QuActive: A Quality of Activities Monitoring and Notification System

Ifat Afrin Emi, Md. Abu Sayeed Mondol, and John A. Stankovic
iae4qb@virginia.edu,mondol@virginia.edu,stankovic@virginia.edu
Department of Computer Science
University of Virginia
85 Engineer's Way
P.O. Box 400740
Charlottesville, Virginia 22904-4740

ABSTRACT

In order to notify users about potentially unsafe situations and to track mistakes or efficiency performing activities, it is important to monitor the quality of performing an activity and identify the missing/wrong steps. However, the state-of-the-art activity recognition frameworks ignore such details and impose constraints on sensor values, the types of detected activities (no parallel/interleaved/joint activities), or the number of users, which reduce the robustness of the system in the real world settings. Therefore, we present *QuActive*, a grammar based general purpose framework for modeling activities and micro-activities that retains the details of the activity steps, quantifies activity quality, and notifies users about missing steps and unsafe situations. In order to show the versatility of *QuActive*, we evaluate the framework on three different public datasets that have interleaved activities, parallel and co-operative activities, and activities of cognitively declined patients with quality information labeled. In all cases, *QuActive* outperforms the state-of-the-art techniques applied on these data sets. In addition, we have deployed the system in a real home and collected data in a semi-controlled setting to evaluate the performance of the system in real settings. The results show that *QuActive* recognizes more than 90% of the defined micro-activities and the grammar detects almost all the defined activities from the recognized micro-activities.

CCS CONCEPTS

•**Human-centered computing** →*Activity centered design ;* •**Applied computing** →**Health care information systems ;** •**Computer systems organization** →*Sensor networks ;* •**Theory of computation** →*Grammars and context-free languages;*

KEYWORDS

Activity Quality, Micro-Activity Modeling, Context Free Grammar, Notification System, Wireless Sensor Network

ICCPS, Pittsburgh, PA USA

DOI: 10.1145/3055004.3055005

ACM Reference format:
Ifat Afrin Emi, Md. Abu Sayeed Mondol, and John A. Stankovic. 2017. QuActive: A Quality of Activities Monitoring and Notification System. In *Proceedings of The 8th ACM/IEEE International Conference on Cyber-Physical Systems, Pittsburgh, PA USA, April 2017 (ICCPS),* 11 pages.
DOI: 10.1145/3055004.3055005

1 INTRODUCTION

In today's smart world, wearable and in-situ sensors are being used to monitor humans and recognize many types of activities. In most cases, the resulting information is not acted upon in any direct or real-time manner. However, by more intimately bringing the human into a feedback loop, there is great potential to use interventions and notifications to improve human activities. For example, by focusing on the micro steps of an activity, it is possible to detect the quality of an activity and dynamically react to improve that activity, if necessary. This human-in-the-loop real-time reaction is important in home health care systems to keep patients safe, in industry process monitoring systems of factory workers to ensure the safety of workers and the quality of products, and so on. Without considering the micro steps of activities, controlling the quality of activities is difficult.

Most current activity recognition systems recognize whether an activity has occurred or not, but do not identify partially completed activities or the missing steps in the overall activity process. Hence, they cannot easily offer notifications and interventions in real-time to improve performance. In addition, many current systems [12, 17] make too many simplifying assumptions about the environment, the number of users etc. that either limit the types of recognized activities, or tailor the system to perform well in simple situations such as single person homes and no concurrent activities [16, 25]. It is necessary to consider interleaved, parallel, and co-operative activities for more robust and realistic activity recognition.

Our first hypothesis is that pushing the activity recognition constraints to a lower (micro) level solves the limitations of many existing systems. In this paper, we consider the fact that activities are composed of micro-activities (μAcs) where a user can perform only one micro-activity at a time, but can switch in between performing micro-activities of a particular activity and do portions of other activities. For example, someone can chop ingredients for a meal, feed the dog, and then come back to use the ingredients for cooking. The μAcs provide important information about the high-level activities, such as whether the activity is complete or partially complete, the different ways a certain activity is performed, whether the activity has missing steps, how a missing step affects the overall activity quality, and how intermediate delays among μAcs might influence the overall activity quality. Therefore, this paper addresses the problem of detecting micro-activities (μAcs) in realistic settings and how to incorporate the human-in-the-loop by offering real-time notifications to improve performance. Since a step of an activity involves using objects resulting from specific gestures by a person, *QuActive* incorporates information from both wearable and in-situ sensors.

One challenge is how to model the activity process in terms of μAcs.The μAcs within an activity can occur in parallel or sequentially. The activity process also varies depending on person, environment, or situation. Different activities often have similar μAcs, and μAcs performed in a different order might result in the same or a different activity. Thus, the process of mapping μAcs to distinct activities capable of handling these variations is vital. Another challenge is addressing the deviation from usual activity processes. For example, if a certain step is missing or performed out of order, then is the activity incomplete, wrongly performed, or still a valid activity performed in a different way? How to keep a general structure of a particular activity which is performed in different ways? How to identify the prospective/incomplete activity when one or more μAcs are missing? Finally, how and when to bring the user more intimately into the loop via notifications and interventions?

To address the mentioned challenges, the *QuActive* framework is created based on a Temporal Probabilistic Context Free Grammar (TPCFG) to define the activity process (details in 4.2 and 4.3). The context free grammar (CFG) follows the basic definition from literature [8] that includes terminals, nonterminals, and rules. However, the terms are tailored for defining particular activities and μAcs. The grammar outlines a general structure for each activity. Activities (nonterminals) and μAcs (nonterminals) are generated from rules. Rules are applied iteratively until terminal symbols (sensor values) are reached. Any future activity instance is recognized from the defined grammar representation. Again, to capture the variation of performing the same activity, multiple rules are added to represent the same nonterminal term. If an activity is performed in several ways, then a probability (P) is associated with each of the rules defining the same nonterminal. The timing parameter (T) is used to capture the time information [21] of each μAc as well as the time difference among two consecutive μAcs. Rules have notifications attached to them.

The main contributions of this work are:

(1) This paper presents *QuActive*, a novel mirco-activity modeling framework that utilizes fine grained information of the activity process and uses that for notifications. *QuActive* is capable of monitoring activity quality and reporting prospective activity in case of missing steps and other realities in contrast to other state-of-the-art detection systems [5, 18].
(2) We implemented a system that incorporates a *QuActive* framework to recognize activity, monitor quality, and notify users. The notification subsystem modifies the latest voice based medication reminder system, Med-Rem [14], to an activity reminder system that provides audio alerts about activities, informs user about missing steps, and stores user feedback.
(3) The *QuActive* framework is applied to three different public datasets of interleaved activities, parallel and co-operative activities, and monitoring cognitive decline (missing steps and activity quality). *QuActive* outperforms the state-of-the-art techniques for all of these datasets.
(4) The system has also been deployed in a real home in a semi-controlled setting. The results show that *QuActive* recognizes more than 90% of the defined μAcs and the grammar detects 98.6% of the defined activities from the recognized μAcs.

2 USAGE SCENARIOS

QuActive provides information about activity quality, such as duration or speed of activity, missing steps, and time taken between steps. Therefore, humans can be more intimately incorporated into real-time intervention loops. To illustrate the value of such a system we describe some of the projected applications below:

- **Dementia and Alzheimer patients:** One of the most important applications of monitoring activity quality is early detection of dementia to prevent the rapid decline of functional and cognitive ability. In the United States, the annual cost of caring for individuals with dementia is $600 billion. Literature [6] shows that one of the early symptoms (functional decline) of dementia is evident in activities of daily living (ADL). For example, functional decline affects the speed of performing an activity or a particular step. Also, the patients often miss a step of an overall activity or forget to finish the work. Since the functional decline of dementia patients often correlates to the cognitive decline, monitoring one provides information about the other. Thus, *QuActive* can be used for the early detection of dementia and for providing notifications to minimize unsafe actions, thereby enabling patients to live alone longer.
- **Worker Training:** Training programs are common in factory, culinary, nursing, laboratory, and many other settings. It is natural for a trainee to miss steps or perform steps in the wrong order. Hence, *QuActive* can help in such scenarios by monitoring the progress of the trainees and providing real-time notifications and immediate feedback that improves the training.
- **Nursing Activity Monitoring:** In smart hospitals, nursing activities are monitored for tracking the quality of service and the well-being of patients. Nurses/physicians perform operations based on the patient chart of previous activity steps and sometimes within a specific time limit. Thus, the grammar can be modeled on the different nursing steps to identify the steps already performed on a patient, then notify the nurses who are responsible for the next steps.
- **Surgical Procedure Monitoring:** Surgical operations are often done by a group of people where each person has a specific role to play. Although the μAc performed by each person is usually predefined, performing it depends on the condition of the patient and the μAcs performed by other physicians and nurses. During long surgical operations, doctors can change shifts or leave temporarily in the middle of the operation. Thus, a system monitoring all the steps and their quality during an operation is extremely helpful for the surgeon group as well as ensures better safety of the patient.

3 RELATED WORK

Many of the existing methods for activity recognition with ubiquitous sensors use different statistical and probabilistic approaches [24, 26]. The common algorithms used are Hidden Markov Model (HMM), semi HMM, Naive Bayes Classifier (NBC), and Conditional Random Field (CRF). However, the problem with techniques like HMM is that when the sub-activities of a complex activity are also complex (such as cooking), they do not work properly, since the hidden layers are not directly observable in such case [2, 4]. Both HMM and CRF are focused on the sensor sequence and are less flexible in incorporating variability of activities. NBCs do not retain any time information which makes them less effective in this scenario. On the other hand, algorithms like item set mining [5] disregard the sequence and repetition information of sensors which is necessary for defining the detailed steps. Although the algorithms are capable of detecting and reporting activities, no information is provided about the activity process. Moreover, for certain closely related activities (brushing teeth and shaving), the systems show poor performance.

Methods that focus on garnering fine-grained information about the activity process are mostly done in the areas of vision and image processing. In [20], the authors describe recognizing composite human actions such as handshaking, pointing, punching, pushing etc. from atomic gestures. Many of the existing works on detecting human actions are focused on surveillance, and they are not directly applicable to fine-grained activity recognition. Fine grained cooking activity recognition from videos are presented in [10, 19], where the authors point out the importance and difficulty of fine grained activity detection. These works using cameras have several challenges including privacy issues, requirements of high processing and storage capacity, environmental constraints such as lighting and angular effect, as well as view obstruction due to other humans or large objects.

Blasco et al. [3] describe a smart kitchen where the appliances communicate wirelessly with each other. The smart kitchen partially automates the later steps of an activity if the initial steps are detected. However, the steps are static, and no framework is defined that works in different scenarios. Patterson et al. [17] describe ADL detection from RFID tags and define models to associate different objects with different activities. However, the focus of the paper is to relate objects with activities, whereas our work concentrates on relating different steps of an activity and extracting more information about each step. FABER [18] is a fine-grained activity recognition system for identifying abnormal behavior. The system uses first order logic and achieves high accuracy in separating abnormal activities from the normal ones. However, it assumes single person settings where only one sensor triggers and only one activity occurs at the same time. Moreover, it does not identify missing steps or the details about activity quality. Cace [1] is a system for multi-inhabitant homes for improved activity recognition with hierarchical dynamic Bayesian networks, but not for different activity types or activity quality.

Context Free Grammars (CFG) is widely used in defining and recognizing human activities. Li and Stankovic [11] present a grammar-based fall detection framework that can recognize slow falls and better differentiate falls from other fall like activities. Dimitrios et al. [13] describes probabilistic CFG to track pathways of a user from one camera to another. In [9], the authors use CFG to recognize human action from video footage by co-relating sequences of human pose. In [12], the authors show how a spatio-temporal pattern matching can be used to find the relationships among the activities of daily livings from motion sensors in a single-person home. Although they explore the probability of a specific activity occurring at a particular time, the focus is not on detecting different activity steps and monitoring activity quality. Moore and Essa [15] use a stochastic CFG in recognizing a multitask activity (playing blackjack) from video. It tracks the hand movements to identify the Player's strategy and compares the behavior of novice and experienced players. However, none of these works addresses the issues of recognizing a variety of activities and monitoring activity process quality from a general framework.

4 QUACTIVE FRAMEWORK

The core of the presented system is the QuActive framework. As mentioned before, the framework is based on Timed Probabilistic Context Free Grammar (TPCFG). The rules of the grammar define micro-activities (μAcs) in terms of processed sensor information and activities in terms of the μAcs. Since grammar rules are applied iteratively, intermediate stages of recognized activities are defined as partial activities. The *QuActive* framework has the following advantages:

- **Manage Variation:** Multiple rules are added to represent the same activity that is performed in different ways. For example, making coffee 'using a coffee maker' or 'using hot water and instant coffee packs' have different grammar representations.
- **Handles randomness:** While making tea, the μAcs of 'adding sugar','adding milk','adding tea', and 'pouring hot water' do not require any specific order. However, 'heating water' must be done before 'pouring hot water', and 'stirring' is always the last μAc. These collections of ordered and unordered terms are handled in the *QuActive* framework.
- **Reusable:** Some μAcs of an activity process are observed in other activities. For example, 'adding sugar' occurs in 'making tea' or 'baking cake'. Thus, μAc definitions are reused in defining new activities.
- **Extensible:** People may perform activities differently due to a change of habit. The activity process may also change when new technology or different appliances/objects are used. These changes can be handled just by adding new rules to *QuActive*, without requiring changes to the overall framework.

Section 4.3 presents the general structure of the *QuActive* framework in terms of TPCFG symbols irrespective of any activity class and section 4.4 describes an example grammar of particular activity class ('Making Coffee'). But before that section 4.1 defines a micro-activity and lists some of its properties, whereas section 4.2 gives a mathematical definition of PCFG from the literature.

4.1 Properties of Micro-Activities

A micro-activity (μAc) is the smallest activity step that cannot be decomposed any further. Therefore, a μAc is equivalent to an atomic activity or a simple activity defined in the state-of-the-art literature. In this paper, the following statements hold true for a μAc:

i An activity can be broken into one or more μAcs. So, a μAc can be an activity itself. For example, 'heating water' can itself be an activity or a μAc of 'making tea'.

ii μAcs can not be done partially, i.e., once started a μAc has to be finished, or otherwise it is disregarded.

iii μAcs can occur in different activities. For example, the μAc 'using water' can be a part of the activity 'washing dishes' or the activity 'mopping the floor'.

iv Although every activity is associated with one or more users, and every μAc is associated with some activit, the μAc itself might be independent of a user. For example, a user triggers the switch to boil water, but water boiling itself is independent and the user may do something else during that time.

4.2 Probabilistic Context Free Grammar

A context free grammar (CFG) is a type of language generator. It is expressed as $< V_N; V_T; Start; R >$, where

- V_N is a finite set of nonterminal symbols. Nonterminals are represented with words starting with a capital letter.
- V_T is a finite set of terminal symbols. Terminals are represented with words starting with lower-case letters.
- $V_N \cap V_T = \emptyset$. $V = V_N \cup V_T$ is called the vocabulary and $V*$ is the set of all strings of symbols in V including the string of length zero.
- $Start \in V_N$ is the start symbol.
- R is a finite nonempty subset of $V_N \times V^*$ called the production rules.

A CFG where multiple rules define the same non-terminal can be extended to a probabilistic CFG [8] as an ordered five-tuple $< V_N; V_T; Start; R; P >$, where

- The production rules are paired with a set of probabilities $\{p_{ij}\}$ that satisfy the following rules.
  - For each production rule $R_{ij} \in R$, there is one and only one probability $p_{ij} \in P$.
  - $0 < p_{ij} \leq 1 \ \forall_{i,j}$
  - For every i with $1 < i \leq |V_n|$, $\Sigma_{1 \leq j \leq n_i} p_{ij} = 1$, where n_i is the number of productions with the i-th nonterminal on the left-hand side.

Timed PCFG is an extension of PCFG where timing information is incorporated into the grammar rules [21].

4.3 TPCFG for Detecting Activity

In this section, a general definition of TPCFG is provided. Each activity class has a separate set of rules that follow this general structure. The terminals of the grammar are sensor values and rules are iteratively applied until the terminals are reached. The activities and μAcs are nonterminals. The following rules show how an activity can be composed of a sequence of μAcs.

Activity	→	MicroActivity TmDiff PartialActivity
Activity	→	MicroActivity
PartialActivity	→	MicroActivity TmDiff PartialActivity
PartialActivity	→	Activity

Here, the rules for 'PartialActivity' are necessary for generating an unambiguous grammar. After iteratively applying the rules, all 'PartialActivity's are decomposed till only the μAcs are left. The nonterminal 'TmDiff' indicates the time difference between two consecutive μAcs. A negative duration value of 'TmDiff' indicates overlap between the two μAcs. For example, heating water while adding coffee can occur in parallel.

The μAc is the smallest activity step that cannot be decomposed any further. However, a μAc can be associated with more than one sensor, since several sensor events can occur at the same time. For example, several motion sensors can be triggered when a user enters a particular location.

MicroActivity	→	Event
Event	→	(Event, Event)
Event	→	(Sensor,Time,Value)

The above rules show how each μAc is associated with one or more sensor events. The comma separated tuple indicates that the sensor events are independent of each other. Each sensor event indicates the change of a particular sensor value at some specific time or during a specific duration.

Sensor	→	InsituSensor | Wearable
InsituSensor	→	motionSensorID | contactSensorID | tempSensorID | pressurePadID
Wearable	→	smartWatchID

Now, a sensor is either a wearable device or an in-situ sensor. The above rules show a sensor setting where a smart watch is used as a wearable device, and motion sensors, contact sensors, temperature sensors, and pressure pads are used as in-situ sensors. The value of

Value	→	num | feature

the sensor is either a numeric value (0 or 1 of a binary pressure pad sensor) or values of list of features extracted from the continous sensor signal (e.g., features calculated from the accelerometer or the gyroscope data of a smart watch).

Although 'Time' and 'Tmdiff' both provide timing information, one is associated to a sensor event and the other indicates the time difference between two μAcs as mentioned earlier. The nonterminal 'Time' is associated with each sensor event. The starttime indicates the time when the sensor value changes, and duration indicates how long the sensor value remained constant (or was above/below threshold in case of continuous data).

Time	→	(starttime, duration)
TmDiff	→	(starttime, duration)

Grammars defined for all the activity classes maintain this described structure. The grammar described here does not show the probability (P) for simplicity. The probability is associated with each rule when multiple rules define the same nonterminal.

4.4 Example Grammar and Parse Tree

Table 1 shows one example of possible different rules for making coffee with start symbol as 'MakingCoffee'. To make the steps in the rules clear, the timing information has not been shown. However, each event is associated with the $Time(starttime, duration)$ information and each rule contains the $TmDiff$ information between two consecutive terms on the right side of the rule. As we can see, multiple rules cover different situations such as when the items or utensils are retrieved from the refrigerator, cupboards, or drawers, as opposed to already being placed on the counter top. The associated probability represents the probability of that situation occurring. If a high probability rule does not match, other relevant rules are applied. One limitation of the system is that it cannot identify the exact added item. For example, if somebody adds only sugar instead of coffee, QuActive still recognizes the Activity 'Making Coffee'. However, the limitation is associated with the sensing system and not directly related to the QuActive framework. In the future, if the sensing capability enables distinguishing each item, then corresponding rules can be added to make the grammar richer.

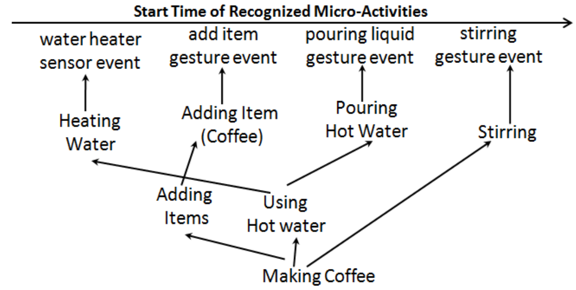
MakingCoffee(p_{11})	$\rightarrow$	(UsingDrinkware) (UsingUtensil) (UsingHotWater) (AddingItems) (Stirring)
MakingCoffee(p_{12})	$\rightarrow$	(UsingDrinkware) (UsingUtensil) (AddingItems) (UsingHotWater) (Stirring)
UsingDrinkware(p_{21})	$\rightarrow$	(MovingObjectGestureEvent)
UsingDrinkware(p_{22})	$\rightarrow$	(OpenCupboardEvent) (MovingObjectGestureEvent)
UsingUtensil(p_{31})	$\rightarrow$	(MovingObjectGestureEvent)
UsingUtensil(p_{32})	$\rightarrow$	(OpenUtensilDrawerEvent) (MovingObjectGestureEvent)
UsingHotWater(p_{41})	$\rightarrow$	PouringWaterGesture
UsingHotWater(p_{42})	$\rightarrow$	HeatingWater PouringWaterGesture
AddingItems(p_{51})	$\rightarrow$	AddingItems*
AddingItems(p_{52})	$\rightarrow$	AddingItem
AddingItem(p_{61})	$\rightarrow$	AddingItemGesture
AddingItem(p_{62})	$\rightarrow$	OpenCupboardEvent RetrievingItemGesture AddingItem
AddingItem(p_{63})	$\rightarrow$	PouringLiquidGesture
Adding Item(p_{64})	$\rightarrow$	OpenRefrigeratorEvent RetrievingItemGesture AddingItem

Table 1: TPCFG for activity 'Making Coffee'.

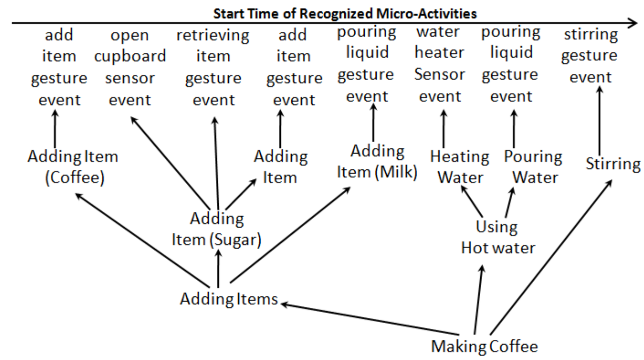
Figure 1 shows two example parse trees depicting 'Making Coffee' in two different situations. In the first tree, only coffee is added whereas in the second tree milk and sugar are added to the coffee. However, in both situations, it is assumed that drinkware (mug) and the utensil (spoon) are already placed on the counter top. A different and larger parse tree will be generated in situations where the drinkware and the utensil need to be retrieved from their storage places. The intermediate levels of the tree shows nodes related to partial/sub activities. However, some order is maintained in all the different situations. For example, heating water always precedes pouring water (although other μAcs can take place in between) in order to identify the higher level activity 'Using Hot Water'. Again, stirring is always performed last. Although timing information has not been shown explicitly, each low level event retains time information which propagates up to the root (highest level activity) of the tree.

5 SYSTEM DESIGN

In this section, we describe the system architecture that uses the QuActive framework to recognize fine grained activity steps, construct high level activities, and extract activity quality parameters.



(a) The parse tree shows a way of 'Making Coffee' in which no sugar or milk is added in the coffee and the coffee is added in between heating and pouring water. It also assumes that the drinkwares, utensils, and ingredients (mug, spoon, and coffee) are already placed on the counter top,i.e., no object is retrieved from the cabinet or the drawer.



(b) The parse tree shows a situation where multiple items (milk, coffee, sugar) are added in the coffee and only one item (sugar) is retrieved from the cabinet.

Figure 1: Example parse trees showing different ways of performing the same activity ('Making Coffee').

5.1 Sensing Layer

The sensing layer consists of both in-situ sensors and wearable sensors for collecting detailed activity information. Wearable sensors are placed on a user's body to collect gesture information related to the activities. The system assumes a smart watch as the wearable device containing accelerometer, gyroscope, and magnetometer. On the other hand, binary contact switches, binary pressure pads, motion sensors, and temperature sensors work as in-situ sensor nodes. Therefore, the sensing layer collects human motion that causes an activity as well as events related to the effect of resulting activities on the surrounding environment.

5.2 Event Layer

This layer preprocesses the sensor data and lists all sensor events. Whenever the status of a sensor is changed, an event is triggered. For example, a pressure pad triggers the event 'occupied' if somebody sits on it and triggers 'empty' whenever the person leaves. The environment sensors are assumed to generate discrete sensor events. On the other hand, the sensors in the smart watch generate continuous data streams at a particular sampling rate. A threshold value is used to filter the normalized time series data where no significant motion is detected by the accelerometer and gyroscope.

The filtered segments denote possible gesture events. Time information from environment sensors is provided to identify segments where gestures are more likely related to some activity step as well as to trim segments to find the approximate start and end time of a gesture. For example, by aligning the contact sensor events 'opening refrigerator' and 'closing refrigerator' with wearable sensor data, QuActive finds signals related to 'storing groceries' and 'retrieving groceries'.

5.3 Micro-activity Layer

This layer constructs the micro-activities (μAc) or possible μAc from the information provided by the event layer. Although a user can leave in the middle of an activity, based on the properties of μAc - a user cannot leave in the middle of a μAc. Therefore, a μAc is either complete or not done at all. Any partial information is ignored. To detect activities from sensors and gesture events, the *QuActive* grammar is applied to event data. Grammar rules associating data with μAcs are created from training data and user labeling. Upon detecting problems (such as taking too long to perform a particular μAc), information is passed to the notification layer.

5.4 Activity Layer

QuActive consists of grammar rules for each activity mapping to one or more μAcs. The rules are defined based on real world observations as well as state-of-the-art definitions (particularly in ADL research in vision) relating activities with micro-activities [10]. Each rule is assigned a default probability. Training is necessary in order to calculate the probability values from a particular real world deployment.

The Activity Layer applies *QuActive* grammar rules to find the activities that occurred. Sometimes low level activities are combined to a higher level activity.So, rules are applied iteratively until no new activities are observed. If an activity rule matches up to a certain level, where the μAcs are not part of any other activity, *QuActive* assumes that a certain activity was started, but not finished. Since the grammar preserves time information and μAc ordering, the activity quality parameters are extracted from these values. The activity complexity is determined based on how many iterations were performed in order to construct the activity. The activity and μAc timing information are used to classify the activity type whereas the quality parameters are used to create notifications. The priority of the notification is relevant to the severity of the problem in terms of safety (e.g., leaving the stove on), inconvenience (e.g., forgetting to put coffee in the coffee machine), or other issues (e.g., taking too long). It should be noted that *QuActive* distinguishes similar μAcs by correlating with different in-situ sensors triggered by the action or matching the μAcs with grammar rules relevant to high level activities. However, if no specific in-situ sensor is triggered and two parallel activities with similar μAcs occur, then *QuActive* is unable to identify the exact μAc.

QuActive Database: The rules for mapping μAcs with each activity class are applied separately for every activity class.

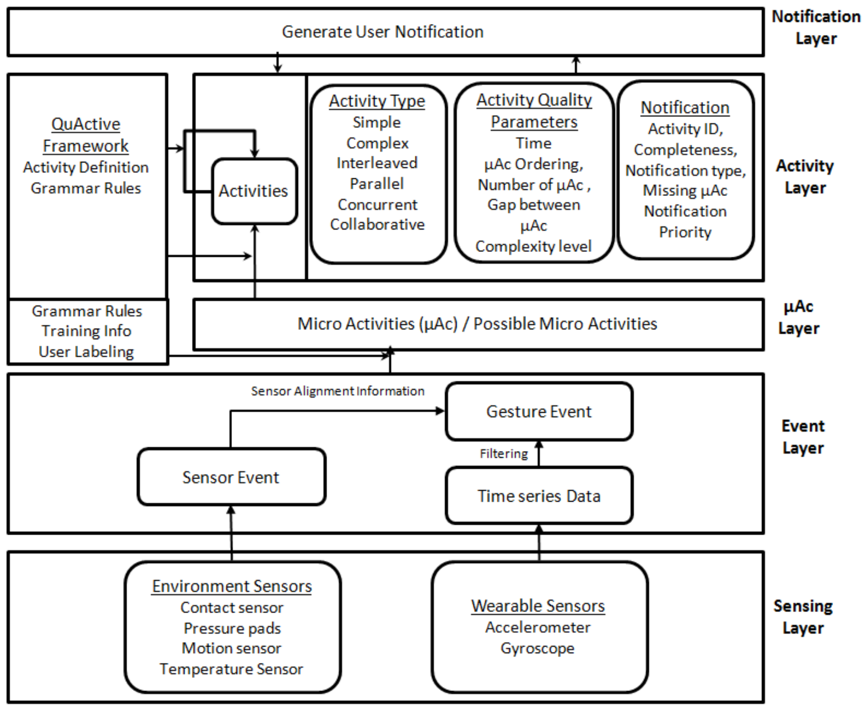


Figure 2: System Architecture for detecting and recognizing micro-activities and high level activities based on QuActive Framework.

In this paper, this mapping has been done manually. However, if the sensor setting and activity list are similar, existing rules are applied for future activity instances. For example, during the evaluation of dataset 6.1.3, the rules created for dataset 6.1.1 were used. On the other hand, once a μAc is defined for a setting, it is not redefined for every activity. For example, in our experiment the 'Stirring' μAc is found in the activities 'Cooking' and 'Making Tea'. The 'Stirring' μAc is mapped to the stirring gesture event, whereas the gesture event detection is done by labeling the smart watch data and applying a decision tree with five-fold cross validation.

5.5 Feedback - A Notification Layer

The smart-watch based notification subsystem is an extension of the MedRem voice based medication reminder system [14]. However, instead of only medication reminders, the system reminds users about activity problems. Moreover, MedRem is a stand-alone system whereas the *QuActive* notification system receives information about the activity parameters assuming that the watch has WiFi capability. In addition to providing reminders about activities, the sub-system generates notifications based on the notification ID.

For example, missing activities trigger question like *"Have you performed 'Activity X'?"* whereas missing *µAcs* trigger question like *"Have you missed 'µAc y' when doing 'Activity X?'"* where the 'y' and 'X' are replaced with the corresponding parameters received from the Activity Layer.User's answer is stored and then sent back to the Activity Layer for further tracking. Therefore, the *QuActive* system intervenes with the user through informed notification.

6 EVALUATION

6.1 Datasets

We used the following datasets to evaluate the performance of *QuActive*. All these datasets have motion sensors, contact sensors, item sensors, temperature sensors, and water sensors installed in a smart home apartment.

6.1.1 Interleaved ADL. Dataset [22] contains activity information from 20 users. Each participant first performs the activities 1 to 8 defined in Table 2 independently, and then multiple activities concurrently. Therefore, the steps of multiple activities are intertwined. The details of the steps are provided to the participants. Table 3 shows an example of given instructions for the activity 'Preparing soup'.

6.1.2 Multiresident ADL . Dataset [23] has information about parallel and co-operative activities. Here, each individual performs activities 1,2,5,9,10,11,12 (Table 2) independently, but two persons act in parallel. Therefore, steps of activities from different users are observed at the same time. In addition, activities 13 to 16 are performed jointly (co-operative activity), where the steps are either done individually (playing checkers) or together (moving furniture).

6.1.3 Cognitive assessment activity data. In dataset [4], 65 healthy and 14 cognitively impaired people are selected for the data collection process based on initial screening and questionnaires. Then, each participant is asked to complete the activities 1 to 8 defined in table 2 step by step. The dataset annotates the ground truth by labeling each sensor value with corresponding activities and sub-activities. Each activity is scored by expert clinicians from 1 to 8 based on the level of completeness. Moreover, the users are also diagnosed by the clinicians as healthy, as patients with mild cognitive impairment (MCI), or as patients with dementia based on the interviews, questionnaires, and performed tasks. All this information is used as the ground truth for the analysis.

6.2 Data Collection

In order to evaluate the performance of *QuActive* in a real home setting, we have deployed the system in a real home and collected data from four users. All the users were healthy young adults from both gender groups (males and females). We used z-wave pressure pads, contact sensors, and motion sensors for collecting environmental information. The pressure pads were placed on the living room sofa, dining room chairs, and study chairs. The contact sensors were attached to the cabinets, microwave, oven, refrigerator, freezer, and closets. The data was collected in a semi-controlled setting and the users were instructed to perform the activities 17-24 (table 2). The experiment was not fully controlled since we did not specify the exact steps to perform the activities or constrain their

	Interleaved Activity List (dataset 6.1.1 and 6.1.3)
1.	Sweeping the kitchen and dusting the living room.
2.	Obtaining medicine containers and a weekly medicine dispenser, filling the dispenser according to the directions.
3.	Writing a birthday card, enclosing a check and writing the address on an envelope.
4.	Finding the appropriate DVD and watching the corresponding news clip.
5.	Obtaining a watering can and watering all plants in the living space.
6.	Answering the phone and responding to questions.
7.	Preparing a cup of soup using the microwave.
8.	Picking a complete outfit for an interview from a selection of clothing.
	Multiresident Independent, Parallel, and Co-operative ADLs (dataset 6.1.2)
	1,2, and 5 (from the above list)
9.	Reading magazine in living room sofa
10.	Preparing dinner
11.	Setting dining table
12.	Hanging up clothes in closet
13.	Moving furniture
14.	Playing checkers
15.	Paying bills
16	Gathering and packing picnic supplies
	Activity List: Collected Data (dataset 6.2)
17.	Study (sitting on study chair, using typing motion)
19.	Watching TV (sitting on living room sofa, occasional hand gesture for using remote)
20.	Making tea (using cabinets and refrigerator, heating water, gesture of stirring and putting items)
21.	Eating (sitting on dining chair, hand gesture of eating)
22.	Washing dishes (using tap, scrubbing dishes, rinsing dishes)
23.	Cooking (using cabinets, refrigerator, microwave, oven, and hand gesture for cutting, stirring)
24.	Dressing (choosing outfit from closet, motion for changing clothes)

Table 2: Activity List used for evaluation in different datasets and the collected data.

Activity Steps: Preparing Soup
1. Retrieve materials from cupboard "A"
2. Fill measuring cup with water
3. Boil water in microwave
4. Pour water into cup of noodles
5. Retrieve pitcher of water from refrigerator
6. Pour glass of water
7. Return pitcher of water
8. Wait for water to simmer in cup of noodles
9. Bring all items to dining room table

Table 3: Example of activity steps within 'Preparing soup' instructed to be performed by a user [22, 23].

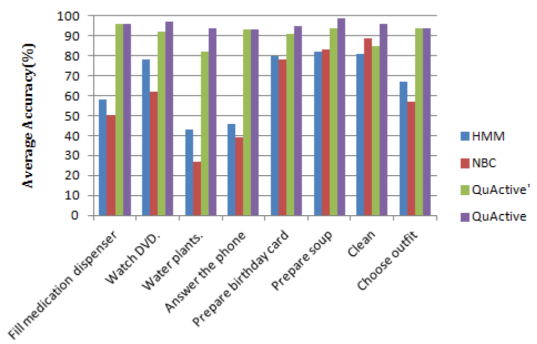


Figure 3: Recognizing interleaved (high level) activities using HMM, NBC [22], QuActive with and without (QuActive') location information incorporated in the grammar. The figure shows the percent of sensor data labeled correctly with respect to ground truth labeling.

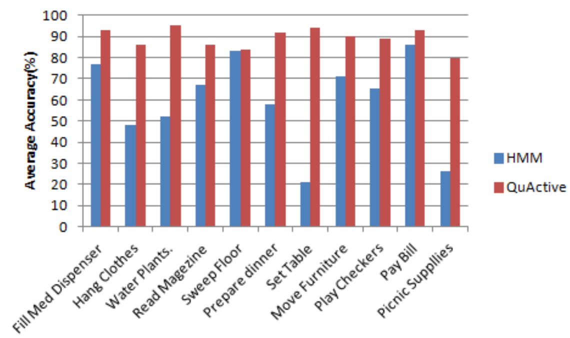


Figure 4: Average accuracy in recognizing instances of independent, parallel, and joint activities using HMM [23] and QuActive on a multiresident dataset.

movement in any way. Users were free to perform the activities in their own way and were not required to start/stop each μAc from a resting position. Before each session the user wore an android smart watch on his/her dominant hand. The accelerometer and the gyroscope sensors (sampling rate 50Hz) in the watch were used to capture the hand gestures relevant to micro-activities (Appendix A). Since each watch has a specific ID, it can be used to identify users in multi-user scenarios. However, in this experiment multiple users did not perform activities at the same time. Therefore, the collected dataset has independent and interleaved activities from a single user setting. Table 2 shows some example steps observed during the activity process.

Another assumption made in this paper is that μAc itself cannot be discontinuous, i.e., a μAc is either done or not. However, the μAcs within an activity can be discontinuous. The framework assumes that each "TmDiff" parameter has a defined limit. QuActive assumes that the activity is not completed if the time is exceeded. The parameter value needs to be defined from training data from long term deployments. However, we did not have enough data necessary to define the parameter. We plan to accomplish this in future. In order to collect the ground truth data, an observer video recorded the session, and the time series sensor data was annotated from the video using 'Chronoviz' software [7].

6.3 Results

6.3.1 Evaluation on the Datasets. Figure 3 shows a comparison of QuActive with NB and HMM classifiers [22] applied to the interleaved ADL dataset. Again, QuActive' disregards the location information where QuActive incorporates location information (using location of the sensors) as part of the grammar. In paper [22], the authors show that Naive Bayes and HMM classifiers achieve average accuracy of 66.08% and 71.01% respectively in detecting activities performed in an interleaved manner. The grammar in QuActive is constructed only with the μAcs defined from instruction steps disregarding the user's location. Although location is not needed to detect most activities, it clarifies when context is important. For example, 'Watering Plants' and 'Sweeping Living Room' use different equipment, but the same closet sensor is triggered while retrieving/storing the equipment. Thus, adding the location information ('kitchen to living' or 'in living') in between the closet sensor trigger provides context about the prospective use of the equipment. With location information incorporated in the grammar, almost all the activities are detected perfectly. However, the accuracy in the graph show less than 100%, since the exact start time and end time of the steps are not always aligned with the ground truth data. Therefore, there is less accuracy in terms of the percentage of sensor values labeled correctly with the corresponding activity.

Figure 4 shows the performance of QuActive in multi-person setting, where some activities are performed jointly and some are done independently but in parallel. Here, the baseline is the average performance of the user-independent HMM classifier. Authors in the paper [23] show that a user specific classifier increases the accuracy of activity detection for that user by about 20% on average, but the performance of the system decreases in detecting the activities of the other user. Thus, the average performances of user-independent and user-specific HMM models are almost the same. On the other hand, QuActive performs very well in this dataset, because it filters irrelevant μAcs that do not match the grammar structure. In other words, unless the μAcs in other users' activity match the μAcs of a particular user, they do not affect the activity detection process of that user and thus yields higher accuracy.

In paper [4], the researchers show the activities of daily living (ADL) as a good predictor of early detection of cognitive impairment. They extract 38 features from the sensor dataset and show that using leave-one-out cross validation accuracy of 86% is achieved in predicting the cognitive state. However, their process requires a lot of data from cognitive impaired and healthy persons for feature generation and training. Although we do not apply QuActive to determine cognitive status of a person, Figure 5 shows how the value of the quality parameters defined in QuActive varies in healthy, MCI, and dementia patients in this dataset.

The solid columns in Figure 5 show the average number of missing steps per activity from the performed activity instances. The striped column shows the average number of missing steps considering the total activities, i.e., the effect of a missing activity is also considered. Similar differences are observed considering the

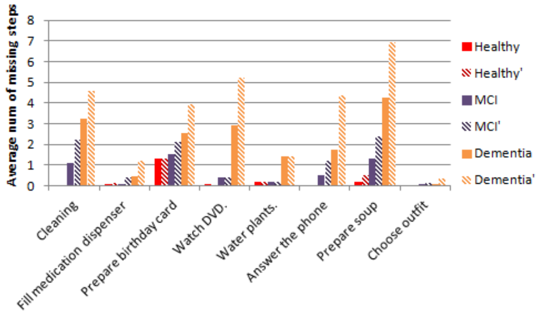


Figure 5: The average number of missing steps in performing activities by healthy, mildly cognitively impaired, and dementia patients. The stripped column values considers the effect of missing activities in calculating missing steps and the solid column values disregards missing activities.

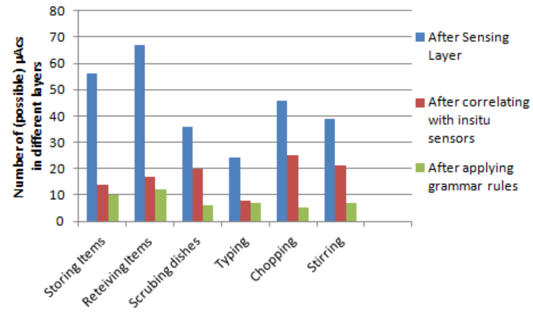


Figure 6: Filtering the falsely recognized micro-activities in subsequent layers of the system.

Activity	Precision	Recall
Making Tea	0.95	0.88
Washing Dishes	0.96	0.87
Cooking	0.91	0.84
Eating	0.93	0.95
Dressing	0.97	0.96
Study	0.98	0.99
Watching TV	0.98	0.98

Table 4: Average performance of QuActive in recognizing activity instances from all users

activity duration and total duration between two consecutive activities. The figure shows the validity of considering missing steps and missing activities in identifying stages of dementia with a general purpose activity recognition framework. Thus, a grammar defined from the descriptions, that does not need huge training data and complex algorithms, can also be a powerful tool in detecting how activity quality degrades over time.

6.3.2 Evaluation on Experimental Data. We collected a total of 67 activity instances of activities 17 to 24 (Table 2) from four users in our experiments. Although only one user performed activity at a time, she/he occasionally performed more than one activity in parallel. The experiment settings have both z-wave sensors and smart watch data for gesture recognition. QuActive recognizes all the μAcs that are defined only in terms of z-wave sensors, such as 'Sitting on the sofa', 'Opening the refrigerator' etc. However, in our experiment only the in situ sensors do not give all information required for detecting all the μAcs.

For example, 'Opening cabinet' and 'Closing cabinet' are detected from the contact sensors, but differentiating between 'storing item in the cabinet' or 'retrieving item from the cabinet' requires additional information which can be extracted from the hand motion. However, detecting μAcs from a continuous stream of sensor data itself is a challenging problem and the accuracy depends on the collected data and the threshold values for determining the cut-off point. In our experiments for detecting μAcs, we choose a lower threshold value to get a high percentage of true positives despite having a high false negative rate and therefore a lower recall. For evaluation purposes, we use a state-of-the-art supervised algorithm (Decision Tree C4.5) for gesture recognition and five fold cross validation irrespective of the user. However, coupling the gesture events with the in situ sensors filters a lot of the falsely recognized gestures. If the same gesture signal indicates more than one possible μAcs, the one matching with the defined grammar is recognized and the rest are eliminated.

Figure 6 shows examples of a number of possible gesture events recognized from raw signal and how the number of irrelevant gestures are filtered at different stages, i.e., after associating with in situ sensors and finally mapping with grammar rules. Figure 7 (Appendix A) shows time series data corresponding to some example gesture events.

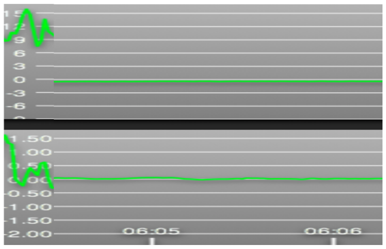
Table 4 shows the precision and recall of activity instances recognized correctly despite each user performing the activities in their own way. The accuracies of detecting high level activities are 91% to 98%, which shows the promise of the *QuActive* system.

6.3.3 Notification Subsystem. The notification subsystem has been implemented and evaluated separately from the activity recognition subsystem. In the implementation, once a notification is required, the notification subsystem delivers it to a smart watch 100% of the time and properly records the user responses. The responses vary depending on the notification type. For example, if the notification is a reminder to add coffee to the coffee maker the user might respond, "OK done." Or if the notification suggests that they forgot to take their noon medication, the user might respond "I'll do it later." However, a user study that shows how effective the notifications are in actually improving health or performance of daily activities is beyond the scope of this paper.

7 CONCLUSIONS

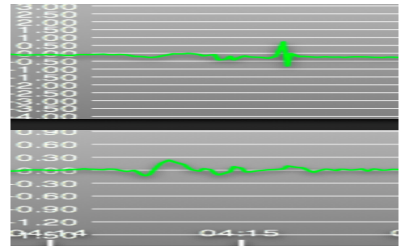
QuActive is a CPS monitoring and notification system for activities of daily living. It is based on a temporal, probabilistic, context free grammar and a smart watch based notification system. It addresses the complexities of concurrent and parallel activities, and multiple person situations. It identifies missing steps, delayed steps, and out of order steps in activities. Using several datasets, the performance of QuActive is shown to be (average accuracy of 95%) significantly above than the two baselines (accuracy of 66% and 71% respectively) from the literature.

A APPENDIX: VISUALIZATION OF TIME SERIES DATA OF MICRO-ACTIVITIES

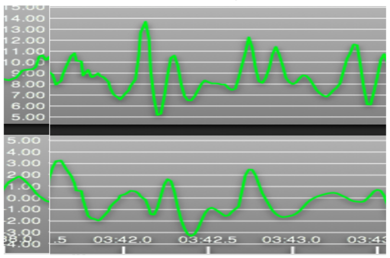


(a) Hand at Resting Position

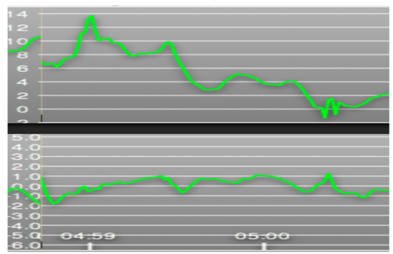
(b) Reading paper



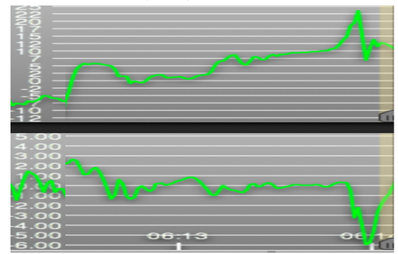
(c) Scrolling Slightly with a Mouse



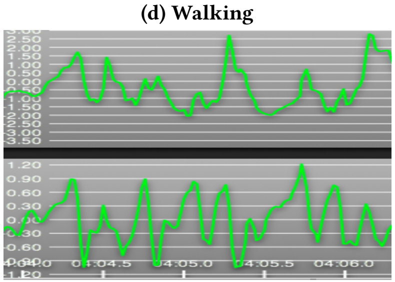
(d) Walking



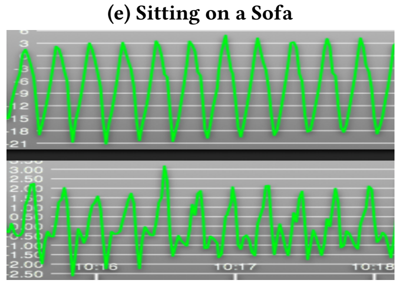
(e) Sitting on a Sofa



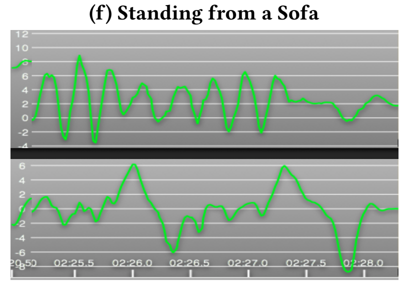
(f) Standing from a Sofa



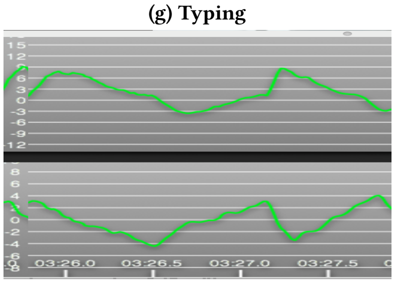
(g) Typing



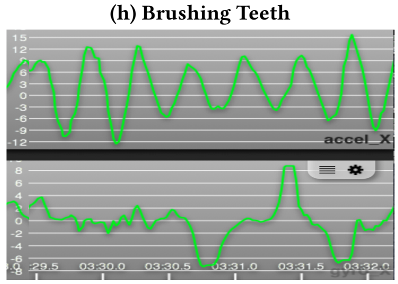
(h) Brushing Teeth



(i) Rinsing a Mug

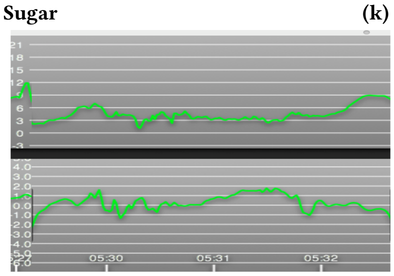


(j) Adding Sugar

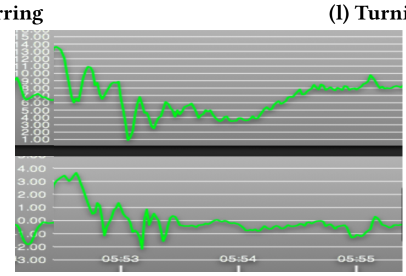


(k) Stirring

(l) Turning Stove On



(m) Opening a Sliding Door



(n) Closing a Sliding Door

Figure 7: The above images show time series data of accelerometer x-axis and gyroscope x-axis corresponding to different gestures. The snapshots are taken from the 'Chonoviz' visualization software tool. The x-axis labels in each image show time in 1s intervals (except (l) which shows in 0.5s intervals). The y-axis show the normalized value of the accelerometer and the gyroscope in a fitted zoomed position.

ACKNOWLEDGMENTS

This work was supported, in part, by NSF Grants CNS- 131930, CNS-1646470, and DGIST Research and Development Program (CPS Global Center) funded by the Ministry of Science, ICT & Future Planning. The authors also thank Ridwan Alam (UVa) for his contribution during the implementation of the z-wave receiver and Sarah Masud (UVa) for her help during the data collection process.

REFERENCES

[1] Mohammad Arif Ul Alam, Nirmalya Roy, Archan Misra, and Joseph Taylor. 2016. CACE: Exploiting Behavioral Interactions for Improved Activity Recognition in Multi-Inhabitant Smart Homes. In *Proceedings of the International Conference on Distributed Computing Systems*. IEEE.

[2] Asma Benmansour, Abdelhamid Bouchachia, and Mohammed Feham. 2016. Multioccupant activity recognition in pervasive smart home environments. *ACM Computing Surveys (CSUR)* 48, 3 (2016), 34.

[3] Rubén Blasco, Álvaro Marco, Roberto Casas, Diego Cirujano, and Richard Picking. 2014. A smart kitchen for ambient assisted living. *Sensors* 14, 1 (2014), 1629–1653.

[4] Prafulla Dawadi, Diane Cook, Carolyn Parsey, Maureen Schmitter-Edgecombe, and Miya Schneider. 2011. An approach to cognitive assessment in smart home. In *Proceedings of the workshop on Data mining for medicine and healthcare*. ACM, 56–59.

[5] Ifat Afrin Emi and John A Stankovic. 2015. SARRIMA: Smart ADL recognizer and resident identifier in multi-resident accommodations. In *Proceedings of the conference on Wireless Health*. ACM, 4.

[6] Chronic Care Networks for Alzheimer's Disease Initiative and others. 2003. Tools for early identification, assessment, and treatment for people with Alzheimer's disease and dementia. *Alzheimer's Association and National Chronic Care Consortium* (2003).

[7] Adam Fouse, Nadir Weibel, Edwin Hutchins, and James D Hollan. 2011. ChronoViz: a system for supporting navigation of time-coded data. In *CHI*. ACM, 299–304.

[8] Stuart Geman and Mark Johnson. 2002. Probabilistic grammars and their applications. *International Encyclopedia of the Social & Behavioral Sciences* (2002), 12075–12082.

[9] Imran N Junejo, Emilie Dexter, Ivan Laptev, and Patrick Perez. 2011. View-independent action recognition from temporal self-similarities. *Pattern Analysis and Machine Intelligence* 33, 1 (2011), 172–185.

[10] Jinna Lei, Xiaofeng Ren, and Dieter Fox. 2012. Fine-grained kitchen activity recognition using rgb-d. In *Proceedings of the Conference on Ubiquitous Computing*. ACM, 208–211.

[11] Qiang Li and John A Stankovic. 2011. Grammar-based, posture and context-cognitive detection for falls with different activity levels. In *Proceedings of Wireless Health Conference*. ACM, 6.

[12] Dimitrios Lymberopoulos, Athanasios Bamis, and Andreas Savvides. 2011. Extracting spatiotemporal human activity patterns in assisted living using a home sensor network. *Universal Access in the Information Society* 10, 2 (2011), 125–138.

[13] Dimitrios Lymberopoulos, Abhijit S Ogale, Andreas Savvides, and Yiannis Aloimonos. 2006. A sensory grammar for inferring behaviors in sensor networks. In *International conference on Information processing in sensor networks*. ACM, 251–259.

[14] Md Abu Sayeed Mondol, Ifat Afrin Emi, and John A Stankovic. 2016. MedRem: An Interactive Medication Reminder and Tracking System on Wrist Devices. In *Proceedings of the Conference on Wireless Health*. ACM.

[15] Darnell Moore and Irfan Essa. 2002. Recognizing multitasked activities from video using stochastic context-free grammar. In *AAAI/IAAI*. 770–776.

[16] Juan Carlos Niebles, Chih-Wei Chen, and Li Fei-Fei. 2010. Modeling temporal structure of decomposable motion segments for activity classification. In *Proceedings of the European Conference of Computer Vision*. Springer, 392–405.

[17] Donald J Patterson, Dieter Fox, Henry Kautz, and Matthai Philipose. 2005. Fine-grained activity recognition by aggregating abstract object usage. In *International Symposium on Wearable Computers*. IEEE, 44–51.

[18] Daniele Riboni, Claudio Bettini, Gabriele Civitarese, Zaffar Haider Janjua, and Rim Helaoui. 2015. Fine-grained recognition of abnormal behaviors for early detection of mild cognitive impairment. In *Proceedings of the International Conference on Pervasive Computing and Communications*. IEEE, 149–154.

[19] Marcus Rohrbach, Sikandar Amin, Mykhaylo Andriluka, and Bernt Schiele. 2012. A database for fine grained activity detection of cooking activities. In *Computer Vision and Pattern Recognition*. IEEE, 1194–1201.

[20] Michael S Ryoo and Jake K Aggarwal. 2006. Recognition of composite human activities through context-free grammar based representation. In *Computer vision and Pattern Recognition*, Vol. 2. IEEE, 1709–1718.

[21] Neda Saeedloei and Gopal Gupta. 2010. Timed definite clause omega-grammars. In *LIPIcs-Leibniz International Proceedings in Informatics*, Vol. 7. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

[22] Geetika Singla, Diane J Cook, and Maureen Schmitter-Edgecombe. 2009. Tracking activities in complex settings using smart environment technologies. *International journal of biosciences, psychiatry, and technology* 1, 1 (2009), 25.

[23] Geetika Singla, Diane J Cook, and Maureen Schmitter-Edgecombe. 2010. Recognizing independent and joint activities among multiple residents in smart environments. *Journal of ambient intelligence and humanized computing* 1, 1 (2010), 57–63.

[24] Emmanuel Munguia Tapia, Stephen S Intille, and Kent Larson. 2004. *Activity recognition in the home using simple and ubiquitous sensors.* Springer.

[25] Max Van Kleek and Howard E Shrobe. 2007. A practical activity capture framework for personal, lifetime user modeling. In *User Modeling*. Springer, 298–302.

[26] Daniel H Wilson and Chris Atkeson. 2005. Simultaneous tracking and activity recognition (STAR) using many anonymous, binary sensors. In *Pervasive computing*. Springer, 62–79.