Recognition of user activity sequences using distributed event detection

Oliver Amft, Clemens Lombriser, Thomas Stiefmeier, and Gerhard Tröster

Wearable Computing Lab., ETH Zurich, Switzerland
{amft,lombriser,stiefmeier,troester}@ife.ee.ethz.ch,
http://www.wearable.ethz.ch

Abstract. We describe and evaluate a distributed architecture for the online recognition of user activity sequences. In a lower layer, simple heterogeneous atomic activities were recognised on multiple on-body and environmental sensor-detector nodes. The atomic activities were grouped in detection events, depending on the detector location. In a second layer, the recognition of composite activities was performed by an integrator. The approach minimises network communication by local activity aggregation at the detector nodes and transforms the temporal activity sequence into a spatial representation for simplified composite recognition. Metrics for a general description of the architecture are presented. We evaluated the architecture in a worker assembly scenario using 12 sensor-detector nodes. An overall recall and precision of 77% and 79% was achieved for 11 different composite activities. The architecture can be scaled in the number of sensor-detectors, activity events and sequences while being adequately quantified by the presented metrics.

Key words: event detection, activity recognition, distributed detectors, activity sensing, inertial sensors, wireless sensor networks, smart objects

1 Introduction

A key challenge for context-aware systems is to unobtrusively recognise complex activities from sensors, distributed in clothing and objects in the environment. Attempts to embed sensors into every-day objects has been proposed for smart environments as the PlaceLab [1]. By using such assistive systems, the user will be comforted with instant and relevant information and coaching support, e.g. advice, when certain task steps have been forgotten by the worker or tasks could be performed more efficiently.

The recognition of activities has been broadly investigated using fusion methods to combine different information sources at the data, feature, or classifier level. Choosing the right fusion approach requires an analysis of resource constraints within the recognition architecture. Wireless sensor networks impose particular limitations on computation and energy resources. Since wireless transmission is the highest power consumer on a sensor node, the required communication bandwidth should be kept as low as possible. However, concerning the processing power of a sensor node, it has been shown, e.g. for sound classification [2],

that sensor nodes can provide important contributions to the recognition. Consequently, for activity recognition in sensor networks, an early aggregation of the sensor data at the node is favourable to minimise bandwidth usage. This, in turn, restricts the fusion approach in the network to the (typically discrete) classifier level.

Fusing detector ensembles using spatially distributed sensing sources helps to assess large sets of activities and more complex activity sequences, demonstrated e.g. in [3]. Such an architecture requires an abstraction model to manage and integrate the nodes in the recognition process with respect to the provided information. Bobick [4] proposed a layered representation of user activities, consisting of movements, activities and interaction or sequences of activities. This hierarchy can serve as basis for a distributed recognition architecture. The lowest recognition layer lends itself ideally for a distributed implementation on sensor nodes, since the activities are aggregated, where the sensor data originates.

In this work we use a two-layered abstraction model based on the concept that activity sequences are composed of *heterogeneous activity events*. Activity events are seen as atomic units of activities, recognised at distributed sensor-detector nodes. The *atomic activities* represent an alphabet of low-level detection entities in the recognition system. A specific set of atomic activities yields a *composite activity*. The recognition of such composite activities is performed by fusing the sensor-based detections at an integrator.
Within this scope the paper makes the following contributions:

1. We present a recognition architecture that can be distributed in a network of wireless sensors. Each sensor node contributes to the activity recognition by detecting heterogeneous activity events in continuously sensed data. The distributed detection simplifies the recognition of composite activities at the integrator. Consequently, simple histogram- and symbol-based algorithms were used for the composite recognition. Section 2 details the deployed algorithms along with descriptive metrics for size and complexity of the architecture.
2. By using domain knowledge we demonstrate that atomic activities can be grouped to detector events. This approach permits the integration of useful information, supporting the recognition of composite activities.
3. We evaluate the architecture in a car assembly scenario using 12 sensor-detector nodes to recognise 11 different composite activities. Section 3 describes the scenario. Evaluation metrics, borrowed from information retrieval, are used to fully assess the recognition performance of the architecture.

1.1 Related work

Recent approaches to recognise interaction or sequences of activities followed the concept of a layered modelling, e.g. [5, 3, 6, 7] as it conveniently partitions the complexity of user activities. However, the solutions differ in the activity granularity of each layer depending on the application and recognition concept. Ryoo and Aggarwal [6] used three categories starting with "atomic actions" extracted from image sequences and modelled the activity sequences using a context-free

grammar. Kawanaka et al. [7] used a hierarchical architecture of interacting hidden Markov models (HMMs) to represent activities and sequences of activities. Oliver et al. [3] aimed to reduce the complexity of hierarchical HMMs by independent training of the layers. Moreover, Dynamic Bayesian networks (DBNs), being a very flexible framework for reasoning, have been applied to the recognition of human activities, e.g. [8].

Uniform HMM- and DBN-based reasoning solutions have some impractical properties for the deployment in distributed sensor networks: 1) a high computational complexity and 2) the need for a large training corpus combined with an extensive parameter search in order to tune the large amount of model parameters.

Distributed recognition in wireless sensor networks poses special requirements on classification algorithms. The main constraints are the limited processing power and the energy resources of wireless sensor nodes. The impact of these limitations have been investigated from different angles, mostly using binary classifiers computing maximum likelihood estimates. Research on distributed estimators includes the minimisation of messages needed for stable decisions [9], the effects of data quantisation and message losses [10] or coding to counteract data corruption during transmission [11]. Thiemjarus and Yang [12] present algorithms for the selection of features depending on sensor location and quality.

Detection of events and inference of composite activities in sensor networks have been addressed by middle-ware solutions like DSWare [13]. In DSWare, compound events can be signalled if a number of atomic events have been observed. A confidence value was assigned to events, indicating the decision confidence. Osmani et al. [14] created overlay networks called "Context Zones" of devices, which could contribute with events for certain activities. Detected events triggered a rule inference engine working on "Activity Maps". The activity maps model possible activities and the associated events.

The recognition approaches for wireless sensor networks described above, have the drawback that they either focus on the low-level classification of relatively simple sensor signals, or work on high-level abstracted events, for which it is not clear how they can be generated. This work aims at integrating low- and high-level activity recognition and investigating their interaction.

2 Distributed Activity Recognition Architecture

The activity recognition architecture is based on a set of networked sensor-detector nodes that perform sensing and recognition of activity events using the locally available sensor data. The detector nodes transfer every event to a central integrator node. The integrator combines the information from multiple detectors and their reported events to recognise composite activities. Figure 1 provides an overview on the recognition architecture. The operation of the detector node is detailed in Section 2.2, of the integrator node in Section 2.3.

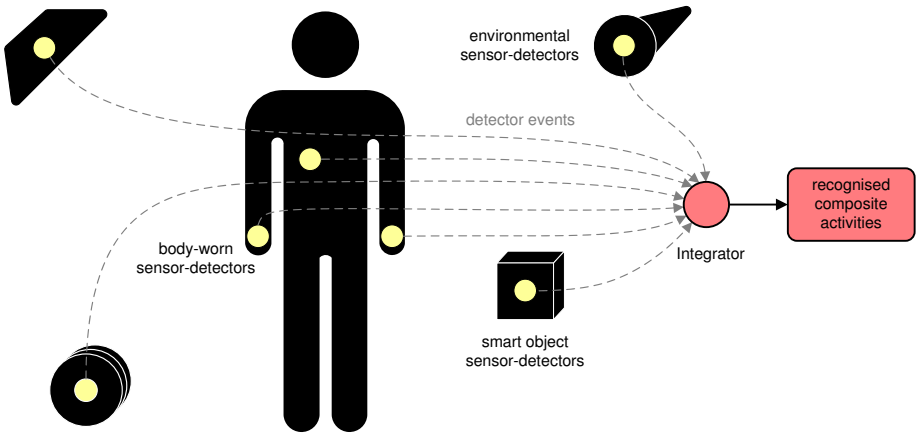

Fig. 1. Schematic of the distributed detection and classification of user activities.

2.1 Activity classification model

The architecture follows the concept of composite activities that can be described by a finite number of atomic activities. The total set of different atomic activities $\mathcal{A}$ describes the basic detection alphabet of the architecture (Eq. 1). The set of composite activities is $\mathcal{C}$. Each composite activity $\mathcal{C}^n$ is composed of unique atomic activities from $\mathcal{A}$.

$$\begin{aligned} &\mathcal{A} = \{a_1, \ldots, a_\alpha\}, \qquad \mathcal{C} = \{\mathcal{C}^1, \ldots, \mathcal{C}^\beta\}, \\ &\mathcal{C}^n \subseteq \mathcal{A}, \quad 1 \leq n \leq \beta \end{aligned} \tag{1}$$

The relations (Eq. 1) indicate some important metrics to describe the size and complexity of the recognition architecture. The *alphabet size* α counts the number of atomic activities used. The *composite class count* β measures the number of activity sequences that are covered by the recognition system.

The distributed detection is formed by the detector nodes and their detector events: a detector node $\mathcal{D}^i$ of the total set of detectors $\mathcal{D}$ contains at least one detector event $\mathcal{E}^i$ (Eq. 2). The *number of detector nodes* ($|\mathcal{D}|$) and the *total number of detector events* ($\sum_{i=1}^{|\mathcal{D}|} |\mathcal{D}_i|$) represent complexity metrics of the implemented architecture.

$$\mathcal{D}^i = \{\mathcal{E}^i_1, \ldots, \mathcal{E}^i_{\gamma_i}\}, \quad \forall i : \gamma_i \geq 1 \tag{2}$$

The primary application goals of the architecture were to use simple motion sensors and include a large activity alphabet. However, some atomic activities cannot be discriminated precisely by every detector. At the individual detector, this is observed as confusion between the atomic activities, indicating that the data patterns for the activities are not separable. Naively, the confused activities would be omitted from the detection for this node. In our approach, the affected activities are grouped to one event of the detector: for each detector D^i, atomic

activities a_j conflicting with each other, are grouped to a single detector event $\mathcal{E}^i_j$ (Eq. 3).

$$\mathcal{E}^i_j \subseteq \mathcal{A}, \quad where \quad \forall i : \mathcal{E}^i_j \cap \mathcal{E}^i_k = \emptyset, \quad for\ j \neq k \tag{3}$$

The combination of multiple, distributed event detectors at the integrator node is used to recognise composite activities, as described in Section 2.3. The event-based composite activity $\hat{\mathcal{C}^n}$ consists of a subset of events reported by different detectors $\mathcal{D}^i$, where the set is empty, if the detector does not contribute to the recognition (Eq. 4).

$$\hat{\mathcal{C}^n} = \bigcup_i \mathcal{D}^i_n, \qquad \forall i : \mathcal{D}^i_n \subseteq \mathcal{D}^i \tag{4}$$

The relationship of atomic and composite activities and the activity grouping at detectors is exemplarily visualised in Figure 2. In the simple example, composite activity $\mathcal{C}^1$ consists of atomic activities $a_1 \ldots a_3$ and corresponds to picking a tool (a_1), using it to manipulate an object (a_2) and returning it (a_3).

Each of the detectors $D^1 \ldots D^3$ uses at least one locally acquired sensor channel and recognises a subset of the atomic activities. In the presented example, acceleration signals from the objects and the user's wrist were used. The event sets for the detectors are: $D^1 = \{\mathcal{E}^1_1\}$, $D^2 = \{\mathcal{E}^2_1, \mathcal{E}^2_2, \mathcal{E}^2_3\}$, and $D^3 = \{\mathcal{E}^3_1\}$. While the events for D^1 and D^2 consist of one atomic activity each, for D^3 two activities are grouped $\mathcal{E}^3_1 = \{a_1, a_3\}$.

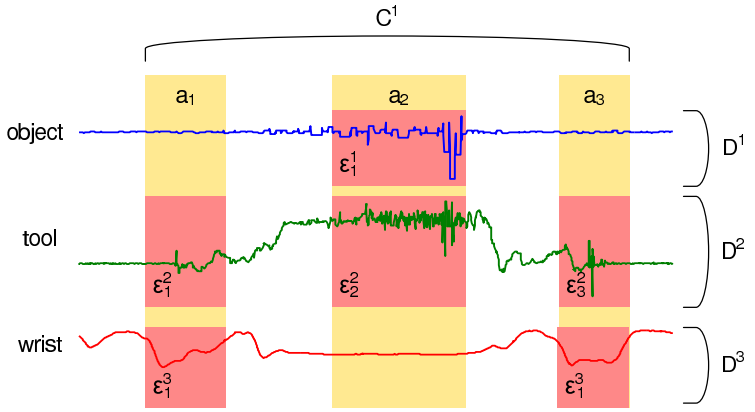

Fig. 2. Relationship of atomic and composite activities and the event grouping at detectors in an object manipulation task. The composite activity C^1 consist of three atomic activities: picking a tool (a_1) using it to manipulate an object (a_2) and returning it (a_3). Each of the detectors $D^1 \ldots D^3$ recognises a subset of the atomic activities, grouped in activity events. Acceleration signals from the users wrist and the involved objects are shown. Please see related text for details.

2.2 Detector node operation

The task of a detector node is to spot activity patterns in the continuous stream of data from the locally acquired sensor data. The detection step at the individual network node is introduced to minimise the transmission bandwidth requirements: instead of raw data or intermediate data compression, only the event detection result is transmitted.

The event recognition performance is a critical parameter for successful detection of composite activities. The most vital aspects for the recognition are: 1) the extraction of valid atomic activities from the embedding continuous data (spotting task) and 2) the disambiguation of different events spotted by the detector (classification task). Both must be achieved within the limited resources of the node. The first problem relates to the embedding of the sporadically occurring user activities into a large set of unknown activities, the NULL class. Due to the large variability of the unknown user activities, the NULL class cannot be modelled reliably. For the detector nodes, a feature similarity search algorithm, based on our previous work [15] was adapted to spot relevant activity events. For the disambiguation of different events, a comparison fusion of the similarity search result was used [16]. Both procedures are briefly summarised below.

Feature Similarity Search The search for potential events was performed by determining the similarity of a data section to a pre-computed pattern of an detector event. Similarity was compared using features computed for a data section when the last sample within a constant-sized window was received. The window size and the pre-computed event pattern were determined in a training step for each detector event $\mathcal{E}_j^i$. The window was moved at a step size of 0.25 s. Every detector $\mathcal{D}^i$ implemented its own feature set F^i.

For the similarity analysis between the feature vectors, the Euclidean distance was used. A threshold on the distance, obtained during training, was applied to omit unlikely sections. The distance of a reported event to the trained pattern was normalised using the distance threshold.

The advantage of this algorithm is that it works as a binary classifier to separate one detector event from the remaining data. Several instances of the feature similarity search were used to spot different events independently.

The computational complexity of the feature similarity search scales with the frequently used sliding window approach, e.g. for sound [2]. Using the detector event notation, the complexity of the sliding window is $O(l(\mathcal{E}_j^i))$, where $\mathcal{E}_j^i$ is the considered event and $l(\mathcal{E}_j^i)$ describes the length of the event in processing units, e.g. data samples. The mean complexity of a detector $\mathcal{D}^i$ is shown in Eq. 5. The complexity is determined by the mean length of all events and the size of the feature set F^i computed for the detector. This result is considered as upper bound, depending on the efficiency of the feature implementation.

$$O\left(\frac{1}{|\mathcal{D}^i|}\sum_{j=1}^{|\mathcal{D}^i|} l(\mathcal{E}_j^i)\cdot|F^i|\right) \tag{5}$$

The independent search instances derive events that can overlap in the time-domain. A filtering was applied to resolve these conflicts. The filter retained events according to the minimal normalised distance among all overlapping events in a buffer. These events were released from the buffer after a timeout.

2.3 Integrator node operation

Similar challenges regarding the detection and classification of composite activities at the integrator node exist, compared to the detector nodes: 1) spotting of relevant activity sequences in continuous data and 2) classifying these sequences. Therefore an analog approach to the detector node operation was taken. However, the analysis at the integrator node is performed using activity events, requiring methods adapted for the type of input. An advantage, compared to the detector node, is that the integrator algorithms run at a massively reduced input rate.

Two different distance algorithms were investigated for the recognition, using the combined input stream from all detectors: 1) an approximate string matching by converting the detector events to characters and comparison to a template string of the activity and 2) a comparison of the event occurrence frequencies to a histogram of the activity. Both approaches require training observations from each composite activity, as detailed below.

Distances were computed between a search window of previously received events and the trained pattern at each new event in the input stream. For the composite activity detection, a variable-sized search window was used. By using a distance threshold, unlikely sections were omitted. Both, the search bounds and the threshold were determined from training instances.

As string matching approach, the *edit distance* [17] was used. The strings were generated by mapping each event type to a unique character. The edit distance returned the minimum number of edit operations (insertions, deletions, substitutions) required, to convert the string inside the search window into the training template of each composite activity. The template itself was found by computing the minimal edit distance among all training instances. This corresponds to the minimum linkage distance of the training instances. The complexity of this algorithm is $O(|\mathcal{C}^n|)$ on continuous character input, where $|.|$ is the number of events in each composite activity $\mathcal{C}^n$.

For the event frequencies approach, an *event histogram* was build for each composite activity. The histogram was initialised with the annotated event occurrences for each composite activity. The histogram was adapted by the event occurrences observed in training instances. A distance between this training histogram and the histogram determined from the search window was computed by summing the absolute differences in each event bin.

If the histograms were identical, the distance is zero. This would indicate that the events would ideally resemble the training distribution. Contrary, a larger distance indicates a different shape of the histogram. For this algorithm, the complexity for testing a composite activity is $O(\sum_{i=1}^{|\mathcal{D}|} |\mathcal{D}^i|)$. The complexity is governed by the total number of different events used in the system.

The computational complexity at the integrator node is determined by the detection algorithm and the window procedure for each composite activity. Eq. 6 shows derived bounds with ordered factors. For simplicity of the notation, the adaptive window size was replaced with the average size of the composite activities, $\frac{1}{\beta}\sum_{n=1}^{\beta}|\mathcal{C}^n|$.

$$Edit : O\left(\frac{1}{\beta}\sum_{n=1}^{\beta}|\mathcal{C}^n|\cdot\sum_{n=1}^{\beta}|\mathcal{C}^n|\right), \qquad Hist : O\left(\sum_{i=1}^{|\mathcal{D}|}|\mathcal{D}_i|\cdot\sum_{i=1}^{\beta}|\mathcal{C}^n|\right) \tag{6}$$

The first factor in each equation indicates the difference between the two algorithms: while the first factor for the edit distance scales with the number of events included in each set $\mathcal{C}^n$, the factor for the event histogram method scales with the total number of different events used in the system.

3 Evaluation

For the evaluation of the proposed architecture, we selected the recognition of car assembly activities. In order to measure the performance of the event detectors and the integrator recognition, detection metrics, commonly used in information retrieval, were introduced. Finally, the feasibility of the architecture was assessed with the metrics.

3.1 Car assembly scenario

A car body, installed in a lab environment, was used to record assembly and testing activities, as they have been observed and annotated beforehand in a car mass production facility. Sensors were worn by the worker and attached to different tools and parts of the car. In total, 12 sensors were used to acquire 3D-acceleration from 47 atomic activities in 11 composite activities. Figure 3 shows a worker during a door attachment activity. The activities are summarised in Tab. 1.

Nine wireless sensor motes were used to record motion of different car parts included in the activities (front light, braking light, front and back doors, the hood and trunk door) as well as tools used for the assembly work (two cordless automatic screwdrivers and a socket wrench, see Figure 3). Three wired sensors were attached to a jacket at the wrist position of both lower arms and the upper back. The body-worn sensors and the tool sensors (screwdrivers and socket wrench) were recorded at 50 Hz, the remaining sensors were sampled at 20 Hz. Two workers each completed 10 repetitions of all composite activities wearing the jacket. A manual annotation of the activities was performed.

A total of 49 different detector events were derived from the atomic activities for all sensor-detector nodes. Tab. 2 presents the mapping along with the sensor locations for reference.

Besides the relevant composite activities, the users performed five additional activities three times during each repetition to enrich the data set diversity,

Fig. 3. Left: worker mounting the front door of the car. Right: socket wrench equipped with acceleration sensing wireless nodes.

Composite activity	Atomic Activities	Description
Mount front door ($\mathcal{C}^1$)	pickup door (a_1), attach door (a_2), fix screws by hand (a_3), pickup socket wrench (a_4), use socket wrench (a_5), return socket wrench (a_6), close door (a_7)	Mount front door of the car and fix screws with a socket wrench.
Mount back door ($\mathcal{C}^2$)	pickup door (a_8), attach door (a_9), fix screws by hand (a_{10}), pickup socket wrench (a_{11}), use socket wrench (a_{12}), return socket wrench (a_{13}), close door (a_{14})	Mount back door of the car and fix screws with a socket wrench.
Test front door ($\mathcal{C}^3$)	open (a_{15}), teeter (a_{16}), close (a_{17})	Test front door hinges.
Test back door ($\mathcal{C}^4$)	open (a_{18}), teeter (a_{19}), close (a_{20})	Test back door hinges.
Test trunk ($\mathcal{C}^5$)	open (a_{21}), teeter (a_{22}), close (a_{23})	Test trunk door.
Mount brake light ($\mathcal{C}^6$)	fetch light (a_{24}), insert light (a_{25}), screw brake light (a_{26})	Install middle brake light at the trunk door.
Test hood ($\mathcal{C}^7$)	open (a_{27}), teeter (a_{28}), close (a_{29})	Test the hinges of the hood.
Mount hood rod ($\mathcal{C}^8$)	fetch rod (a_{30}), open hood (a_{31}), install rod (a_{32})	Open the hood and install a supporting rod to keep it open.
Mount water tank ($\mathcal{C}^9$)	fetch tank (a_{33}), hand screw tank (a_{34}), pickup driver (a_{35}), screw (a_{36}), return driver (a_{37})	Install a water tank in the hood room using screwdriver 1.
Mount bar ($\mathcal{C}^{10}$)	fetch bar (a_{38}), hand screw bar (a_{39}), pickup driver (a_{40}), screw (a_{41}), return driver (a_{42})	Mount a bar under the hood using screwdriver 2.
Mount light ($\mathcal{C}^{11}$)	fetch light (a_{43}), install light (a_{44}), pickup screwdriver (a_{45}), fix screw (a_{46}), return screwdriver (a_{47})	Install left front light using screwdriver 2.

Table 1. Composite activities $\mathcal{C}^n$ and the associated atomic activities.

including writing on a notepad, working at a computer, drinking, tieing shoes and scratching the head. The activities were not further considered in the evaluation.

Sensor location	Detector events
Right lower arm ($\mathcal{D}^1$)	$\mathcal{E}_1^1 = \{a_{27}, a_{21}\}$, $\mathcal{E}_2^1 = \{a_{28}, a_{22}\}$, $\mathcal{E}_3^1 = \{a_{29}, a_{23}\}$, $\mathcal{E}_4^1 = \{a_{30}\}$, $\mathcal{E}_5^1 = \{a_{32}\}$, $\mathcal{E}_6^1 = \{a_{34}, a_{39}, a_3, a_{10}\}$, $\mathcal{E}_7^1 = \{a_{35}, a_{40}, a_{45}, a_{37}, a_{42}, a_{47}, a_4, a_6, a_{11}, a_{13}\}$, $\mathcal{E}_8^1 = \{a_{43}\}$, $\mathcal{E}_9^1 = \{a_1, a_8\}$, $\mathcal{E}_{10}^1 = \{a_5, a_{12}\}$
Left lower arm ($\mathcal{D}^2$)	$\mathcal{E}_1^2 = \{a_{27}, a_{31}, a_{21}\}$, $\mathcal{E}_2^2 = \{a_{28}, a_{22}\}$, $\mathcal{E}_3^2 = \{a_{29}, a_{23}\}$, $\mathcal{E}_4^2 = \{a_{32}\}$, $\mathcal{E}_5^2 = \{a_1, a_8\}$, $\mathcal{E}_6^2 = \{a_7, a_{14}, a_{17}, a_{20}\}$, $\mathcal{E}_7^2 = \{a_{15}, a_{18}\}$, $\mathcal{E}_8^2 = \{a_{16}, a_{19}\}$, $\mathcal{E}_9^2 = \{a_{25}\}$
Upper back ($\mathcal{D}^3$)	$\mathcal{E}_1^3 = \{a_{34}, a_{36}\}$, $\mathcal{E}_2^3 = \{a_{35}, a_{37}, a_{40}, a_{42}, a_{45}, a_{47}, a_1, a_4, a_6, a_8, a_{11}, a_{13}\}$, $\mathcal{E}_3^3 = \{a_{46}\}$
Front light ($\mathcal{D}^4$)	$\mathcal{E}_1^4 = \{a_{43}\}$, $\mathcal{E}_2^4 = \{a_{46}\}$
Brake light ($\mathcal{D}^5$)	$\mathcal{E}_1^5 = \{a_{24}\}$, $\mathcal{E}_2^5 = \{a_{25}\}$
Hood ($\mathcal{D}^6$)	$\mathcal{E}_1^6 = \{a_{27}, a_{31}\}$, $\mathcal{E}_2^6 = \{a_{28}\}$, $\mathcal{E}_3^6 = \{a_{29}\}$
Trunk ($\mathcal{D}^7$)	$\mathcal{E}_1^7 = \{a_{21}\}$, $\mathcal{E}_2^7 = \{a_{22}\}$, $\mathcal{E}_3^7 = \{a_{23}\}$
Screwdriver 1 ($\mathcal{D}^8$)	$\mathcal{E}_1^8 = \{a_{45}\}$, $\mathcal{E}_2^8 = \{a_{46}\}$, $\mathcal{E}_3^8 = \{a_{47}\}$
Screwdriver 2 ($\mathcal{D}^9$)	$\mathcal{E}_1^9 = \{a_{35}, a_{40}\}$, $\mathcal{E}_2^9 = \{a_{36}, a_{41}\}$, $\mathcal{E}_3^9 = \{a_{37}, a_{42}\}$
Front door ($\mathcal{D}^{10}$)	$\mathcal{E}_1^{10} = \{a_1\}$, $\mathcal{E}_2^{10} = \{a_2\}$, $\mathcal{E}_3^{10} = \{a_{15}\}$, $\mathcal{E}_4^{10} = \{a_{16}\}$
Back door ($\mathcal{D}^{11}$)	$\mathcal{E}_1^{11} = \{a_8\}$, $\mathcal{E}_2^{11} = \{a_9\}$, $\mathcal{E}_3^{11} = \{a_{18}\}$, $\mathcal{E}_4^{11} = \{a_{19}\}$
Socket wrench ($\mathcal{D}^{12}$)	$\mathcal{E}_1^{12} = \{a_4, a_{11}\}$, $\mathcal{E}_2^{12} = \{a_5, a_{12}\}$, $\mathcal{E}_3^{12} = \{a_6, a_{13}\}$

Table 2. Location of the sensor-detectors $\mathcal{D}^i$ and the 49 corresponding detector events $\mathcal{E}_j^i$. The detector events were derived by grouping atomic activities for each sensor-detector node.

In total, 4.8 hours of data were recorded, containing 2.3 hours of relevant composite activities. The data from sensors mounted on the car parts and tools were transmitted wirelessly to a root sensor node. No special synchronisation procedure was applied, since the delay of the data was found to be negligible for our purposes. In the recording sessions, an average message loss of 6.7% was observed. For some sessions the loss rate was 16.4%.

3.2 Analysis procedure

Simple time-domain features were derived from the three acceleration signals available and optimised for each detector, such as signal sum, mean, sum of differences, maximum and minimum. The features were computed for the entire signal section of each activity and partitions of it. By using partitions of the signal section, the temporal structure within an atomic activity was captured.

A four-fold cross-validation was applied on the data set to select training and validation set for the event detection and composite activity evaluation. For this purpose the data set was partitioned into four sections. For each cross-validation iteration three data sections were used for training and one for validation. In this way, each section was used once for testing.

The detection performance was analysed with the metrics *Precision* and *Recall*, commonly used in information retrieval (Eq. 7). *Relevant activities* corre-

spond to the actually conducted and manually annotated activities, *retrieved activities* are all objects returned by the algorithms and *recognised activities* represent the correctly returned activities.

$$Recall = \frac{Recognised\ activities}{Relevant\ activities}, \qquad Precision = \frac{Recognised\ activities}{Retrieved\ activities} \tag{7}$$

To identify an activity as correctly returned by the algorithms, time-domain overlap check between the annotation and returned event was used.

3.3 Detector node performance

The recognition performance was analysed for all 12 detectors and its respective detector events. Figure 4 presents the overall performance of each detector. The best results were achieved for the detector $\mathcal{D}^7$ attached to the trunk. Figure 5 exemplarily shows the performance of the individual events for the socket wrench detector ($\mathcal{D}^{12}$).

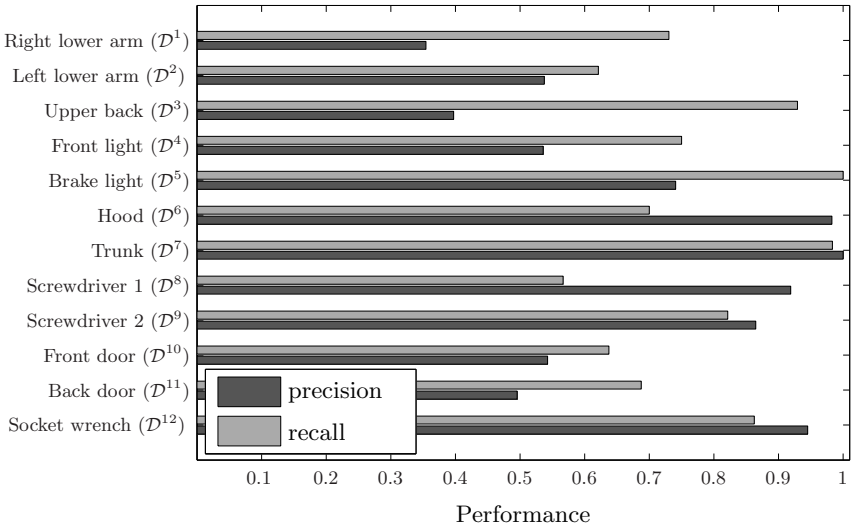

Fig. 4. Overall recognition performance of the 12 detectors. (Best performance is found towards high precision and high recall.)

Although the right and left arm detectors ($\mathcal{D}^1$ and $\mathcal{D}^2$) contained many event types (ten and nine respectively), a meaningful discrimination was achieved for both. The front and brake light detection performance ($\mathcal{D}^4$, $\mathcal{D}^5$) suffered from highly variable movements during the mounting activity. For the upper back detector ($\mathcal{D}^3$), often event durations did not match with the annotation of the instances. The annotation was not optimised for events from the upper back

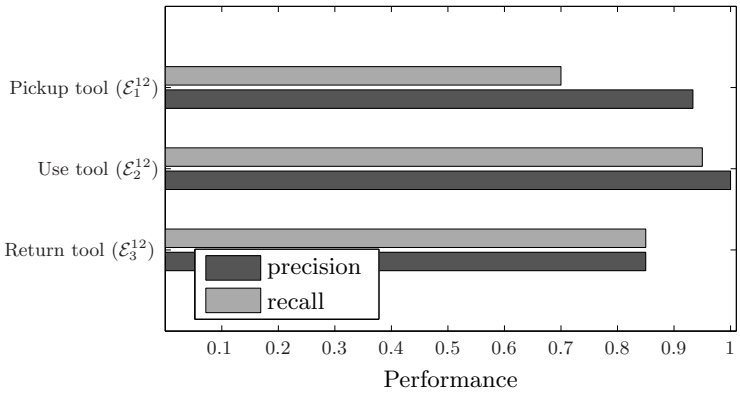

Fig. 5. Detector event recognition performance. This example shows the socket wrench detector ($\mathcal{D}^{11}$).

detector. The weak detection at the front and back door were influenced by low amplitude of the acceleration signals during the door opening and closing activities. Finally, the data loss of all wireless sensors ($\mathcal{D}^3$-$\mathcal{D}^{12}$) observed during the recordings degraded the performance results for these detectors.

3.4 Integrator node performance

Figure 6 shows an example histogram plot of the detector events for the composite activity $\mathcal{C}^9$ ("Mount water tank"). While the upper histogram presents the event frequencies according to the manual annotation, the lower plot shows the event frequencies returned by the detectors. Deviations between the two charts indicate insertion or deletion errors of the detectors. The bar at $\mathcal{E}_1^4$ represent event insertions, since the event corresponding to "Fetch front light" are not included in the "Mount water tank" composite. (In the experiment it happened, that the water tank and the front light were placed on the same carriage waggon. When the tank was fetched, the waggon may have moved slightly, which resulted in the fetching activity insertion.)

For the edit distance algorithm an overall recall and precision of 0.77 and 0.26 was achieved. For the event histogram algorithm the results are 0.77 and 0.79. The event histogram based approach clearly outperformed the edit distance in precision while both algorithms return a similar number of correctly recognised activities. The strict event sequence requirement imposed by the edit distance, was often disturbed by the independent and unordered event reports of the detectors. In contrast, supported by the adaptive training, the histogram algorithm was able to handle detector errors, such as insertions better.

Figure 7 presents the final recognition result for the individual composite activities. For the edit distance, a large variation in precision was observed. This indicated a low performance for several of the composite activity classes, e.g. $\mathcal{C}^{11}$ while a few other, e.g. $\mathcal{C}^6$ ("Mount brake light") achieved a high performance. We

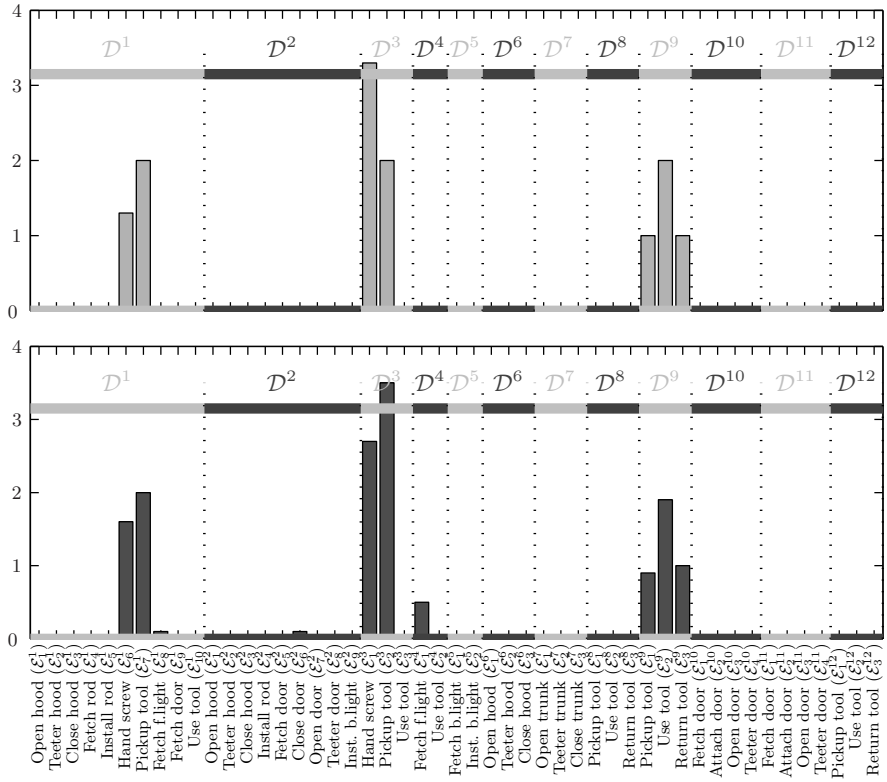

Fig. 6. Histogram plot of the detector events for the composite activity $\mathcal{C}^9$ ("Mount water tank"). Upper plot: event frequencies according to the manual annotation. Lower plot: event frequencies returned by the detectors.

attributed this result to the different training and event sequence requirements of the algorithms, as discussed before.

4 Discussion and conclusion

In this work, an online architecture for the distributed recognition of complex activity sequences was introduced. The architecture was chosen to reflect a two-layer abstraction of user activities. In the first layer, a distributed event aggregation from atomic activities was performed. In the second layer, activity sequences were recognised from the events. Our work was driven by the observation, that distributed detectors, at the user's body or the surrounding environment, sense activities in a location-dependent granularity. As a consequence, neither each detector can spot every activity in continuous sensor data nor uniquely discriminate all activities.

As a solution, an intermediate grouping of atomic activities to detector-specific events was used. With this approach, the information from each detec-

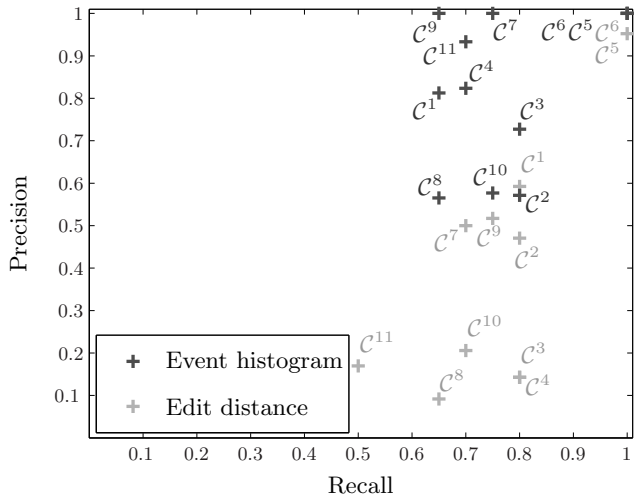

Fig. 7. Recognition performance of the composite activities in the car assembly scenario. Result for two event fusion methods are shown: edit distance and event histograms.

tor was adequately included in the second layer recognition, instead of naively omitting less-specific detectors. By performing the grouping manually, available domain knowledge on the relation of activity events was integrated (as detailed in the example of Section 2.1, Figure 2). Moreover, the grouping simplified the detector nodes, since less events had to be discriminated. Further work will aim at automating the manual grouping of atomic activities, while still incorporating the domain knowledge.

Moreover, the grouping allowed the transformation of temporal activity sequences into a spatial recognition model. The purely spatial recognition algorithm based on event histograms, showed even better performance for the composite activity detection than the strictly sequential edit distance. This result shows, that the distributed solution strongly simplified the complex task of activity sequence recognition.

Data loss due to wireless transmission of the sensor nodes using a standard MAC layer peaked at more than 16% in the car assembly scenario. This data loss influenced the achieved recognition performance in this work. By performing the recognition at the node directly, sensor data loss is avoided.

In the present work, a central integration and recognition of composite activities was used. However, the recognition based on the histogram algorithm could be solved by a distributed implementation of wireless sensor networks. This solution would increase redundancy in the recognition architecture.

The complexity of the detection step scales with the length of the event and the number of features used. The computational requirements are further decreased by sharing intermediate results during feature processing. The complexity of the single sliding window algorithm is a lower bound for this optimisation.

Notably, the sliding window approach was even used on sensor nodes at data rates of more than 4 kHz, for spectral audio data processing [2]. Given these relations, we are confident that the distributed architecture is feasible for an online implementation on low-power sensor nodes. Similarly, we believe that the simple histogram algorithm used for the event-processing at the integrator could be processed on a wireless sensor node.

The influence of recognition errors by detector nodes is often underestimated. In some works, different error types, e.g. insertion errors, were not considered at all [14, 13]. The results of the architecture evaluation showed that these errors cannot be avoided for activity recognition. Hence, theoretic or simulation-based investigations should consider them more carefully.

For detector nodes that communicate recognition results to the network, the detection metric *precision* indicates the node's bandwidth requirements. A low precision corresponds to a high number of insertion errors and increases the number of required event transmissions above the relevant events.

The car assembly scenario was selected based on the need for novel worker support systems. Such assistive systems track the task steps and report, whether tasks have been executed precisely enough. A clear emphasis of the work was to implement systems, that help the user in increasing task performance, e.g. if the tightening of screws for a car part was not detected, the worker could be asked to verify that the part was attached correctly. Moreover, using this system, the worker could train tasks at an individual pace or refresh skills when working with different car models. All of these applications require monitoring of activities, that involve multiple connected task steps.

We introduced the detection architecture along with descriptive metrics for a general quantification of its size and complexity. For the presented car assembly tasks, the architecture was composed of 11 composite classes, an alphabet size of 47 atomic activities and 49 detector events originating from 12 sensor-detector nodes. Similar scenarios can be found in other application areas, e.g. in monitoring of daily activities for health care and behaviour analysis in smart homes [1]. The presented approach could be applied to these applications as well. The metrics provide a convenient way to assess size and complexity of the distributed recognition architecture.

Acknowledgements The work was supported by the Swiss State Secretariat for Education and Research (SER) as well as the 6th European Framework Programme Integrated Projects e-SENSE, http://www.ist-e-SENSE.org, contract number 027227 and WearIT@Work, http://www.wearitatwork.com, contract number 004216.

References

1. Intille, S.S., Larson, K., Tapia, E.M., Beaudin, J.S., Kaushik, P., Nawyn, J., Rockinson, R.: Using a live-in laboratory for ubiquitous computing research. In: Pervasive 2006: Proceedings of the 4th International Conference on Pervasive Computing. (2006) 349–365

2. Stäger, M., Lukowicz, P., Tröster, G.: Implementation and evaluation of a low-power sound-based user activity recognition system. In: ISWC 2004: Proceedings Eigth International Symposium on Wearable Computers. (2004)
3. Oliver, N., Garg, A., Horvitz, E.: Layered representations for learning and inferring office activity from multiple sensory channels. Comput Vis Image Und **96**(2) (2004) 163–180 Special issue on event detection in video.
4. Bobick, A.: Movement, activity, and action: The role of knowledge in the perception of motion. Philos T Roy Soc B **352**(1358) (1997) 1257–1265
5. Bao, L., Intille, S.S.: Activity recognition from user-annotated acceleration data. In: Pervasive 2004: Proceedings of the International Conference on Pervasive Computing. Volume 3001 of Lecture Notes in Computer Science., Springer Berlin / Heidelberg (2004) 1–17
6. Ryoo, M., Aggarwal, J.: Recognition of composite human activities through context-free grammar based representation. In: CVPR 2006: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Volume 2. (2006) 1709–1718
7. Kawanaka, D., Okatani, T., Deguchi, K.: Hhmm based recognition of human activity. IEICE T Inf Sys **E89-D**(7) (2006) 2180–2185
8. Du, Y., Chen, F., Xu, W., Li, Y.: Recognizing interaction activities using dynamic bayesian network. In: ICPR 2006: Proceedings of the 18th International Conference on Pattern Recognition. Volume 1. (2006) 618–621
9. Predd, J.B., Kulkarni, S.R., Poor, H.V.: Distributed learning in wireless sensor networks. IEEE Signal Proc Mag **23**(4) (2006) 56–69
10. Saligrama, V., Alanyali, M., Savas, O.: Distributed detection in sensor networks with packet losses and finite capacity links. IEEE T Signal Proces **54**(11) (2006) 4118–4132
11. Wang, T.Y., S Han, Y., Varshney, P.K., Chen, P.N.: Distributed fault-tolerant classification in wireless sensor networks. IEEE J Sel Areas Comm **23**(4) (2005) 724–734
12. Thiemjarus, S., Yang, G.Z.: An automatic sensing framework for body sensor networks. In: BodyNets 2007: Proceedings of the Second International Conference on Body Area Networks. (2007)
13. Li, S., Lin, Y., Son, S.H., Stankovic, J.A., Wei, Y.: Event detection services using data service middleware in distributed sensor networks. Telecommun Syst **26**(2) (2004) 351–368
14. Osmani, V., Balasubramaniam, S., Botvich, D.: Self-organising object networks using context zones for distributed activity recognition. In: BodyNets 2007: Proceedings of the Second International Conference on Body Area Networks. (2007)
15. Amft, O., Junker, H., Tröster, G.: Detection of eating and drinking arm gestures using inertial body-worn sensors. In: ISWC 2005: IEEE Proceedings of the Ninth International Symposium on Wearable Computers. (2005) 160–163
16. Bannach, D., Amft, O., Kunze, K.S., Heinz, E.A., Tröster, G., Lukowicz, P.: Waving real hand gestures recorded by wearable motion sensors to a virtual car and driver in a mixed-reality parking game. In: CIG 2007: Proceedings of the IEEE Symposium on Computational Intelligence and Games. (2007) 32–39
17. Navarro, G., Raffinot, M.: Flexible Pattern Matching in Strings. Cambridge University Press, New York, NY, USA (2002)