to be robust against unrelated motion such as camera motion. This problem has been addressed recently by Wang and Schmidt for camera motion [111] and by Bilinski et al. for unrelated body motion [6].

Several improvements over the bag-of-words representation have been proposed, including methods for soft-assignment (such as sparse coding) [113], supervised dictionary learning [53] and Fisher vectors [73]. In order to better capture the spatio-temporal structure of local features in videos, descriptors encoding second-order statistics have been proposed [66, 86]. As the number of relations between codewords grows exponentially with codebook size and with the number of discrete relational predicates, the main challenges are to estimate the distributions over relations reliably and to describe discriminative higher-order statistics in a compact form. Taking a radical approach to this problem, Savarese et al. [86] proposed a descriptor for pair-wise relations of codewords whose size is independent of the size of the codebook. For each pair of codewords, co-occurrence

statistics are estimated in multiple concentric spatio-temporal cuboids. Similar to bag-of-visual-words, these features called correlatons are are clustered to build a codebook of correlatons during training. Correlatons from each video are aggregated into histograms essentially discarding information about the identity of the visual words whose pairwise relation correlatons represent.

On a dataset of food preparation activities, Rohrbach et al. [83] compared features from body-poses and local features along dense tracklets [110], which they refer to as holistic features. Evaluation results suggest body-pose is rather indiscriminative for food preparation activities as holistic features performed comparably well (49.4% precision at 44.8% recall compared to 28.6% precision at 28.7% recall with body-pose features), but leave substantial room for improvement. Kuehne et al. [52] reported similarly poor performance using HOG/HOF [55] features in a sequential inference framework that involves HMMs and context-free grammars on their Breakfast dataset.

2.3 Activity Recognition from Embedded Sensors

Inertial sensors have been used for several pervasive computing applications including self-localization [31, 41], action recognition [75, 76] and skill assessment [35]. Whereas carefully engineering or learning discriminative features are research foci in the computer vision community, activity recognition from accelerometers commonly involves standard statistical features in the temporal [75] or frequency domain [46], and string-matching techniques applied to raw data [60]. For an overview of features proposed for accelerometer-based activity recognition, the interested reader is referred to the survey by Figo et al. [23]. An exception is the recent work of Plötz et al. [77] in which deep belief networks were used for feature learning.

Recognition of complex manipulation activities such as those involved in kitchen tasks from embedded sensors alone appears to require the environment to be highly instrumented.

For example, the *Ambient Kitchen* [72] in CultureLab[1] at Newcastle University is equipped with RFID readers and tags associated with movable objects, cameras, accelerometers attached to kitchen objects, cupboard doors and utensils, and pressure sensitive flooring. Based on sensors such as these the wider research community has developed some methods for recognizing manipulation activities of which a selection are reviewed here.

Lukovicz et al. [63] modelled workshop actions such as filing, drilling and screwing with hidden Markov models using the raw accelerometer data as emissions. They reported near-perfect recognition accuracy on segmented activities and mixed results ranging from 83% precision at 56% recall obtained for screwing to 100% correct for, e.g., filing on an activity detection task in unsegmented sequences.

Pham et al. [76] developed a method for recognizing food preparation actions such as *chopping*, *peeling*, *stirring* and *scooping* using dynamic time warping (DTW) on raw accelerometer data from sensors embedded in knifes and spoons. Evaluation was performed on a real world dataset including 20 people performing 10 actions involved in preparing salads and sandwiches. On this dataset the proposed method achieved an average precision of about 92% with about 92% recall. These results are good considering the inter-person and intra-action variations included in this dataset. The promising results suggest that the proposed method may be very useful to recognize *actions* from raw sensor data in a complex activity recognition system.

Rather than selecting features from accelerometer data depending on the target application, Hammerla et al. [34] proposed a generic feature representation that showed state-of-the-art performance on six datasets covering physical activities, daily living activities, and gestures at the workplace. The empirical cumulative density function (ECDF) is estimated from each axis of raw accelerometer data over a fixed temporal window. The function values at a fixed number of equidistant locations along the inverse of the ECDF (the quantile function) form the feature descriptor This representation essentially characterizes the shape of the ECDF non-parametrically while avoiding issues of over- and

[1] http://www.ncl.ac.uk/culturelab/

under-sampling commonly observed with histograms.

Hoey et al. [39] proposed a method for the specification of automatic prompting systems that incorporates an activity hierarchy to reason about activities at multiple levels of granularity and a mechanism for prompting a subject when help is needed. They illustrated their approach on the task of making a cup of tea. A complex activity is specified in terms of goals (sub-tasks) and behaviors (activities a person needs to perform in order to reach a goal, e.g. open the sugar container in order to have sugar in a cup of tea). Subsequently, cognitive abilities necessary for the specific behaviors are identified. Sensors are selected to detect all states of involved objects that need to be recognized and the reliabilities of these sensors are determined. These specifications are semi-automatically converted into a separate partially observable Markov decision process (POMDP) [12] for each goal. Each goal is associated with a set of preconditions, which allows for defining a partial ordering of goals. A simple controlling policy switches between POMDPs according to these preconditions whenever a goal is reached or a sensor measurement contradicts the currently active goal. In a different context, Hoey et al. [40] used POMDPs to guide a person through the task of hand-washing with a camera mounted above a sink.

2.4 Fusing Vision with Inertial Sensors and RFID

Fusing vision with other sensor modalities has previously been investigated for tasks including activity recognition [5, 118], people tracking [41] and object tracking [96]. Behera et al. [5] recognized assembly tasks by concatenating histograms of visual and inertial sensor features in an early fusion approach. Specifically, pairwise distances and changes of distance between objects recognized from a body-worn camera were encoded in a histogram as were pairwise body-part relations estimated from inertial data. Their experimental results confirmed that visual features and features from inertial data were complementary as recognition performance observed with a combination of these features was considerably higher than using either feature type independently. Recognition of

complex kitchen activities has been addressed by Wu et al. in [118]. Complex activities such as *make coffee*, *make salad* and *pack lunch* are described in terms of the set of objects involved in each task. Observations from an RFID reader in a bracelet attached to the subject's arm and from a video camera are used in a Dynamic Bayesian Network (DBN) [50] to infer manipulated objects and performed complex activities. Visual object representations are modeled as bags of SIFT features and are trained using RFID readings to infer object labels. Using learned visual object models and active RFID readings, the authors report recognition rates of 73% for complex activities and 71% for objects on a dataset including 16 complex activities and 33 objects. The authors also experiment with using RFID readings only to train the model and perform recognition exclusively based on the learned visual representations. Unfortunately, they present test results of this approach only on the same data as used for training. Although complex kitchen activities are recognized, the applicability of this method in a situated support system is very limited. In order to guide a cognitively impaired person through a complex activity such as preparing a sandwich it is essential for the system to know which particular activity and action the subject is currently trying to perform. While knowledge about the identity of objects involved might be discriminative enough to distinguish multiple complex activities, differences between food preparation activities on a lower level are more subtle. A spoon, for example, can be used for stirring, scooping and scraping, and knowledge about the motion of the spoon is necessary to distinguish these actions.

2.5 Discussion

Methods for visual activity recognition are particularly appealing as they do not require objects or people to be equipped with sensors. While methods that reason about interactions of objects with each other and with hands showed good results on recognizing manipulation activities, training visual object detectors is challenging and acquiring training data of labeled object examples is costly and time-consuming. Generic local features

have the advantage that they are applicable to a wide range of contexts, but have shown poor performance on recognizing manipulation activities. Accelerometer data seem to better capture the subtleties in motion that discriminate manipulation activities, but lack information on interactions between accelerometer-equipped objects and objects that are not equipped with sensors. Methods for combining features extracted from video and accelerometers have shown promising results, which motivate an investigation of more effective sensor fusion approaches. Associating data captured by accelerometers with locations in the video, as proposed in this thesis, is key to developing feature descriptors that model interactions between accelerometers and other entities observed in the video. Capturing properties of visual entities that are not equipped with sensors through generic local features and encoding relational properties between accelerometer-equipped objects and local features has the potential to combine the advantages of both sensor domains while avoiding their respective disadvantages.

Chapter 3

Accelerometer Localization and Tracking

This thesis investigates multi-modal activity recognition by combining information from different sensor types during feature extraction. In order to combine data from accelerometers and video, it would be useful to associate accelerometers with locations in the camera's frame of reference. This chapter proposes one method for localizing and tracking accelerometers in video.

The proposed method utilises acceleration data captured by accelerometers and visual motion estimated from video data. Because it does not rely on the visual appearance of accelerometer equipped objects, this method is largely agnostic to the objects to which accelerometers are attached or in which they are embedded; it can be used to track new objects without the need for reconfiguration.

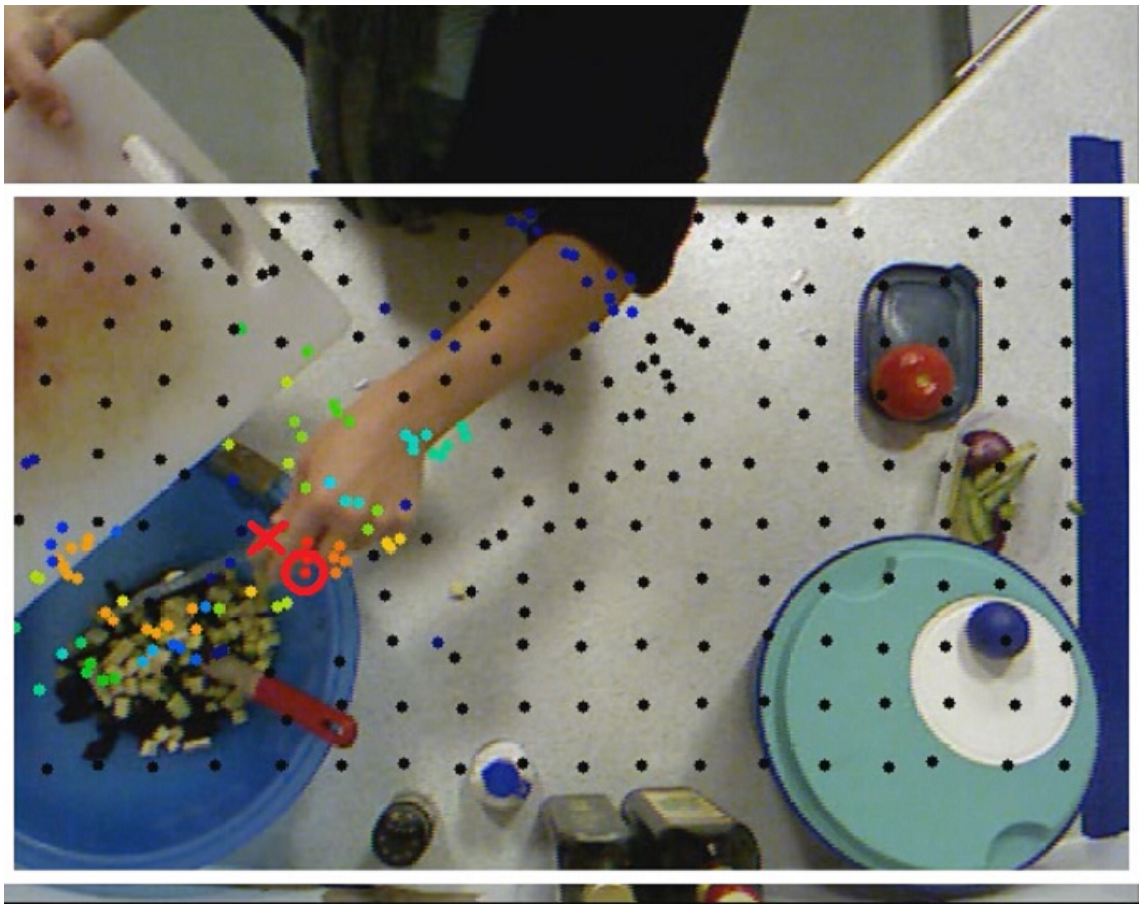


Figure 3.1: Accelerometer Localization. By measuring similarity of accelerations along trajectories of point features (colored dots) with accelerometer data (black indicates weakest, red indicates strongest similarity), the algorithm estimates the accelerometer location (red circle). A red cross marks the ground-truth.

3.1 Background

Accelerometers are becoming increasingly ubiquitous (through, e.g., smart phones, smart watches, tablets and specialized fitness devices) and there is a growing interest in fusing accelerometer data with visual data. Combining these sensor types may have strong potential as they provide complementary information. While accelerometers capture subtleties in the movement of the device, computer vision may, for example, put this information into spatial context and into relation with other entities.

Localizing an accelerometer in the visual field of a camera has a wide range of applications beyond multi-modal activity recognition. For example, detecting and localizing

accelerometers embedded into personal devices in the camera view may reveal a person's identity, which can be useful for collaborative table-top interactions, personalized information delivery and cross-device synchronization.

This chapter proposes an accelerometer localization pipeline that is compatible with multiple techniques for estimating visual trajectories from video data. Here two methods are considered, namely trajectories from sparse optical flow (KLT) [62] and trajectories sampled on a regular grid from a dense optical flow field [19]. An incremental similarity measure for matching visual trajectories with accelerometer data is defined that is easy to implement and fast to compute. The output of the algorithm proposed in this chapter is illustrated on an example video frame in Fig. 3.1. The algorithm assigns a similarity score to each hypothesized visual trajectory and estimates an accelerometer's location as the location of the best matching trajectory in the latest frame.

In the remainder of this section the main challenges for localizing and tracking accelerometers in video are discussed and relevant related work is reviewed.

3.1.1 Challenges

The general problem of fusing visual data with accelerometer data and some potential solutions are discussed in detail by Corke et al [15]. Here, the key challenges for accelerometer localization and tracking are highlighted that motivate algorithm design decisions made in subsequent sections.

Occlusion: Accelerometers are usually visually occluded. An accelerometer may be occluded by the object it is attached to or embedded into, in which case the motion observed at the visible location of the object is likely to be similar to the motion captured by the accelerometer. It may, however, also be occluded by a different visual entity in which case the visual motion at the accelerometer's location and the accelerometer's motion projected in the image plane are likely to differ. In the sensor setup considered in this thesis accelerometers are embedded into kitchen utensils to unobtrusively capture utensil

movements. While for some utensils accelerometers are embedded in their handles, for others the sensors are attached in locations that are not necessarily close to a person's hand (e.g., on the rim of a bowl). Because of these frequent and diverse occlusions, a localization algorithm that relies on the visual appearance of the accelerometer or of the enclosing device is expected to be unreliable and would require some reconfiguration or re-training for each new device. The method presented in this chapter proposes therefore to extract local accelerations from video, which are subsequently matched to data captured by an accelerometer.

Gravity: An accelerometer measures proper acceleration, that is acceleration relative to free-fall. If such a device rests on the ground it measures acceleration due to the mechanical force from the ground which is proportional to gravity g but points in the opposite direction. Coordinate acceleration, on the other hand, describes acceleration relative to a coordinate system. In this case a resting object has zero acceleration. Accelerations along trajectories extracted from video for example represent coordinate accelerations relative to the 3D coordinate system attached to the camera. In order to compare accelerations from vision and accelerometer data it is useful to convert visually estimated accelerations into proper accelerations. The gravity vector, which is necessary to perform this conversion, is generally unknown in the 3D coordinate system attached to the camera. This chapter proposes to estimate the direction of gravity from annotated depth maps from an RGB-D camera. This estimate is subsequently used to transform visual acceleration estimates to proper accelerations.

Accelerometer's local coordinate system: Accelerometers capture tri-axial translational acceleration with respect to a local reference frame; in general an accelerometer's orientation is unknown and changes over time, making alignment with the camera's frame of reference problematic. As accelerometers measure only translational and not angular acceleration it is infeasible to calibrate and subsequently track their relative orientation. A sensor device containing 3 orthogonal accelerometers and 3 orthogonal gyroscopes,

providing acceleration information in all six degrees of freedom, would therefore facilitate the fusion task. The lack of knowledge about an accelerometer's orientation relative to the camera makes it difficult to match visual acceleration estimates with accelerometer data. A simple work-around employed in this chapter is to only match acceleration norms. While matching the norms of acceleration vectors is orientation invariant, a substantial portion of information that could be valuable for matching is discarded.

Synchronization: Although camera frames and accelerometer readings may be timestamped, we may not generally assume that their clocks are synchronized. Similarly, if timestamps are generated by a server receiving new data from the client devices (cameras and accelerometers), the timestamps are synchronized, but they do not reflect latencies in data transmission.

Different sensor frequencies: Different sensor frequencies of accelerometers and cameras impose another problem as accelerometer frames and images are not acquired at the same time, even if the streams are synchronized. For example, there may be multiple samples of accelerometer data associated with the interval between subsequent video frames. As the local coordinate systems of accelerometers may change in between time instances at which accelerometer measurements are taken, simple integration over time is a suboptimal solution to data aggregation for comparison with camera observations. A related problem is that accelerometers measure instantaneous acceleration, whereas acceleration estimated from consecutive frames represents mean acceleration over the time duration between frames. The expected estimation error, i.e. the difference between the visually estimated acceleration and instantaneously measured acceleration, is proportional to the mean gradient magnitude of the instantaneous acceleration between subsequent frames. Therefore, estimation errors are expected to be larger when an accelerometer is moving strongly than when it is stationary. When designing a similarity measure there is a trade-off to be made between falsely penalizing expected estimation errors and under-representing true mismatches.

3.1.2 Related Work

Inertial sensor localization in a camera's field of view is primarily being addressed in the context of tracking people [64, 91, 102]. For example, Teixeira et al. [102] used magnetometers and accelerometers in mobile phones to identify and localize multiple people tracked from CCTV cameras. Sensor measurements from phones carried by the subjects were associated with tracked body positions in a hidden Markov model for localization and to resolve ambiguities in visual tracks. This approach makes strong assumptions about the appearance of the object (person) that is to be tracked. Shigeta et al. [91] make similar assumptions localizing an accelerometer using normalized cross-correlation (NCC) of accelerometer data with trajectories of a jacket and a hand. Maki et al. [64] proposed point feature trajectories in combination with NCC for accelerometer localization without prior knowledge of object appearance. Unfortunately, those authors did not present quantitative data on localization performance. The method for accelerometer localization presented in this chapter has been previously published in [95]. As an extension to [95], this chapter proposes methods for long-term accelerometer tracking in Section 3.6.

3.2 Localization Pipeline

The proposed method for accelerometer localisation involves

- generating trajectory hypotheses in videos,
- estimating local visual accelerations along these trajectories and
- matching visual acceleration estimates to data captured by accelerometers.

The flow from raw input data to location estimates is illustrated in Fig. 3.2.

The visual data, i.e. the colour frames and depth maps, are geometrically and temporally aligned such that at each pixel, RGB values and depth is available. The OpenNI SDK[1], which was used to acquire Kinect colour and depth images, provides a method for automatic geometrical alignment. In practice, there is a small time delay between colour and depth frames. With depth from structured light the depth measurement for some pixels is invalid. This is observable at depth discontinuities, at non-opaque or reflective surfaces, and in areas that lie from the viewpoint of the sensor in the shadow of the projected light pattern. Colour images were transformed to grayscale prior to processing.

Trajectory hypotheses are generated by sampling and tracking points in the video. The sampled points are extended to trajectories in order to estimate acceleration for matching with accelerometer data. This chapter considers generating point trajectories by tracking either salient features [89] with pyramidal KLT [7] or by tracking points on a regular grid with dense optical flow [19]. Both techniques are described in Sec. 3.3.

In order to match visual trajectories with accelerometer data, locations in image space need to be transformed into world coordinates, which requires the camera to be calibrated. Additionally, sequences of locations need to be differentiated twice to accelerations. In order to transform coordinate acceleration to proper acceleration, the gravity vector needs to be estimated in the view of the camera and added to visual acceleration estimates. This procedure is described in detail in Sec. 3.4.

[1] OpenNI SDK: `http://structure.io/openni`

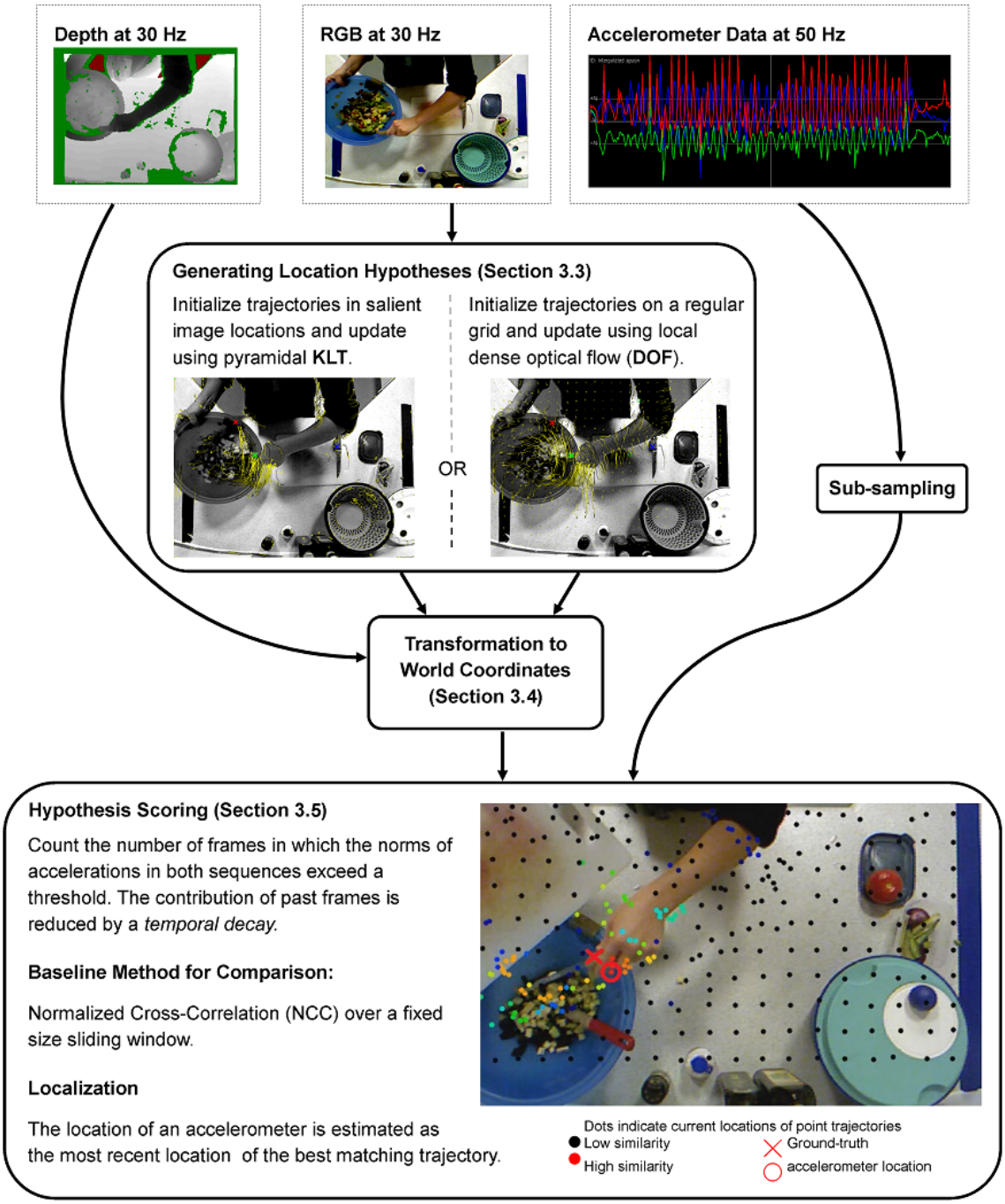


Figure 3.2: Accelerometer localization pipeline: Data flow from raw input streams generated by accelerometers and the camera to location estimates for each accelerometer in the current video frame.

The resulting acceleration sequences are not directly comparable to the captured accelerometer data, as the frequencies of the camera and the accelerometers are different. Therefore, the acceleration data from accelerometers is sub-sampled such that each video frame is uniquely matched to one sample from each accelerometer.

Once these processing steps are completed a scoring function is applied to create a similarity map which is used to determine trajectories that best match the accelerometer data. Similarity scores are computed exhaustively between all current visual trajectories and all accelerometers. Based on the resulting similarity map between devices and trajectories, the location of each accelerometer in the current video frame is estimated as the location of the best matching trajectory in the most recent frame. An example similarity map representing similarity scores for all trajectory hypotheses is shown in Figures 3.1 and 3.2 with estimated and ground-truth accelerometer locations also marked.

3.3 Generating Hypotheses

Visual localization of accelerometers involves matching accelerometer data to local motion properties in a video. In order to solve this correspondence problem, acceleration data need to be extracted from video and compared to data captured from accelerometers. In this context, visual motion estimation may generally be performed on three levels of granularity.

At the lowest level, dense optical flow methods may be used to estimate a velocity for each location in an image from two or more subsequent frames. Using these velocity estimates to track surface points over time, however, is not advised as tracks produced in this manner tend to drift away from the locations on objects they are initialized on with increasing track length and a very large number of tracks is produced, which is associated with a high computational cost for track correspondence validation.

At the highest level, a track may be extracted from each moving object using standard object tracking techniques. These models usually make strong assumptions about the

appearance and geometry of the tracked object. For localizing an accelerometer, however, we want a model that is not object specific.

Based on these observations, the most promising approach to extracting acceleration information from visual data seems to be at an intermediate level through the use of point tracking methods such as KLT [62, 89, 105] or subsequently proposed feature trackers (e.g. [14, 17, 51, 108, 121]). By considering a relatively small number of points compared to dense optical flow, the number of location hypotheses and the computational cost associated with their validation is drastically reduced. In comparison to tracking objects, point trajectories make no assumptions about the objects' appearance, providing a high degree of flexibility and generality. Some methods for extracting point trajectories have already been successfully applied to action recognition (e.g. [9, 66, 67, 99, 110]). Shigeta et al. [64] present a first attempt at localizing an accelerometer in a camera field of view using KLT and normalized cross-correlation (NCC).

The remainder of this section describes two methods for trajectory construction: sampling the image at salient (highly textured) locations [89], which are tracked over time using sparse optical flow (pyramidal KLT [7]), and sampling the image on a regular grid with point locations updated over time using Farnebäck's dense optical flow algorithm [19]. While the first method extracts trajectories at feature locations that are expected to be easily tracked individually, the latter evenly samples the image space and uses a spatially smooth flow field. Both methods are explained in detail below.

3.3.1 Trajectories from Feature Point Tracking

In this first trajectory construction method trajectories are initialized at keypoint locations, defined as image locations whose covariance matrix of local image gradients has two large eigenvalues [89]. Two large eigenvalues in this matrix indicate that the local region around an image location is highly textured and therefore more likely to be tracked robustly in the next frame than homogeneous image patches. The smaller eigenvalue of the covariance matrix is subsequently called *cornerness*.

A fixed maximum number of trajectories N_T^{max} is maintained. From the first frame of a video, a trajectory at the image location with highest cornerness is initialized. The algorithm proceeds by initializing trajectories at additional locations in order of decreasing cornerness subject to the following constraints.

- The total number of trajectories is not greater than N_T^{max}.
- The Euclidean distance of a point to the locations in the current frame of all other trajectories is greater than a threshold d_I^{min}.
- The cornerness of the point is greater than a fixed fraction of the maximum cornerness value in the current image.

The minimum distance d_I^{min} prevents initialization of large numbers of trajectories in a small area of the image.

For each subsequent frame, currently maintained point features are attempted to be tracked using pyramidal KLT [7]. All trajectories for which locations have been tracked successfully in the new frame are updated by appending their respective new locations. Trajectories that could not be tracked in the current frame are discarded. The set of trajectories that are tracked successfully to the current frame or that were newly initialized are referred to as *active* trajectories. If after an update step there are less than N_T^{max} active trajectories, new trajectories are initialized at keypoint locations subject to the constraints above.

The implementations for feature extraction as proposed by Shi and Tomasi [89], and tracking as proposed by Bouguet [7] provided by the OpenCV library [29] were used.

3.3.2 Trajectories from Grid-Sampling and Dense Optical Flow

The second method for constructing trajectories from video initializes trajectories at locations on a regular grid with horizontal and vertical spacing d_G between points. For each new frame a dense flow field is computed as described by Farnebäck [19] and provided

by OpenCV [29]. Point feature locations are then updated through shifting previous locations by the horizontal and vertical flow estimated for their respective positions. In the case where a newly estimated location lies outside the image region the corresponding point trajectory is discarded. Otherwise the trajectory is updated by appending the newly estimated location to its respective location sequence. Point trajectories have variable length and are extended indefinitely. New trajectories are initialized at grid point locations whose minimum distance to current locations of active trajectories is greater than d_G.

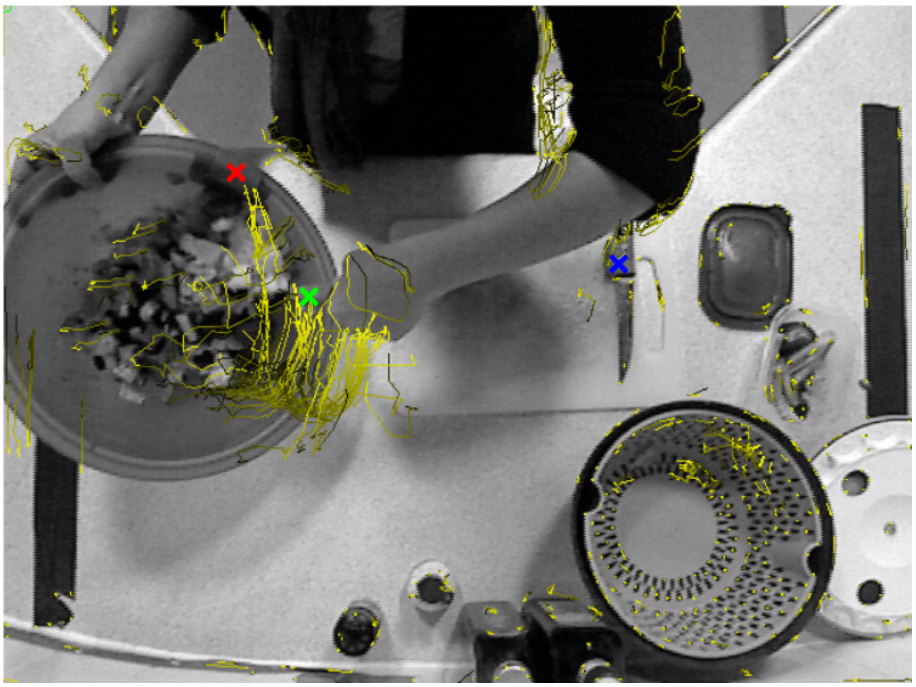
Following this approach the number of tracked points may grow indefinitely as pairwise distances become diminishingly small. This is avoided by *terminating* a track if its location in the most recent frame becomes closer than some threshold τ_d to another track's location and it is younger than that other track (i.e., has been tracked for a smaller number of frames). The grid spacing d_G and the threshold τ_d provide bounds on the spatial resolution of maintained location hypotheses.

Figure 3.3 illustrates trajectory hypotheses extracted with sparse feature point tracking (KLT) and dense optical flow (DOF). Notice that KLT trajectories are clustered in highly textured locations whereas DOF trajectories densely cover the image. Dense coverage of the image could be advantageous in situations where accelerometers are occluded by surfaces that are not highly textured. Due to the dense flow field, DOF trajectories are also smoother and more robust to false correspondences [110]. These expected advantages come at the expense of an increased computational cost of estimating a dense flow field. While the computational cost of KLT is proportional to the number of tracked points, the cost of DOF is proportional to the number of pixels in the image.

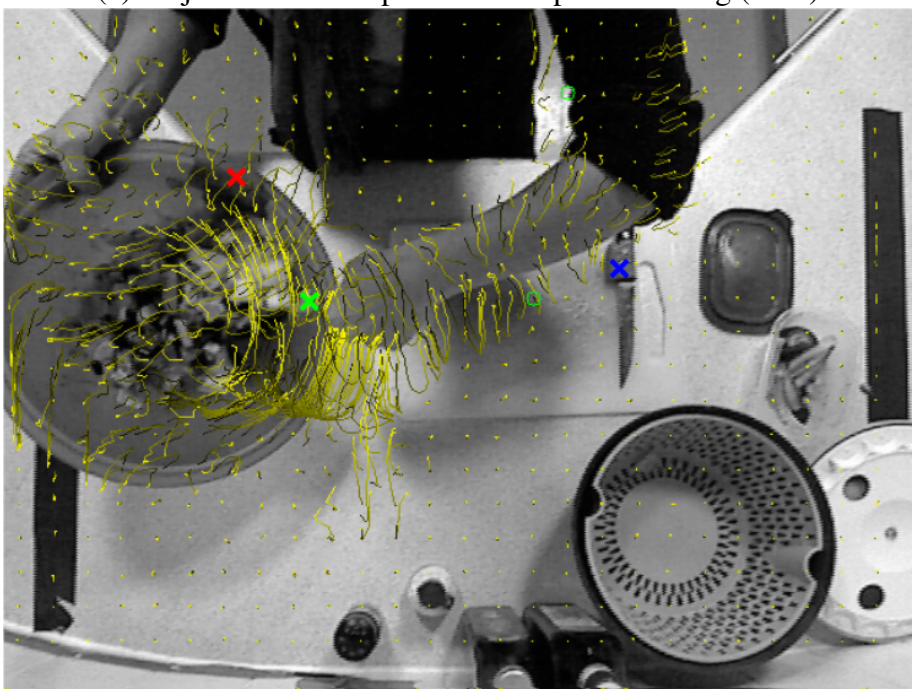
3.4 Transformation to World Coordinates

Two transformations are needed in order to match accelerations estimated along point trajectories with data captured by accelerometers:

1. Point trajectories in image coordinates need to be transformed into world coordinates



(a) Trajectories from sparse feature point tracking (KLT)



(b) Trajectories from grid-sampling and dense optical flow (DOF)

Figure 3.3: Comparison of KLT (a) and DOF (b) for trajectory hypothesis construction. Trajectories are visualized as yellow line sequences corresponding to the last twenty point displacements and overlaid on top of the most recent frame converted into a gray-level image. The most recent displacements are shaded brighter than older ones. The green circles indicate trajectories initialized in the most recent frame. The red, green and blue crosses indicate the ground-truth locations of three accelerometers.

(i.e., metric values with the center of the image plane at $\mathbf{0}$).

2. Gravitational effects in world coordinates need to be simulated to transform coordinate acceleration to proper acceleration (relative to free-fall).

The relationship between image and world coordinates is governed by the distance of objects from the camera, lens distortion, and the camera's intrinsic parameters.

Distance from the Camera

Due to perspective projection, acceleration magnitudes depend on the location of points along trajectories in the normal direction of the camera's image plane (depth). Two methods for assigning depth values to pixels are investigated. First, a constant uniform depth for all pixels in the image is assumed. This configuration may be particularly interesting if depth data is noisy or unreliable, or if a combined colour and depth sensor is simply unavailable. Second, the value from the depth map provided by a range sensor is assigned to each pixel in the colour image. Surprisingly, lower localization accuracy was observed using depth maps than using a constant fixed depth.

In practical scenarios the depth estimate at some pixels in a frame may become invalid. This happens for example frequently with cameras that use structured light for constructing depth maps. If the situation occurs that the depth at the current image location of a point feature trajectory is invalid, a depth estimate from previous depth values and velocities is provided. The current depth z_t is estimated from the previous depth z_{t-1}, the estimated velocity v_z and the video frame rate f_{vid} with a linear predictor:

$$z_t = z_{t-1} + \frac{v_z}{f_{vid}} \tag{3.1}$$

Note that 3D trajectories could also be constructed from scene flow [107]. However, estimating dense 3D scene flow is too computationally expensive for online accelerometer localization [33].

Lens Distortion and Camera Intrinsic Parameters

Assuming a pinhole camera model, distortion coefficients and intrinsic parameters are determined as proposed by Brown [10] and Zhang [126], respectively.

The Brown-Conrady [10] model with radial and tangential lens distortions is employed for determining the intrinsic camera parameters. Let the focal length be f, the dimensions of the image elements be s_x, s_y and the principal point be (c_x, c_y). Then the ideal model for the projection without distortion of a 3D point (described relative to the camera) onto a 2D surface is given by

$$\begin{pmatrix} x'_p \\ y'_p \\ s \end{pmatrix} = \begin{pmatrix} fs_x & 0 & c_x \\ 0 & fs_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix}, \tag{3.2}$$

which is unique up to a scale parameter s. For a given scale parameter the location of a point projected onto the image is $(x_p, y_p) = (x'_p/s, y'_p/s)$. Radial distortion with coefficients k_1, k_2, k_3 and tangential distortion with coefficients p_1, p_2 transform these locations to

$$\begin{pmatrix} x \\ y \end{pmatrix} = (1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \begin{pmatrix} x_p \\ y_p \end{pmatrix} + \begin{pmatrix} 2p_1 x_p y_p + p_2(r^2 + 2x_p^2) \\ p_1(r^2 + 2y_p^2) + 2p_2 x_p y_p \end{pmatrix}, \tag{3.3}$$

where r is the point distance from the centre of the image.

Functions provided by OpenCV [29] were used to determine intrinsic parameters from multiple views of a chessboard pattern. The library also provides functions to estimate undistorted image locations that were used when transforming image locations into world coordinates.

Given an undistorted image location $\mathbf{x} : (x, y)$, estimated depth z, focal length f, imaging element dimensions s_x, s_y, and principal point (c_x, c_y), image locations are transformed to world coordinates using (3.4).

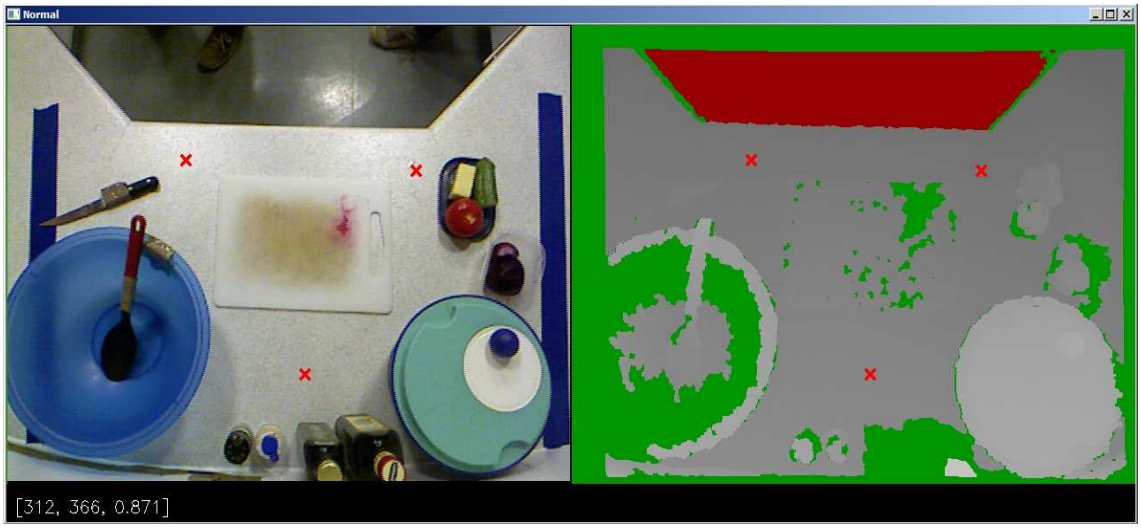


Figure 3.4: Direction of gravity: Three points are marked (red crosses) in the colour image (left) on a surface perpendicular to gravity, here the work surface. Their locations in the depth map (right) are used to extract the surface normal, which is assumed to be in the direction of gravity.

$$x' = \frac{(x - c_x)z}{fs_x} \qquad y' = \frac{(y - c_y)z}{fs_y} \qquad z' = z \tag{3.4}$$

Estimation of Proper Acceleration

Let $P'_i : (\mathbf{x}'_0, \ldots, \mathbf{x}'_t)$ denote a point trajectory represented as a sequence of locations in world coordinates, $\mathbf{x}'_j : (x'_j, y'_j, z'_j)$. Velocities $\mathbf{v}'_j$ and accelerations $\mathbf{a}'_j$ are approximated using discrete differences $\mathbf{v}'_j = f_{vid}(\mathbf{x}'_j - \mathbf{x}'_{j-1})$ and $\mathbf{a}'_j = f_{vid}(\mathbf{v}'_j - \mathbf{v}'_{j-1})$, respectively, where f_{vid} is the video frame rate. Locations $\mathbf{x}'_j$ are smoothed over time with a zero-mean isotropic Gaussian with some small variance to avoid instabilities in the approximation [80]. Note that estimates $\mathbf{v}'_j$ and $\mathbf{a}'_j$ are only available with short, fixed delays due to the smoothing kernel.

Ideally, one would transform accelerometer data to coordinate acceleration by subtracting acceleration measured due to resisting gravity[2], but as accelerometer orientation relative to the direction of gravity is unknown and changing over time this is not possible.

[2]This is equivalent to adding acceleration due to gravity.

Fortunately, acceleration due to gravity can be simulated and subtracted from acceleration estimated along point trajectories if the direction of gravity in video can be estimated. While the magnitude of gravity at the earth's surface is well known[3], the direction relative to the camera's coordinate system depends on the positioning of the camera.

This chapter proposes to determine the direction of gravity in the camera's field of view by estimating surface normals from depth maps. Assuming there is a planar surface in the scene that is aligned with gravity (e.g., a floor, a ceiling, a work surface or a tabletop) a normal is estimated pragmatically from a set of at least three manually marked points on the surface (see Fig. 3.4). The marked points are transformed into world coordinates using (3.4). Given three such points in world coordinates $\mathbf{x'}_0, \mathbf{x'}_1, \mathbf{x'}_2$, the direction of gravity is given by the cross-product of the co-planar vectors $\mathbf{s} = \mathbf{x'}_1 - \mathbf{x'}_0$ and $\mathbf{t} = \mathbf{x'}_2 - \mathbf{x'}_0$, and the magnitude follows from standard gravity (3.5).

$$\hat{\mathbf{g}} = 9.80665 \frac{\mathbf{s} \times \mathbf{t}}{||\mathbf{s} \times \mathbf{t}||} \tag{3.5}$$

This result may be refined by sampling the vectors $\mathbf{s}$ and $\mathbf{t}$ multiple times from a larger number of marked points and averaging the estimated surface normals. Depending on the order in which the points have been marked, the estimated direction of gravity may need to be inverted. The estimated proper acceleration $\mathbf{a}_t$ of a point trajectory at time t is given by $\mathbf{a}_t = \mathbf{a'}_t + \hat{\mathbf{g}}$.

Instead of manually marking points on the work surface, this task could be automated. For example, a horizontal plane could be fitted to depth data using Random Sample Consensus (RANSAC) [24]. The estimated plane could be further refined through least-squares estimation from the set of points that RANSAC identified as inliers.

[3]Here, standard gravity $|\mathbf{g}| = 9.80665 m/s^2$ is used.

3.5 Hypothesis Scoring

3.5.1 Sub-Sampling

Sequences of acceleration estimates along point trajectories $A_t^{vid} : (\mathbf{a}_{t-j}^{vid}, \ldots, \mathbf{a}_t^{vid})$ are matched with the sequence of accelerometer data $A_t^{acc} : (\mathbf{a}_{t-j}^{acc}, \ldots, \mathbf{a}_t^{acc})$. The video frame rate f_{vid} is usually lower than the accelerometer sampling rate f_{acc}. For a one-to-one association the accelerometer data needs to be sub-sampled. Acceleration as measured by an accelerometer corresponds to instantaneous properties of the sensor. Because acceleration experienced by the sensor *in between* subsequent samples is unknown, it is advisable to match visual acceleration estimates to the temporally closest accelerometer sample rather than using interpolation. Thereby accelerometer data is implicitly sub-sampled as some samples remain unmatched. This preference has been confirmed by comparative empirical evaluation.

3.5.2 Temporally Decaying Thresholding (TDT)

Since accelerometer orientation is unknown and changing over time, a similarity between acceleration *norms* is established. A moving object's visual trajectory is most easily discriminated from those of other objects during periods when its velocity changes frequently. Unfortunately, the similarity of raw acceleration sequences during such periods is sensitive to synchronization errors and to differences between instantaneous acceleration measured by accelerometers and mean acceleration between frames as estimated from video. In order to address this issue a radical approach is taken, transforming sequences A_t^{vid} and A_t^{acc} into binary sequences B_t, where each element b_{t-j} is non-zero if and only if the absolute difference between the acceleration norm and the magnitude of standard gravity exceeds some noise threshold τ_a, as in (3.6) where $1[\cdot]$ is the indicator function. While this transformation discards most information on acceleration magnitudes it preserves local extrema and saddle points in the corresponding velocity sequences.

$$b_t = 1\left[|\mathbf{a}|_t - |\hat{\mathbf{g}}| \geq \tau_a\right] \tag{3.6}$$

An efficient, recursive similarity measure between pairs of binary sequences which gives higher weight to recent frames using a multiplicative temporal decay $\alpha \in [0, 1)$ is defined in (3.7) and (3.8). Similarity is thereby proportional to the number of frames in which both sensors capture significant acceleration, reducing the impact of samples in the past through temporal decay.

$$S_0^{(i)}(B_0^{vid}, B_0^{acc}) = 0 \tag{3.7}$$

$$S_t^{(i)}(B_t^{vid}, B_t^{acc}) = \alpha S_{t-1}^{(i)} + b_t^{vid} b_t^{acc} \tag{3.8}$$

This similarity measure is subsequently being referred to as Temporally Decaying Thresholding (TDT).

As new trajectories get initialized others have already accumulated a potentially high similarity score over time. Through empirical evaluation it was found that the algorithm becomes more effective if after the first frame the similarity score of new trajectories is initialized to the score of the closest trajectory hypothesis in the most recent frame.

3.5.3 Normalized Cross-Correlation (NCC)

In the most relevant related work, Maki et al. [64] used normalized cross-correlation (NCC) as a scoring function. It is therefore sensible to introduce NCC here and use it as a baseline for comparison of the proposed similarity measure. Normalized cross-correlation (3.9) is a standard measure in signal processing for estimating the correlation between two signals of fixed dimensionality D. Incorporating the mean and standard deviation of both signals makes NCC invariant to differences in scale and translation.

$$NCC(A_t^{vid}, A_t^{acc}) = \sum_{i=t-D+1}^{t} \frac{(|\mathbf{a}|_i^{vid} - \mu_t^{vid})(|\mathbf{a}|_i^{acc} - \mu_t^{acc})}{\sigma_t^{vid}\sigma_t^{acc}} \tag{3.9}$$

$$\mu_t = \frac{1}{D} \sum_{i=t-D+1}^{t} |\mathbf{a}|_i \tag{3.10}$$

$$\sigma_t = \sqrt{\frac{1}{D-1} \sum_{i=t-D+1}^{t} (|\mathbf{a}|_i - \mu_t)^2} \tag{3.11}$$

Considering the difficulties of signal synchronization and systematic errors arising from comparing mean acceleration between video frames to instantaneous accelerations captured by accelerometers, it is easy to see that normalized cross-correlation is not a good similarity measure for accelerometer localization. When accelerometer data oscillates with a high frequency, minor synchronization errors have a strong impact on the measured signal similarity. NCC is particularly sensitive to systematic estimation errors as it is dominated by extreme values (values far from the mean). As a result, the best matching trajectory for a moving accelerometer corresponds often to a badly tracked point in the background exhibiting random motion.

3.5.4 Point Location Estimate

Finally, an accelerometer's location is estimated as the location of the highest scoring hypothesis. In the rare case where all active trajectories have the same similarity (usually occurring right after initialization), the algorithm does not give a location estimate. When two or more hypotheses have the same highest score, the last location of the trajectory that has been initialized earlier is chosen.

Note that by enforcing a unique point estimate for the location of an accelerometer a lot of useful information provided by a similarity map is discarded. If for example none of the point trajectories show a strong similarity with an accelerometer after an extended period of time, this could indicate that the accelerometer is outside the camera's field of

view. If, on the other hand, trajectories in two (or more) distinct image locations show strong and almost equal similarity to an accelerometer it would be sensible to consider multiple location hypotheses in subsequent processing steps.

3.6 Long-Term Accelerometer Tracking

If an occluding object's motion differs from the accelerometer motion, a previously correctly matched trajectory is likely to drift away from the correct location as it tracks the occluder. This scenario frequently occurs, for example, after an accelerometer-equipped utensil has been released and the hand that previously held the device moves away. In this case ($b_t^{acc} = 0$), the similarity scores of all point trajectories are updated to $S_t^{(i)} = \alpha S_{t-1}^{(i)}$ and, as the ranking of hypotheses does not change, the estimated accelerometer location diverges from the true location. For this reason the localization method discussed above is extended to long-term accelerometer tracking by (i) detecting when an accelerometer is not moving, (ii) taking a snapshot of the similarity map, and (iii) re-initializing similarity scores once an accelerometer starts moving again. The accelerometer tracking algorithm is described in pseudo-code in Algorithm 1 and discussed below.

A sample $b^{acc} = 0$ indicates approximately constant velocity. As it is unlikely that motion induced by a human exhibits constant velocity over an extended period of time, it is likely that any sequence $(b_s^{acc}, \ldots, b_{t-j}^{acc}, \ldots, , b_t^{acc})$ with $s \ll t$ and $b_{t-j}^{acc} = 0$ for all $j \in [0, t-s]$ is generated by a stationary device.

At each time instant s with $b_{s-1}^{acc} = 1$ and $b_s^{acc} = 0$ a similarity map $\mathcal{M} : \{(S_s^{(i)}, \mathbf{x}_i)\}$ of pairs of hypothesis scores and trajectory locations in the current frame is generated and stored temporarily (Algorithm 1.13). The transition from $b_{s-1}^{acc} = 1$ and $b_s^{acc} = 0$ indicates that the accelerometer potentially stopped moving.

Once the length of the interval $[s, t]$ in which an accelerometer continuously measures no significant acceleration exceeds a threshold τ_t, the location of that accelerometer is temporarily estimated from the latest stored similarity map $\mathcal{M}$ (Algorithm 1.24). The

Algorithm 1 Accelerometer Tracking Algorithm

Input: Sequence of accelerometer data $(\mathbf{a}_1, \dots \mathbf{a}_T)$, where $\mathbf{a}_t : (a_x^t, a_y^t, a_z^t)$ and sequence of RGB-D images $(I_0, \dots, I_T)$.

Output: Sequence of estimated accelerometer locations $(\hat{\mathbf{x}}_1, \dots, \hat{\mathbf{x}}_T)$, where $\hat{\mathbf{x}}_t : (x_t, y_t)$.

$t \leftarrow 0$, where $t \in \mathbb{N}_0^+$
$b_t \leftarrow 0$, where $b_t \in \{0, 1\}$
$n \leftarrow 0$, where $n \in \mathbb{N}_0^+$ ▷ counts consecutive samples with approx. constant velocity
$\{P_i\} \leftarrow InitializeTrajectories(I_0)$, where $P_i : (\mathbf{x}_{t-j}^{(i)}, \dots, \mathbf{x}_t^{(i)})$
$\{S^{(i)}\} \leftarrow InitializeSimilarityScores(\emptyset)$, where $S^{(i)} \in \mathbb{R}_0^+$
$\mathcal{M} \leftarrow SimilarityMap(\{P_i\}, \{S^{(i)}\})$, where $\mathcal{M} : \{(S^{(i)}, \mathbf{x}_t^{(i)})\}$

for $t \leftarrow 1, \dots, T$ **do**
    $b_t \leftarrow 1\left[|\mathbf{a}_t| - |\hat{\mathbf{g}}| \geq \tau_a\right]$
    $\{P_i\} \leftarrow UpdateTrajectories(\{P_i\}, I_t)$
    $\{S^{(i)}\} \leftarrow UpdateSimilarityScores(\{P_i\}, \{S^{(i)}\}, b_t)$

    if $b_t = 0 \wedge b_{t-1} = 1$ **then** ▷ Accelerometer potentially stopped moving
        $\mathcal{M} \leftarrow SimilarityMap(\{P_i\}, \{S^{(i)}\})$

    if $b_t = 1 \wedge n > \tau_t$ **then** ▷ Accelerometer stops being stationary
        $\{S^{(i)}\} \leftarrow InitializeSimilarityScores(\mathcal{M})$

    if $b_t = 0$ **then**
        $n \leftarrow n + 1$
    else
        $n \leftarrow 0$

    if $n > \tau_t$ **then** ▷ Accelerometer is likely to be stationary
        $\hat{\mathbf{x}}_t \leftarrow PointEstimate(\mathcal{M})$
    else
        $\hat{\mathbf{x}}_t \leftarrow PointEstimate(\{P_i\}, \{S_t^{(i)}\})$

accelerometer's location $\hat{\mathbf{x}}$ is estimated as the constant location $\mathbf{x}_i$ of the entry i in $\mathcal{M}$ with highest similarity score $S_s^{(i)}$.

When the accelerometer starts moving again, i.e., $b_{s+k}^{acc} = 1$ for some positive k, the hypothesis scores of trajectories P_i are re-initialized (Algorithm 1.16). Three methods for re-initialization were considered and compared empirically.

Cold Start: All similarity scores are initialized to zero, i.e., $S_t^{(i)} = 0$ for all i.

Nearest Neighbour: The similarity score $S_t^{(j)}$ of trajectory hypothesis j with current location $\mathbf{x}_j$ is initialized to the similarity score $S_s^{(i)}$ in $\mathcal{M}$ whose corresponding location $\mathbf{x}_i$ is closest to $\mathbf{x}_j$.

Kernel Density Estimation: As similarity scores are only available for a sparse set of points in the image, selecting the score of a single nearest neighbor can be rather arbitrary. One method for combining the similarity scores of multiple nearby hypothesis locations is kernel density estimation (KDE). Kernel density estimation with a Gaussian kernel is a special case of a Gaussian mixture-model with equal weights assigned to all Gaussians (3.12). For similarity score initialization, 2D isotropic Gaussians are centered at all locations $\mathbf{x}_i$ in the similarity map $\mathcal{M}$.

$$S_t^{(j)} = \frac{1}{|\mathcal{M}|} \frac{1}{2\pi\sigma^2} \sum_i S_s^{(i)} exp\left(-\frac{(\mathbf{x}_j - \mathbf{x}_i)^T (\mathbf{x}_j - \mathbf{x}_i)}{2\sigma^2} \right) \tag{3.12}$$

It was found empirically that all re-initialization methods performed comparably, with nearest neighbour exhibiting slightly better localization precision on average.

3.7 Summary and Outlook

This chapter introduced an accelerometer localization pipeline in which sequences of accelerometer data are matched to sequences of local visual accelerations. Two methods for estimating acceleration sequences from visual data were discussed, using either a)

trajectories initialized at keypoint locations and tracked via KLT or b) trajectories initialized on a regular grid and updated using a dense optical flow field. An incremental similarity measure was developed that is arguably more robust than normalized cross-correlation against minor misalignment errors and errors resulting from the mismatch between instantaneous acceleration data from accelerometers and estimated mean acceleration between subsequent video frames. In order to address the problem of occlusion when an accelerometer-equipped object is stationary, a method was proposed that detects object-stationarity and re-initializes hypothesis scores once an object starts moving. The methods presented in this chapter are evaluated in chapter 7.1.

In the context of multi-modal activity recognition, visual accelerometer tracking will be used to establish correspondences between accelerometer data and locations in the visual field of the camera. Through these correspondences it is possible to model relations between accelerometer data and visual features extracted in spatial proximity to the estimated accelerometer locations. The next chapter introduces a feature representation that encodes such relations, and it will be argued that such an encoding can capture important properties of interactions between accelerometer-equipped objects and other visual entities.

Chapter 4

Relational Histograms for Activity Recognition

The main hypothesis of this thesis is that extracting features independently from accelerometer data and from video discards important information on cross-modal relations that is useful for activity recognition. The previous chapter introduced a method for associating data captured by accelerometers with locations in every frame of a video. This chapter investigates features that use this information. A family of feature descriptors called object-generic relational histograms is formalized, and a method for multi-modal recognition of activities from accelerometers and video data is described. Central to the proposed recognition method is one instance from the family of object-generic relational histograms: histograms of relative tracklets (RETLETS). It captures cross-modal relations by encoding relational properties between accelerometer displacements and generic visual motion descriptors (dense tracklets). A method for online activity recognition is proposed, in which RETLETS are combined with other feature descriptors prior to classification. Additional feature descriptors and multiple classifiers are described, which will be used for comparative evaluation.

4.1 Background

Information on the location and movement of key objects involved in activities can be highly descriptive for visual activity recognition. Recognition of complex interactions based on visually tracking all objects of interest often relies on high-level reasoning methods which are computationally demanding and domain specific [2, 68, 69, 84, 85]. These approaches require object detectors and tracking methods for all objects of interest. Acquisition and annotation of sufficient training data to build reliable object detectors is a very time-consuming and costly process, and certain classes of objects are hard to track visually (objects that are non-rigid, frequently subject to (partial) occlusion, or that occupy a small number of pixels, for example). Relying exclusively on object tracking for visual activity recognition is therefore impractical in many application areas. These include recognition of food preparation activities, which involve a potentially large number of different objects, and activity recognition in uncontrolled environments such as movies, YouTube videos, and sports recordings, where state-of-the-art methods employ sub-symbolic, generic local features (e.g., [111]). Methods for modelling activities from local features usually follow a bag-of-words approach encoding the occurrence frequency of codewords, essentially discarding spatial relations between features. Spatio-temporal pyramids address this issue to some extent by coarsely encoding feature co-occurrence, but they are very limited in accurately capturing interactions that span across spatial segmentation boundaries [55].

In some application areas where it is practically infeasible to track all objects it may however be possible to visually detect and track *some* objects reliably, or to acquire object location and displacement information through other means that do not rely on objects' visual appearance (e.g., accelerometer tracking). For these cases it would be useful to have a feature descriptor that encodes relations between properties of tracked objects and generic local descriptors. Section 4.3 introduces object-generic relational histograms, a family of descriptors that captures relations between generic local features and *reference features* extracted from some objects. This feature representation adapts the bag-of-words model

Figure 4.1: Example snaphots from the activities *place something into the bowl* and *mix the salad*. Tracklets (green) in the left and right image have similar shape. However, relative to the trajectory of the spoon (red), points in the left image move *towards* whereas most points in the right image move *in parallel to* this reference trajectory.

to scenarios in which some objects can be detected or tracked, and facilitates recognition of complex interactions with standard classification algorithms such as support vector machines (SVM).

Consider, for example, the activities *place something into the bowl* and *mix the salad*. Example snapshots from this pair of activities are depicted in Figure 4.1, which also shows dense tracklets ending in the depicted frames and trajectories of the spoon estimated via accelerometer localization. Examining the shapes of dense tracklets in both images, they appear to be similar. Relative to the trajectory of the spoon, however, points in the left image move *towards* whereas most points in the right image move *in parallel to* this trajectory. Exploiting this difference, a suitable relational feature descriptor could allow a classifier to easily discriminate this pair of activities in this case. One example of the family of object-generic relational histograms, histograms of RElative TrackLETS (RETLETS), which encodes dense tracklets relative to fixed length displacement sequences of tracked objects (subsequently called *reference tracklets*) is described in detail in Section 4.4.

For activity recognition, this chapter proposes to combine RETLETS with (i) features extracted from each sensor modality independently (Sections 4.2 and 4.5), and with (ii) statistical features from the visual trajectories of localized accelerometers (Section 4.5)

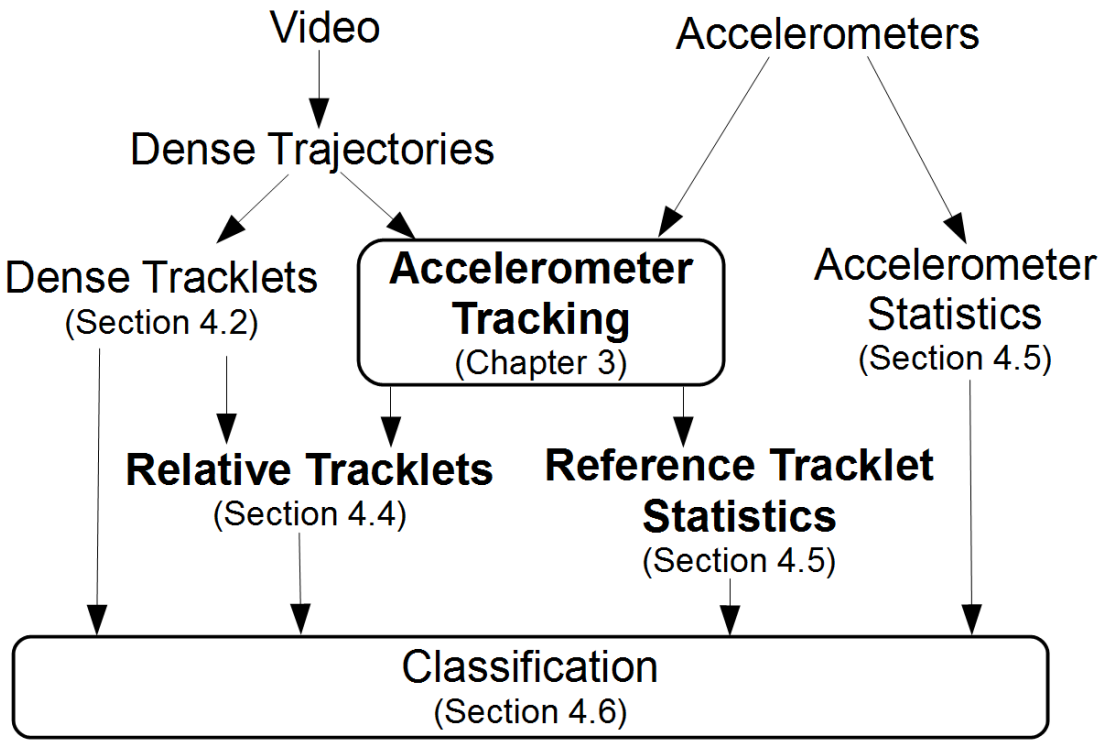


Figure 4.2: Overview of data flow in the proposed method. Stages involved in encoding cross-modal properties are highlighted in bold.

prior to classification (Section 4.6). The data flow is shown diagrammatically in Figure 5.1.

4.2 Dense Tracklets

The methods for activity recognition presented in this chapter are based on the concept of point tracking, specifically dense tracklets as proposed for visual activity recognition by Wang et al. [110]. In contrast to sparse point tracking, where tracks are initialized sparsely at salient image locations [89] and tracked by matching the image content in a confined local neighborhood [7], dense tracks are initialized on a regular grid and extended to the next frame using a dense optical flow field [19]. Generating point trajectories from dense optical flow produces fewer strong outliers compared to sparse point tracking due to the smoothness of the flow field [110]. Furthermore, as tracked points are not bound to be initialized at salient locations in the image, dense point tracking enables better coverage

of the motion of weakly textured surfaces by uniformly sampling locations in the image space.

As proposed in [110], point trajectories are initialized at locations $g \in G$ on a regular grid (with horizontal and vertical displacement d_G between grid locations) in each frame iff two conditions are satisfied: (i) none of the locations of active trajectories are within a $d_G \times d_G$ neighborhood around the grid point g and (ii) the minimum eigenvalue $min(\lambda_1^{(g)}, \lambda_2^{(g)})$ of the auto-correlation matrix of the image at location g is larger than the threshold $\tau_\lambda = 0.001 \cdot \quad max_{h \in G} min(\lambda_1^{(h)}, \lambda_2^{(h)})$. The displacement of a point from one frame to the next is estimated as the median-filtered dense optical flow field in a 3×3 neighborhood around the point's location in the previous frame.

Tracklets encode point trajectories $P : (\mathbf{x}_0, \ldots, \mathbf{x}_{L-1})$ of image coordinates $\mathbf{x} : (x, y)$ with fixed length L as $L-1$ displacements $\Delta\mathbf{x}_j : (x_{j+1} - x_j, y_{j+1} - y_j)$ that are normalized by the total length of displacements along the trajectory (4.1).

$$T = \frac{(\Delta\mathbf{x}_0, \cdots, \Delta\mathbf{x}_{L-2})}{\sum_{j=0}^{L-2} ||\Delta\mathbf{x}_j||_2} \tag{4.1}$$

Normalizing trajectories by their total length yields a scale-invariant representation and thereby emphasizes the trajectory's shape. Tracklets extracted from a spatio-temporal video window are encoded as a histogram over a dictionary of tracklets for classification. Tracklet dictionaries are obtained via k-means clustering of tracklets from a training set.

4.3 Object-Generic Relational Histograms

Consider a set $\{(\mathbf{f}_m, \mathbf{x}_m)\}_{m=1}^M$ of M local features $\mathbf{f}_m$ and corresponding locations in the image $\mathbf{x}_m$, and a set $\{(\mathbf{f}_n^{ref}, \mathbf{x}_n^{ref})\}_{n=1}^N$ of N reference features extracted from N tracked objects. Local features could be, for example, colour histograms, histograms of oriented gradients (HOG), histograms of optical flow (HOF), or motion boundary histograms (MBH).

This section proposes to encode interactions between local features and reference

features as pairwise relations $R(\mathbf{f}_n^{ref}, \mathbf{f}_m)$. Pairwise relations are mapped onto a codebook $\mathcal{C}$ using a quantization function $q(R(\mathbf{f}_n^{ref}, \mathbf{f}_m)) : \mathbb{R}^{|\mathbf{f}|} \times \mathbb{R}^{|\mathbf{f}|} \rightarrow [0,1]^{|\mathcal{C}|}$. The codebook could, for example, encode feature co-occurrence, joint local appearance, relative location or relative motion. A total of N histograms H_n are constructed, one for each reference feature, that encode frequencies of relations between one reference feature and all local features.

The contribution of each pairwise relation to a histogram is weighted by the likelihood of a meaningful interaction $w_{n,m}$. Weighted relational histograms $H_n \in \mathbb{R}^{|\mathcal{C}|}$ are constructed using (4.2) and (4.3).

$$H_n = \sum_{m=1}^{M} w_{n,m} q(R(\mathbf{f}_n^{ref}, \mathbf{f}_m)) \tag{4.2}$$

All weighted histograms are individually L_1-normalized. Each histogram H_n provides a different representation of the set of local features by encoding their relationship to one reference feature. Depending on the choice of relational codebook $\mathcal{C}$ and spatial weighting function this descriptor can encode meaningful interactions between a reference object and local visual features in its proximity. The presence of a meaningful interaction between a local feature and a reference feature is intuitively related to their spatial separation. It was chosen to weight the contribution of a feature $\mathbf{f}_m$ to a histogram H_n using a Gaussian function (4.3) with Euclidean distance for point features. For distances between point trajectories we use (4.6).

$$w_{n,m} = \exp(-\frac{D(\mathbf{x}_n^{ref}, \mathbf{x}_m)^2}{2\sigma^2}) \tag{4.3}$$

This formulation provides a generic model of relational histograms that can be used with a wide variety of local feature descriptors and pair-wise relations. One instance of this family is described in the next section, which is used for modelling interactions between dense tracklets and accelerometer-equipped objects.

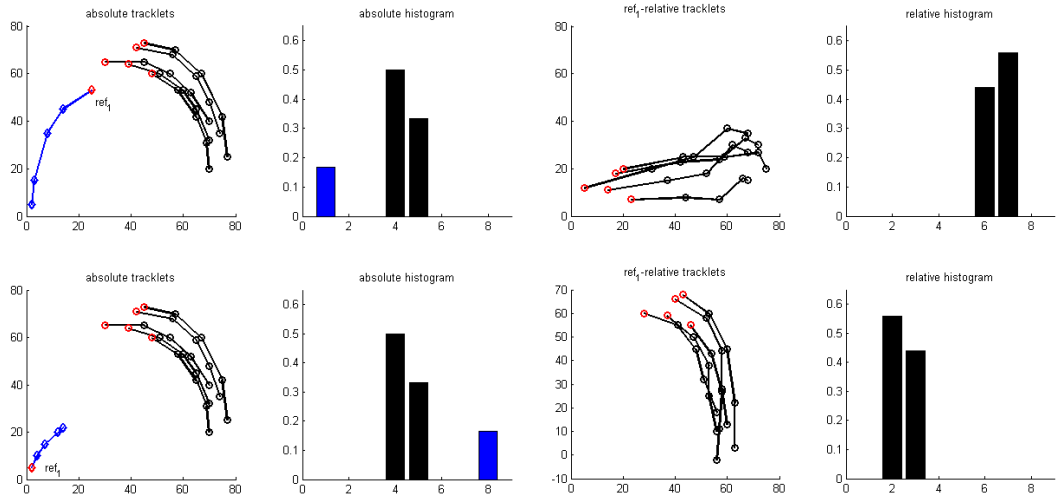
One type of relation - local co-occurrence - has been widely studied in the computer

vision literature (e.g., [42, 87, 100, 122, 125]). Correlograms [42] encode for each pair of codebook features a histogram over a fixed number of spatial relations. The length of this descriptor grows exponentially with the codebook size, which makes it very costly to use with common numbers of codewords between 400 and 4000 for local features. Several methods have been proposed to address this issue, including correlatons, which discard information on codeword identity [87], mining frequent [125] and mining discriminative [122] patterns of correlations. The model presented here offers a new way to reduce the descriptor length by considering pairwise relations between codewords of local features and features of a small fixed number of tracked objects. Here, the descriptor length is proportional to $N|\mathcal{C}|$, which is manageable if the number of tracked objects N is small. Another important difference of this model to the methods in the literature is the distance-dependent weighting function that emphasizes relations between features that are spatially close. Note, however, that through the weighting function in Equation (4.3) scale-invariance is lost.

4.4 Histograms of Relative Tracklets (RETLETS)

While feature co-occurrence may be a suitable second-order statistic for local appearance features, a *relative* description of local visual motion features better captures, in qualitative terms, interactions such as visual entities moving *towards*, *away from* and *around* each other (see Fig. 4.1). A descriptor encoding generic video tracklets relative to semantically meaningful reference tracklets acquired by tracking some objects is therefore more informative for complex interactions of multiple objects (see Fig. 4.3). This section proposes relational histograms using densely sampled fixed length point trajectories P_m as local features $\mathbf{f}_m$, using fixed length reference trajectories P_n^{ref} acquired through some form of object tracking as reference features $\mathbf{f}_n^{ref}$ and using relative tracklets as pair-wise relations.

Given a fixed length point trajectory P_m and a fixed length reference trajectory P_n^{ref} as defined on page 49 above Equation (4.1), the relative trajectory P_m^{rel} is defined as the



(a) (b)

Figure 4.3: Absolute tracklets and RETLETS. A toy example in which (a) histograms of absolute tracklets differ in only two bins whereas (b) tracklets relative to a reference tracklet (ref_1-relative) change their shape entirely and make the corresponding histogram representation more discriminative. Distance-based re-weighting of histogram entries adds further to discriminative power.

sequence of differences between point locations (4.4). The difference between a pair of point locations $(\mathbf{x}_0^{(m)}, \mathbf{x}_0^{ref})$ describes the location $\mathbf{x}_0^{(m)}$ relative to the location $\mathbf{x}_0^{ref}$, and the sequence of relative locations describes the motion of the visual entity tracked by P_m from the perspective of the reference feature P^{ref}. This relative motion is illustrated in Figure 4.3(b).

$$P_m^{rel} = ((\mathbf{x}_0^{(m)} - \mathbf{x}_0^{ref}), \cdots, (\mathbf{x}_{L-1}^{(m)} - \mathbf{x}_{L-1}^{ref})) \tag{4.4}$$

Similar to Equation (4.1), the relative tracklet R is defined as the sequence of normalized displacements along the relative trajectory P^{rel} as in Equation (4.5).

$$R = \frac{(\Delta\mathbf{x}_0^{rel}, \cdots, \Delta\mathbf{x}_{L-2}^{rel})}{\sum_{j=0}^{L-2} ||\Delta\mathbf{x}_j^{rel}||_2} \tag{4.5}$$

As tracklets are extracted along a sequence of points in the image, weights in Equation (4.3) are determined based on the mean pair-wise distance between locations along the corresponding point trajectories (Equation 4.6).

$$D(P_n^{ref}, P_m) = \frac{1}{L} \sum_{l=0}^{L-1} ||\mathbf{x}_l^{(m)} - \mathbf{x}_l^{ref}|| \tag{4.6}$$

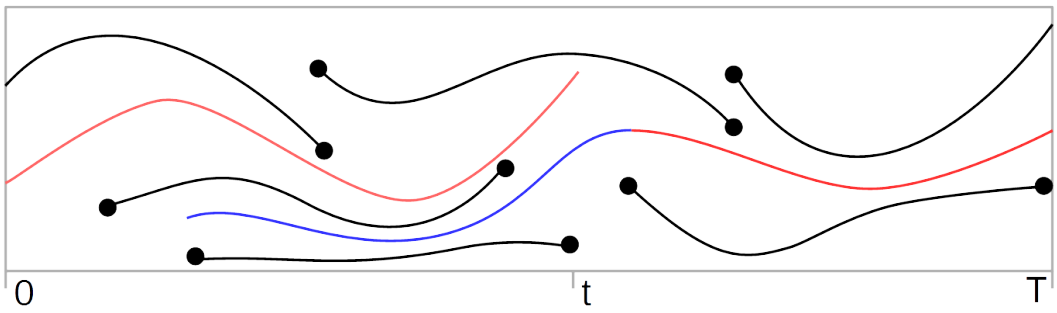
The relational codebook $\mathcal{C}$ is trained using k-means clustering on a training set of relative tracklets R_m. The Voronoi cells defined by the k cluster centers define the quantization function q. In every frame of a video, dense tracklets are extracted and reference tracklets are newly determined. For each reference tracklet, relative tracklets are constructed using all dense tracklets ending in the same frame using (4.4). All relative tracklets are then mapped onto the codebook $\mathcal{C}$ using quantization function q, and inserted to their respective histogram with their contribution weighted using (4.3) and (4.6). The process of extracting a histogram of relative tracklets from a temporal window within a video is illustrated in Figure 4.4.

4.4.1 Discussion of Related Work

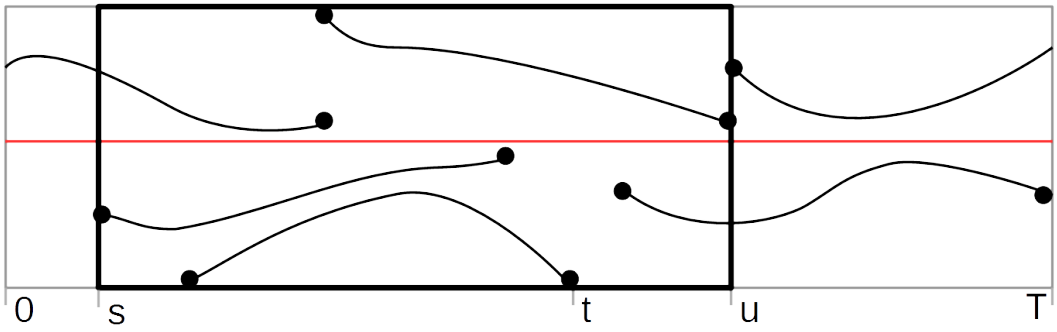
A degenerate case of this method, tracking a single object, a person's head, has been independently proposed by Bilinski et al. [6] in order to extract tracklets that are invariant to the person's translational motion in the image plane. Their motivation for employing relative trajectories is different in that their model aims to correct for unrelated motion of the person's body rather than to capture activity-specific interactions between the tracked object and dense tracklets. In that regard their approach is closely related to Wang and Schmidt [111], where dense tracklets are improved by correcting for camera motion and by discarding tracklets on the background.

4.4.2 Reference Tracklets from Accelerometer Localization

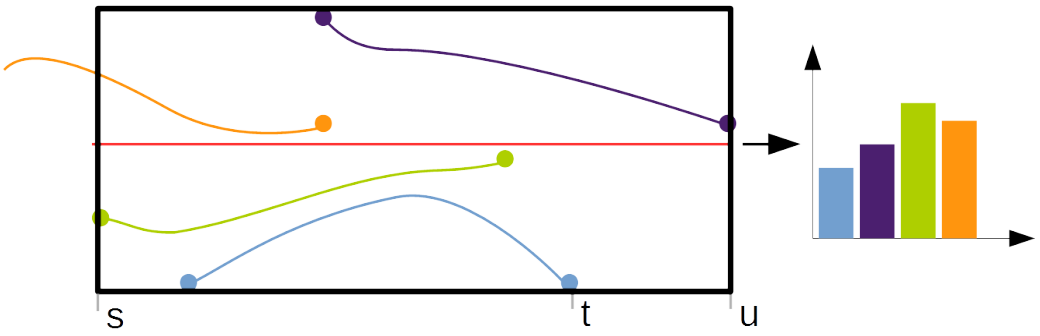
This thesis proposes to generate multiple reference tracklets using visual accelerometer localization as a method for tracking accelerometer-equipped objects. For each accelerometer-equipped object one reference tracklet is created from the most recent $L-1$ displacements of the point trajectory that best matches the data captured by its accelerometer. One



(a) Generic fixed length point trajectories (black) and estimated object trajectories (red). The blue line indicates the history of the point trajectory associated with the tracked object after time t before it becomes the best match.



(b) Relative point trajectories wrt. the tracked object. The first object trajectory serves as reference trajectory for all point trajectories ending before time t. The second object trajectory and its history are used for all point trajectories ending after time t.



(c) When encoding RETLETS over a temporal subwindow from time s to time u, all relative trajectories ending in the interval $[s, u]$ are inserted into a histogram.

Figure 4.4: Construction of RETLETS within the temporal window $[s, u]$ from generic fixed length point trajectories and object trajectories.

histogram over relative tracklets is constructed for each accelerometer equipped object and the final descriptor is the concatenation of these histograms. Note that the number of accelerometer-equipped objects is assumed to be known and fixed. Analysis of relations between accelerometer motion and visual features in spatial proximity to the device provides context for accelerometer motion and allows for joint reasoning about *how* other objects in the scene move relative to accelerometer-equipped objects. This information provides a crucial link between dense tracklets in the camera view and features extracted from accelerometer data, that enables a classifier to discover descriptive cross-modal relations.

4.4.3 Discussion of Joint Appearance using Relational Histograms

In order to illustrate how relational histograms can be used with other common local visual features, let us consider a joint appearance model with local HOG descriptors extracted along point trajectories as proposed by Wang et al. [110]. The dense local HOG descriptors would serve as generic features $\mathbf{f}_m$, and a HOG descriptors along the point trajectory of each tracked object would represent the reference features $\mathbf{f}_n^{ref}$. The joint appearance $R(\mathbf{f}_n^{ref}, \mathbf{f}_m)$ could be modeled by concatenating both local appearance descriptors, optionally reducing the dimensionality of the relational descriptor using principal component analysis (PCA). The relational codebook $\mathcal{C}$ would be learned using k-means clustering applied to a training set of joint appearance descriptors. As local HOG descriptors in this example are extracted along point trajectories, Equation (4.6) would be equally applicable to determine the weights of histogram entries here. If local HOG descriptors were extracted around fixed locations in the space-time volume, the Euclidean distance between these locations would be used to determine weights instead.

4.5 Auxiliary Features

The feature descriptor presented in the previous section - RETLETS - provides a rich representation of the motion in the scene by describing local motion features relative to the motion of tracked objects. Since objects are tracked using accelerometer localization, which works well but fails in some cases, it is advisable to also use the original dense tracklets, which capture motion relative to the camera viewpoint (subsequently called *Absolute Tracklets*) for recognition. Note that up until this point, little motion information captured by accelerometers is encoded in the set of feature descriptors. Even the displacement of the visual trajectory that best matches the accelerometer's data is only used as a reference frame for dense tracklets, and none of the trajectory's characteristics are encoded explicitly. Both features encoding the data captured by accelerometers and features extracted from the associated visual trajectory are expected to provide complementary information to RETLETS and absolute tracklets. Therefore, additional features, *Accelerometer Statistics* and *Reference Tracklet Statistics* are included for activity recognition. Two further feature types are considered as simple baseline features for comparative evaluation: *Object Use* extracted directly from accelerometer data and *Device Locations* extracted from accelerometer localization results. All of these feature types are described below.

4.5.1 Accelerometer Statistics

Accelerometer features that have previously been shown to give good performance on a recognition task involving food preparation actions are used [75]. Statistical features are estimated from acceleration sequences $A : ((a_x^0, a_y^0, a_z^0), \ldots (a_x^t, a_y^t, a_z^t))$ over a temporal window of fixed length N. The following statistics are estimated from each 3D component of A individually, formalized using the a_x-component as an example:

- Mean: $\mu_x = \frac{1}{N} \sum_{i=t-N+1}^{t} a_x^i$
- Standard Deviation: $\sqrt{\frac{1}{N-1} \sum_{i=t-N+1}^{t} (a_x^i - \mu_x)^2}$

- Energy: $\sum_{i=t-N+1}^{t} |a_x^i|^2$
- Entropy: $-\sum_{j=0}^{K-1} p_x^j log_2(p_x^j)$,

  where p_x is a normalized K-dimensional histogram over a_x^i with $i = t-N+1, \ldots, t$.

Additionally, pitch and roll are estimated from four temporal subwindows evenly spaced with 50% overlap. Pitch and roll encode the device's orientation relative to the direction of gravity and can be estimated from accelerometer data because the data represents proper acceleration (relative to free fall). The yaw angle cannot be recovered from accelerometer data because the yaw angle describes rotation around the axis that is aligned with the direction of gravity. This set of features encodes each accelerometer's motion with a 20-dimensional feature vector, and features extracted from all devices are concatenated.

4.5.2 Reference Tracklet Statistics

Global characteristics of the shape of reference tracklets are encoded by *Reference Tracklet Statistics* with one feature vector extracted for each accelerometer-equipped object. From the most recent $L-1$ displacement vectors of the point trajectory that best matches an accelerometer's data, the mean, standard deviation, energy and entropy are estimated separately for displacements in x- and y-coordinates in the image. These features are concatenated to form a single feature vector characterising the visual motion of all devices.

4.5.3 Object Use

For benchmarking purposes simple features are used that encode whether an accelerometer-equipped object is used at a given time. This feature type will be used to evaluate the importance of identifying the objects involved in activities for activity recognition. Following the argumentation in Section 3.6, an accelerometer is stationary if it measures no significant acceleration over an extended period of time. Assuming an accelerometer is in use if and only if it is moving, object use at time t can be defined as in (4.7).

$$InUse(B^{acc}) = \left(\sum_{j=0}^{\tau_t - 1} b_{t-j} \right) > 0, \tag{4.7}$$

where b and τ_t are defined as in Sections 3.5.2 and 3.6, respectively.

In order to extend battery life, the wireless accelerometers used in the experiments of this thesis stop data transmission if the measured magnitude of acceleration does not exceed a noise threshold over a fixed number of consecutive samples N^*. Whether an accelerometer is not $InUse$ at a given time could therefore be detected by checking whether the accelerometer sent any data. With this approach not *InUse* is detected with a delay of $N^* - 1$ samples. This delay could be reduced using Equation (4.7) with $\tau_t < N^*$. However, decreasing the temporal interval also increases the uncertainty about whether an object is stationary or moving with constant velocity. Preliminary evaluation results have shown that recognition performance does not change significantly with $\tau_t \neq N^*$.

4.5.4 Device Locations

In order to gain further insight into which type of information is important for recognizing food preparation activities, a feature type that encodes for each accelerometer-equipped object its location (x, y) in the most recent frame is encoded. An accelerometer's location is set to the most recent location of the best matching point trajectory identified by the accelerometer localization algorithm.

4.5.5 Discussion

Note that *Absolute Tracklets* encode information extracted from video data, *Accelerometer Statistics* and *Object Use* encode information extracted from accelerometer data, and *RETLETS*, *Reference Tracklet Statistics*, and *Device Locations* capture information from both sensor types. By evaluating each feature type independently and different combinations of these feature types with respect to activity recognition performance we will gain a detailed understanding of (i) which types of information are important for recognizing

food preparation activities, and importantly, whether including *RETLETS* does enable the classifier to discover strong cross-modal relations.

4.6 Online Activity Recognition

4.6.1 Temporal Sliding Windows

Using the features described in the previous sections this thesis aims to recognize activities online. Online recognition involves recognition of activities in videos containing sequences of multiple activities without knowing the temporal extent of each activity instance, i.e. the start and end frames of activity instances are unknown. This thesis employs a sliding window approach in which features are extracted from a fixed length temporal window (windows with temporal length of 5s were used, and tracklets with length L=16 were extracted). For inference, the temporal window is shifted by a single frame at a time and features from each temporal window are classified independently. The classification results of each temporal window are not combined or processed further. For classifier training an overlap between temporal windows of up to 75% is considered rather than extracting training samples from temporal windows placed at every frame. This substantially reduces the number of training samples while maintaining recognition performance and better approximates the independence assumption of training samples made by common classifiers.

4.6.2 Classifiers

Support Vector Machine

The support vector machine (SVM) is a margin-maximizing classifier that is particularly popular in the computer vision community. Kernel-SVMs learn to separate training samples from different classes in a possibly infinite dimensional hyperspace without explicitly projecting the training data into this high-dimensional space. Instead, optimization is

performed based on a kernel matrix, which specifies for each training sample its similarity to a set of training functions. This thesis uses the Gaussian radial basis function (RBF) kernel and the exponential χ^2 kernel. The RBF kernel estimates the similarity between all pairs of training samples using the squared Euclidean distance (Equations (4.8) and (4.9)). The exponential χ^2 kernel uses the χ^2 distance, which is more suitable to comparing normalized histograms (Equations (4.10) and (4.11)).

$$K_{RBF}(\mathbf{f}_i, \mathbf{f}_j) = exp\left(-\gamma_{RBF} D_{RBF}(\mathbf{f}_i, \mathbf{f}_j)\right), \text{ with} \tag{4.8}$$

$$D_{RBF}(\mathbf{f}_i, \mathbf{f}_j) = ||\mathbf{f}_i - \mathbf{f}_j||^2 \tag{4.9}$$

$$K_{\chi^2}(\mathbf{f}_i, \mathbf{f}_j) = exp\left(-\gamma_{\chi^2} D_{\chi^2}(\mathbf{f}_i, \mathbf{f}_j)\right), \text{ with} \tag{4.10}$$

$$D_{\chi^2}(\mathbf{f}_i, \mathbf{f}_j) = \sum_{k=0}^{|\mathbf{f}|} \frac{(\mathbf{f}_i^k - \mathbf{f}_j^k)^2}{\mathbf{f}_i^k + \mathbf{f}_j^k} \tag{4.11}$$

SVMs are binary classifiers for which two multi-class extensions have been proposed. One-vs-one multi-class SVMs train one binary classifier for each pair of classes, and all classifiers vote for one class during inference. A test sample is classified as the class that receives most of the votes. One-vs-rest multi-class SVMs train one binary classifier for each class with negative training samples from all other classes. Using classification scores, a test sample is classified as the class with the highest score.

In this thesis, features extracted from temporal sliding windows are classified using one-vs-one multi-class SVMs. For *Absolute Tracklets* and *RETLETS* the exponential χ^2 kernel (4.10) is used with $\gamma_{\chi^2} = \frac{1}{A}$, where A is the average distance between training histograms [124]. For *Accelerometer Statistics*, *Reference Tracklet Statistics*, *Object Use* and *Device Locations*, features extracted for each accelerometer were concatenated and feature vectors were compared using the RBF kernel (4.8) after scaling all dimensions

individually to $[-1, 1]$. For these feature types γ_{RBF} was determined by cross-validation.

Kernel Combination SVM classification from multiple feature types involves combining the kernel matrices computed for each feature type. This thesis considers three methods for kernel combination. Given a set of F feature types $\mathbf{f}^{(f)}$, $f = 1, \ldots, F$, distance matrices $D^{(f)}$, scaling parameters $\gamma^{(f)}$, and kernel matrices $K^{(f)}$ the following kernel combinations are considered:

- $SumDistance : exp(-\sum_{f=1}^{F} \gamma^{(f)} D^{(f)}(\mathbf{f}_i^{(f)}, \mathbf{f}_j^{(f)})$
- $MeanDistance : exp(-\frac{1}{F} \sum_{f=1}^{F} \gamma^{(f)} D^{(f)}(\mathbf{f}_i^{(f)}, \mathbf{f}_j^{(f)})$
- $SumKernel : \sum_{f=1}^{F} K^{(f)}(\mathbf{f}_i^{(f)}, \mathbf{f}_j^{(f)})$

As $e^{(a+b)} = e^a \cdot e^b$, $SumDistance$ and $MeanDistance$ assume individual feature types to be independent. It is expected that $SumKernel$ performs better when combining strongly correlated feature types such as *Absolute Tracklets* and *RETLETS*.

Random Decision Forest

Random decision forests are non-linear classifiers that naturally extend to multi-class classification and produce well calibrated posterior probabilities. These characteristics make this classifier favorable compared to the popular support vector machine (SVM) [16], if the whole class posterior distribution is used for further processing. This thesis investigates feature fusion at different stages of the recognition pipeline, and random decision forests will be used as base classifiers for late fusion, i.e. combining feature types after classification.

A random forest is an ensemble of random decision trees, where each tree is trained in isolation [16]. Each internal node of a decision tree represents a weak classifier in the form of a binary decision function. Starting at the root node, a random subset of the set of weak learners is selected. This random subset of learners is evaluated in combination with a small number of randomly selected thresholds against the information gain criterion.

The weak classifier with highest information gain on the training data for this node is selected. The weak classifier divides the training data into two partitions. The left and right child node are subsequently trained based on their respective training data partitions. Leaf nodes store the distribution of training samples arriving in a given node over classes. This thesis uses axis-aligned weak classifiers, which simply compare the value of a single dimension of a feature vector with a threshold. Using axis-aligned weak classifiers makes it particularly easy to combine multiple types of feature vectors; the vectors only need to be concatenated. Using more complex classifiers in the tree nodes is associated with a significantly higher computational cost for training. Preliminary evaluation on our data showed that more complex classifiers did not improve recognition performance.

In order to deal with unbalanced training data, the contribution of samples from different classes to (i) information gain and (ii) class distributions in the leaf nodes are weighted differently. The weight of a sample from class $c \in C$ is set to be inversely proportional to the number of samples n_c from that class in the training set:

$$w_c = max_{u \in C}(n_u)/n_c \tag{4.12}$$

In the limit of an infinite number of training samples this approach is equal to stratification, i.e., selecting an equal number of samples per class.

In the inference stage a feature vector traverses all trees starting at the root node, descending to the next node depending on the evaluation of the weak classifiers in the current node on the test sample. The class distributions of the destination leaf nodes are summed and normalized yielding a posterior distribution over activity classes given the test sample.

The meta-parameters of a random forest specifying (i) the number of decision trees, (ii) the maximum depth of each tree, (iii) the number of randomly selected features tested in each node, and (iv) the number of thresholds tested per feature need to be set prior to forest training. We attempt to find good values for these parameters through model selection. Automatic model selection involves choosing the model that minimizes a loss-function.

As our random forests select features for each node to maximize information gain, the cross-entropy error is used as the loss-function for model comparison. The cross-entropy error for a single datapoint is defined as

$$H(p,q) = -\sum_{c} p(c) log_2(q(c)) = -log_2(\hat{p}(c_{gt}|o)), \tag{4.13}$$

where $p(c)$ is the ground-truth class distribution (delta-function with peak at true class label c_{gt}) and $q(c)$ is the class posterior $\hat{p}(c|o)$ estimated by the recognition algorithm. In this special case the sum only contains a single non-zero element, which is the log of the estimated probability for the ground-truth class. We compute the per class cross-entropy error and sum over all classes in order to obtain a single performance measure given in (4.14).

$$H_m = \sum_{c} \frac{1}{n_c} \sum_{i:p_i(c)=1} H(p_i, q_i) \tag{4.14}$$

Since cross-entropy is estimated from cross-validation it can be regarded as a random variable with fluctuation around the mean. Treating model-selection as a regression problem with cross-entropy as its error function, we handle the bias-variance trade-off by selecting the model that minimizes $mean(H_m)^2 + variance(H_m)$.

Fusion Methods

In addition to combining information from accelerometers and video data through accelerometer localization to extract *RETLETS*, *Reference Tracklet Statistics*, and *Device Locations* (*data-level fusion*), fusion at feature level by concatenating feature vectors (*early fusion*) and fusion at classifier level (*late fusion*) are considered. For classifier fusion, the sum-rule and the product-rule [48] as well as Random Decision Forests are considered as a non-linear combination method.

4.7 Summary

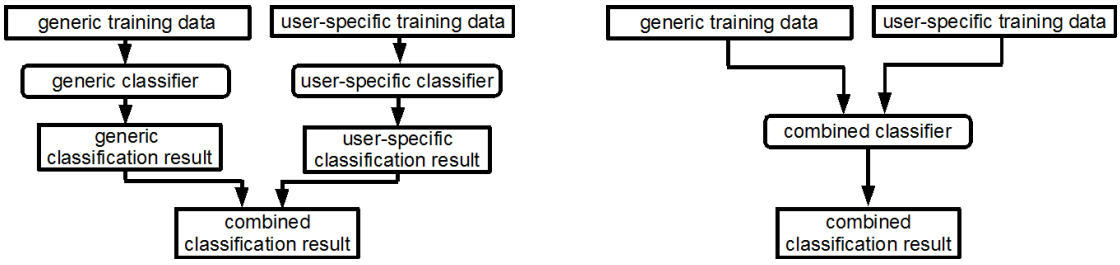
This chapter introduced a family of feature descriptors, object-generic relational histograms, that model relations between generic local features and a set of reference features using histograms over a bag of relations. One example from this family - RETLETS - was proposed, which models relations between dense tracklets and reference tracklets using relative tracklets. For activity recognition from video and accelerometer data it was proposed to encode RETLETS using reference tracklets from accelerometer-equipped objects, which are determined via the accelerometer tracking algorithm developed in Chapter 3. Additional features were discussed that will be either combined with RETLETS for multi-modal activity recognition or used as baseline features for comparative evaluation. Finally, two classifiers, SVMs and random decision forests were discussed, and it was described how multiple features are combined for early fusion using these classifiers.

Chapter 5

User-Adaptive Classification

One dominant characteristic of manipulation activities is that individual tasks such as "mix the ingredients" are often defined by the states of the manipulated objects after successful task completion (the ingredients are mixed), and one can choose from a potentially wide spectrum of strategies (e.g., mixing ingredients by hand, using a pair of spoons, or by shaking the bowl). As we are commonly believed to be creatures of habit, it is reasonable to hypothesize that each person tends to choose the same strategy each time he is confronted with the same task. If a recognition system is used primarily by a single individual, it could be possible to improve recognition accuracy for that individual by *tailoring* the system to his idiosyncrasies.

This chapter investigates ways of exploiting personal preferences and habits in task execution for activity recognition. The proposed methods adapt a discriminative activity model that has been trained on data from multiple individuals using some small amount of labeled data from an additional individual such as, for example, the primary end-user of the recognition system. While the main contribution of this thesis is the application of these methods to activity recognition, they do not exploit domain knowledge and are therefore easily transferrable to other domains (in which the success of these methods may, of course, vary).



(a) Classifier combination (b) Joint classifier training

Figure 5.1: User-adaptive recognition. (a) Classifier combination in which the classification results of generic and user-specific discriminative classifiers are combined at test time. (b) A single classifier is trained on both generic and user-specific data.

5.1 Background

5.1.1 Motivation

Usually a classifier is trained with the goal of maximizing generalization performance. An activity recognition system would be trained on data from a large sample of individuals that are assumed to be drawn from the same distribution as the individuals whose activities the system is to recognize after deployment. With a potentially large inter-person variability in activity execution, the problem of distinguishing a pair of activities performed by two randomly drawn individuals is comparably harder than distinguishing that pair of activities performed by the same individual. For example, if an activity may be performed with a wide range of strategies in general, but as a creature of habit the individual always chooses the same one, the intra-class variability of activities performed by that individual will be smaller. If a recognition system is expected to be used primarily by that individual, it would be sufficient to generalize well across variations of his chosen strategy, which can be easier than attempting to achieve good generalization performance across all possible strategies.

In order to discover and train a classifier on intra-person variations one would need data from that individual. In practice, acquiring this data is often costly and limited to a small number of samples. It is therefore likely to be insufficient to train a classifier on this data exclusively. A more promising approach, which has been applied successfully in domains

such as speech recognition [101] and handwriting recognition [71], is to fine-tune or adapt a classifier that was trained on generic data using some small amount of data from the end user.

This chapter explores the potential of user-specific classification of manipulation activities with limited data from the target individual. Three simple methods for adapting generic, stereotypical activity models to specific individuals whose data are not available when training the generic models are investigated. Firstly, a user-adaptive discriminative recognition model is presented in which a generic and a user-specific classifier are trained independently and merged at test time by combining class posterior probabilities (see Figure 5.1a). This method does not require all generic training data to be available at the time the model is adapted to a particular subject, which can be desirable for practical reasons of system deployment. The second model involves training a single support vector machine jointly on generic and user-specific data (see Figure 5.1b). This approach requires the whole model to be retrained every time new user-specific data become available. Thirdly, a K-Nearest-Neighbour classifier is proposed in which different probability mass is assigned to user-specific and generic training samples (the data flow in this model can be also represented by the diagram in Figure 5.1b) . While K-Nearest-Neighbour classification allows user-specific data to be added at any time with no cost for retraining, maintaining the entire set of generic data may be practically infeasible. The contributions of this chapter have been previously published in part in [97].

5.1.2 Related Work

User-adaptive classification has been an active research area in handwriting [71], speech [36, 101], gesture recognition [58, 60], and recognition of activities from accelerometer data [4]. Existing methods vary widely in the level of user effort required for adaptation, both initially (when the user interacts with the system for the first time) and continuously (during normal everyday interaction with the system).

User-directed personalization [58, 60] requires the user to provide example activity

executions and the corresponding labels. These approaches can be initialized with as little as a single training example per activity and provide maximum flexibility by allowing users to freely add new activities. During normal interaction these methods depend on the user to communicate recognition errors back to the system and to provide the intended/ correct label for falsely recognized activities, which can be extremely burdensome. Reducing the effort of providing feedback during normal interaction, Foster et al. [26] proposed to automatically adapt to recognition failures that are detected from error signals in electroencephalography (EEG) data.

Initially driven by the speech recognition community, major advances have been made in fully automatic adaptation of recognition systems to target users and their environments. He and Zhao [36] propose a method using dynamic model selection, which identifies one out of a large set of pre-trained classifiers that matches best a speaker's characteristics. Tang et al. [101] represent speaker idiosyncrasies by a vector in a low-dimensional space representing variation across speakers, which is adapted using maximum-likelihood estimation on user-specific data. While these approaches could be similarly applicable to activity recognition, a large number of classifiers needs to be trained which cover all possible user-idiosyncrasies and all possible environments in which the system shall be used. This inherent inability to adapt to new environments severely limits the applicability of this approach to manipulation activities such as preparing food. Recently, Wen-Sheng et al. [114] introduced a promising method that can adapt to unforeseen contexts, requires less training data, and also does not require labelled data from the end user. Their Selective Transfer Machine trains a new classifier for the end user from generic training data. Individual samples are re-weighted in an unsupervised way based on unlabelled user specific data, which effectively adapts the recognition system to the end-user's idiosyncrasies. Although this method has been proposed for facial action unit recognition, it may be similarly applicable to multi-modal activity recognition. For adapting to gradual changes in sensor placement, Forster et al. [27] proposed a method that tracks clusters of activities in feature space. This method seems particularly promising for online adaptation

of situational support systems for people with impairments that become more severe or improve over time.

This chapter focuses on methods for one-off user-adaptation based on a small set of labeled user-specific training examples. As labeled data contains more information than unlabeled data, a supervised setting is well suited to exploring the potential of user-adaptation by providing an indication of an upper bound for achievable recognition performance increase. Furthermore, initial adaptation based on a small set of labelled data appears to be more acceptable and less distracting in practice than continuously prompting the user for labels and confirmations of recognized activities. Although not considered in this thesis, a combination of initial adaptation using labelled data with continuous refinement based on unlabeled data is expected to increase recognition performance further without additional effort required from the end-user.

5.2 Adaptation by Classifier Combination

Before introducing the first method for classifier adaptation it is useful to establish some naming conventions. Data from a set of individuals that does not include the end-user is called *generic* data, and data from the end-user is referred to as *user-specific* data. Analogously, a generic classifier is trained exclusively on generic data, and a user-specific classifier is trained exclusively on user-specific data. In mathematical notation, expressions related to generic and user-specific data are indicated with subscript g and subscript s, respectively. Note that the terms user-specific classification and classifier adaptation are used interchangeably, as all methods subsequently discussed in this chapter address user-specific classification by adapting a recognition system through modifying or adding classifiers.

The problem of adapting a generic classifier to a target user can be formulated as classification with multiple experts, a problem that is well established in the machine learning literature [48]. One expert, a generic classifier, is trained to generalize well across

individuals, and another expert, a user-specific classifier, establishes a good model of the activity patterns observed from the target user. From this perspective, user-adaptive classification involves training a separate classifier from *user-specific* training data and combining the classification results obtained by the generic and user-specific classifiers at test time. Given two classifiers that estimate posterior probabilities $p(c|\mathbf{f})$ of activity-classes c given feature vectors $\mathbf{f}$, these distributions can be combined by taking their weighted sum,

$$p(c|\mathbf{f})_{comb} = w_g p(c|\mathbf{f})_g + (1 - w_g) p(c|\mathbf{f})_s, \tag{5.1}$$

where $w_g \in [0, 1]$ is the weight of the contribution the generic classifier makes to the combined classification result. This weight is a free parameter that can be used to control over-fitting and to adjust the relative confidence in the classification results of generic and user-specific classifiers, which depends, for example, on the number of available user-specific training data points. The activity class with maximum $p(c|\mathbf{f})_{comb}$ is considered to be the recognized activity. This adaptation method was evaluated with support vector machines, whose outputs were transformed to posterior probabilities using Platt scaling as provided by the LibSVM[1] library.

5.3 Adaptation by Joint Classifier Training

Adapting a recognition system to a target user by combining generic and user-specific classifiers does not require the full set of generic training data to be available at the time the system is adapted to a target user. The late combination of recognition results may, however, yield suboptimal recognition performance and jointly training a single classifier from both generic and user-specific training data may prove to be superior. This section introduces two models for joint classifier training and shows how the relative contribution of generic and user-specific data may be modified.

[1] LibSVM: `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`

5.3.1 Joint SVM Training

As SVMs have shown competitive performance on many recognition tasks and are particularly popular in the computer vision community, it is of particular interest to analyze this classification model with respect to user-adaptation. The primal optimization problem of the C-SVM formulation with weight vector $\mathbf{w}$, kernel function $\phi(\mathbf{f})$, bias b, feature vector $\mathbf{f}_i$, labels $y^{(i)}$ and slack-variables $\xi^{(i)}$ for $i = 1, \ldots, N$ training samples is given by Equation (5.2).

$$\begin{aligned} min_{\mathbf{w},b,\xi} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N}\xi^{(i)}, \\ s.t. \quad & y^{(i)}(\mathbf{w}^T\phi(\mathbf{f}_i) + b) \geq 1 - \xi^{(i)}, \\ & \xi^{(i)} \geq 0 \end{aligned} \tag{5.2}$$

The contribution of the set of generic training data and the set of user-specific training data to the objective function can be controlled by the relative number of samples in these subsets. If the number of user-specific samples is greater than the number of samples from generic training data the relative cost of misclassifying user-specific data is greater than the cost of misclassifying generic data. If the number of training samples are equal, generic training samples are taken from a large number of different activity-sequences, and user-specific training data is only available for, e.g., a single sequence, then the cost of misclassifying samples from the user-specific sequence is proportionally higher than the cost of misclassifying samples from each sequence in the generic training data set. Alternatively, one can define a different cost of misclassification C_g and C_s for generic and user-specific training samples (5.3). The sets of generic and user-specific training samples are denoted as $\{\mathbf{f}\}_g$ and $\{\mathbf{f}\}_s$, respectively.

$$min_{\mathbf{w},b,\xi} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} \quad + C_g\left(\sum_{\{i|\mathbf{f}_i\in\{\mathbf{f}\}_g\}}\xi^{(i)}\right) \quad + C_s\left(\sum_{\{i|\mathbf{f}_i\in\{\mathbf{f}\}_s\}}\xi^{(i)}\right) \tag{5.3}$$

5.3.2 Weighted K-Nearest-Neighbour

A simple yet effective discriminative classification algorithm is K-Nearest-Neighbour [70]. This algorithm does not involve any learning but needs all training data to be available at test time. The class-posterior distribution given an observation is estimated by identifying the set $NN_K(\mathbf{f})$ of K training samples that are closest to the observation in feature space given some distance metric. Let K_c be the number of training samples from class c in this set. The class posterior distribution is defined as the proportion of samples from class c in the set, i.e.,

$$p(c|\mathbf{f}) = \frac{K_c}{K}. \tag{5.4}$$

This treats all training samples as being equally important. A generalization of Equation (5.4) assigns a different probability mass to each training sample, i.e.,

$$p(c|\mathbf{f}) = \left(\sum_{\{i:\mathbf{f}_i \in NN_K(\mathbf{f}), \mathbf{f}_i \in \{\mathbf{f}\}_c\}} m^{(i)} \right) / \left(\sum_{\{j:\mathbf{f}_j \in NN_K(\mathbf{f})\}} m^{(j)} \right), \tag{5.5}$$

where the set of training samples from class c is denoted as $\{\mathbf{f}\}_c$. Equation (5.5) is equivalent to Equation (5.4) if $m^{(i)} = 1/N$ for all training samples. For user-adaptive classification, the relative contributions of generic and user-specific data to the estimated class-posterior probability can be adjusted using Equation (5.5) by assigning probability mass m_g to all generic samples and probability mass m_s to all user-specific samples. If the weight for generic training samples is much greater than the user-specific weight ($m_g \gg m_s$) then generic samples dominate Equation (5.5). Therefore, the impact of user-specific training samples on the recognition result is negligible if there is at least one generic sample among the K nearest neighbours. In the inverse case ($m_s \gg m_g$) the recognition result depends almost exclusively on user-specific training samples assuming at least one user-specific sample is among the K nearest neighbours.

While multiple feature types for SVM classification are combined via kernel combination (see Section 4.6.2, p. 61), a combined distance metric needs to be defined for K-Nearest-Neighbour classification from multiple feature types. For evaluation in this

Adaptation Method	All generic training data required for user-adaptation	Training time for user-adaptation	Test time after user-adaptation
Classifier Combination	no	medium	medium
Joint SVM Training	yes	high	low
Weighted K-Nearest-Neighbor	yes	low	high

Figure 5.2: Qualitative comparison of the three methods for user-adaptive classification methods discussed in this chapter.

thesis a weighted sum of individual distances in each feature space as in Equation (5.6) was computed, with feature-type dependent weighting coefficients γ_f and distance metrics D_f as described in Section 4.6.2, p. 60.

$$D(\mathbf{f}, \mathbf{f}_i)_{comb} = \sum_f \gamma_f D_f(\mathbf{f}^f, \mathbf{f}_i^f), \tag{5.6}$$

5.4 Comparison of User-Adaptation Methods

The three methods for user-adaptive classification discussed in this chapter have different characteristics with respect to the amount of generic training data required for user-adaptation, the amount of time required for user-adaptation, and the amount of classification time required after user-adaptation. A qualitative comparison of these characteristics is shown in Figure 5.2. Note that statements about computation time assume sequential processing and estimates for K-Nearest-Neighbour assume that an exact search is performed.

In contrast to joint SVM training and weighted K-Nearest-Neighbour, classifier combination does not require all generic training samples to be available for user adaptation. As a result, such a system may have substantially lower requirements on data storage (depending on the ratio of model parameters to the number of training samples times the feature dimensionality), which is particularly important for mobile and pervasive embedded systems.

As K-Nearest-Neighbour does not require any model training, the time required for

user adaptation in this model is comparably low. Among the three models considered, the time required for user adaptation is highest when training a joint kernel-SVM from generic and user-specific data as $(N_g + N_s)^2$ entries of the kernel matrix need to be computed. For combining generic and user-specific kernel-SVMs, the kernel matrix of user-specific data of size N_s^2 can be substantially smaller.

In the inference stage, exact K-Nearest-Neighbour requires most computation time as the distance between the test sample and all training samples need to be computed . Computation time for SVM classification is proportional to the total number of support vectors. As SVM-classifier combination employs two disjoint sets of support vectors their total number is generally higher than the number of support vectors in a jointly trained SVM but usually smaller than the total number of training examples. This analysis aims to give a general intuition for relative computational cost; in practice the ranking of classifiers with respect to computational cost varies with classifier implementations and datasets.

2D Toy Example

In order to give a better understanding of the adaptive behavior exhibited by classifier combination (here, K-Nearest-Neighbour is used as base classifier) and weighted K-Nearest-Neighbour, Figure 5.3 illustrates these methods applied to a toy example of a two-class classification problem in two dimensions. The generic training data was generated by sampling $N_g = 2000$ samples from Gaussian distributions with parameters (5.7) and (5.8). The user-specific training data was generated by sampling $N_s = 20$ from Gaussians with parameters (5.9) and (5.10). The test data contains N_s datapoints sampled from Gaussians with parameters (5.11) and (5.12).

The top row in Figure 5.3 shows the distribution of generic training data (left) and user-specific training data (right) with the decision boundary acquired when training a K-Nearest-Neighbour classifier with $K = 10$ exclusively on the generic or the user-specific data, respectively. It can be seen that the problem of distinguishing the two classes on the user-specific data is substantially easier as the sample distributions do not overlap.

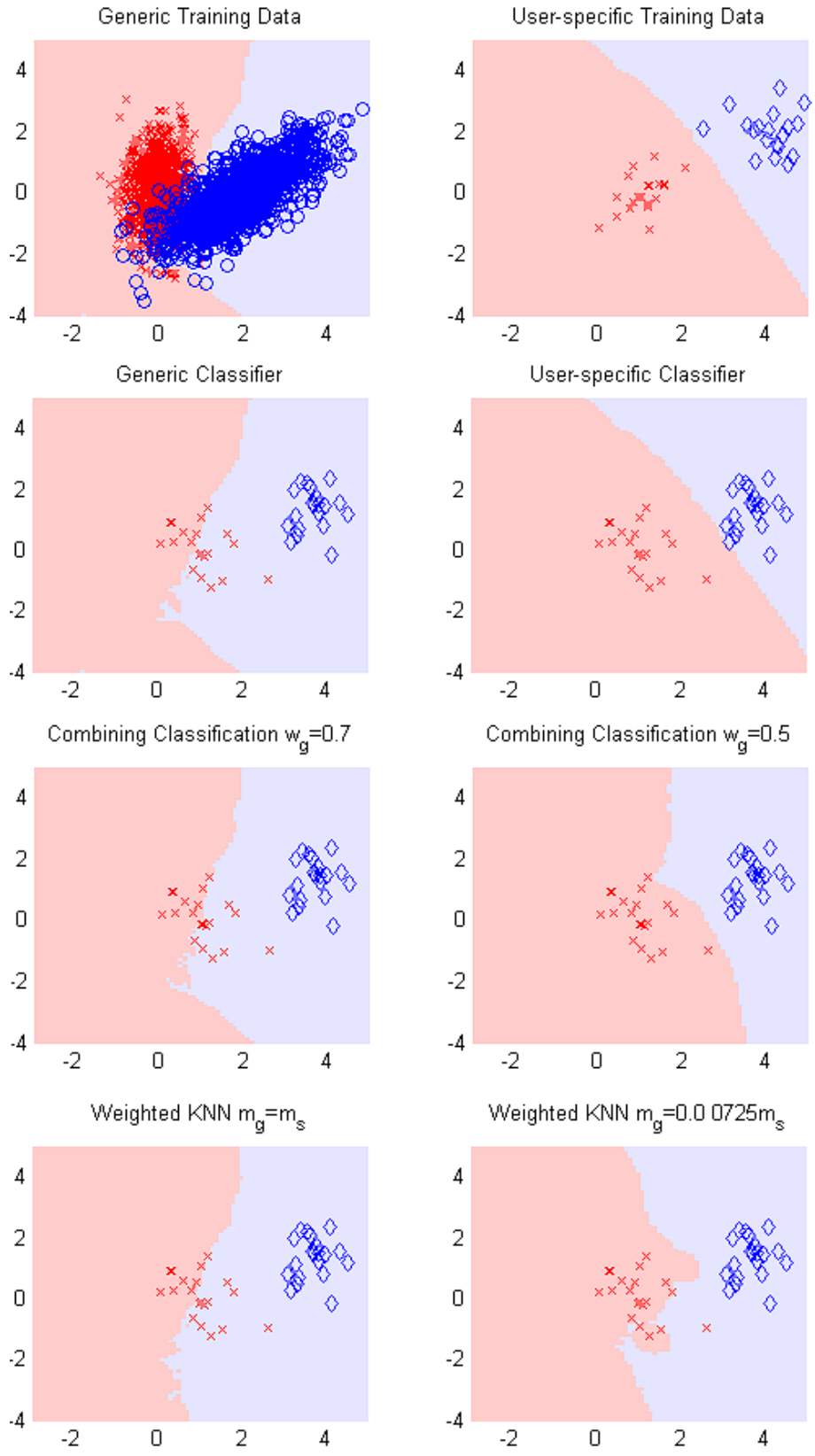


Figure 5.3: Comparison of classifier combination using K-Nearest-Neighbour as base classifier and Weighted K-Nearest-Neighbour on a 2D toy example. See text in Section 5.4 for discussion.

$$\mu_g^{red} = (0,0)^T \qquad \Sigma_g^{red} = \begin{pmatrix} 0.15 & 0.00 \\ 0.00 & 1.00 \end{pmatrix} \tag{5.7}$$

$$\mu_g^{blue} = (2,0)^T \qquad \Sigma_g^{blue} = \begin{pmatrix} 1.00 & 0.75 \\ 0.75 & 1.00 \end{pmatrix} \tag{5.8}$$

$$\mu_s^{red} = (1,0)^T \qquad \Sigma_s^{red} = \begin{pmatrix} 0.30 & 0.00 \\ 0.00 & 0.60 \end{pmatrix} \tag{5.9}$$

$$\mu_s^{blue} = (4,2)^T \qquad \Sigma_s^{blue} = \begin{pmatrix} 0.55 & 0.00 \\ 0.00 & 0.55 \end{pmatrix} \tag{5.10}$$

$$\mu_s^{red} = \mu_s^{red} \qquad \Sigma_{test}^{red} = \Sigma_s^{red} \tag{5.11}$$

$$\mu_s^{blue} = (3.5,1.5)^T \qquad \Sigma_{test}^{blue} = \Sigma_s^{blue} \tag{5.12}$$

The second row of Figure 5.3 illustrates the generic and the user-specific classifier applied to some new user-specific test data. As can be seen, the generic classifier falsely classifies many samples from the red class, whereas the user-specific classifier makes only few errors. Comparing the decision boundary of the user-specific classifier to the generic training data, one can note that the bottom right quadrant is being falsely associated with the red class, which could become problematic if the user changes his execution strategy for the activity corresponding to the blue class. If only data from, e.g., a single activity execution by the target user is available for training, it is unlikely that data points from more than one execution strategy are present in the training data. Given the small number of available user-specific labeled training data, it is therefore not advisable to rely on the user-specific classifier exclusively. It is, however, possible to adapt over time as more user-specific data becomes available.

The third row of Figure 5.3 shows two combined classifiers with different weights w_g. While a high weight of $w_g = 0.7$ on the generic classifier (left) changes the decision

boundary only minimally, a lower weight of $w_g = 0.5$ seems to strike a good compromise between generalization across individuals (the bottom left quadrant is now correctly associated with the blue class) and modelling user-idiosyncratic behavior (most samples from the red class are classified correctly). The bottom row shows the decision boundaries of weighted K-Nearest-Neighbour classifiers with different mixing weights. Similar to classifier combination, a high relative probability mass assigned to generic training data changes the decision boundary of the original generic classifier minimally, but with an appropriate choice of relative mass the classifier can adapt to the idiosyncrasies of the target user.

5.5 Summary and Outlook

As manipulation activities allow for a potentially wide range of execution strategies and recognition systems may be used by only a single individual, methods for adapting to the target user's idiosyncrasies seem promising. This chapter proposed three simple methods for adapting a recognition system to a target user: classifier combination, joint SVM training, and weighted K-Nearest-Neighbour. The relative advantages of these methods with respect to the amount of training data, and time required for user-adaptation and inference were discussed, and the behavior of user-adaptive classification methods has been illustrated on a small toy example. This chapter concludes the discussion of the methodology for multi-modal activity recognition proposed in this thesis. The next chapter introduces datasets of manipulation activities that will be used for evaluation.

Chapter 6

Datasets and Annotations

The methods for multi-modal activity recognition proposed in this thesis assume that accelerometers are attached to objects that participate in manipulation actions, and that those objects are in the field of view of a camera. The proposition of attaching accelerometers to objects as opposed to a person's body only started to gain interest in the wider research community in recent years. As a result, none of the datasets that were publicly available at the start of this project were suitable for evaluating the proposed methods.

As part of this project, two annotated datasets of food preparation activities were created and were made publicly available for evaluation purposes[1]. More than $4h$ of data were acquired and annotated, consisting of RGB-D video with a top-down view onto a work surface and readings from tri-axial accelerometers attached to kitchen objects. The main dataset which is called **50 Salads** includes 50 sequences of people preparing a mixed salad with two sequences per subject. Activities were split into a preparation, a core and a post-phase, and these phases were annotated as temporal intervals. A snapshot from this dataset is depicted in Fig. 6.1. The second dataset was annotated with ground-truth accelerometer locations of all accelerometers in every frame, which enables quantitative evaluation of accelerometer localization and tracking performance. This chapter reviews related public datasets and discusses design decisions, the experimental setup, and the

[1] Datasets *50 Salads* and *Accelerometer Localization* available at `http://cvip.computing.dundee.ac.uk/datasets/`

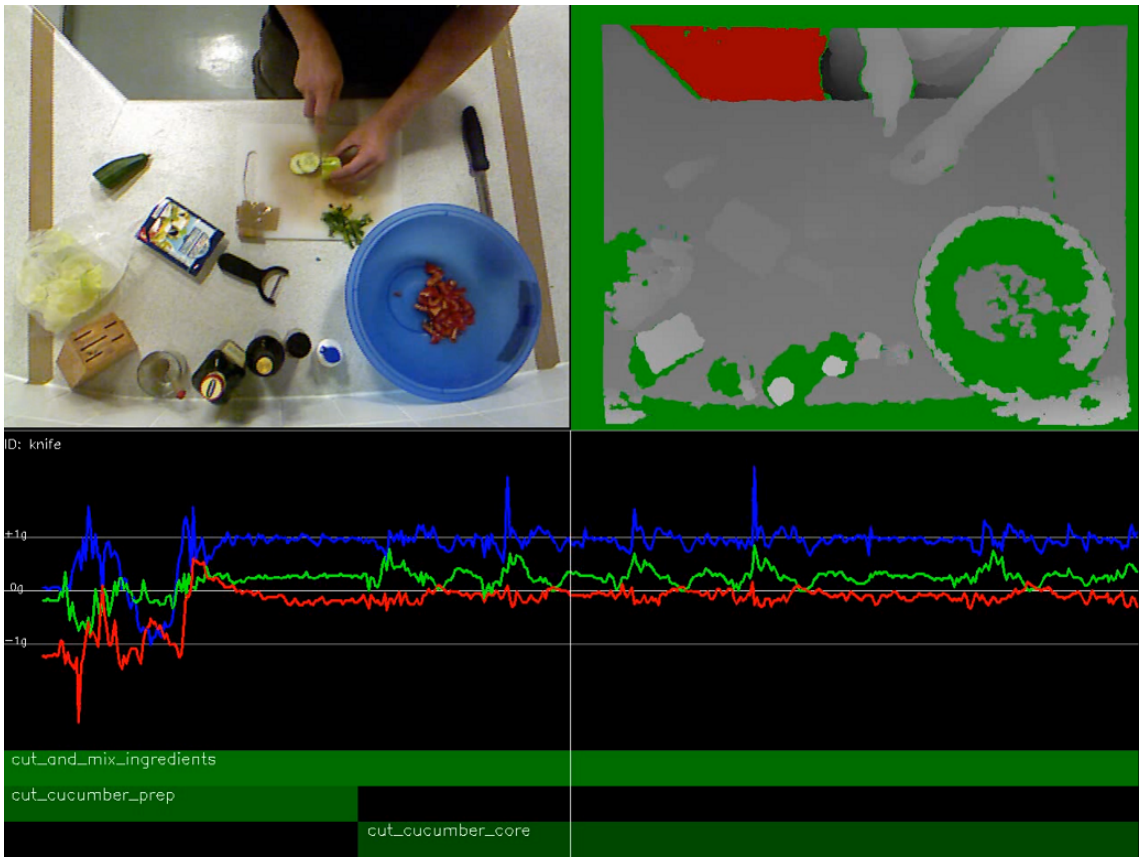


Figure 6.1: Snapshot from the 50 Salads dataset. Data from an RGB-D camera (top) and from accelerometers attached to kitchen objects (middle) were recorded while 25 people prepared two mixed salads each. Activities were split into preparation, core and post-phase, and these phases were annotated as temporal intervals (bottom).

annotation protocol.

6.1 Existing Public Datasets

Several public datasets for benchmarking activity recognition algorithms exist in the fields of wearable computing [43, 75, 82, 123] and computer vision [18, 59, 65, 67, 83, 88, 103]. One reason for the multiplicity of datasets is that the terms *activity* and *recognition* are used for varied concepts. In many cases *recognition* means offline classification, where data from an entire video clip is used to determine its activity class (e.g., KTH [88], YouTube [59], Hollywood2 [65] and URADL [67]). In others, however, *recognition* additionally includes identifying the temporal (and spatial) extent of an action, also referred to as

activity detection or spotting (e.g., Darmstadt Daily Routines [43], AmbientKitchen [75], TUM Kitchen [103], CMU-MACC [18], Opportunity [82], ICPR-KSCGR[2] and MPII-Cooking [83]). Datasets supporting activity spotting have the benefit that they can also be used for pure classification experiments. The main dataset presented in this chapter comprises sequences of activities that can be used for classification, activity spotting and progress tracking experiments.

The term *activity* is used even more broadly and may refer to atomic gestures (e.g., grasp) [82], simple, repetitive, articulated full body motions [88], fine-grained hand gestures [75], or complex interactions of multiple objects [43]. Activities are also described with varying level of detail and may contain only a verb (e.g., boxing, waving, cutting, adding) or may additionally include the objects interacted with (e.g, picking up cafeteria food) [43]. Furthermore, the total set of activities considered in a dataset may be very broad (e.g., actions in movies [65] or web video clips [59]) or scenario-specific (e.g., car assembly-line checkpoint [123] or food preparation [75]). This choice affects intra- and inter-class variability. A classification problem with high inter-class variability and low intra-class variability is easier than one with low inter-class variability and high intra-class variability. The *50 Salads* dataset contains activities with low inter-class variability and high intra-class variability as well as detailed activity annotations including the identities of objects involved (e.g., *place tomato into bowl*).

Recently, several datasets of kitchen activities have been released that combine visual and non-visual sensor types. The CMU-MMAC dataset [18] contains data from multiple cameras and body worn IMUs, BodyMedia and an eWatch. The cameras were placed to overlook large parts of the kitchen as in a surveillance scenario. Participants wore a suit with IMUs placed at joint locations and a helmet with an attached camera. Such a sensor placement is arguably infeasible for practical assisted living solutions as the camera positioning is obtrusive and wearing the suit is strongly disruptive. The TUM Kitchen dataset [103] contains video and RFID data of people dressing a table. Cameras were

[2] `http://www.murase.m.is.nagoya-u.ac.jp/KSCGR/index.html`

placed similarly to the CMU-MMAC dataset and RFID sensors were embedded at three locations in the kitchen. Although participants did not have to wear a sensor-suit in this scenario, the camera positioning was intrusive. The activities captured in this dataset did not involve objects that cannot be equipped with RFID tags. One major challenge for recognizing food preparation tasks is, however, that sensors cannot be attached to food. The use of RFID readers in the kitchen is also very limited due to the fact that there is no small number of distinct strategic locations that would suffice to be equipped with RFID readers.

Pham et al. [75] replaced the handles of kitchen utensils with Wii-controllers capturing tri-axial accelerometer data. The sensor setup for the datasets discussed in this chapter is an extension of their approach with accelerometers attached to various kitchen objects and an RGB-D camera facing the work-surface. Such a sensor setup is affordable and feasible to integrate into a home kitchen.

6.2 50 Salads Dataset

6.2.1 Sensor Setup

Accelerometers were embedded in the handles of a knife, a mixing spoon and a peeler (see Fig. 6.2). Further accelerometers were attached to a small spoon, a glass, an oil bottle, and a pepper dispenser. Axivity WAX3 wireless accelerometers were used, which are equipped with a rechargeable battery, microprocessor, 3-axis accelerometer, IEEE 802.15.4-2006 radio and a micro-USB port for recharging and configuration. These devices transmit acceleration data at 50Hz with 16-bits per axis resolution. All samples were timestamped upon arrival at the server. The recorded times were therefore subject to jitter and did not correspond exactly to the time at which the samples were captured. Gyroscopes were also considered, but as they significantly reduce battery life and their data exhibit strong artifacts resulting from magnetic interference with kitchen equipment, they were not included in this experimental setup.

Figure 6.2: Wireless accelerometers (top left) were embedded into the modified handles of kitchen utensils.

Figure 6.3: Camera placement in the laboratory kitchenette (left) and demonstration of inconspicuous camera placement inside a wall cabinet (right).

An RGB-D camera (Microsoft Kinect version 1) was mounted on the wall to have a top-down view onto the work surface. The camera was attached to an adjustable arm on a vertical bar, providing a high degree of flexibility in positioning (see Fig. 6.6). In a domestic kitchen the camera could be placed inconspicuously inside (or with a wide-angle lens potentially underneath) a wall cabinet (see Fig. 6.3). The camera was adjusted manually to roughly align the work surface with the image plane. At an operating height (distance between sensor and work surface) of 100cm and with 22cm distance to the wall, the camera captured a 108cm wide area of the work surface. Visually aligned colour and depth images were recorded at a resolution of 640×480 pixels and at about 30 frames per second. While the device captured colour and depth images usually a few milliseconds apart, a given depth map was associated with the colour image whose timestamp was closest in time in a post-processing step. Each pair of colour and depth images was subsequently treated as if both images were taken at the same time.

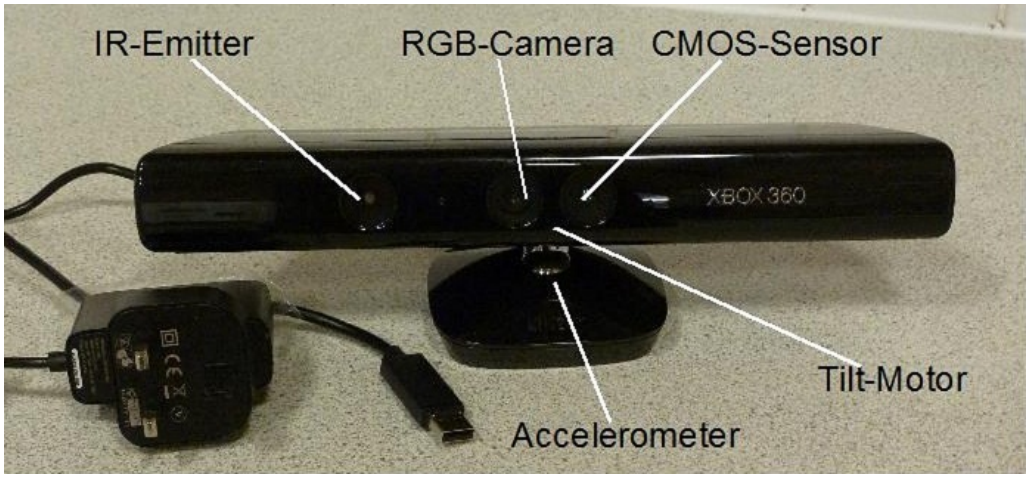


Figure 6.4: Microsoft Kinect device illustration

The Kinect was initially released as a hardware extension for the XBox360 video game console. The device has an RGB camera, a laser diode emitting near-infrared structured light, a CMOS sensor, a microphone array, a motorized (vertically adjustable) head and a 3-axis accelerometer in the foot (see Fig. 6.4). The horizontal field of view is specified as 58°. An integrated system on a chip controls the IR emitter and computes a depth image by stereo correlation of the CMOS sensor data and a prerecorded image of the

emitted light pattern projected onto a planar surface at a known distance. The resulting depth map has a per pixel resolution of 11 bits. A non-linear function maps these depth values to metric units. After this non-linear mapping, the experimental operating range of $[0.50m, 1.00m]$ is represented by only about 250 distinct values. A USB 2.0 port serves as data interface and power supply. Probably the main motivational factor for using the Kinect is its compelling price. At the time of writing, the device is sold by major internet vendors for U.S. $120, which is one order of magnitude lower than the price for a stereo camera such as the Point Grey Bumblebee 2. This development suggests that 3D cameras will soon find their way into a large number of homes and that the presence of these devices at such private locations as the living room may find a growing toleration and acceptance.

6.2.2 Experimental Protocol

A pool of 27 subjects of varied age, ethnic background and cooking experience was recruited from staff of the School of Computing at the University of Dundee. All subjects prepared a mixed salad twice totalling 54 sequences. The data of two subjects had to be excluded from the final dataset due to partial data loss. Preparing the mixed salad involved preparing a dressing with salt, pepper, olive oil and balsamic vinegar, cutting ingredients (cucumber, tomato, feta cheese and lettuce) into pieces, mixing ingredients, adding the dressing to the salad and serving the salad onto a plate. Participants were given a specific but varied task order to follow in each run. They were also told to perform all activities within a fixed area on the work surface delimited with tape, that marked the border of the camera's field of view. While no specific quantities for ingredients were given, participants were asked to prepare a single portion of salad for one person.

Repeated Task Execution

The participants were asked to prepare a salad twice for two reasons. Firstly, people exhibit varying degrees of disorientation while preparing food if they are not within their usual kitchen setting and without access to their own utensils. A laboratory kitchen setup may

add to that further through the knowledge of being recorded. Therefore, it was expected that subjects would behave more naturally in the second session as they would have had time to get used to the laboratory kitchen in the first run. Secondly, recording subjects multiple times enables the study of idiosyncrasies and the comparison of different learning scenarios, e.g., same subject included in training data against cross-subject generalization.

Task Order Sampling

When observing a person performing multi-step activities that involve interacting with a number of different objects, different orderings in which these steps are carried out induce strong variation in the configuration and appearance of objects in the scene. In the context of preparing a mixed salad the scene looks different after preparing the dressing, depending on whether the ingredients of the salad have been cut and mixed already. In order to build robust activity models for recognition it is convenient to have a balanced dataset that contains roughly the same number of examples for all likely task-orderings. In practice it is costly to acquire annotated video data of a large number of people performing the same multi-step activity. Additionally, the task-orderings that the recorded sample population chooses naturally are potentially highly imbalanced. Therefore, this chapter proposes to sample task-orderings from a statistical activity model and ask participants to follow the steps of a recipe in orderings generated by the model.

The statistical activity model that was used for the preparation of mixed salads is illustrated in Figure 6.5. The model is based on *Activity Diagrams* used in computational process specification and analysis. Every choice-node of the diagram (represented by a horizontal bar with multiple outgoing arcs) is augmented by a probability distribution over all options representing the probability of choosing each option when that choice-node is reached. The probabilities in Figure 6.5 are set to be uniform to ensure a balanced distribution of task-orderings.

Each participant was given a different ordering of tasks to follow in each session. Surprisingly, few participants precisely followed the task ordering given to them although

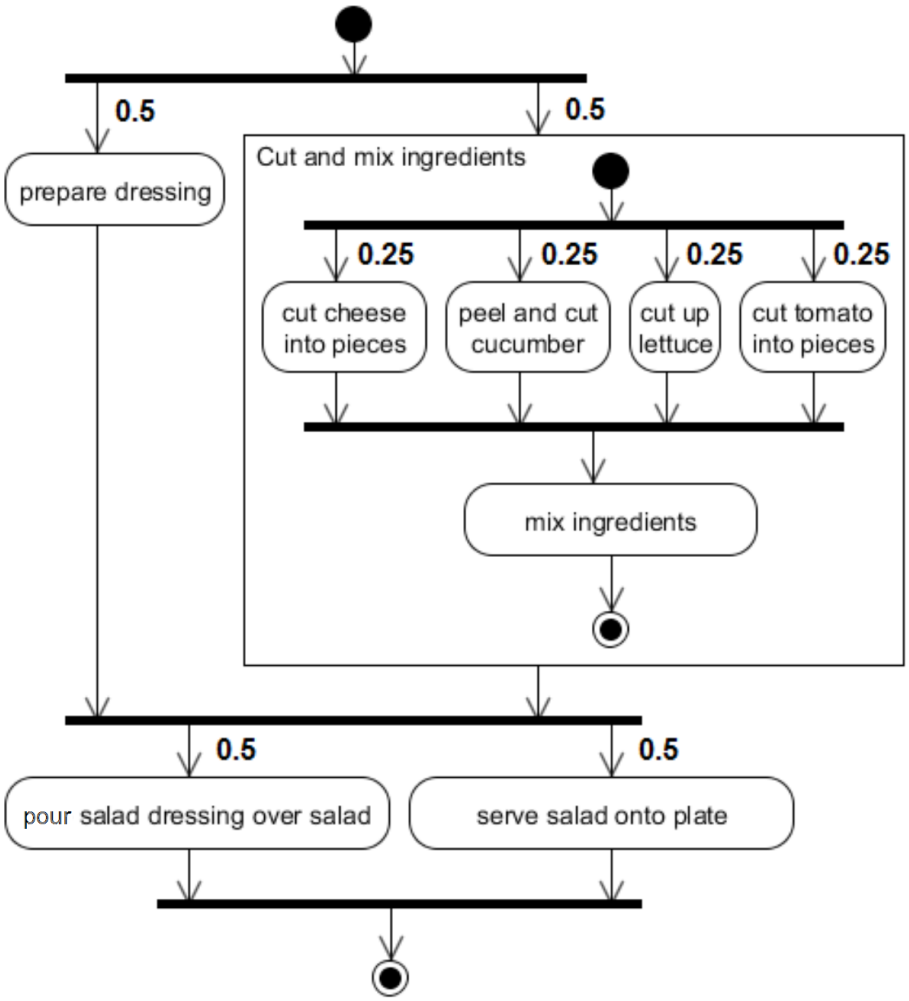


Figure 6.5: Activity Diagram: Task orderings were sampled from this model and given to participants in order to increase variability of task orderings in the dataset.

they were specifically instructed to do so. Our hypothesis is that ordering tasks within a food preparation activity is strongly governed by habit and personal reasoning. In cases where subjects failed to correctly follow the instructions in the second session, this error may also be due to the subject following the memorized task ordering of the previous session. Although this behavior was unintended, the availability of the instructed task ordering together with the annotated activities may be used to experiment with detecting deviations from the given task order.

Sensor Synchronization

As the timestamps of video frames and accelerometer samples are not synchronized, the data were synchronized in a post-processing step. To facilitate sensor synchronization an action was performed at the start and the end of all sequences that simultaneously produces strong signals in the video and the accelerometer data; the experimenter beat an accelerometer-equipped kitchen utensil repeatedly on the work surface. An annotator manually marked the correspondences at the start and the end of all sequences. Using least squares estimation, two temporal offsets per sequence were estimated based on these correspondences, one for the start and one for the end. Linear interpolation was used for temporal alignment within this interval.

6.2.3 Activity Annotation

The following 17 activities were annotated: *add oil*, *add vinegar*, *add salt*, *add pepper*, *mix dressing*, *peel cucumber*, *cut cucumber*, *place cucumber into bowl*, *cut cheese*, *place cheese into bowl*, *cut lettuce*, *place lettuce into bowl*, *cut tomato*, *place tomato into bowl*, *mix ingredients*, *serve salad onto plate* and *add dressing*. Each activity was split into three phases which were annotated individually in the form of a start time and an end time corresponding to their temporal extent: *pre-*, *core-* and *post-phase*. Each activity was associated with one of three stages in the recipe which were also annotated: *prepare dressing*, *cut and mix ingredients* and *serve salad*. In total 966 activity instances were

Activity	#Inst.	#Frames	Core	Label in Experiments
add oil	55	27813	8161	add oil
add vinegar	54	23657	6572	-
add salt	53	11369	4995	-
add pepper	55	12912	6123	give pepper
mix dressing	61	19492	14295	mix dressing
peel cucumber	53	62141	38613	peel cucumber
cut cucumber	59	49853	38787	cut into pieces
cut cheese	56	50680	26001	cut into pieces
cut lettuce	61	53313	28847	cut into pieces
cut tomato	63	68347	50768	cut into pieces
place cucumber into bowl	59	16800	9071	place into bowl
place cheese into bowl	53	12753	6305	place into bowl
place lettuce into bowl	61	15159	7856	place into bowl
place tomato into bowl	62	13547	6418	place into bowl
mix ingredients	64	22917	16050	mix incredients
serve salad onto plate	53	35230	19110	serve salad
add dressing	44	22428	12092	add dressing
Total	**966**	**518411**	**300064**	

Table 6.1: Dataset size in terms of activity instances and video frame counts.

annotated. Annotations spanned more than 500k video frames of which more than 300k frames represented the core-phase of an activity. Table 6.1 lists the numbers of instances and frames for all activities.

The pre- and post-phases of an activity include grabbing, moving and placing utensils and ingredients. The core phase captures actions that are essential. Taking *add oil* as an example, the pre-phase might consist of grabbing the oil bottle, moving it over the dressing glass and screwing it's lid off. The core phase represents tilting the bottle and pouring oil into the glass. Screwing the lid back on, moving the bottle and placing it on the work surface would be annotated as the post-phase of the *add oil* activity. The annotated temporal extent of a phase of an activity is delimited by distinct events at the start and end of these phases. The start of the pre-phase of the *add oil* activity, for example, would be marked as the frame in which the oil bottle was first touched as opposed to the frame in which the hand reaches out in order to grab the bottle. Annotations are therefore unambiguous and repeatable.

6.2.4 Use Cases

This dataset can be split in various ways into training and test data in order to investigate different generalization problems. Cross-subject generalization is a common problem investigated by the activity recognition community, where all data of any subject is used either for training or for testing but not both. This is a hard problem in the context of food preparation due to strong personal preferences regarding how activities are executed (idiosyncrasies). Intra-subject generalization is a comparably easier problem given the same amount of training data. Here, only very limited data of a single subject is available. Nevertheless it would be interesting to evaluate intra-subject generalization on this dataset as having very limited training data of the target subject might be a good representation of real-world conditions for, e.g., a situated prompting system.

As the dataset consists of sequences of steps involved in the recipe, it can be useful for evaluating activity spotting, activity recognition, sequence analysis and progress tracking. As it involves data from accelerometers and RGB-D video data, the dataset can be used for evaluating sensor fusion methods and methods for transfer learning. Further potential use cases include user-adaptation and skill assessment.

While inferring the full specification of an activity (verb, phase and objects involved) is a long-term objective, it is questionable whether current activity recognition methods are powerful enough to do this. In order to gradually approach this goal, however, the available detailed annotations may be used to automatically formulate easier classification problems. For example, parts of the description such as the ingredients or the activity-phase may be ignored, mapping distinct annotations onto the same label. One such simplified recognition problem that will be used in the evaluation chapter of this thesis is discussed below.

The choice of ontology by which activities are categorized can substantially influence recognition performance. Consider, for example, the two activities mixing the salad dressing and mixing the completed salad. They may be regarded as belonging to the same general activity of mixing ingredients. The exhibited motion pattern when performing these activities is similar. Based on low-level motion features alone it would therefore

be difficult to differentiate between mixing the dressing and mixing the completed salad, and combining these activities in the general activity *mixing ingredients* simplifies the recognition task. However, the kitchen utensils used while performing these two activities are different. While participants tend to move the dressing glass and the small spoon when mixing the dressing, the large spoon and the salad bowl are moved when the final salad is being mixed. As the sets of utensils moved in these two activities are mutually exclusive, the task of differentiating these activities based on object use is comparably easy. Therefore, assigning a common class label to these activities renders the recognition task more difficult.

For comparison of features from accelerometers and vision, and an investigation of various combinations of feature types, a recognition problem with $C = 10$ activity classes was used in this thesis: *add_oil, give_pepper, mix_dressing, mix_ingredients, cut_into_pieces, place_into_bowl, peel_cucumber, serve_salad, dress_salad* and *NULL*, where *NULL* indicates that none of the activities of interest is happening. The mapping from annotated labels to activity classes used for evaluation is shown in Table 6.1. Each activity class includes the pre-, core- and post-phases of each annotated label. It was decided to exclude *add pepper*, as there was no accelerometer attached to the pepper dispenser and it would therefore be impossible to recognize this activity from accelerometer data. Where annotated labels only differed in the maninpulated ingredient, these labels were mapped to the same activity class (i.e., *cut_into_pieces* and *place_into_bowl*). While recognizing the manipulated ingredient is ultimately necessary to track through a recipe, this thesis focusses on ingredient-agnostic activity models and leaves this aspect to future work. This mapping is of course only one of many recognition problems that could be investigated using this dataset.

6.3 A Dataset for Accelerometer Localization

Prior work by other authors on accelerometer localization [64, 91] evaluated accelerometer localization only qualitatively. In order to compare different localization algorithms,

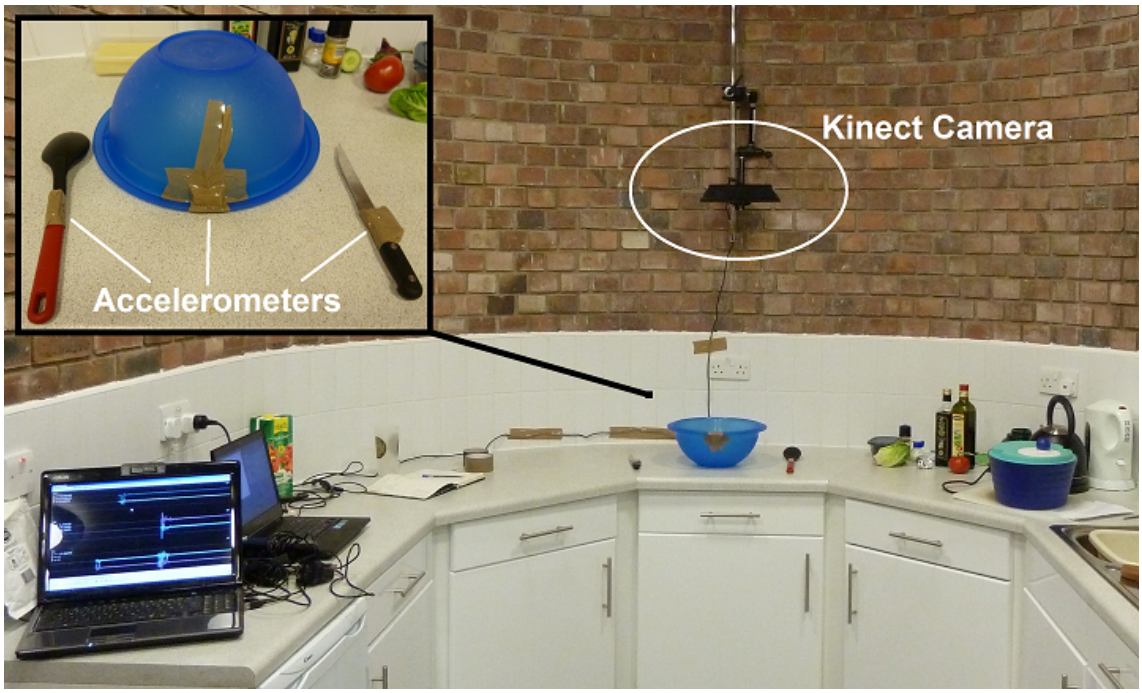


Figure 6.6: Accelerometer Localization: slightly modified experimental setup. Laboratory kitchenette with Kinect facing the work surface and accelerometers attached to three kitchen utensils with tape (top left).

there is a strong need for datasets that provide the necessary annotations for quantitative evaluation. This thesis makes one contribution in this direction by offering a challenging dataset with ground-truth accelerometer location annotations.

The experimental setup was similar to the one discussed in the previous section (see Fig. 6.6). RGB-D video data and data from three accelerometers that were attached to a knife, a spoon and the rim of a bowl, respectively, were recorded while a person prepared a mixed salad. Note that some devices were sometimes moved at the same time (e.g., the spoon and the bowl move concurrently while the salad is being mixed before serving). Ten ingredients were used in total to prepare the mixed salad: beetroot, cheese, courgette, lettuce, purple onion, and tomatoes, as well as balsamic vinegar, olive oil, salt and pepper for the dressing.

The recorded sequence contains 13,263 frames of combined colour and depth data and 31,346 samples of 3-axis acceleration data. The locations of the three accelerometers were annotated in all frames by hand. For each pair of video frame and accelerometer, a

point location in image coordinates was marked by manually clicking on the image at the estimated location of the geometric centre of the device. As accelerometers are covered by tape and frequently occluded by the participant's hand, it cannot be assumed that these labels are exact. Nevertheless, from visual inspection it can be confirmed that the marked locations are reasonably close to the true centre locations for our purposes.

6.4 Summary and Outlook

This chapter introduced two datasets with a novel combination of RGB-D video data and data from accelerometers attached to objects. *50 Salads* is a large dataset of 50 sequences of complex manipulation activities in the kitchen. It can be used to evaluate a wide variety of machine learning problems including sensor fusion, activity recognition and user-adaptation, which are of particular interest for this thesis. The second dataset includes ground-truth annotations of accelerometer locations, which are necessary for quantitative evaluation and comparison of visual accelerometer localization algorithms. The next chapter presents evaluation results of the methods presented in this thesis using the datasets presented in this chapter.

Chapter 7

Evaluation

This chapter validates and comparatively evaluates the methods proposed in this thesis on the datasets introduced in the previous chapter. Section 7.1 evaluates the accuracy of the accelerometer tracking algorithm introduced in Chapter 3 and quantitatively compares it to alternatives with different methods for hypothesis generation, similarity estimation, and similarity re-initialization. Section 7.2 formulates an activity recognition problem with 10 activity classes that were derived from the detailed annotations of the *50 Salads* dataset, and compares the recognition performance of features from different sensor types and different sensor fusion techniques. The models proposed for user-adaptive classification are evaluated in Section 7.4.

ID	Description	#Frames
1	Knife Synchronization Signal	203
2	Moving Spoon From Bowl To Worktop	25
3	Spoon Mixing Dressing	310
4	Knife Slicing Beetroot	296
5	Knife Scraping Beetroot Into Bowl	125
6	Knife Scraping Courgette Into Bowl	45
7	Knife Scraping Cheese Into Bowl	106
8	Knife Scraping Green Salad Into Bowl	85
9	Knife Slicing Green Salad	45
10	Knife Slicing Green Salad 2	170
11	Knife Scraping Green Salad Into Bowl	123
12	Knife Dicing Tomatoes And Moving Into Bowl	1133
13	Spoon Mixing Salad	375
14	Spoon Mixing Salad 2	95
15	Spoon Serving Salad Into Plates	798
16	Knife Synchronization Signal 2	233
	Total	4167

Table 7.1: Test Sequences: Subsequences with at least one accelerometer measuring substantial accelerations over an extended period of time.

7.1 Accelerometer Localization and Tracking

In contrast to published work by other authors on accelerometer localization [64, 91] which evaluated localisation performance only qualitatively, this thesis reports quantitative evaluation using the accelerometer location annotations for the dataset introduced in Section 6.3, a single 13,263 frame sequence of a person preparing a mixed salad. For quantitative evaluation of different algorithm configurations the average per frame Euclidean distance of an estimated accelerometer location to the ground truth point in terms of pixels in the image plane was used.

First, evaluation was performed on a set of *easy* subsequences from the dataset. An accelerometer is arguably easier to localize when it measures strong acceleration than when it is stationary and may not send any data. A set of 16 subsequences was identified in the video during which at least one accelerometer measured strong acceleration over an extended period of time. These subsequences are listed in Table 7.1. The localization performances reported in the following sub-sections were based on testing each of these

subsequences separately, re-initializing the algorithm at the start of each subsequence. The algorithm was tested on the task of localizing the single accelerometer that is in motion throughout the entire subsequence. Trajectories from feature point tracking were compared with trajectories from grid-sampling and dense optical flow. The distance of points from the camera along the normal of the image plane was acquired from depth maps (variable) or a constant depth for all pixels was defined manually (fixed).

In a second experiment, the accelerometer tracking algorithm that showed best performance in the previous experiment was evaluated on the entire video on the task of localizing all accelerometers in every frame. In this experiment different methods for similarity re-initialization were compared.

7.1.1 Sparse vs. Dense Optical Flow

For the first stage of the localization pipeline two different methods to extract point feature trajectories from video data were compared: sparse point feature tracking and dense optical flow (see Section 3.3). For a fair comparison, the parameters of each method were optimized empirically. Dense trajectories were initialized on a grid with $d_G = 24$ pixels, initializing 336 trajectories at equidistant locations, and were terminated based on a threshold $\tau_d = 5$ pixels. For sparse feature point tracking, the maximum number of trajectories was set to $N_T^{max} = 96$ with a minimum distance at initialization of $d_I^{min} = 14$ pixels.

The fixed depth was set to the estimated operating height $\hat{z} = 0.9m$, which corresponds roughly to the operating height of the camera. Recall that this distance is used to convert image into world (metric) coordinates in Eq. (3.4). In all experiments accelerations were estimated from Gaussian-filtered locations with $\sigma = \frac{0.3}{f_{vid}}$ and a temporal decay $\alpha = 0.9982$ was used.

Table 7.2 shows evaluation results for these configurations using the depth provided by the camera (variable) and using a fixed depth. While the region of interest as indicated in Fig. 3.1 was used to compare effects of fixed and variable depth on localization

		KLT			Dense Optical Flow		
		Depth		ROI	Depth		ROI
ID	#Frames	fixed	var.	off	fixed	var.	off
1	203	74	56	65	**32**	50	42
2	25	206	313	217	**92**	99	182
3	310	**23**	28	24	25	35	33
4	296	**19**	28	59	49	32	61
5	125	143	215	139	53	**46**	66
6	45	89	259	118	**67**	94	152
7	106	**67**	79	128	89	84	**67**
8	85	138	146	181	**48**	69	73
9	45	131	213	**71**	92	74	165
10	170	66	304	210	**36**	91	196
11	123	**65**	81	132	93	114	96
12	1133	60	184	209	23	179	**20**
13	375	**62**	63	66	97	105	173
14	95	81	87	235	**54**	76	150
15	798	135	265	141	**52**	104	108
16	233	**38**	338	57	74	85	80
Total	4167	76	167	134	**49**	106	79

Table 7.2: Sparse vs. Dense Optical Flow: Different trajectory construction methods with depth from the camera variable and fixed. Performance is reported as average Euclidean distance of estimated accelerometer location to ground truth in pixels.

performance, results based on the entire image region (ROI off) with fixed depth are also reported.

It is obvious that the effect of frames far in the past on the present similarity measure needs to be limited, but there are several different ways to model this behavior. Section 3.5.2 introduced a temporal decay through which the contribution of past frames to the similarity measure is reduced by a multiplicative factor α. Figure 7.1 shows average localization accuracy obtained for various values of the decay parameter α. The average Euclidean distance of estimated accelerometer location to the ground-truth in pixels reported is the average distance over all subsequences. The result of each test was weighted by the number of frames in the corresponding subsequence.

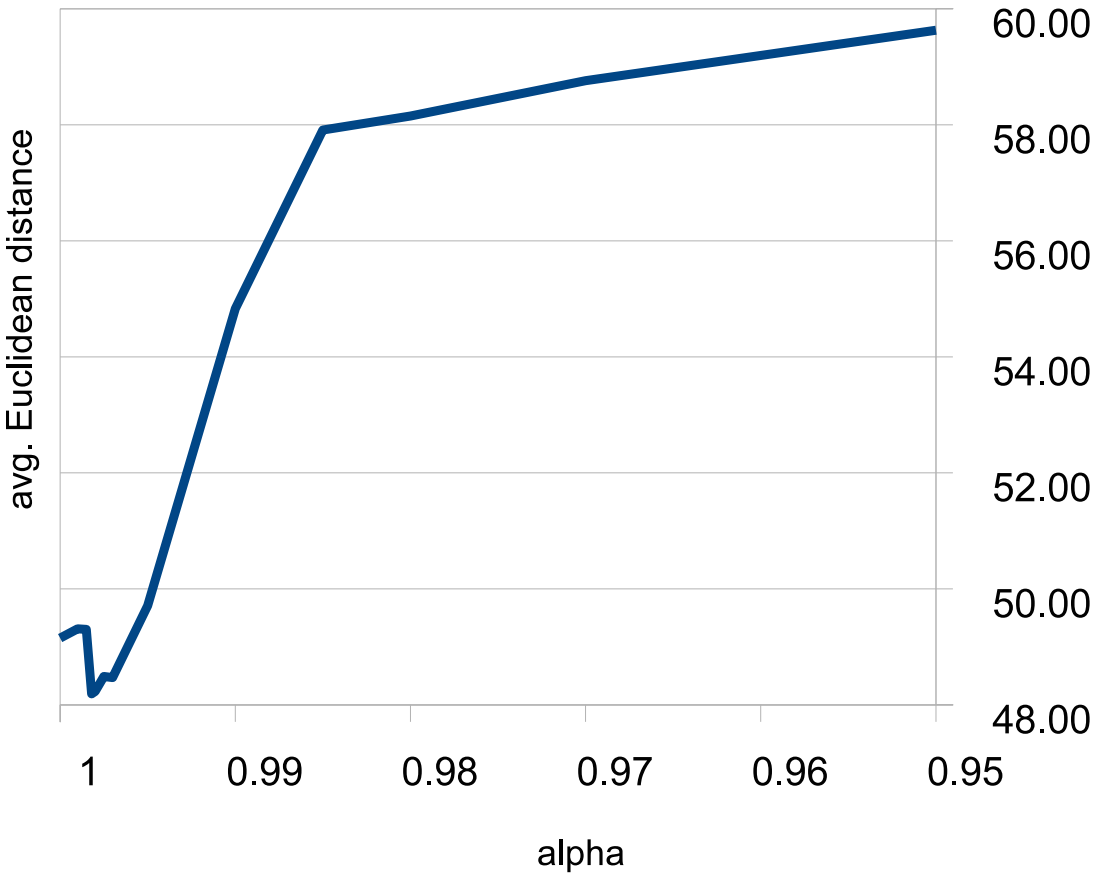


Figure 7.1: Temporal Decay: the effect of temporal decay on accelerometer localization accuracy was evaluated by varying the multiplicative factor α.

Discussion

Intuitively, a larger number of trajectories is expected to increase localization accuracy as *the true trajectory* is more likely to be found in the sample. Nevertheless, in preliminary experiments, using more than 96 active trajectories with KLT and more than 336 initial trajectories with dense optical flow has not been beneficial to localization performance. Presumably a lower number of trajectories increases robustness against artifacts in the estimated flow fields.

There are four major observations resulting from these experiments.

1. Trajectories constructed from dense optical flow estimates are shown to perform considerably better than those from sparse tracking. It is suspected that the substantial difference in performance to KLT is due to the smoothness of the dense flow field which reduces the number of false feature correspondences, and the better coverage of low-texture regions obtained with uniform sampling.

2. The 3D extension of trajectories as proposed here - annotating trajectories with the depth from the Kinect - does not outperform localization based on a user-defined constant depth. Acceleration sequences constructed with measured depth performed considerably worse than those assuming a constant depth, for trajectories from KLT tracking and dense optical flow alike. The depth maps produced by the camera are clearly not reliable enough for extending point trajectories to 3D. This might be due, in part, to noisy depth measurements, holes in the depth maps on areas that lie in the shadow of the structured light pattern, and point trajectories frequently crossing depth-discontinuities. A more sophisticated method of extending point feature trajectories to the 3D case, in which depth-discontinuities and noise in the raw depth maps are specifically taken into account, could improve these results, especially for scenarios in which the accelerometer is moved along the camera's view axis.

3. Narrowing down the location search space with a smaller region of interest in the

image increases localization accuracy as expected. While the whole image region also covers some space in front of the work surface, which is in large parts occupied by the participant's body, this region is completely cropped in the latter configuration, eliminating a large fraction of sources of confusion for the localization algorithm.

4. Localization accuracy decreased drastically with $\alpha \ll 1$. The performance gain compared to no temporal decay was rather small (1.04 pixels on average), which might be due to the limited length of the test-sequences. A more significant improvement is expected when running the accelerometer localization algorithm over an extended period of time.

As point trajectories from dense optical flow with fixed depth outperformed all other configurations on average this configuration was used in all subsequent experiments.

7.1.2 Comparison to Normalized Cross-Correlation

The similarity measure proposed in this thesis, subsequently called *Temporal Decay Thresholding* (TDT), was also compared to normalized cross-correlation (NCC) as used by Maki et al. [64]. For a fair comparison between NCC and the proposed similarity measure, NCC was applied to trajectories constructed from dense optical flow. Estimated accelerations along point trajectories were correlated with the sub-sampled acceleration data captured by an accelerometer. Results of accelerometer localization using NCC with varying length of temporal sliding window are presented in Figure 7.2. The average Euclidean distance of estimated accelerometer location to the ground-truth in pixels here is the average over all frames in all subsequences. Localization accuracy reached its maximum at a window size of about 150 frames with an average Euclidean distance to the ground-truth of 114 pixels. This time window corresponds to an interval of five seconds. TDT substantially outperformed NCC with an average distance to the ground truth of 49 pixels.

Cumulative distributions over the distances of estimated locations to ground-truth are

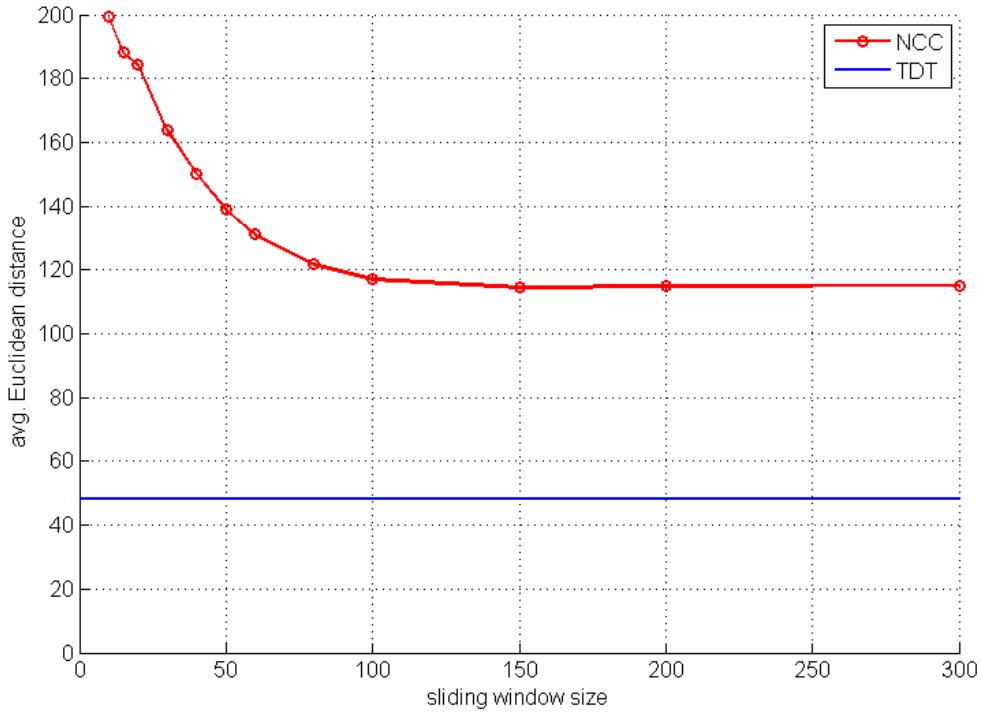
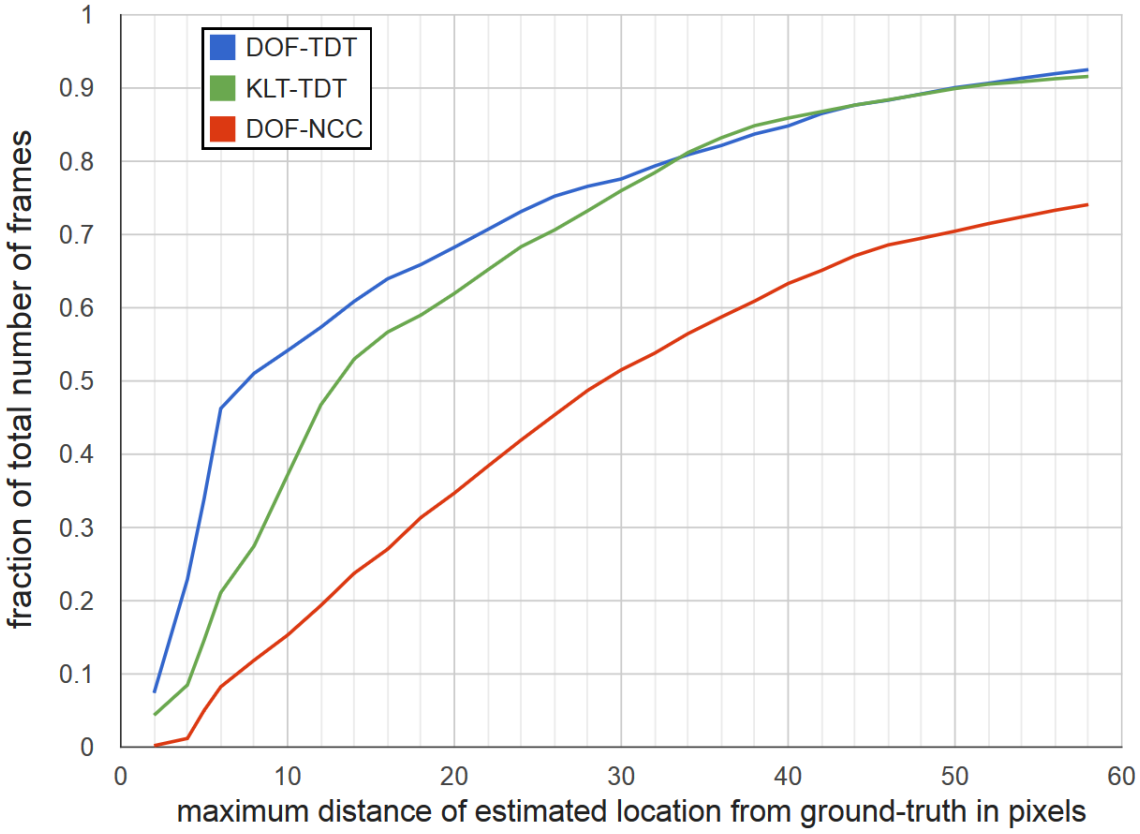


Figure 7.2: Localization accuracy of NCC with varying length of temporal sliding window. The best performance obtained with TDT is plotted for comparison.

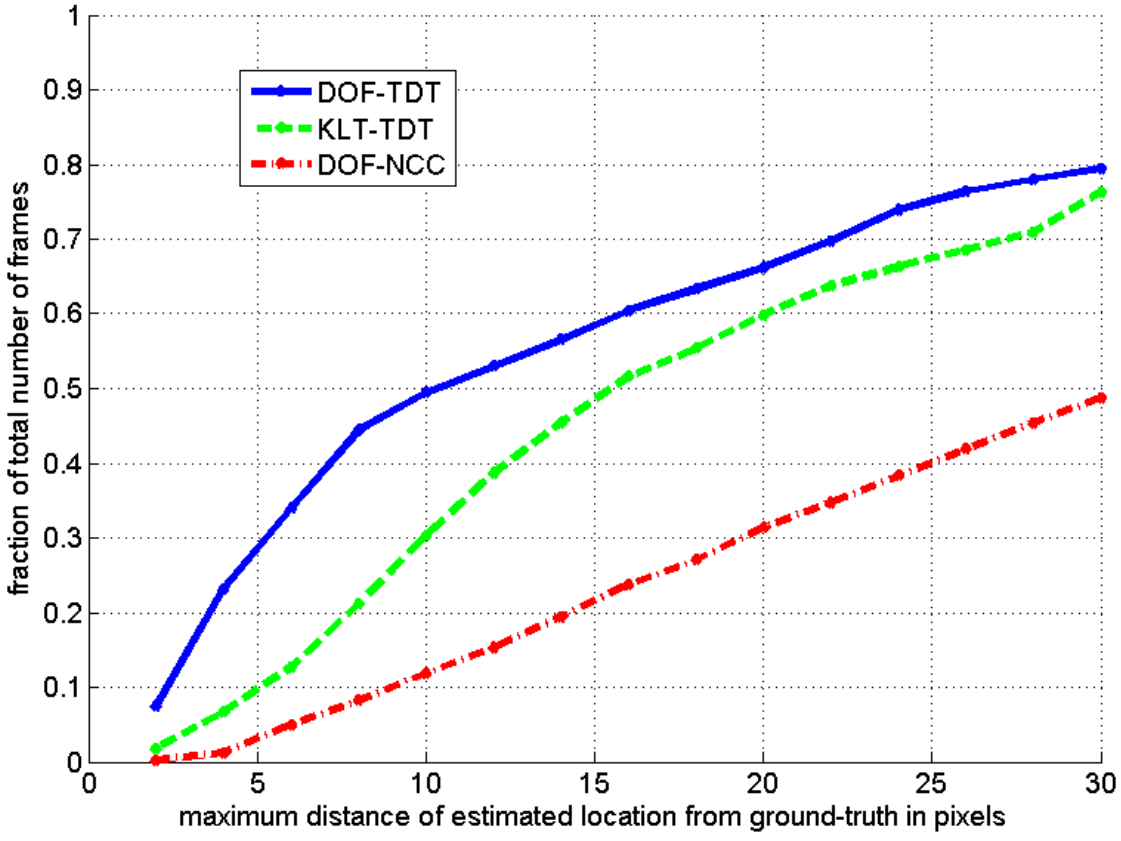
given in Figure 7.3 for the proposed method (DOF and KLT with fixed depth) and NCC using DOF trajectories. The plotted NCC result was obtained using a temporal window size of 150 frames; this gave the lowest average distance from ground-truth (114 pixels) of all the window sizes tried in the previous experiment. Although the proposed method leaves room for improvement, it clearly outperforms NCC, increasing the fraction of predictions within a 10 pixel radius from 12% to 50% and with a 30 pixel radius from 50% to 80%.

7.1.3 Long-Term Accelerometer Tracking

In order to compare the accuracy of long-term accelerometer tracking with different methods for re-initializing hypothesis scores after an accelerometer has been stationary, the estimated accelerometer location in every frame of the entire video was evaluated (Figure 7.4). *No Re-Initialization* and *Cold Start* (initialization to $S_t = 0$) were compared to two methods for re-initialization based on the similarity map $\mathcal{M}$: (i) assigning the



(a) the big picture



(b) enlarged view on interval [0,30]

Figure 7.3: Comparison of accelerometer localization precision evaluated as the cumulative distribution of distances of estimated locations to the ground-truth with the proposed method (DOF-TDT and KLT-TDT) and normalized cross-correlation (DOF-NCC).

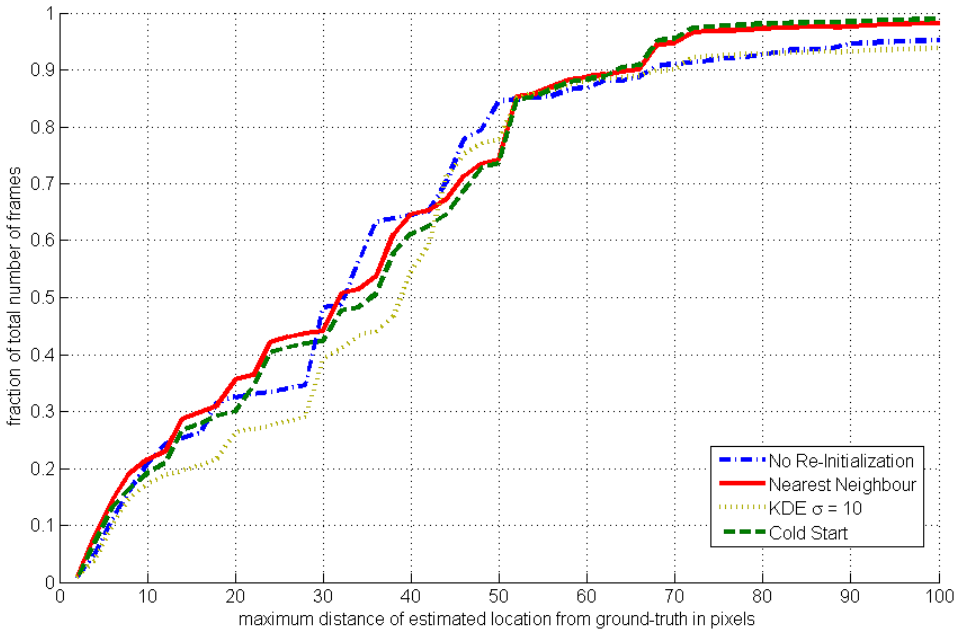


Figure 7.4: Long-term accelerometer tracking accuracy obtained with multiple methods for similarity re-initialization.

score of the *Nearest Neighbour* and (ii) kernel density estimation (*KDE*) (see Section 3.6, p. 43). KDE did not perform better than other methods with $\sigma = 1, \ldots, 10$. (Only KDE results for $\sigma = 10$ were plotted for clarity). While none of the explored strategies clearly outperformed the others, re-initialization from the *Nearest Neighbour* in $\mathcal{M}$ and *Cold Start* showed a considerably higher fraction of estimates in the range up to 25 pixels and approached saturation closer to the ground-truth location. As the *Nearest Neighbour* strategy slightly outperformed cold start, this method was employed in all subsequent experiments.

7.2 Single Modality Activity Recognition

7.2.1 Evaluation Protocol

For activity recognition it is assumed that no two activities occur simultaneously. In rare cases the annotated temporal intervals of subsequent activities overlap. As the frames in

which this situation occurs account for only 0.09% of the data, these samples were skipped for both training and testing. All remaining frames are unambiguously associated with a single class label.

In order to test cross-subject generalization, algorithms were evaluated by 5-fold cross-validation. Although 10-folds are often used (as argued in [49]), 5-fold cross-validation was chosen to keep the computational cost manageable. Each test set consisted of two sequences of each of five participants; the corresponding training set consisted of two sequences of each of the remaining 20 participants.

Performance Measures Performance was measured as mean precision, mean recall and their harmonic mean (f-measure). For an unbiased estimate of recognition performance based on unbalanced test data, class precision and recall were weighted inversely proportional to their occurrence in the test set when aggregated. Given the number of true positive (TP), false positive (FP) and false negative (FN) classification results, precision is defined as $\frac{TP}{TP+FP}$ and recall is defined as $\frac{TP}{TP+FN}$. Mean precision and recall over all classes and cross-validation partitions were computed by first summing TP, FP and FN over all partitions for each class separately, then applying the formulas for precision and recall on the sums and finally estimating the arithmetic mean over all classes, as argued in [25]. The arithmetic mean over all classes assigns equal importance to all classes regardless of their prevalence in the test data.

Codebook Learning Codebooks for *Absolute Tracklets* and *RETLETS* were learned from a subsample of 100k tracklets extracted from training data. K-means clustering was applied to the sampled data using a GPU implementation of exact nearest neighbor search. K-means was initialized 8 times and the dictionary with minimal reconstruction error kept.

Feature Extraction Features were extracted from temporal windows of 154 video frames or 256 accelerometer samples ($\sim 5s$) centred on each video frame. Temporal windows were shifted by one frame at a time, and features were extracted from each

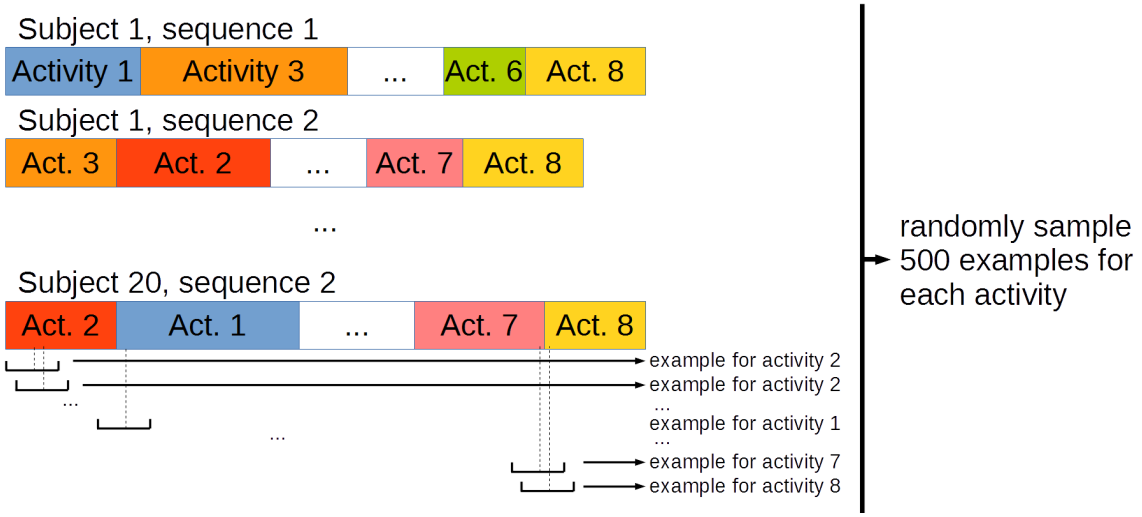


Figure 7.5: Training sets were constructed by extracting features from temporal sliding windows shifted by one frame at a time, treating each window independently. The feature vector extracted from a temporal window was assigned to the ground-truth label at the window's center. From each activity class 500 examples were sampled at random.

window independently. Each temporal window was assigned to the ground-truth label at the window's center. This process is illustrated for the training data of one test partition in Figure 7.5. For *Reference Tracklet Statistics* a temporal window of 16 video frames was used, a trajectory length commonly used for visual action recognition [110]. In order to create a balanced training set for SVM training, a stratified random sub-sample of all training data points was used which contained 500 samples from each class and 5000 samples in total.

Parameter Search The free parameters of SVM and random decision forest classifiers were determined via 5-fold cross-validation on each training set. Each training set was split into 5 partitions containing both sessions of 4 subjects. Each of these partitions was used for validating a model that has been trained on the remaining 16 subjects. Random Decision Forest model parameters were selected out of a set of models with 20 trees, the values $\{8, 10, 12\}$ for the maximum number of tree levels, $\{\frac{1}{3}d, \frac{2}{3}d, d\}$ for the number of weak classifiers tested per node (where d is the number of feature dimensions) and 10 thresholds per weak classifier. For *Reference Tracklet Statistics* and *Accelerometer Statistics* parameter search for the RBF kernel parameter γ_{RBF} was performed through

Feature Type	Precision	Recall	f-measure
Absolute Tracklets	0.37 ± 0.03	0.40 ± 0.02	0.38
Object Use	0.43 ± 0.03	0.50 ± 0.02	0.46
Accelerometer Statistics	$\mathbf{0.63 \pm 0.07}$	$\mathbf{0.65 \pm 0.03}$	**0.64**

Table 7.3: Comparison of recognition performance using visual features (*Absolute Tracklets*, $D = 500$ visual words) and accelerometer features (*Object Use* and *Accelerometer Statistics*).

5-fold cross-validation on the training set of the first test partition prior to evaluation. It was set to $\gamma_{AS} = 0.03125$ for *Accelerometer Statistics* and $\gamma_{RTS} = 2$ for *Reference Tracklet Statistics* in all experiments.

7.2.2 Accelerometer vs. Visual Features

Recognition performance obtained with *Accelerometer Statistics*, *Object Use* and *Absolute Tracklets* was compared using SVM classifiers. For *Absolute Tracklets* a grid distance of $d_G = 5$ pixels was used. The results are presented in Table 7.3. The results confirm that the problem under investigation is sufficiently challenging. The best performance of 63% precision at 65% recall was obtained with *Accelerometer Statistics*, whereas chance is 10%. This result confirms that acceleromter-based activity recognition can yield good performance using features that are fast to extract from the temporal domain and require no learning as proposed in [77]. Visual motion features showed surprisingly low recognition performance of 37% precision at 40% recall.

7.3 Multi-Modal Activity Recognition

This section investigates the combination of data from accelerometers and video through features from accelerometer localization and by combining multiple feature types at different stages in the recognition pipeline.

Features from Individual Sensor Types	**Precision**	**Recall**
Object Use	0.41 ± 0.03	0.48 ± 0.02
Acceleration Statistics	$\mathbf{0.62 \pm 0.05}$	$\mathbf{0.64 \pm 0.04}$
Features from Accelerometer Localization		
*Device Locations**	0.26 ± 0.02	0.22 ± 0.03
*Reference Tracklet Statistics**	$\mathbf{0.52 \pm 0.05}$	$\mathbf{0.49 \pm 0.04}$
Feature Vector Concatenation		
Object Use + *Accelerometer Statistics*	0.63 ± 0.05	0.66 ± 0.03
*Reference Tracklet Statistics** + *Accelerometer Statistics*	0.67 ± 0.05	0.67 ± 0.03
Object Use + *Reference Tracklet Statistics** + *Accelerometer Statistics*	$\mathbf{0.67 \pm 0.05}$	$\mathbf{0.68 \pm 0.03}$

Classifier Combination	**Comb. Method**		
Object Use + *Acceleration Statistics*	Sum	0.43 ± 0.07	0.49 ± 0.03
	Product	0.44 ± 0.02	0.50 ± 0.03
*Reference Tracklet Statistics** + *Acceleration Statistics*	Sum	0.63 ± 0.04	0.67 ± 0.03
	Product	$\mathbf{0.65 \pm 0.03}$	$\mathbf{0.67 \pm 0.03}$
Object Use + *Reference Tracklet Statistics** + *Acceleration Statistics*	Sum	0.62 ± 0.04	0.65 ± 0.02
	Product	0.64 ± 0.03	0.66 ± 0.03
	RDF	$\mathbf{0.65 \pm 0.05}$	$\mathbf{0.67 \pm 0.03}$

* sensor-fusion via accelerometer localization

Table 7.4: Activity recognition performance (mean precision and mean recall) achieved with various features, fusion methods and Random Decision Forest classifiers. Intervals represent $\pm$ one standard deviation. Device Locations and Reference Tracklet Statistics use multi-modal fusion by accelerometer localization. Sum, product and RDF (Random Decision Forest) indicate classifier combinations by aggregating class posterior distributions.

Modality Fusion with Random Decision Forests

Random Decision Forest classifiers are used to evaluate (i) fusion via accelerometer localization, (ii) early fusion via feature vector concatenation, and (iii) late fusion using sum-rule, product-rule, and a secondary Random Decision Forest classifier.

The recognition results obtained with various features and fusion methods are shown in Table 7.4. The top section shows the recognition performance for features from accelerometers. Performance obtained with *Object Use* and *Accelerometer Statistics* is comparable

to classifying the same features via SVMs. Unfortunately, Random Decision Forest classifiers applied to histograms (*Absolute Tracklets* and *RETLETS*) did not perform considerably better than chance (not shown). Decision trees with axis-aligned weak learners and a depth in the order of ten seem to be not flexible enough to learn realistic decision boundaries. This suggests that a subset of $10 - 20$ histogram bins are insufficient to distinguish the patterns of activity under investigation.

The second section in Table 7.4 shows recognition performance with two feature types that rely on multi-modal fusion using accelerometer localization. Recognition accuracy for *Device Locations* was lowest among all configurations tested (26% precision and 22% recall). This illustrates that objects are positioned quite freely on the work surface and that their locations do not provide strong cues for activity recognition even in this dataset where all sequences were recorded in a single (confined) kitchen setup involving the same utensils. The lower performance of *Reference Tracklet Statistics* compared to *Acceleration Statistics* can be attributed to the shorter temporal window used for feature extraction (0.53s compared to 5.12s for *Acceleration Statistics*) and imperfect localization.

The combination of different features prior to classification, shown in the third section of Table 7.4, consistently improved recognition performance compared to the individual feature types. These observations strongly support the hypothesis that robust activity recognition benefits from integrating multiple types of cues. It is particularly notable that *Reference Tracklet Statistics* considerably increase recognition performance when combined with *Accelerometer Statistics*. As both feature types encode motion characteristics of accelerometer-equipped objects, one could have expected that these feature types capture no complementary information and recognition performance would not improve when combined. This observation suggests that an accelerometer's motion with respect to other entities in the scene, e.g. a stationary camera or objects interacting with the accelerometer, represents useful information that is not captured by acceleration data with respect to the accelerometer's local reference frame.

Combining class posterior distributions (bottom section of Table 7.4) using the sum-rule

	NULL	add_oil	give_pepper	dress_salad	mix_dressing	mix_ingredients	peel_cucumber	cut_into_pieces	place_into_bowl	serve_salad
NULL	42	10	4	3	7	3	2	4	17	8
add_oil	3	89	2	2	1	0	1	0	1	0
give_pepper	5	1	89	1	2	0	0	0	0	1
dress_salad	1	1	0	72	8	6	1	0	4	8
mix_dressing	6	8	2	6	56	7	1	0	0	14
mix_ingredients	11	0	0	3	2	49	0	1	19	15
peel_cucumber	1	0	0	0	0	0	73	10	15	0
cut_into_pieces	4	0	0	0	0	0	1	69	25	0
place_into_bowl	6	0	0	0	0	0	1	25	67	0
serve_salad	7	0	1	4	1	11	1	1	4	70

Figure 7.6: Confusion Matrix for the method that achieved highest activity recognition accuracy among all configurations considered in Table 7.4. A Random Decision Forest classifier was trained on concatenated *Object Use*, *Acceleration Statistics* and *Reference Tracklet Statistics* features (early fusion). Rows and columns represent ground-truth and predicted class labels, respectively. Numbers represent frequencies in percent and cell gray-levels linearly encode frequencies from 0% (black) to 100% (white).

or the product-rule only showed minor improvement compared to the individual feature types. Note, however, that inference in Random Decision Forests trained on combined features is faster by a factor K than combining classifier outputs of K feature types. Due to the computationally demanding model selection experiments using Random Decision Forests as secondary classifiers were only performed on the combination of features that achieved highest recognition accuracy with early fusion. Here, recognition accuracy was similar to that obtained with early fusion of the same features.

These results strongly suggest to combine feature types prior to classification: early fusion consistently outperformed performance obtained with individual feature types.

The confusion matrix of test results obtained with a Random Decision Forest classifier trained on concatenated *Object Use*, *Acceleration Statistics* and *Reference Tracklet Statistics* features is illustrated in Figure 7.6. This configuration achieved highest activity recognition accuracy among all configurations considered in Table 7.4. For almost all activ-

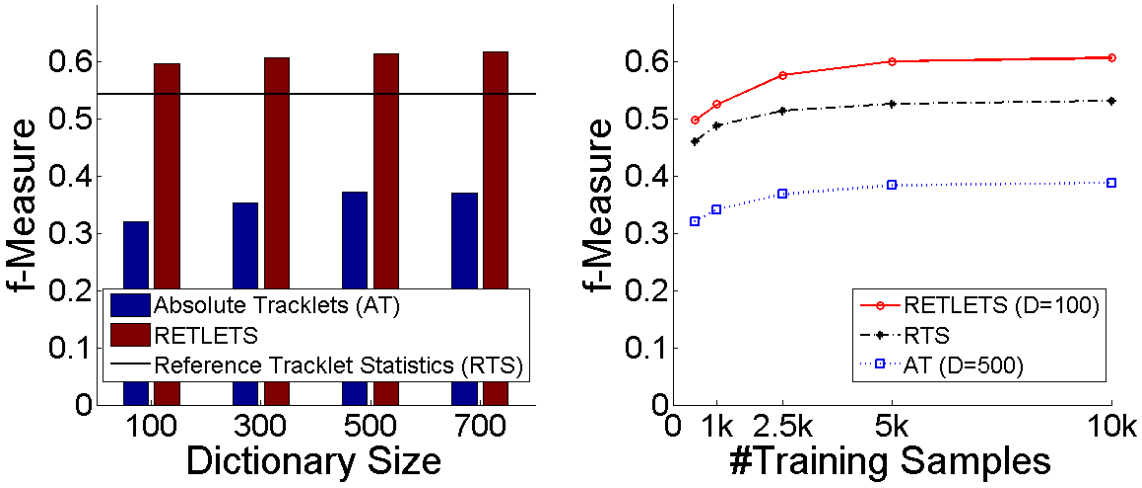


Figure 7.7: Recognition results using Reference Tracklet Statistics, Absolute Tracklets and RETLETS with variation in dictionary size and number of training samples. Note that as *Reference Tracklet Statistics* are not histograms their performance is constant across dictionary sizes (represented as a horizontal line in the Figure on the left).

ities recall was above 50%, except for the $NULL$-activity (42%) and $mix\_ingredients$ (49%). Considering the high intra-class variability and the fact that no temporal activity modelling was used these results are very promising. As the pre- and post-phases of activities involve re-organizing objects on the work surface, there is strong confusion between $NULL$ and all other activities. The large spoon was often used to carry out the $mix\_ingredients$ and $serve\_salad$ activities. As the way the large spoon was moved during these activities was also very similar they were frequently confused. The noticeable confusion between cut_into_pieces and $place\_into\_bowl$ may be due to the knife often being used to scrape chopped ingredients off the chopping board into the bowl. Stronger motion features or a representation of spatial relations between objects might help distinguish these activities.

Absolute Tracklets vs. RETLETS

The three feature types encoding visual motion *Reference Tracklet Statistics*, *Absolute Tracklets*, and *RETLETS* were comparatively evaluated with varied dictionary size for *Absolute Tracklets* and *RETLTS*, and varied number of training samples using SVM classifiers. As shown in Figure 7.7, the feature types that exploit information from accelerometer loc-

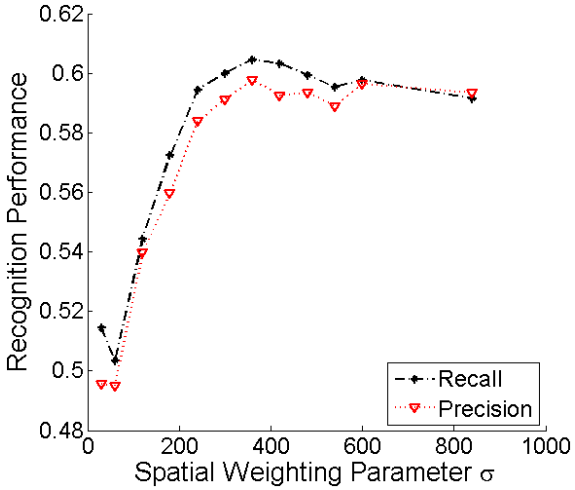


Figure 7.8: Average precision and average recall using RETLETS with varied spatial weighting parameter σ (in pixels).

alization (*Reference Tracklet Statistics* and *RETLETS*) considerably outperformed *Absolute Tracklets*, which are extracted exclusively from video. The proposed feature representation (*RETLETS*) strongly outperformed *Absolute Tracklets* and *Reference Tracklet Statistics*. This result strongly suggests that *relative* motion is important for recognizing the activities under investigation, and suggests that *RETLETS* capture this information well. Dictionary size had less effect on recognition performance for *RETLETS* compared to *Absolute Tracklets*. Performance saturated at about 10k training samples, corresponding to an expected overlap of $\sim 75\%$ between temporal windows.

Spatial Weighting

The impact of applying spatial re-weighting to relative tracklets on recognition performance was evaluated by constructing relative histograms over 100 dictionary words with varied spatial weighting parameter σ (see Equation (4.3), p. 50). Average precision and average recall are plotted in Figure 7.8. Performance rose sharply from $\sigma = 30$ to $\sigma = 360$ pixels. From that point onwards recognition performance was relatively unaffected, falling a little. A possible explanation is that trajectories that were very close to the reference trajectory were likely to exhibit motion similar to the reference trajectory. A relative description of such motion is therefore uninformative. Trajectories that were very far away from the

Individual Features	**Precision**	**Recall**	**f-measure**
Absolute Tracklets	0.37 ± 0.03	0.40 ± 0.02	0.38
Reference Tracklet Statistics	0.54 ± 0.04	0.51 ± 0.03	0.53
RETLETS	0.61 ± 0.02	0.61 ± 0.03	0.61
Accelerometer Statistics	$\mathbf{0.63 \pm 0.07}$	$\mathbf{0.65 \pm 0.03}$	**0.64**
Feature Vector Concatenation			
Baseline			
Accelerometer Statistics + *Absolute Tracklets*	0.65 ± 0.04	0.66 ± 0.03	0.66
Accelerometer Statistics + *Reference Tracklet Statistics*	0.66 ± 0.06	0.67 ± 0.03	0.67
Accelerometer Statistics + *RETLETS* + *Absolute Tracklets*	0.67 ± 0.03	0.69 ± 0.03	0.68
Accelerometer Statistics + *RETLETS* + *Absolute Tracklets* + *Reference Tracklet Statistics*	$\mathbf{0.69 \pm 0.04}$	$\mathbf{0.71 \pm 0.03}$	**0.70**

Table 7.5: Recognition performance obtained using combinations of *Absolute Tracklets* and *RETLETS* with *Reference Tracklet Statistics* and *Accelerometer Statistics*.

reference trajectory on the other hand were less likely to interact with the reference object, justifying the spatial weighting applied here. In all subsequent experiments, *RETLETS* are encoded with 100 codewords and spatial weighting parameter $\sigma = 360$ pixels.

Feature Combination with Support Vector Machines

This section investigates recognition performance using SVMs and various combinations of feature types. Table 7.5 shows recognition results obtained by combining *Absolute Tracklets* and *RETLETS* with *Reference Tracklet Statistics* and *Accelerometer Statistics* using the $MeanKernel$ (see Section 4.6.2, p. 61). Among the individual feature types under investigation, *Accelerometer Statistics* achieved highest recognition performance. Combinations of feature types consistently showed a considerable performance increase compared to individual features. The best performance was achieved by combining *Accelerometer Statistics* with *RETLETS* and *Reference Tracklet Statistics*. This approach of fusing information from video and accelerometer data clearly outperformed the combina-

tion of features extracted from each sensor type independently (*Accelerometer Statistics + Absolute Tracklets* in Table 7.5), the traditional early fusion approach, which showed lowest performance among all feature combinations investigated. This result confirms the hypothesis of this thesis that careful combination of data from video and accelerometers via accelerometer localization can considerably improve activity recognition performance.

It is interesting to note that *RETLETS* performed slightly worse than *Accelerometer Statistics* individually, which may suggest that the additional cost associated with extracting dense tracklets from video, localizing accelerometers, and extracting relative tracklets is unjustified. The traditional sensor fusion approach of concatenating features extracted from each sensor type independently, however, only showed comparable performance to *Accelerometer Statistics*, and strong improvements are only observed with combinations of features that utilize accelerometer localization. This comparison strongly suggests that *features from accelerometer localization establish a crucial link* between features extracted from each sensor type independently, which enables the classifier to discover cross-modal relations.

Kernel Combination The results from comparing three methods for kernel combination $SumDistance$, $MeanDistance$, and $SumKernel$ are shown in Table 7.6. While the effect of kernel combination on recognition accuracy appears to be minor, $SumKernel$ outperforms the other two methods for two combinations of feature types and shows competitive performance for the other two combinations.

7.4 User-Adaptive Classification

This section reports evaluation results on the user-adaptive classification methods introduced in Chapter 5. For generic classifier training a stratified sample of $N_g = 5000$ data-points was used. User-adaptive classifiers were trained on the generic training set and an additional user-specific training set consisting of datapoints from one sequence of the target user. For each user in the test set, two models were evaluated using datapoints from

Feature Type	$SumDistance$	$MeanDistance$	$SumKernel$
Acceleration Statistics + *Absolute Tracklets*	0.66	0.66	0.66
Acceleration Statistics + *Reference Tracklet Statistics*	0.64	0.65	0.67
Acceleration Statistics + *RETLETS*	0.68	0.67	0.68
Acceleration Statistics + *Reference Tracklet Statistics* + *RETLETS*	**0.68**	**0.69**	**0.70**

Table 7.6: Recognition performance as f-measure obtained with different methods for kernel combination.

one sequence of the target user as user-specific training data and the other sequence for testing, and vice versa. As some activities only last a few frames and some might not occur at all in any particular sequence, the user-specific training data was highly imbalanced. An approximately-stratified sample was taken from the user-specific training sequences. Let N_s denote the target size of the user-specific training set and $C = 10$ denote the number of activity classes. An exactly-stratified sample would include N_s/C datapoints from each class. The approximately-stratified sample included all datapoints from activity classes for which less than N_s/C datapoints were available, and a stratified sample of datapoints from all other classes. Let N_c denote the number of datapoints from each class $c = 1, \ldots, C$ in the approximately-startified sample. For SVM training the cost parameter was set differently for each class to be inversely proportional to the number of datapoints from that class in the training set: $C_c = \frac{N_s}{|\mathcal{C}|N_c}$.

7.4.1 Adaptation by Classifier Combination

Adding more training data to a learning algorithm may improve recognition accuracy regardless of whether these additional training data are from the user the system is evaluated on or not. To investigate whether the model actually learns user-specificities a randomized control trial was performed, in which the adaptive model trained on the same subject was compared with an adaptive model trained on a randomly selected other subject from the test

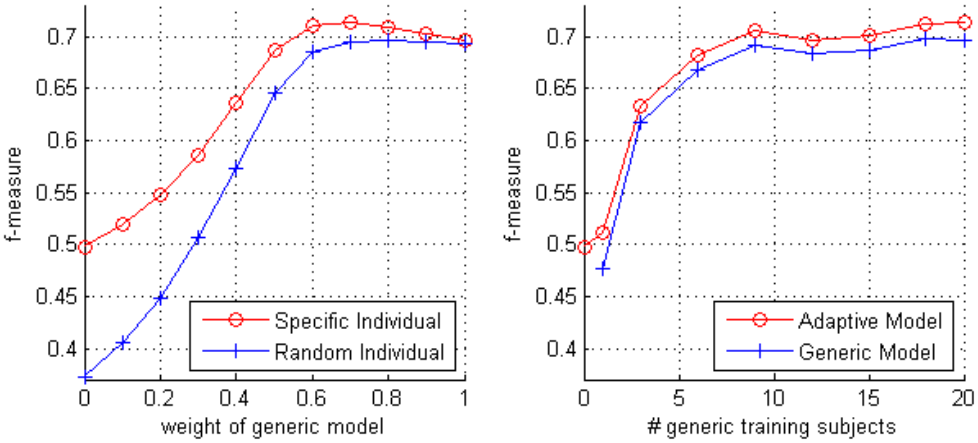


Figure 7.9: Recognition performance of user adaptation via classifier combination with varied mixing weight w_g (left), and varied number of generic training subjects using $w_g = 0.7$ (right).

set. These cases are referred to as *Specific Individual* and *Random Individual*, respectively. Any performance increase observed after training on a *Random Individual* can be attributed to the availability of additional generic training data. The degree by which performance increases further when training the model on the *Specific Individual* compared to training on a *Random Individual* indicates learning of user-specificities.

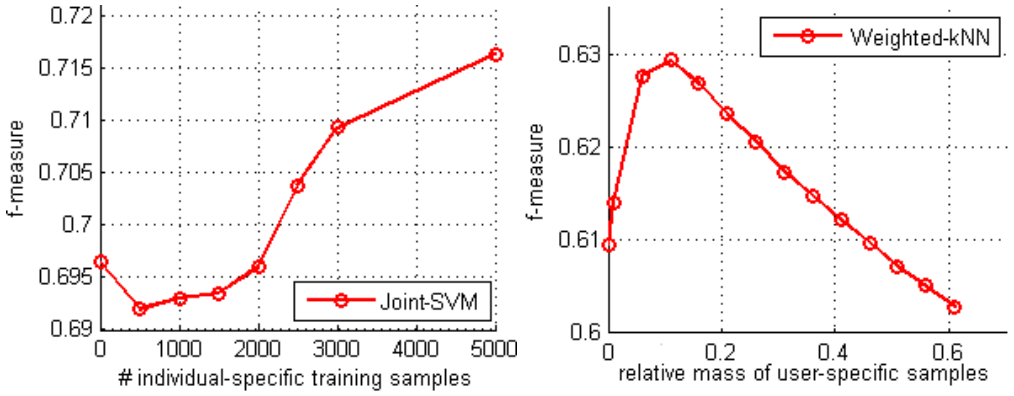
Recognition performance as f-measure with varying weight w_g for mixing the generic with the individual model is plotted in Figure 7.9(left). Combining the generic model with the specific individual model improved performance by 0.017 to 0.714 with a peak at $w_g = 0.7$, whereas the combination with a model trained on a random individual did not increase recognition performance. This experiment confirms that the model learns idiosyncrasies of the test subject. Note that this improvement is based on user-adaptation using a single training sequence and is expected to increase with additional user-specific data.

When training data are scarce, e.g., only a single training sequence is available, it seems particularly valuable for this data to be obtained from the target user. When $w_g = 0$ in Figure 7.9(left), which represents training a single classifier from a single training sequence, we observe 0.125 absolute and 0.335 relative performance increase switching from training on a random subject to training on the target user. This hypothesis was

verified by training a classifier from data of a varied number of random subjects prior to user-adaptation. The generic classifier was combined with an individual-specific classifier using a fixed weight $w_g = 0.7$, whereby the ratio of training sequences from random subjects to training sequences from the target user was varied. Results in Figure 7.9(right) show that the performance gain from user-adaptation reached its minimum when training the generic classifier on data from three subjects, and that the improvement remained approximately constant even after adding data from a further 17 subjects. This experiment confirmed that high gains from user-adaptation can be expected when training data are only available from a few subjects. Surprisingly, the maintained performance gain indicates that user-adaptation is beneficial even if generic data are gathered from a large number of subjects.

7.4.2 Adaptation by Joint SVM Training

User adaptation by training a single SVM on data from randomly selected subjects and on the target user was evaluated by varying the number of user-specific training samples. Results in Figure 7.10(a) show that the performance initially dropped but then continuously rose with added user-specific data to 0.716. The initial drop is due to the employed method for estimating class-dependent cost parameters C_c. In the presence of imbalanced data, which only occurred when user-specific data were added to the training set, the cost parameters only provide valid *relative* weights between classes. The *absolute* cost of misclassification for a given binary optimization problem can however be greater or smaller than with a stratified sample. The highest performance gain (0.020) was observed when 5000 user-specific training samples were added to the generic training data. While absolute performance after user-adaptation by jointly training a single SVM was higher compared to combining classifiers, retraining with generic data considerably increased the training time.



(a) Joint SVM (b) Weighted kNN

Figure 7.10: Recognition performance of user adaptation via joint classifier training (a) SVM with varied number of training samples from the target user, and (b) weighted K-Nearest-Neighbor with varied probability mass m_s of user-specific samples.

7.4.3 Adaptation with Weighted K-Nearest-Neighbour

For weighted K-nearest-neighbour classification, 5000 user-specific samples were added to the generic training data and the number of samples in the local neighbourhood of a test sample was set to $K = 64$ (which had shown good recognition performance in cross-validation on the training data). Evaluation results with varied relative probability mass for user-specific training samples are shown in Figure 7.10(b). Recall that m_s and m_g denote the probability mass of user-specific and generic training datapoints, respectively (see Section 5.3.2, p. 72). Although the highest observed performance increase (0.020) after adding user-specific data with $m_s = 0.11 \cdot m_g$ was comparable to SVM joint training, the absolute recognition performance of 0.63 was not competitive. As there was considerable overlap between temporal windows sampled from user-specific data, the i.i.d. assumption was strongly violated. This explains why the optimal probability mass m_s was below m_g.

7.4.4 Variation across Individuals and Activities

The previous experiments showed that the proposed methods for user-adaptation are well suited to capturing idiosyncrasies. The benefit of adapting a recognition model to a particular subject intuitively depends on the difference of their task-execution *style* from

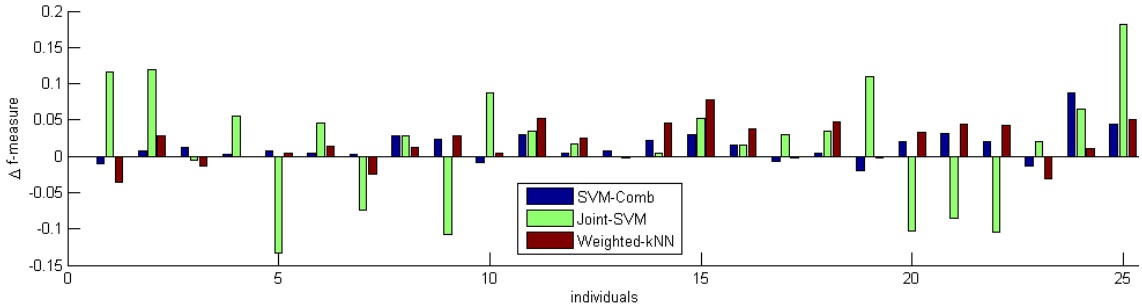


Figure 7.11: Variation in gain from user-adaptation across individuals

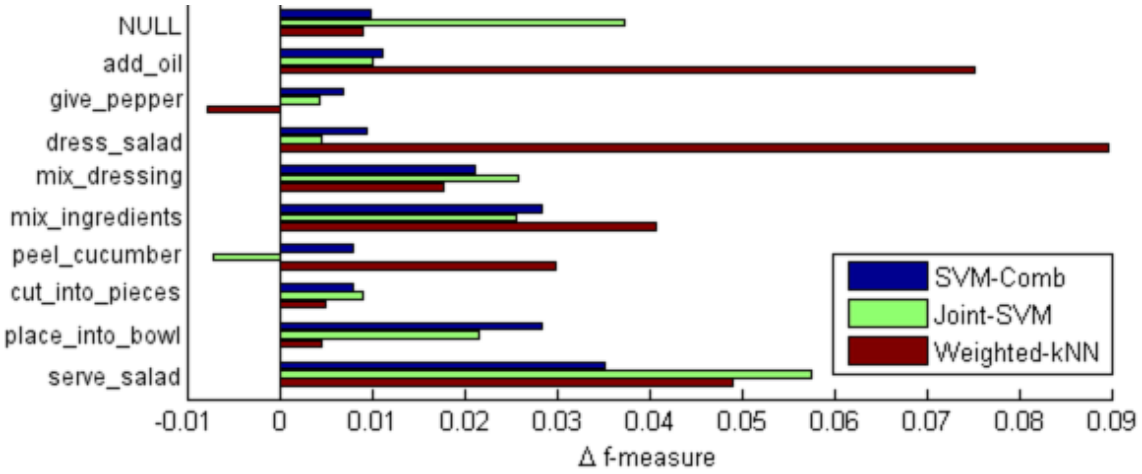


Figure 7.12: Performance gains for each activity.

the norm, and on their *consistency*. The recognition performance after user-adaptation is expected to increase if the execution style is particular to the user, and to decrease after training on one sequence if the target user is inconsistent. Figure 7.11 illustrates the variation in performance gained after user-adaptation across individuals. It provides evidence that could support the intuition, showing that user-adaptation is beneficial for most users, particularly beneficial for some, and has adverse effects for others.

Similarly, the performance gain from user-adaptation varies across activities. Some activities may be performed in many different ways potentially involving different utensils. Additionally, as the amount of available user-specific data may vary across activities, higher gains are expected for activities for which more user-specific data are available. Figure 7.12 shows performance gain per activity obtained with the proposed methods. Particularly high gains were observed for activities that can be executed with a wide range of strategies. Placing ingredients into the bowl, for example, can be performed by grasping

the cut ingredients by hand and placing them into the bowl, picking up the chopping board and scraping the ingredients into the bowl by hand, or scraping the ingredients off the chopping board using a knife. In contrast, the gain from user-adaptation for *cutting into pieces* was below average, indicating only minor variation in strategy across users; more than 40% of the time during which data were acquired consisted of this activity.

7.5 Summary and Discussion

This chapter reported evaluation results on accelerometer tracking, multi-modal activity recognition, and user-adaptive classification. Through quantitative comparative evaluation, it was shown that the proposed accelerometer tracking algorithm outperforms previously proposed methods considerably. The observed performance increase was shown to be primarily due to the proposed similarity measure.

Evaluating an online activity recognition problem using the *50 Salads* dataset, it was discovered that the visual features *Absolute Tracklets* performed considerably worse than *Accelerometer Statistics*, which further motivate a multi-modal approach, and help to justify a sensor-rich environment for certain applications including recognition of manipulation activities.

It may seem surprising that the simplest type of feature considered here, *Object Use*, compared favourably with *Absolute Tracklets* which have shown competitive performance on several standard activity recognition datasets. This result indicates that, in this experimental scenario at least, the identity of objects involved in an activity is more discriminative than a generic description of motion in the scene. The relative importance of the identity of objects involved in an activity and motion descriptors matches our intuition, considering that differences in visually observable motion across food preparation activities are very subtle, and knowledge about the involvement of specialized tools in an activity can strongly reduce the number of possibly occurring activities. Furthermore, the considerable margin between the results using *Accelerometer Statistics* and *Object Use* indicates that object

involvement and motion characteristics are strongly complementary. Scaling a recognition system beyond a single recipe, however, would drastically increase the number of possible activities involving any single object, rendering *Object Use* features less informative.

From a traditional computer vision perspective these results might suggest to use a method in which objects involved in activities of interest are detected and tracked over time, and activities are recognized by reasoning about these object's (relative) position and motion. As argued in Chapter 4, such an approach is problematic for reasons of scalability with the number of objects and reliability. Learning detectors for all objects involved in food preparation activities requires substantial amounts of additional labeled training data for each object class, which is costly to obtain in practice. As large portions of kitchen objects are usually occluded when in use, tracks obtained by visual object detection are expected to be highly unreliable and are therefore of limited value for motion analysis. It is arguably easier to attach accelerometers to additional objects than to train a large number of object detectors that deal well with occlusion. In order to scale the recognition system to a larger number activity classes, it would be useful to incorporate additional visual features and relational histograms that use these as base features. For more fine grained activity recognition where different manipulated ingredients are to be distinguished, local appearance features that encode shape, texture and colour are expected to be particularly beneficial. Encoding visual appearance in spatial proximity to tracked objects through relational histograms promises to facilitate reasoning about utensil-ingredient interactions.

Although the proposed feature representation *RETLETS* did not outperform *Accelerometer Statistics* individually, features from accelerometer localization and *RETLETS* in particular seem to capture key information for discovering cross-modal dependencies: while traditional modality fusion methods such as feature vector concatenation and classifier combination showed comparable recognition performance to *Accelerometer Statistics*, the combination of *Accelerometer Statistics* with *RETLETS* and *Reference Tracklet Statistics* showed a small but noticeable performance increase. All methods for user-adaptive classification increased recognition performance compared to their generic counterparts,

and it was demonstrated that user-specific training data helps even in the presence of a large amount of generic data. The randomized control trial using classifier combination for user-adaptation showed that the model indeed learned user-specific activity characteristics. Results on variation of performance gain via user-adaptation suggest that expected gains depend strongly on the variety of execution styles with which an activity can be performed, the deviation of the target user's style from the norm, and the user's consistency in activity execution.

Chapter 8

Conclusions and Recommendations

8.1 Summary of Contributions

This thesis proposed a novel approach for multi-modal recognition of manipulation activities that combines information from video and accelerometer data through accelerometer tracking and relative motion descriptors.

A method for localizing an accelerometer in the field of view of a stationary camera was introduced and extended to enable long term tracking across multiple episodes in which accelerometer-equipped objects are used. The proposed method is easy to implement, fast to compute, and strongly outperforms previous methods. Quantitative evaluation confirmed superior performance of trajectories from dense optical flow compared to sparse feature point tracking. A first attempt at extending point feature trajectories to 3D was presented. However, evaluation suggests that a more effective way of combining color and depth information is necessary in this regard. Finally, comparison of the proposed similarity measure (TDT) with NCC convincingly showed TDT's superior performance. With the progress towards accurate accelerometer tracking made in this thesis, using embedded accelerometers becomes a reasonably robust approach to visually tracking *arbitrary* objects of sufficient size without relying on their visual appearance.

For multi-modal recognition of manipulation activities it was shown that relative motion descriptors provide a key link between features from different sensor types. Features encoding *Object Use* showed considerable discriminative power, highlighting the importance of identifying the objects involved in manipulation activities. Similar information might have been obtained using RFID tags as an alternative to accelerometers. However, with a recognition accuracy below 50% it is clearly insufficient to solely rely on this type of feature. Motion features extracted from accelerometer data provided the strongest cues among individual feature types investigated. Comparative evaluation of motion features from accelerometers and video highlighted the large performance gap between visual and accelerometer-based motion features. This result should motivate future research on motion descriptors extracted from video. By encoding dense tracklets relative to the visual trajectories of accelerometer-equipped objects as *RETLETS* a considerably better activity

recognition performance was obtained compared to their absolute representation. The combination of this novel feature representation with features extracted from accelerometer data and video showed a higher recognition performance compared to combining only features extracted from each modality independently, which is the traditional approach to sensor fusion for activity recognition. These results justify a multi-modal approach and indicate the importance of developing methods for effective modality fusion, of which one example was presented in this thesis. Note that the choice of activities used for evaluation here was deliberately made to exclude important factors such as the manipulated ingredients. If the task was set out to differentiate between different ingredients being cut into pieces or placed into the bowl, additional (visual) features would be necessary to robustly recognize such activities. It is expected that the integration of visual information will be even more beneficial for reasoning about interactions between multiple entities such as *moving the chopped tomato from the chopping board into the bowl*.

Specifically addressing scenarios in which a recognition system is primarily used by a single person (e.g., cognitive situational support), this thesis investigated methods for adapting activity models to a target user based on user-specific training data. Three methods for user-adaptive classification were presented: combining classifiers that were trained separately on generic and user-specific data, jointly training a single SVM from generic and user-specific data, and a weighted K-nearest-neighbour formulation with different probability mass assigned to generic and user-specific samples. The experiments confirmed that adapting an activity recognition system to a target user can considerably increase recognition accuracy. Via randomized control trials it was further shown that this performance increase is indeed attributable to user idiosyncrasies as opposed to being a mere consequence of additional data available for training. Variation in performance increase after user adaptation across individuals indicates that users with execution style that deviates from the norm benefit most from adaptation. Similarly, user-adaptation is particularly advantageous for tasks that may be accomplished with a wide range of execution strategies, such as many tasks involved in food preparation. Which of the

presented methods for user-adaptation to apply in a particular scenario depends on the specific constraints on storage space and computation time. If there are no constraints on storage space and time for classifier retraining, jointly training a single SVM from generic and user-specific data is expected to achieve higher recognition performance than the other presented methods. The weighted K-nearest-neighbour approach would be recommended in the extreme case where only minimal computation time for initial classifier training and user-adaptation is available. If both storage space and retraining time are limited but not negligible, combining classification results from separately trained SVMs should be chosen out of the three methods discussed in this paper. While it showed slightly lower performance *gain* from user-adaptation than joint SVM training and weighted K-nearest-neighbour classification, *final* recognition performance was comparable to the best observed performance. This model provides a reasonable compromise that may be particularly useful in practical applications.

All proposed methods were evaluated on two new challenging datasets of food preparation activities that have been made publicly available. Both datasets feature a novel combination of video and accelerometers attached to objects. The *Accelerometer Localization* dataset is the first public dataset that enables quantitative evaluation of accelerometer tracking algorithms. The *50 Salads* dataset with its detailed activity annotations may be used for - and hopefully stimulates - research in areas such as activity recognition, activity spotting, sequence analysis, progress tracking, sensor fusion, transfer learning, and user-adaptation.

Food preparation activities share many characteristics with other manipulation activities in, .e.g., manufacturing, assembly, and repair tasks. For example, they involve complex interactions of a large number of different objects, which makes an approach that exclusively relies on embedded sensors impractical, and the exhibited motion can be subtle, which makes it hard to distinguish activities visually. Therefore it is expected that the methods proposed in this thesis, although only evaluated on food preparation activities, generalize well to activity recognition in other contexts that share these characteristics.

8.2 Recommendations

8.2.1 Accelerometer Localization and Tracking

The proposed method for accelerometer tracking has several limitations. It assumes that the camera is stationary, which is reasonable if the camera is installed as part of the environment, but makes the algorithm in its current form unusable with hand-held or body-worn cameras. With a moving camera it is necessary to correct for camera motion when estimating acceleration along point trajectories. This problem has been addressed by Wang et al. [111] who propose to estimate a homography from consecutive video frames and warp the later frame before extending point trajectories. If the camera device also includes an accelerometer (e.g., Microsoft Kinect or Google Glass), it could be an alternative solution to subtract device acceleration from acceleration estimated along point trajectories prior to matching. As accelerometers measure proper acceleration relative to free fall, the challenge here is to factor acceleration due to camera motion and acceleration due to resisting gravity.

The localization algorithm also assumes that accelerometer-equipped objects do not enter or leave the camera view, and that the distance of objects from the camera is roughly known and does not change over time. A probabilistic formulation through, e.g., particle filtering could potentially relax these assumptions. With hidden variables that indicate presence/ absence and 3D location of an accelerometer-equipped object with respect to the camera reference frame modeled as latent variables, and accelerometer data and acceleration along point trajectories as observations, a set of particles could model the maximum a posteriori distribution suggesting likely distances from the camera and how likely the accelerometer is in the camera view. By drawing particles from a mixture of proposal distributions that model prior knowledge about object presence/ absence, object occlusions, object stationarity, and object motion when in use, all of these issues could be addressed explicitly. Some stronger method for temporal filtering would also reduce *jumping* artifacts in location estimates.

Modelling and recovering accelerometer orientation could also enable stronger matching criteria that use 3D acceleration as opposed to acceleration norms. If an accelerometer's orientation relative to the camera view could be recovered, its local coordinate system could be rotated to align with the camera reference frame, and 3D acceleration estimates along point trajectories could be directly matched to 3D acceleration captured by the accelerometer. One way to recover accelerometer orientation is to integrate accelerometers with other sensors such as gyroscopes and magnetometers. While there are currently issues of battery consumption with gyroscopes and issues of magnetic interference of electric appliences with magnetometers, these may be overcome in the future. Data from these additional sensors would not only be helpful for pose recovery but could also be used in similarity estimation, which may further increase localization performance.

While the proposed similarity measure has been shown to outperform normalized cross-correlation, it leaves room for improvement in localization performance. One direction for potential improvement is the exploration of similarity measures that are based on temporal statistics of acceleration sequences as opposed to aggregates over point-wise comparisons such as TDT and NCC. For example, mutual information and empirical cumulative distribution functions estimated over temporal windows may prove to be even more robust to synchronization errors and artifacts arising from the comparison of instantaneous acceleration captured by accelerometers and mean acceleration estimated between consecutive video frames.

Finally, it would be interesting to evaluate the proposed method in other application scenarios.

8.2.2 Relational Histograms

The descriptive power of *RETLETS* as used in this thesis relies on accurate accelerometer tracking. There are at least two ways in which this dependency can be weakened. First, by enforcing a unique match between an accelerometer and one point trajectory large parts of useful information provided by a similarity map are discarded. A method for encoding

relative tracklets with respect to multiple reference tracklets for each accelerometer could lead to a more robust descriptor. If for example trajectories in two (or more) distinct image locations show strong and almost equal similarity to an accelerometer, relative tracklets should be extracted relative to both of these trajectories In order to maintain a constant feature vector length, histograms of relative tracklets with respect to multiple reference tracklets per accelerometer could be aggregated. If, on the other hand, none of the point trajectories show a strong similarity with an accelerometer after an extended period of time, this could indicate that the accelerometer is outside the camera's field of view. In that case it would not be sensible to estimate relative tracklets with respect to any reference tracklet. As the accelerometer is potentially far away from the locations of tracklets in the video, the hypothetical weights on relative tracklets would be diminishingly small, which would suggest a uniform histogram for that accelerometer. Second, object locations estimated via accelerometer tracking could be used to bootstrap a visual object model similarly to Wu et al. [118]. Accelerometer-equipped objects could then be tracked using information captured by the accelerometer *and* the appearance model, which should give a more robust estimate, or accelerometers could be removed from the objects once appearance models have been learned. In the second scenario, accelerometers could replace the tedious manual task of labeling object locations in video. Accelerometers could be attached to all objects whose locations are to be annotated,and accelerometer data could be captured while a video is being recorded. Object locations could then be automatically estimated for every frame in the video using the accelerometer tracking algorithm proposed in this thesis.

The proposed feature representation can be used with reference tracklets obtained in any way. These could, for example, be trajectories of visually tracked objects, or prominent point trajectories. Future work could evaluate RETLETS with reference tracklets from other sources such as visual object tracking.

This thesis only investigated a single example from the family of object-generic relational histograms. Descriptors that encode other information than motion relative to the locations of tracked objects promise to capture additional valuable information for recog-

nizing manipulation activities. For example, appearance features such as texture or colour could be spatially weighted depending on their distance to reference locations in order to characterize *what* is in spatial proximity to objects. Giving a higher weight to appearance features in spatial proximity to the objects would provide some focus of attention, which could be useful to infer the objects accelerometer-equipped objects interact with. If, for example, there is a reddish object with a tomato-like texture close to the estimated location of a knife, such a descriptor could help inferring that the knife is indeed cutting a tomato.

The length of the feature vector of object-generic relational histograms grows linearly with the number of tracked reference objects. This could be a major limitation if there is a large number of tracked objects, or if the size of the relational codebook is large. In these cases some feature selection through, e.g., frequent or discriminative item-set mining could be useful to identify sets of relational codewords that are - for each individual reference object - particularly frequent or discriminative [22, 53].

8.2.3 User-Adaptive Classification

The methods for user-adaptive classification presented in this thesis adapt to a target user offline in one shot, assuming that adaptation is complete once the available set of labeled user-specific data has been processed. However, their adaptation to user-specific data becoming available successively (online) is trivial: new user-specific samples could be included in the weighted k-nearest-neighbour classifier as they become available, and the user-specific classifier for classifier combination, and the joint SVM could be retrained periodically. The main challenge is finding good values for the free model parameters. In this thesis, good values have been found through extensive cross-validation. While this approach is also possible with repeated online training of a user-adaptive classifier, line- or grid-search with cross-validation is resource intensive and does not exploit an expected gradual change in parameter values. Recently, Bayesian methods have shown promising results for optimizing hyper-parameters in machine learning by modeling classification accuracy obtained with different hyper-parameters as samples from a Gaussian process

[94]. This approach could be adapted to support gradually changing numbers of training samples, which would help finding good parameters for a new ratio of generic to user-specific training data.

Acquiring labeled training data from the target user is very costly and might in some scenarios be practically infeasible. Recent research on unsupervised user-adaptation in one shot [114] and gradual adaptation over time [27] could be equally well applied in the context of manipulation activities. In conjunction with gradual adaptation over time, one interesting direction of future research is the assessment of a user's behavioral change in order to infer degradation of cognitive capabilities (e.g., in situational support systems [40]) or improvement of some skill (e.g., for training applications [35]).

8.2.4 Recognition of Manipulation Activities

The methods for multi-modal activity recognition presented in this thesis only include features that encode motion information. As mentioned in the Introduction, evaluating appearance models on the *50 Salads* dataset is expected to be strongly biased as these would model the relatively stable appearance of a person's clothing and the appearance of particular object instances (e.g., the blue bowl or the green olive oil bottle). Evaluation results would give an overly optimistic estimate of generalization performance. On a dataset that includes a wider variety of objects one should, however, investigate other types of features such as colour and texture, and relational histograms using these features. These are expected to further improve recognition performance, and might enable recognition of the manipulated object (i.e., the ingredient). Yang et al. [120] recently argued that the consequence of a manipulation activity, i.e. the state into which a manipulated object is being transformed, is a robust cue for recognition. If this target state could be detected reliably, it would certainly provide important complementary information that could be easily combined with the methods proposed in this thesis.

As this thesis builds on research in two rapidly evolving research areas, computer vision and ubiquitous computing, the base features used for evaluation (*Dense Tracklets* and

Accelerometer Statistics) are not any more the state-of-the-art for visual and accelerometer-based activity recognition by the time of writing. Note, however, that the methods presented here can be easily combined with additional visual features, e.g. HoG, HoF and MBH, and *Accelerometer Statistics* can be replaced by other accelerometer-based features such as [77] without invalidating the arguments and findings presented in this thesis. Although recognition performance achieved with stronger accelerometer-based and visual features may be higher, modelling cross-modal dependencies through accelerometer localization and relational histograms is nevertheless expected to improve recognition performance compared to using either of these sensor types individually and compared to traditional sensor fusion techniques.

The 50 Salads dataset has richer annotation than used here. Specifically, activities were split into preparation, core and post-phases, and these phases were annotated as temporal intervals. Each activity annotation also includes the ingredient acted upon (e.g., *cut tomato into pieces*). These detailed annotations may be used in future work to investigate the main sources of confusion errors between activities and for evaluating methods that simultaneously reason about motion and objects acted upon.

For future work, evaluating the proposed method in different scenarios such as assembly tasks, repair tasks, sports (e.g., climbing), surgical applications (e.g., suturing), and social interactions would be desirable to further support the effectiveness of our method. All of these scenarios could benefit from multi-modal activity recognition using accelerometer localization. Currently, there is a strong unmet need for multi-modal activity recognition datasets. This is partly due to the substantial effort necessary for careful planning, data acquisition and annotation. Due to the novelty of the proposed sensor setup, specifically embedding accelerometers into key objects involved in manipulation activities as opposed to placing them on a person's body, most of the publicly available datasets are unsuitable for evaluating the proposed method for multi-modal activity recognition. As the accelerometer tracking algorithm in its current form requires the camera intrinsic parameters to be known, and the distance of accelerometers from the camera to be fixed and known, the only other

dataset that includes accelerometers attached to objects, the Opportunity dataset [82], is also not usable.

While this thesis has shown that manipulation activities can be recognized more robustly with a multi-modal approach using accelerometer localization, relational histograms, and user-adaptive classification, the proposed methods did not exploit the temporal structure in which activities occur. Although food preparation activities involved in preparing a recipe do not have to follow a unique order, one can usually define a temporal partial ordering over activities. For example, ingredients are only mixed after all ingredients have been cut into pieces and placed into a bowl, and the final salad is only served onto a plate after all other steps of the recipe have been completed. Exploiting this structure could not only improve recognition performance further but may also be used to detect a person's deviation from a valid order of steps, which could be a trigger for issuing a guiding prompt back to a user. Preliminary evaluation (not reported in this thesis) suggests that simple temporal models such as Hidden Markov Models (HMMs) applied to the recognition activity scores obtained by classifying subsequent temporal windows are not powerful enough to model the complex temporal structure of food preparation activities. Research by Iscen and Duygulu [44] suggests that modelling transitions between pre-segmented temporal blocks of activities would be more promising. Recipe structures commonly have a checklist-type structure in which activities that have been performed in the past will not occur again, which could be exploited to effectively prune the search space for future activities.

Bibliography

[1] J. K. Aggarwal and M. S. Ryoo. Human activity analysis: A review. *ACM Computing Surveys*, 43(3):16:1 – 16:43, 2011.

[2] M. Albanese, R. Chellappa, N. Cuntoor, V. Moscato, A. Picariello, V. S. Subrahmanian, and O. Udrea. PADS: a probabilistic activity detection framework for video data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(12):2246–2261, 2010.

[3] S. Bansal, S. Khandelwal, S. Gupta, and D. Goyal. Kitchen activity recognition based on scene context. In *Proceedings of the International Conference on Image Processing (ICIP 2013), Melbourne, Australia*, pages 3461–3465, 2013.

[4] L. Bao and S. S. Intille. Activity recognition from user-annotated acceleration data. In *Proceedings of the 2nd International Pervasive Computing Conference, Linz/ Vienna, Austria*, pages 1–17, 2004.

[5] A. Behera, D. C. Hogg, and A. G. Cohn. Egocentric activity monitoring and recovery. In *Proceedings of the 11th Asian Conference on Computer Vision (ACCV 2011), Daejeon, Korea*, pages 519–532, 2012.

[6] P. Bilinski, E. Corvee, S. Bak, and F. Bremond. Relative dense tracklets for human action recognition. In *Proceedings of the 10th IEEE International Conference on*

Automatic Face and Gesture Recognition (FG 2013), Shanghai, China, pages 1–7, 2013.

[7] J.-Y. Bouguet. Pyramidal implementation of the Lucas Kanade feature tracker. In *Proceedings of the USENIX Annual Technical Conference, Monterey, California, USA*, pages 1–9, 1999.

[8] M. Brand, N. Oliver, and A. Pentland. Coupled hidden Markov models for complex action recognition. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR 1997), San Juan, Puerto Rico*, pages 994 – 999, 1997.

[9] M Bregonzio, J. Li, S. Gong, and T. Xiang. Discriminative topics modelling for action feature selection and recognition. In *Proceedings of the British Machine Vision Conference (BMVC 2010), Aberystwyth, UK*, pages 8.1–8.11, 2010.

[10] D. C. Brown. Decentering distortion of lenses. *Photometric Engineering*, 32(3):444–462, 1966.

[11] A. Burns and P. Rabins. Carer burden in dementia. *International Journal of Geriatric Psychiatry*, 15(S1):S9–S13, 2000.

[12] A. R. Cassandra, L. P. Kaelbling, and M. L. Littman. Acting optimally in partially observable stochastic domains. In *Proceedings of the twelfth Conference on Artificial Intelligence (AAAI 1994), Seattle, Washington, USA*, pages 1–6, 1994.

[13] L. Chen, C. D. Nugent, J. Biswas, and J. Hoey, editors. *Activity Recognition in Pervasive Intelligent Environments*. Springer/Atlantis Press, 2011.

[14] D. Chetverikov and J. Verestoy. Tracking feature points: a new algorithm. In *Proceedings of the IEEE International Conference on Pattern Recognition (ICPR 1998), Brisbane, Australia*, volume 2, pages 1436–1438, 1998.

[15] P. Corke, J. Lobo, and J. Dias. An introduction to inertial and visual sensing. *International Journal of Robotics Research*, 26(6):519–535, 2007.

[16] A. Criminisi, J. Shotton, and E. Konukoglu. Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning. *Foundations and Trends in Computer Graphics and Vision*, 7(2–3):81–227, 2012.

[17] A. Dame and E. Marchand. Optimal detection and tracking of feature points using mutual information. In *Proceedings of the IEEE International Conference on Image Processing (ICIP 2009), Cairo, Egypt*, pages 3601–3604, 2009.

[18] F. de la Torre, J. Hodgins, J. Montano, S. Valcarcel, R. Forcada, and J. Macey. Guide to the Carnegie Mellon University multimodal activity (CMU-MMAC) database. Technical report, Robotics Institute, Carnegie Mellon University, 2009.

[19] G. Farnebäck. Two-frame motion estimation based on polynomial expansion. In *Proceedings of the 13th Scandinavian Conference on Image Analysis*, LNCS 2749, pages 363–370, Gothenburg, Sweden, June-July 2003.

[20] A. Fathi, A. Farhadi, and J. M. Rehg. Understanding egocentric activities. In *Proceedings of the International Conference on Computer Vision (ICCV 2011)*, pages 407–414, Barcelona, Spain, 2011.

[21] A. Fathi, X. Ren, and J. M. Rehg. Learning to recognize objects in egocentric activities. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2011)*, pages 3281–3288, Colorado Springs, Colorado, USA, 2011.

[22] B. Fernando, E. Fromont, and T. Tuytelaars. Effective use of frequent itemset mining for image classification. In *Proceedings of the European Conference on Computer Vision (ECCV 2010), Hersonissos, Heraklion, Crete, Greece*, pages 214–227, 2010.

[23] D. Figo, P. C. Diniz, and D. R. Ferreira. Preprocessing techniques for context recognition from accelerometer data. *Personal and Ubiquitous Computing*, 14(7):645 – 662, 2010.

[24] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[25] G. Forman and M. Scholz. Apples-to-apples in cross-validation studies: Pitfalls in classifier performance measurement. *ACM SIGKDD Explorations*, 12(1):49–57, 2010.

[26] K. Förster, A. Biasiucci, R. Chavarriaga, JdR. Millan, D. Roggen, and G. Tröster. On the use of brain decoded signals for online user adaptive gesture recognition systems. In *Proceedings of the International Conference on Pervasive Computing*, pages 427–444, Helsinki, Finland, May 2010.

[27] K. Förster, D. Roggen, and G. Tröster. Unsupervised classifier self-calibration through repeated context occurences: Is there robustness against sensor displacement to gain? In *Proceedings of the International Symposium on Wearable Computing (ISWC 2009)*, pages 77–84, Linz, Austria, 2009.

[28] A. Gaidon, Z. Harchaoui, and C. Schmid. Recognizing activities with cluster-trees of tracklets. In *Proceedings of the British Machine Vision Conference (BMVC 2012), Dundee, Scotland, UK*, pages 30.1–30.13, 2012.

[29] Willow Garage. Opencv, 2012 (accessed January 06, 2012). `http://opencv.willowgarage.com/wiki/`.

[30] T. Giovannetti, D. J. Libon, L. J. Buxbaum, and M. F. Schwartz. Naturalistic action impairments in dementia. *Neuropsychologia*, 40(8):1220–1232, 2002.

[31] P. Goel, S. I. Roumeliotis, and G. S. Sukhatme. Robust localization using relative and absolute position estimates. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 1999), Kyongju, South Korea*, pages 1134–1140, 1999.

[32] A. Gupta, A. Kembhavi, and L. S. Davis. Observing human-object interactions: Using spatial and functional compatibility for recognition. *IEEE Transactions On Pattern Analysis and Machine Intelligence*, 31(10):1775–1789, 2009.

[33] S. Hadfield and R. Bowden. Kinecting the dots: Particle based scene flow from depth sensors. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV 2011), Barcelona, Spain*, pages 2290–2295, 2011.

[34] N. Y. Hammerla, R. Kirkham, P. Andras, and T. Plötz. On preserving statistical characteristics of accelerometry data using their empirical cumulative distribution. In *Proceedings of the International Symposium on Wearable Computing (ISWC 2013)*, pages 65–68, 2013.

[35] N. Y. Hammerla, T. Plötz, P. Andras, and P. Olivier. Assessing motor performance with PCA. In *Proceedings of the International Workshop on Frontiers in Activity Recognition using Pervasive Sensing*, pages 18–23, 2011.

[36] X. He and Y. Zhao. Fast model selection based speaker adapation for nonntative speech. *IEEE Transactions on Speech and Audio Processing*, 11(4):298–307, 2003.

[37] S. Henderson and S. Feiner. Exploring the benefits of augmented reality documentation for maintenance and repair. *IEEE Transactions on Visualization and Computer Graphics*, 17(10):1355–1368, 2011.

[38] J. Hoey, T. Plötz, D. Jackson, A. Monk, C. Pham, and P. Olivier. Rapid specification and automated generation of prompting systems to assist people with dementia. *Pervasive and Mobile Computing*, 7(3):299–318, 2010.

[39] J. Hoey, P. Poupart, A. v. Bertoldi, T. Craig, C. Boutilier, and A. Mihailidis. Automated handwashing assistance for persons with dementia using video and a partially observable Markov decision process. *Computer Vision and Image Understanding*, 114(5):503–519, 2010.

[40] J. Hoey, A. v. Bertoldi, P. Poupart, and A. Mihailidis. Assisting persons with dementia during handwashing using a partially observable Markov decision process. In *Proceedings of the International Conference on Computer Vision Systems (ICVS 2010), Bielefeld, Germany*, pages 1–10, 2007.

[41] C.-H. Hsu and C.-H. Yu. An accelerometer based approach for indoor localization. In *Proceedings of the 2009 Symposia and Workshops on Ubiquitous, Autonomic and Trusted Computing (UIC-ATC 2009), Brisbane, Queensland, Australia*, pages 223–227, Washington, DC, USA, 2009.

[42] J. Huang, S. R. Kumar, M. Mitra, W.-J. Zhu, and R. Zabih. Image indexing using color correlograms. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 1997), San Juan, Puerto Rico*, pages 762–768, 1997.

[43] T. Huynh, M. Fritz, and B. Schiele. Discovery of activity patterns using topic models. In *Proceedings of the 10th International Conference on Ubiquitous Computing (UbiComp 2008), Seoul, South Korea*, pages 10–19, 2008.

[44] A. Iscen and P. Duygulu. Knives are picked before slices are cut: Recognition through activity sequence analysis. In *Proceedings of the 5th International Workshop on Multimedia for Cooking & Eating Activities (CEA 2013)*, pages 3–8, 2013.

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

[46] Y. Kawahara, N. Ryu, and T. Asami. Monitoring daily energy expenditure using a 3-axis accelerometer with a low-power microprocessor. *e-Minds*, pages 1–10, 2009.

[47] S.-R. Ke, H. L. U. Thuc, Y.-J. Lee, J.-N. Hwang, J.-H. Yoo, and K.-H. Choi. A review on video-based human activity recognition. *Computers*, 2(2):88–131, 2013.

[48] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas. On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20:226–239, 1998.

[49] R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI 1995), Montreal, Quebec, Canada*, pages 1137–1143, 1995.

[50] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles And Techniques*. MIT Press, November 2009.

[51] M. Krinidis, N. Nikolaidis, and I. Pitas. 2-D feature-point selection and tracking using physics-based deformable surfaces. *IEEE Transactions on Circuits and Systems for Video Technology*, 17(7):876–888, 2007.

[52] H. Kuehne, A. B. Arslan, and T. Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2014), Columbus, Ohio, USA*, pages 780–787, 2014.

[53] B. G. V. Kumar and Ioannis Patras. Supervised dictionary learning for action localization. In *Proceedings of the International Conference on Automatic Face and Gesture Recognition (FG 2013), Shanghai, China*, pages 1–8, 2013.

[54] I. Laptev. On space-time interest points. *International Journal of Computer Vision*, 64(2/3):107–123, 2005.

[55] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2008), Anchorage, Alaska, USA*, pages 1–8, 2008.

[56] I. Laptev and P. Perez. Retrieving actions in movies. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV 2007), Rio de Janeiro, Brazil*, pages 1–8, 2007.

[57] J. Lei, X. Ren, and D. Fox. Fine-grained kitchen activity recognition using RGB-D. In *Proceedings of the International Conference on Ubiquitous Computing (UbiComp 2013), Pittsburgh, Pensylvania, USA*, pages 208–211, 2012.

[58] A. Licsar and T. Sziranyi. User-adaptive hand gesture recognition system with interactive training. *Image and Vision Computing*, 23:1102–1114, 2005.

[59] J. Liu, J. Luo, and M. Shah. Recognizing realistic actions from videos "in the wild". In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2009), Miami Beach, Florida, USA*, pages 1–8, 2009.

[60] J. Liu, Z. Wang, L. Zhong, J. Wickramasuriya, and V. Vasudevan. uWave: Accelerometer-based personalized gesture recognition and its applications. *Pervasive and Mobile Computing*, 5(6):657–675, 2009.

[61] B. Logan, J. Healey, M. Philipose, E. Munguia Tapia, and S. Intille. A long-term evaluation of sensing modalities for activity recognition. In *Proceedings of the International Conference on Ubiquitous Computing (UbiComp 2007), Innsbruck, Austria*, pages 483–500, 2007.

[62] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI 1981), Vancouver, British Columbia, Canada*, pages 674–679, 1981.

[63] P. Lukowicz, J. A. Ward, H. Junker, M. Stäger, G. Tröster, A. Atrash, and T. Starner. Recognizing workshop activity using body worn microphones and accelerometers. In *Proceedings of the International Conference on Pervasive Computing, Linz, Austria*, 2004.

[64] Y. Maki, S. Kagami, and K. Hashimoto. Accelerometer detection in a camera view based on feature point tracking. In *Proceedings of the IEEE/SICE International Symposium on System Integration, Sendai, Japan*, pages 448–453, 2010.

[65] M. Marszałek, I. Laptev, and C. Schmid. Actions in context. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2009), Miami, Florida, USA*, pages 2929–2936, 2009.

[66] P. Matikainen, M. Hebert, and R. Sukthankar. Trajectons: Action recognition through the motion analysis of tracked features. In *Proceedings of the International Conference on Computer Vision (ICCV 2009), Kyoto, Japan*, pages 514–521, 2009.

[67] R. Messing, C. Pal, and H. Kautz. Activity recognition using the velocity histories of tracked keypoints. In *Proceedings of the International Conference on Computer Vision (ICCV 2009), Kyoto, Japan*, pages 104–111, 2009.

[68] V. I. Morariu and L. S. Davis. Multi-agent event recognition in structured scenarios. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2011), Colorado Springs, Colorado, USA*, pages 3289–3296, 2011.

[69] N. Nejati and T. Könik. Probabilistic relational learning of human behavior models. In *AAAI Spring Symposium: Human Behavior Modeling, Stanford, California, USA*, pages 62–67, 2009.

[70] A. Niculescu-Mizil and R. Caruana. Obtaining calibrated probabilities from boosting. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI 2005), Edinburgh, UK*, pages 413–420, 2005.

[71] A. Nosary, L. Heutte, and T. Paquet. Unsupervised writer adaptation applied to handwritten text recognition. *Pattern Recognition*, 37(2):385–388, 2004.

[72] P. Olivier, A. M. Monk, J. Hoey, and G. Xu. Ambient kitchen: Designing situated services using a high fidelity prototyping environment. *Workshop on Affect & Behaviour Related Assistance in the Support of the Elderly (PETRA-09), Corfu, Greece*, pages 47.1–47.7, 2009.

[73] D. Oneata, J. Verbeek, and C. Schmid. Action and event recognition with Fisher vectors on a compact feature set. In *Proceedings of the International Conference on Computer Vision (ICCV 2013), Sydney, Australia*, pages 1817–1824, 2013.

[74] M. Pei, Y. Jia, and S.-C. Zhu. Parsing video events with goal inference and intent prediction. In *Proceedings of the International Conference on Computer Vision (ICCV 2011), Barcelona, Spain*, pages 487–494, 2011.

[75] C. Pham and P. Olivier. Slice&Dice: recognizing food preparation activities using embedded accelerometers. *Ambient Intelligence, LNCS*, 5859:34–43, 2009.

[76] C. Pham, T. Plötz, and P. Olivier. A dynamic time warping approach to real-time activity recognition for food preparation. *Ambient Intelligence, LNCS*, 6439:21–30, 2010.

[77] T. Plötz, N. Y. Hammerla, and P. Olivier. Feature learning for activity recognition in ubiquitous computing. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1729 – 1734, 2012.

[78] M. E. Pollack. Intelligent technology for an aging population: The use of AI to assist elders with cognitive impairment. *AI Magazine*, 26(2):9–24, 2005.

[79] R. Poppe. A survey on vision-based human action recognition. *Image and Vision Computing*, 28(6):976–990, 2010.

[80] C. Rao, A. Yilmaz, and M. Shah. View-invariant representation and recognition of actions. *International Journal of Computer Vision*, 50(2):203 – 226, 2002.

[81] P. Rhienmora, P. Haddawy, S. Suebnukarn, and M. N. Dailey. Intelligent dental training simulator with objective skill assessment and feedback. *Artificial Intelligence in Medicine*, 52(2):115–121, 2009.

[82] D. Roggen, A. Calatroni, M. Rossi, T. Holleczek, K. Förster, G. Tröster, P. Lukowicz, D. Bannach, G. Pirkl, A. Ferscha, J. Doppler, C. Holzmann, M. Kurz, G. Holl,

R. Chavarriaga, M. Creatura, and J. del R. Millan. Collecting complex activity data sets in highly rich networked sensor environments. In *Proceedings of the 7th International Conference on Networked Sensing Systems (INSS), Kassel, Germany*, pages 233–240. IEEE Computer Society Press, June 2010.

[83] M. Rohrbach, S. Amin, M. Andriluka, and B. Schiele. A database for fine grained activity detection of cooking activities. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June, Providence, United States*, pages 1194–1201, 2012.

[84] M. S. Ryoo and J. K. Aggarwal. Hierarchical recognition of human activities interacting with objects. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2007), June, Austin, Texas, USA*, pages 1–8, 2007.

[85] M. S. Ryoo and J. K. Aggarwal. Spatio-temporal relationship match: Video structure comparison for recognition of complex human activities. In *Proceedings of the International Conference on Computer Vision (ICCV 2009), Kyoto, Japan*, pages 1593–1600, 2009.

[86] S. Savarese, A. del Pozo, J. C. Niebles, and L. Fei-Fei. Spatial-temporal correlatons for unsupervised action classification. In *Proceedings of the IEEE Workshop on Motion and Video Computing, Copper Mountain, Colorado*, pages 1–8, 2008.

[87] S. Savarese, J. Winn, and A. Criminisi. Discriminative object class models of appearance and shape by correlatons. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2006), New York City, New York, USA*, pages 2033–2040, 2006.

[88] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: A local SVM approach. In *Proceedings of the 17th International Conference on Pattern Recognition (ICPR 2004), Cambridge, UK*, pages 32–36, 2004.

[89] J. Shi and C. Tomasi. Good features to track. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 1994), Seattle, Washington, USA*, pages 593–600, 1994.

[90] Q. Shi, L. Cheng, L. Wang, and A. Smola. Human action segmentation and recognition using discriminative semi-Markov models. *International Journal of Computer Vision*, 93(1):22–32, 2001.

[91] O. Shigeta, S. Kagami, and K. Hashimoto. Identifying a moving object with an accelerometer in a camera view. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS 2008), Nice, France*, pages 3872–3877, 2008.

[92] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, and M. Finocchio. Real-time human pose recognition in parts from single depth images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2011), Colorado Springs, Colorado, USA*, pages 1297–1304, 2011.

[93] Z. Si, M. Pei, B. Yao, and S.-C. Zhu. Unsupervised learning of event AND-OR grammar and semantics from video. In *Proceedings of the International Conference on Computer Vision (ICCV 2011), Barcelona, Spain*, pages 41–48, 2011.

[94] J. Snoek, H. Larochelle, and R. P. Adams. Practical Bayesian optimization of machine learning algorithms. In *Proceedings of the Conference on Neural Information Processing Systems (NIPS 2012), Lake Tahoe, Nevada, USA*, pages 1–9, 2012.

[95] S. Stein and S. J. McKenna. Accelerometer localization in the view of a stationary camera. In *Proceedings of the 9th Conference on Computer and Robot Vision (CRV 2012), Toronto, Ontario, Canada*, pages 109 – 116, 2012.

[96] S. Stein and S. J. McKenna. Combining embedded accelerometers with computer vision for recognizing food preparation activities. In *Proceedings of the ACM*

International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp 2013), Zürich, Switzerland, pages 729–738, 2013.

[97] S. Stein and S. J. McKenna. User-adaptive models for recognizing food preparation activities. In *Proceedings of the 21st ACM International Conference on Multimedia, 5th International Workshop on Multimedia for Cooking & Eating Activities (CEA'13), Barcelona, Spain*, pages 39–44. ACM, October 2013.

[98] T. Stiefmeier, D. Roggen, G. Tröster, G. Ogris, and P. Lukowicz. Wearable activity tracking in car manufacturing. *IEEE Pervasive Computing*, 7(2):42–50, 2008.

[99] J. Sun, X. Wu, S. Yan, L.-F. Cheong, T.-S. Chua, and J. Li. Hierarchical spatio-temporal context modeling for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2009), Miami Beach, Florida, USA*, pages 2004–2011, 2009.

[100] Q. Sun and H. Liu. Learning spatio-temporal co-occurrence correlograms for efficient human action recognition. In *Proceedings of the International Conference on Image Processing (ICIP 2013), Melbourne, Australia*, pages 3220–3224, 2013.

[101] Y. Tang and R. Rose. Rapid speaker adaptation using clustered maximum-likelihood linear basis with sparse training data. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(3):607–616, 2008.

[102] T. Teixeira, D. Jung, and A. Savvides. Tasking networked CCTV cameras and mobile phones to identify and localize multiple people. In *Proceedings of the 12th ACM International Conference on Ubiquitous Computing (UbiComp 2010)*, pages 213–222, 2010.

[103] M. Tenorth, J. Bandouch, and M. Beetz. The TUM Kitchen Data Set of Everyday Manipulation Activities for Motion Tracking and Action Recognition. In *IEEE International Workshop on Tracking Humans for the Evaluation of their Motion in*

Image Sequences (THEMIS). In conjunction with ICCV 2009, pages 1089–1096, 2009.

[104] M. Tenorth, F. de la Torre, and M. Beetz. Learning probability distributions over partially-ordered human everyday activities. In *Proceedings of the International Conference on Robotics and Automation (ICRA 2013), Karlsruhe, Germany*, pages 4539–4544, 2013.

[105] C. Tomasi and T. Kanade. Detection and tracking of point features. *Carnegie Mellon University Technical Report CMU-CS-91-132, April*, 1991.

[106] P. Turaga, R. Chellappa, V. S. Subrahmanian, and O. Udrea. Machine recognition of human activities: A survey. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(11):1473–1488, 2008.

[107] S. Vedula, P. Rander, R. Collins, and T. Kanade. Three-dimensional scene flow. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 27(3):475–480, 2005.

[108] C. J. Veenman, E. A. Hendriks, and M. J. T. Reinders. A fast and robust point tracking algorithm. In *Proceedings of the IEEE International Conference on Image Processing (ICIP 1998), Chicago, Illinois, USA*, volume 3, pages 653–657, 1998.

[109] S. Vishwakarma and A. Agrawal. A survey on activity recognition and behavior understanding in video surveillance. *The Visual Computer*, 29(10):983–1009, 2013.

[110] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu. Action recognition by dense trajectories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2011), Colorado Springs, Colorado, USA*, pages 3169–3176, 2011.

[111] H. Wang and C. Schmid. Action recognition with improved trajectories. In *Pro-*

ceedings of the International Conference on Computer Vision (ICCV 2013), Sydney, Australia, pages 3551–3558, 2013.

[112] H. Wang, M. M. Ullah, A. Kläser, Ivan Laptev, and C. Schmid. Evaluation of local spatio-temporal features for action recognition. In *Proceedings of the British Machine Vision Conference (BMVC 2009), London, UK*, pages 124.1–124.11, 2009.

[113] X. Wang, L. Wang, and Y. Qiao. A comparative study of encoding, pooling and normalization methods for action recognition. In *Proceedings of the Asian Conference on Computer Vision (ACCV 2012)*, pages 572–585, 2012.

[114] C. Wen-Sheng, F. de la Torre, and J. F. Cohn. Selective Transfer Machine for personalized facial action unit detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2013)*, pages 3515–3522, 2013.

[115] J. P. Wherton and A. F. Monk. Designing cognitive supports for dementia. *SIGACCESS Accessibility and Computing*, 86:28–31, 2006.

[116] J. P. Wherton and A. F. Monk. Problems people with dementia have with kitchen tasks: The challenge for pervasive computing. *Interacting with Computers*, 22(4):253–266, 2010.

[117] Anders Wimo and Martin Prince. *World Alzheimer Report 2010*, 2010. `http://www.alz.co.uk/research/worldreport/`.

[118] J. Wu, A. Osuntogun, T. Choudhury, M. Philipose, and J. M. Rehg. A scalable approach to activity recognition based on object use. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV 2007), Rio De Janeiro, Brazil*, pages 1–8, 2007.

[119] J. Yamato, J. Ohya, and K. Ishii. Recognizing human action in time-sequential images using hidden Markov model. In *Proceedings of the Conference on Computer*

Vision and Pattern Recognitino (CVPR 1992), Champaign, Illinois, USA, pages 379–385, 1992.

[120] Y. Yang, C. Fernmuller, and Y. Aloimonos. Detection of manipulation action consequences (MAC). In *Proceedings of the IEEE International Conference on Computer Vision andn Pattern Recognition (CVPR 2013), Portland, Oregon, USA*, pages 2563–2570, 2013.

[121] Y.-S. Yao and R. Chellappa. Dynamic feature point tracking in an image sequence. In *Proceedings of the IEEE International Conference on Pattern Recognition (ICPR 1994), Jerusalem, Israel*, volume 1, pages 654–657, 1994.

[122] J. Yuan, M. Yang, and Y. Wu. Mining discriminative co-occurrence patterns for visual recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2011), Colorado Springs, Colorado, USA*, pages 2777–2784, 2011.

[123] P. Zappi, C. Lombriser, T. Stiefmeier, E. Farella, D. Roggen, L. Benini, and G. Tröster. Activity recognition from on-body sensors: accuracy-power trade-off by dynamic sensor selection. In *Proceedings of the European Conference on Wireless Sensor Networks, Bologna, Italy*, pages 17–33, 2008.

[124] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: A comprehensive study. *International Journal of Computer Vision*, 73(2):213–238, 2007.

[125] L. Zhang, X. Zhen, and L. Shao. High order co-occurrence of visual words for action recognition. In *Proceedings of the International Conference on Image Processing (ICIP 2012), Orlando, Florida, USA*, pages 757–760, 2012.

[126] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.

[127] B. Zhou, X. Wang, and X. Tang. Understanding collective crowd behaviors: Learning a mixture model of dynamic pedestrian-agents. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2012), Providence, Rhode Island*, pages 2871–2878, 2012.