Assurance-Oriented Activity Recognition

Eunju Kim
University of Florida
ejkim@cise.ufl.edu

Sumi Helal
University of Florida
helal@cise.ufl.edu

Chris Nugent
University of Ulster
cd.nugent@ulster.ac.uk

Jae Woong Lee
University of Florida
jwlee@cise.ufl.edu

ABSTRACT
Activity recognition (AR) research promises to enable a multitude of human-centric applications in smart environments. Nevertheless, application developers will require assurance mechanisms before they can confidently use and apply AR in real-world pervasive systems. In this work we propose an extension of an existing AR approach in which richer recognition semantics that address confidence and assurance are provided. Our approach differentiates between an activity and its effect and subsequently relies on verifying an activity by recognizing its effect. We present our approach along with a comparative experimental evaluation.

Author Keywords
Activity recognition, Activity verification, Activity effect, Recognition Assurances, Situation, Activity effect knowledge base, Activity assurance.

ACM Classification Keywords
I.2.4 [Knowledge Representation Formalisms and Methods]: Frame and scripts, Representation (procedural and rule-based), Temporal logic

General Terms
Verification, Design

INTRODUCTION
Activity recognition (AR) is a key technology used in developing intelligent services in many human-centric applications. The existing body of AR research has greatly contributed to improvements in intelligent services technology. Nevertheless, despite this progress, current AR systems are not adequate for practical use in real world applications for a number of reasons, the most notable being a lack of general confidence in its usage and assurances or guarantees of the outcomes produced by the AR process. In this paper, we analyze confidence and assurance issues in the context of practical application development of AR processes. We propose a new approach that provides much richer recognition semantics, which embeds a level of confidence and assurance into the AR system. Our approach utilizes the activity model itself and the AR system monolithically to verify a given recognized activity. Prior to presenting the details of our approach we firstly present the current limitations of the recognition semantics of existing AR systems.

Confidence Level of Recognized Activities
For practical applications, an activity should be recognized with the highest possible level of confidence. In other words, whenever an AR system recognizes an activity, it may be necessary to verify that a certain level of confidence is satisfied before taking an action. Confidence is crucial especially for safety- or security-critical applications such as in instances of healthcare or eldercare service provision [1][2][11]. For example, complying with a medication regime is extremely important. Issues of non-compliance may result in serious health problems. Therefore, whenever an AR system recognizes the activity "taking medicine", it is necessary to minimize any error in the recognition of this specific activity.

AR systems generally use statistical performance only to quantify the accuracy of their recognition. When stationary (steady state) average performance is used to characterize an AR system, high recognition performance does not provide a guarantee that any single activity is recognized at or above the statistical performance level. In other words, the measurement of stationary performance only provides an indication of overall system performance. Nevertheless, AR system performance in reality is dynamic, which is dependent on many factors such as activity type, sensor condition at the recognition time, the user's attitude to perform an activity, to name but a few. To illustrate this concept, consider the scenario when the same activity may be performed by the same user differently at different times depending on the user's mood. This inconsistency can cause problems for an AR system especially when a high confidence level is required. Therefore an AR system should provide a method to confirm the confidence level of a recognized activity.

Activity Effect Recognition vs. Activity Procedure Recognition
Activity effect recognition recognizes the related effects of an activity whereas activity procedure recognition recognizes the steps of an activity based on the observation from a series of sensor events. For example, an eating activity is composed of several sub-step actions such as serving food, using a spoon or fork, cutting meat, or drinking water. For the purposes of recognizing an eating activity, the AR system is required to detect sensor events which can be directly associated with the actions involved. The eating activity will have effects such as "increasing glucose level" or "increasing body temperature". Even

though activity effect recognition and activity procedure recognition are different, they are used together in many AR systems without notable distinction. The two processes should, however, be separated for a number of reasons.

Firstly, activity effects usually occur after an activity has been performed and with the time gap varying significantly depending on the activity. If activity effects and activity procedures are used together, some effects may not be related to the activity. For example, both "eating a meal" and "having chocolate" increase the glucose level of a person. The effect of chocolate does, however, have a faster effect on glucose level that a standard meal. If an AR system does not distinguish between activity procedure and effect, when a person has chocolate prior to eating a meal, it is difficult to know which activity has resulted in increase in blood glucose level. Therefore, activity procedure and activity effect should be distinguished and their relationship should also be analyzed.

Secondly, one of the aims of AR is to determine if the goal of an activity has been achieved or not. For example, an AR system recognizes an eating activity to determine if the person takes their food correctly. The inherent assumption in this approach is that if a person has food, they are also being provided with the necessary nutrition. Nevertheless, without careful distinction between activity procedure and activity effect, it will be easy to apply only one of these recognition approaches and this will result in a low recognition performance. To illustrate this point further, when an AR system recognizes an activity using activity procedure recognition, it does not ensure that the final goal of the activity is achieved. Nevertheless, simply performing an activity does not guarantee that its goal is achieved given that sometimes people do not complete their activities. For example, if the subject performed some of the actions of the eating activity, however, did not swallow food or ate too little, the goal of eating activity is not achieved.

Similarly, some AR systems recognize an activity through activity effect recognition. In this case the system assumes that effect detection implies that the activity was performed. This assumption is in "fallacy of Inference". For example, the assumption that a rise in blood glucose level may or may not imply that eating activity has been performed given that blood glucose may increase due to several reasons such as chocolate consumption or glucose ingestion. Therefore, both activity effect recognition and activity procedure recognition are required in an AR system.

To address the aforementioned issues, we propose a new activity verification approach. The remainder of the paper is organized as follows. Related work is presented in Section 2. Section 3 introduces our proposed approach. Section 4 outlines an implementation of the proposed approach. Validation and comparisons of experimental results are presented in Section 5.

RELATED WORKS

In this section, we present the details of previous works related to AR and situation recognition, which are related to activity verification.

Activity Recognition

In object-use based AR systems [10][11], activities are modeled based on activity theory proposed and developed by psychologists [8][9]. In [8], activity theory is defined as: "a philosophical and cross-disciplinary framework for studying different forms of human practices as development processes, both individual and social levels interlinked at the same time." The activity theory contains four components (subject, tool, objective and outcome) [8][9]. A subject is a participant of an activity. An objective is a plan or common idea that can be shared for manipulation and transformation by the participants of the activity. A tool is an artifact a subject uses to fulfill an objective. An outcome is another artifact or activity that is the result of the activity. Even though activity theory is well known and is often used in AR research, it has some limitations. Firstly, activity theory does not distinguish between tool and object. These two parameters, however, need to be distinguished given the same item may be used as a tool or object. For example, when a dish is used as a tool for serving, it implies it contains food. On the other hand, if it is an object for a dishwashing activity, it means that it is an empty dish. Secondly, a temporal relationship between activities is difficult to represent in activity theory due to the fact that activity theory focuses on the relationship between components such as subject, activity objective, tool and outcome rather than the relationship between activities.

In some other approaches [11][13], it is assumed that human activities are continuously performed and each activity is a sequential composition of activity components like actions according to a temporal sequence. Several probabilistic models have been used to build an activity model based on this idea. The Hidden Markov Model (HMM) and the Conditional Random Field (CRF) are amongst the most popular modeling techniques. HMM is a probabilistic function of Markov chains and is based on the first order Markov assumption of transition [2]. The basic idea of a Markov chain of order *m* is that the future state depends on the past *m* states. To illustrate, for the first order Markov assumption, the future state depends only on the current state, not on past states [2]. A HMM determines the hidden state sequence that corresponds to the observed sequence and previously determined hidden state. In AR, the hidden state is the human activity and the HMM recognizes activities from both sensor observation and the previous activity according to the first order Markov chain. The HMM is also a generative, directed graph model [2][11]. Being a generative model means that observation data is randomly generated. A directed graph is used to capture the order between states. Hence, a generative and directed graph model in AR implies it should find all possible sequences of observations. Many activities, however, may have a non-deterministic nature in practice, wherein some steps of the activities may be performed in any order. Therefore, enumerating all possible observation cases and orders is difficult for a practical system. Furthermore, missing an observation or an order will cause the HMM to produce errors in the model.

A CRF is a more flexible alternative to the HMM because it relaxes the strict assumptions of the HMM [11]. A CRF solves the problems associated with the HMM by neglecting the order constraint. Similar to the HMM, CRF also determines a hidden state transition from randomly generated observation sequences. Nevertheless, a CRF is a discriminative and an undirected acyclic graph, flexibly capturing any relation between an observation variable and a hidden state [11][13]. Given that CRF does not consider order, it considers only relationships such as state feature function (e.g. relationship between observations over a period of time and activities) and transition feature functions (e.g. relationship between past activities and future activities). Even though CRF removes order constraint from an activity model, it has been shown to be capable to outperform HMM [13].

Situation Recognition

Different researchers have defined situation differently. In addition to the variety, there are some ambiguities relating to the differentiating situation from context given that some definitions of context contain situations within [14] [15]. In many studies [15][16][17][18], context is used to describe or characterize a situation. For example, in [15], a situation is representing, understanding and developing a phenomenon. An action is triggered if a problem is detected by recognizing a set of context information. Therefore, in this paper, situation-awareness is defined as the recognition of problems by context-awareness along with the ability to discover a sequence of actions for solving problems also by means of underlying context-awareness. This relationship between situation, context and action is formalized as follows: situation is defined as a quadruple of context information, history of actions, problem and plan (set of actions). In [16], situation is defined as information that is interesting to the user and related to the goals and the decision tasks for a job. Even though situation is related to the decision, more situation information or improved situation awareness does not imply an improved decision given that highly situation-aware systems may also make poor decisions. Therefore, this paper stipulates that a decision should be separated from the situation-aware system because a computer system has limitations in making a decision compared with human beings who have much experience in the process. In [17], episodes and events are situation and the situation is defined as a triple of desire, actions and set of contexts. A situation is used to recognize human intention of an activity. The recognized human intention is used to help an application service to evolve into a human-intention driven service. In [18], situation is a device-action recorded over a period of time and/or the variation of a set of contexts relevant to the application software on the device. Context is defined as any detectable attribute of a device, its interaction with other devices, or its environment at an instant of time. The work in [18] focused on situation-awareness of device-user action due to the fact that the purpose of situation-awareness was the development of situation-aware applications (e.g. smart classroom) that can adequately capture and analyze combinations of multiple contexts and user's actions over a period of time.

There are also several situation recognition (SR) systems. In [19], the SR system receives as input a stream of time-stamped events and performs recognition of occurring situations. It generates deduced events and triggered actions as output. The situation model of this system is a set of event patterns and a set of constraints. The SR system detects a subset of an events stream and finds the subset of situation patterns in predefined situation models. If a complete match is found, the situation is considered as being recognized. In [22], a SR system provides a categorization of approaches for situation recognition. There are four different categories depending on how knowledge is gathered (data driven or knowledge driven) and how knowledge is represented (observable state space or abstract state space). Based on the four categories, the SR system uses templates for describing situations. A situation template is a pattern capturing the most essential parts of a typical situation.

ACTIVITY, EFFECT AND SITUATION

As mentioned in Section 2, the definition of activity or situation is slightly different and depends on the research area and group. In this section, we define activity, effect, situation and their relationship within the context of the current work. Also a model and a recognition approach of activity, effect and situation will be discussed, respectively.

Definition of Activity, Situation and Effect

We define activity, effect and situation as follows:

Activity = {(Subject, Actions)}
Effect $_{(Activity)}$ = {Situation$_{t1}$ $\rightarrow$Situation$_{t2}$}
Situation = {(TemporalCondition, Activities, Contexts)}
t1, t2: time window of situations, t1 <= t2

Figure 1. Definitions of activity, effect and situation.

Activity. Activity is a collection of actions that are performed by a subject. For example, the eating activity is a set of actions such as serving food, selecting food, scooping food. An action is also composed of components such as operation and object.

Situation. Situation is a state of the pervasive space. In this paper, situation is defined as a set of activities and contexts. Activities are a sequence of performed activities. Contexts are a sequence of detected changes of interesting contexts. In situation, context changes are tracked and evaluated. Situation also represents the history of recognized activities.

Effect. Effect is caused by an activity. The effect changes the situation. Effect recognition identifies which activity causes a situation change from an activity set and context change set.

Figure 2 presents the relationships among activity, effect and situation. An activity causes effects and situations are changed due to the effects. There is a temporal gap between activity and situation because it takes time for an activity to cause effects and effects will happen only when the activity is performed properly. This relationship between activity and situation is presented in the Figure by the gray arrowed line. Activities and situations are recognized from different sensor observations respectively. The recognized activities and situations are utilized to determine the effect of the activities.

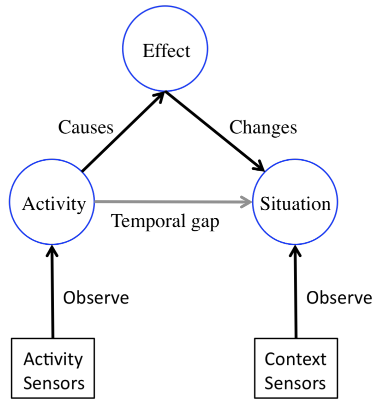

Figure 2. Activity-Effect-Situation relationship.

Table 1 presents a set of examples of activity, effect and situation.

	Activity	Effect	Situation
1	Eating food	Generating glucose	Change of glucose level
2	Making hot tea	Increasing temperature and humidity in kitchen	Change of temperature
3	Brushing teeth	Increase of fluorine in sink tube	Change of the fluorine amount in sink tube

Table 1. Examples of activity, effect and situation.

Before developing the recognition algorithm, it is necessary to establish the models for activity, effect and situation. A well-designed model significantly affects the recognition algorithm and the overall system performance.

Activity Model and Activity Recognition

Our activity model is based on a generic activity framework previously introduced in [4]. Activity components are classified into two categories: primary components and composed components. Primary components are presented in ellipses and composed components are operation, action, activity and meta activity in Figure 3. The detailed description of these components can be found in [4].

To recognize an activity, activity sensors detect events and send the detected sensor events to the AR system. The AR algorithm determines activities from the sensor observation. A multilayer neural network was used as the basis for the AR algorithm [5].

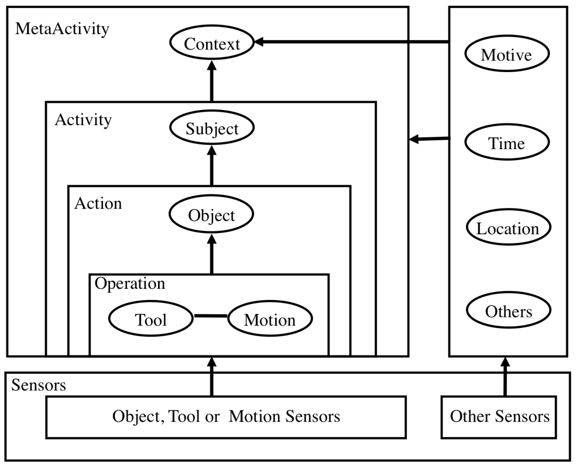

Figure 3. Activity composition framework.

Situation Model and Situation Recognition

A situation is interesting information that is acquired through evaluation of the rule with collected activities and context parameters [15][16][20]. As mentioned in the related work section, the definition of situation is different depending on the purpose of situation awareness. Our requirement for situation recognition is for the purposes of verifying recognized activities as opposed to finding specific problems. Therefore, our situation has a simplified definition that is limited to the state of space where the activities are performed. The ontology of a situation is shown in Figure. 4.

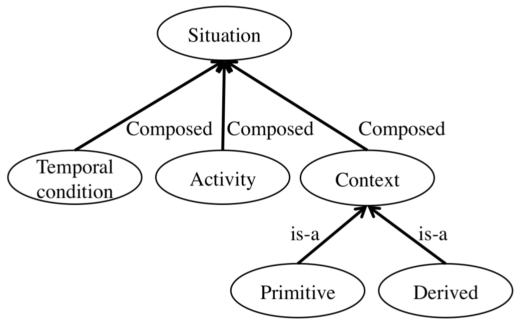

Figure 4. Ontology of situation and situation components.

A situation is composed of three components: temporal condition such as duration, recognized activities and recognized contexts. Context is classified as being primitive context and derived context. Primitive contexts are collected from sensor events such as temperature or humidity. Derived contexts like temperature difference between two moments are acquired through processing primitive contexts. Situation is a phenomenon that has been observed for certain duration.

Also situation is determined through analyzing several contexts together. For example, temperature or humidity in a kitchen is a primitive context. The temperature difference or humidity between two points in time is derived context. If the temperature and humidity in the kitchen are kept high for a period of time, it is a situation such as "stuffy kitchen". Similarly, situation is also determined through observing several activities together. For example, "making hot tea" is a recognized activity whereas "need to turn off range" is a situation because it is determined after tracking several activities. Another example of situation is glucose change in body. A measured glucose level is a context. The difference between glucose levels between two points in time is a derived context. If people have food, the glucose level will exhibit a characteristic change. For example, two hours after having food, the level increases significantly to a level of 140 mmol/L [22]. Following this period it starts to decrease. Observing these changes and deciding if the glucose level is normal according to a situation specification and finding the relationship between an activity and the situation changes is activity effect recognition. A situation can be specified using expressions or conditional rule sentences as shown in Figure 5. Temporal conditions can be represented with a variety of expressions according to a situation. It can be duration of two points in time, or time related to a context event.

```
Class Situation {
      private TemporalCondition tc;
      private ActivityList activities;
      private ContextList contexts;
      private Situation(activities, contexts);

      public boolean tooCold(){
      if (contexts.temperature < 30 and tc > 5 min. and
       activities != null)
         return true;
      } …….
}
Situation kitchen = Situation (activities, contexts)
If (kitchen.tooCold() == true)  …….
```

Figure 5. An example of a situation specification.

For recognizing a situation, an activity and contexts need to be recognized given that they are components of a situation. An activity will be recognized in the AR system as previously mentioned. For context recognition, context sensors observe their related contexts regularly. Based on the observation, situation will be recognized through inference or analyzing. Figure 5 shows an example of a situation. The situation ontology and a kitchen situation are specified using class and instance, respectively.

Activity Effect Model and Effect Recognition

Unlike activities or contexts that are recognized from related sensor observations, effect recognition requires finding changes of situations as shown in Figure 2. Also activity effect takes time to cause situation change, and this temporal gap between activity and situation change varies depending on activity. For instance, sound effect of an activity happens almost immediately when the activity is performed whereas some effect like temperature or glucose level appears after a period of time.

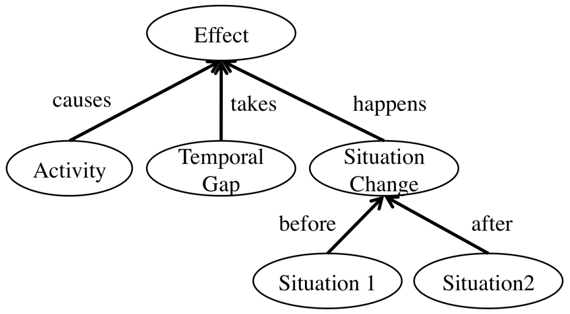

Figure 6. Ontology of activity effect.

According to the temporal gap, T, between an activity and the effects of the activity, situation recognition has different algorithms.

$T = 0$ *or* $T \approx 0$: if there is no gap, situations related to the effect should be ready for detection. For example, a chewing sound is an effect of the eating activity. It happens immediately after a person starts to chew food.

$T > 0$: If there is a gap between activity and effect, when the activity is recognized, effect recognition is triggered and related situations are also collected.

To identify information related to the temporal gap and the potential effect of an activity, it is necessary to take into account domain knowledge from experts. We can also utilize an expert's knowledge to determine the priorities of the effects of an activity. Among several possible activities, some effects serve as stronger evidence compared to others. The strength of the recognized effect will affect the verification result of an activity.

Verification of Recognized Activities

In this Section we explain our proposed approach to utilize activity effect recognition for verifying recognized activities. Activity verification is performed when an AR system recognizes an activity. When an AR system recognizes an activity the Effect Recognition (ER) system begins to recognize related situation changes. The detailed description of the activity verification procedure is presented in Figure 7. In Step1, we prepare a list of target activities that need to be recognized and verified. In Step2, we identify the goal and effect of each activity. If the goal and effect are defined well, detecting the effect subsequent to the goal will minimize errors and maximizes confidence in the goal recognition. The activity goal provides information when deciding which effect to be recognized among several possible effects of an activity. In other words, there can be several effects of an activity. Among the effects, the effects that are more related to the goal of an activity have stronger information.

For example, if we verify a child brushed their teeth really well, the best effect will be how much their mouth will be clean because the goal of “brushing teeth” is “cleaning mouth and teeth”. Nevertheless, sometimes it is difficult to find a way to recognize the effect given issues related to current sensing technology limitations or privacy issues. Under these conditions, we choose alternative effects according to the priority of evidential power of effects. In Step3, we decide which method will be used to find the effect. Step 4 shows two methods to find effect. Firstly, effect will cause situation change. By direct observation of situation change, we can know the effect, which are in hidden states. Secondly, Some effect will be acquired by effect inference. For example, when glucose level increases, body temperature increases also after having food [23]. In Step 5, verification of a recognized activity is performed based on the effect identified in Step 4.

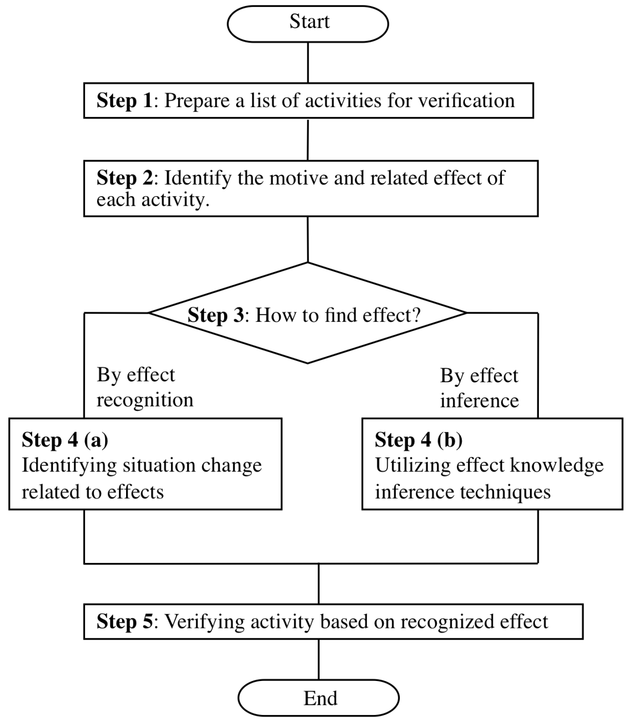
Start
Step 1: Prepare a list of activities for verification
Step 2: Identify the motive and related effect of each activity.
Step 3: How to find effect?
By effect recognition
By effect inference
Step 4 (a) Identifying situation change related to effects
Step 4 (b) Utilizing effect knowledge inference techniques
Step 5: Verifying activity based on recognized effect
End

Figure 7. Flowchart for the proposed process of activity verification.

The result of verification will be one of five possible states as presented in Table 2.

Definitely performed. refers to an activity that is completely performed and the effect of the activity is also satisfied and the effect is related to the goal of an activity.

Performed. refers to an activity that is completely performed and the related effects are recognized, however, the recognized effect is not a strong effect.

Performing. Activity is started, and continuing. Effects are partially recognized. After the activity is finished, this activity will be verified again to determine if this is not completed or performed.

Incomplete. Activity is started, however, stopped without completing it and effects are not recognized.

Definitely not performed. Activity is not recognized and the effect is also not recognized.

Activity Effect Verification	Effect (Situation Change)	
	Activity	Situation
Definitely performed	√	√
Performed	√	√
Performing		√
Incomplete	√	
Definitely not performed		

Table 2. Relationship between activity verification results and effects .

IMPLEMENTATION OF ACTIVITY VERIFICATION SYSTEM

In this section, we describe the structure and API of the proposed activity verification system.

Activity recognition and verification system

The activity verification system (Figure 8) is composed of three recognizers, one activity verifier and a knowledge base in Figure 8.

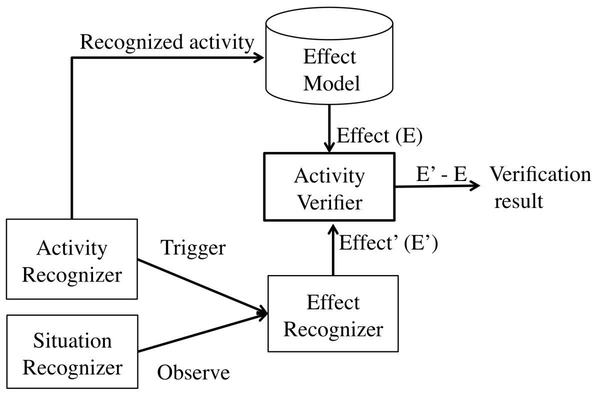

Figure 8. Structure of activity verification system.

Activity recognizer. This component recognizes an activity based on sensor observation. We implemented this using several three-layered neural networks, whose algorithm is described in detail in [4]. In our activity model shown in Figure 3, tool and motion are inputs of operation. Therefore, the neural network will have the same number of input with the sum of the numbers of tools and motions. Also the number of output of the neural network will be equal to the number of operations. There are three neural networks because our activity model has a hierarchy with operation, action and activity.

Situation recognizer. Situation recognizer collects contexts for a certain period of time and finds if there is a matched situation among predefined situations in the knowledgebase.

Effect recognizer. It recognizes the combined change (effect) of both activity and situation.

Activity verifier. It compares the inferred activity effect from the knowledge base (based on the activity recognizer input) against the activity effect from the effect recognizer. According to the comparison result, it returns the verification result.

Knowledge base. This is a repository of models for activities, situations and effects. It also stores predefined situation specifications.

Activity Verification API Design

We designed two interfaces that can have several methods for programmers as shown in Figure 9. The interface methods will have date and time as parameters. "recognizeActivity" will return the activity list performed between the time periods being considered. "verifyActivity" will verify if the activity is really performed between the specific time periods.

```
interface ActivityRecognition {
//recognize activities in start time ~ end time
//s: start date e: end date  format: mm/dd/yy,hh:mm:ss
Vector<Activity> recognizeActivity (Date s, Date e);
//recognize an activity that is currently performing
Vector<Activity> recognizeActivity ();
   …….
}
interface ActivityVerification {
//verify an activity. Return value is integer as follows
//1: definitely performed, 2: performed, 3: performing
//4: not completed, 5: definitely not performed
int verifyActivity(Date s, Date e, String activityName);
int verifyActivity(String activityName);
   …….
}
```

Figure 9. APIs for AR and activity verification.

EXPERIMENTS AND COMPARISONS

We validate our approach experimentally. Our aim is to measure the increase in confidence level of recognized activities achieved by using activity effect recognition.

Experiment setup

We implemented the AR and verification systems in Figure 8 and developed two applications that use the systems.

Application 1: recognizes activities from sensor data. It utilizes only the activity recognition interface in Figure 9.

Application 2: this is the same as Application 1 except that both activity recognition and activity verification interfaces are utilized.

Our experiment scenario is as follows: we first decide target activities and identify related situations and effects of every activity, and then we install sensors accordingly in a smart house. Following this, users who are activity performers and programmers who are activity recognizers play a game. Users will perform target activities to collect sensor data. When they perform the activities, they will decide randomly whether they will perform an actual activity or they will just pretend performing the activity. Sometimes, they can also perform incomplete activities. Whenever the user performs an activity they will annotate the activity performed (actual or pretending). Programmers will execute Application 1 and Application 2 for labeling activities with the collected activity datasets. We will then compare the following performance metrics:

(a) Recognition accuracy of Applications 1 and 2.

(b) Confidence levels of Applications 1 and 2, for each activity.

Following this, we will analyze the relationship between system accuracy and confidence level performance. The detailed steps of the experiment are given below.

Step1. Target Activities

The target activities of our experiments are listed in Table 3.

Activity	Goal	Effect
Making hot drink	Drinking hot tea	Increasing temperature and humidity in kitchen
Using restroom	Using toilet	Changing body weight, body fat, and body water
Eating	Taking food	Generating Glucose

Table 3. Target activities and effects.

Step2. Modeling activities

We created an activity model for target activities based on the example shown in Table 4.

Hierarchies	Model Description
Activity	making hot tea = {(subject, {boiling water, opening tea bottle, pouring water to the cup, adding sugar or cream}}
Action	opening tea bottle = { (holding, tea bottle) } boiling water = {(boiling), (turning on, range)} …….
Operation	boiling = {(kettle, moving on a range)} turning on, range ={range, turning on}…
Object	{tea bottle, sugar, cream}
Tool	{kettle, range, cup}
Motion	{moving, lifting, holding, stirring}

Table 4. An example of "making hot tea" activity model.

Step3. Generating situation specifications

We generated specifications of possible situations based on the situation model.

```
Situation1: IncreasingTemperature
= (duration (30 sec.), Temperature(t1) < Temperature(t2))
Situation2: DecreasingHumidity
= (duration (30 sec.), Humidity(t1) < Humidity(t2))
```

Figure 10. Examples of predefined situations.

Step4. Modeling activity effects

We found possible effects of each activity and built an activity effect knowledgebase.

Effect(Making hot tea) =
(3min, {IncreasingTemperature, IncreasingHumidity})

Effect(Using restroom) =
(2sec, {DecreasingBodyFat, DecreasingWeight})

Effect(Eating) = { (5min, IncreasingGlucose)
(2hours, DecreasingGlucose)}

Figure 11. Examples of effects of target activities

Step5. Sensor Instrumentation and sensor data collection

Table 5 shows the sensors that are used for activity recognition or activity effect recognition.

	Sensors and devices used	Bluetooth devices
Making hot tea	- RFID reader & tags [24] (Bluetooth enabled) - Touch sensor [25] - Temperature/humidity sensor - Light sensor [25]	
Using restroom	- Pressure sensor - Vibration sensor - Body composition monitor (Bluetooth enabled) [26]	
Eating	- RFID reader & tags [24] (Bluetooth enabled) - Glucose meter [27] (Bluetooth enabled)	

Table 5. Sensors for activity and effect recognition

We used several Bluetooth enabled devices and smart phones for collecting the activity and effect dataset. The Bluetooth RFID reader is adequate to collect the activity data set given that it is light enough to wear (75g) and has an appropriate range of 50cm [24]. A glucose meter can be used to detect changes in glucose levels. It may not, however, be appropriate for daily usage given that it has a limitation in that it may not be convenient to carry and to use in situations beyond the home. In such situations, it is necessary either to use another device to measure the glucose levels, or to choose another parameter to be measured. To collect the dataset, two people performed target activities for four days and collected approximately 6600 sensor data readings. During this period they performed totally 51 activities (Making hot tea: 17, Using toilet: 28, Eating: 6). The activities are performed in random ways (concurrently or interleaving) by two people. However, the datasets of the activities are collected separately to avoid gathering mixed, complicated data sets. This is possible because the activities are unrelated to each other. Among 51 activities, 16 activities (Making hot tea: 6, Using toilet: 7, Eating: 3) are actually performed and the remaining 35 (Making hot tea: 11, Using toilet: 21, Eating: 3) are partially completed or pretend activities.

Step6. Application Implementation and execution

The two interfaces depicted in Figure 9 are implemented into classes as shown in Figure 12.

```
Class ActivityRecognizer extends ActivityRecognition {
//implementations …}
Class ActivityVerifier extends ActivityVerification {
//implementations …}
```

Figure 12. Examples of interface implementation

Comparison and Analysis

Two metrics were measured for Applications 1 and 2: the statistical accuracy of the AR system, and the confidence level of each activity.

To measure the performance of our AR approach/system, we measured accuracy, sensitivity, and specificity in terms of true positive, true negative, false positive and false negative [6][7]. The accuracy is the proportion of true results for both true positives and true negatives [7]. Sensitivity is the proportion of true positives and specificity is the proportion of true negatives [6]. These statistical measures are defined below:

$$Accuracy = \frac{tp + tn}{tp + tn + fp + fn} \quad (1)$$

$$Sensitivity = \frac{tp}{tp + fn} \quad (2)$$

$$Specificity = \frac{tn}{tn + fp} \quad (3)$$

True positive (tp): the number of correctly recognized cases for activities that were really performed

True negative (tn): the number of correctly recognized cases that are not performed

False positive (fp): the number of cases in which activities are recognized, but actually were not actually performed

False negative (fn): the number of cases that were not recognized even though they were actually performed

Table 6 shows the number of cases of true positive, true negative, false positive, and false negative for each activity.

Activities	tp	tn	fp	fn
Making hot Tea	4 (3)	8 (11)	3 (0)	2 (3)
Using restroom	7 (3)	2 (15)	19 (6)	0 (4)
Eating	3 (3)	0 (3)	3 (0)	0 (0)
Average	0.89 (0.64)	0.27 (0.9)	0.73 (0.1)	0.11 (0.36)

Table 6. The number of true or false cases of Application 1 and Application 2. The numbers in brackets are data for Application 2.

Based on Table 6, we measured accuracy, sensitivity, and specificity as shown in Table 7.

Activities	Accuracy	Sensitivity	Specificity
Making hot Tea	0.71 (0.82)	0.67 (0.5)	0.73 (1)
Using restroom	0.32 (0.64)	1 (0.43)	0.1 (0.71)
Eating	0.50 (1.0)	1 (1)	0 (1)
Average	0.51 (0.82)	0.89 (0.64)	0.27 (0.9)

Table 7. The comparison of accuracy, sensitivity, and specificity of Application1 and Application 2. The data of Application 2 is shown in brackets.

Table 7 compares the recognition accuracy, sensitivity, and specificity of Applications 1 and 2. Application 1 recognizes an activity using only the AR interface whereas Application 2 performed both AR and activity verification. The accuracy of Application 1 is 51% whereas Application 2 has 82% accuracy. Application 1 shows better sensitivity than Application 2 for recognizing activities that are truly performed. It is because some activities do not produce significant effect when they are actually performed. Nevertheless, Application 2 reduces false positive error tremendously.

We used the AR system in [5] that has greater than 88% accuracy. Nevertheless, Application 1 shows only 51% accuracy in Table 6. The reason for this low performance is due to many activities not being actually performed or not completed in this experiment. Even though activities are not really performed, their datasets are very close to the real activity dataset because people pretend to perform the activities. Therefore, the AR system shows a high false positive error here. If an activity dataset is collected only from performed activities that are actually performed, the measured accuracy performance is higher. As shown in Table 6, true positive performance is 89%.

Figure 13 presents the comparison results. In every activity, Application 2 shows better accuracy.

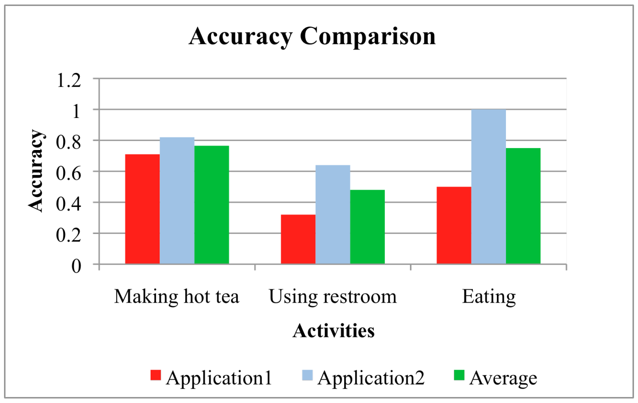

Figure 13. Comparison of accuracy between Application 1 and Application 2

Table 8 presents the results of activity verification. “Definitely performed” or “Performed” are highly confident verification results for activities that are actually performed. Also “Not performed” gives a high confidence when an activity is not actually performed. In Table 8, for “Making hot tea”, 9 out of 16 activities are verified with high confidence. For “Using restroom”, 11 from 28 activities are verified with high confidence. It seems like that the verifier provides too low a confidence value for this activity. Nevertheless, the verifier found 17 mistakes made by the AR system. All “Eating” activities are verified with high confidence.

Verification results	Making hot tea	Using restroom	Eating
Definitely performed	2	2	3
Performed	1	7	0
Performing	4	0	0
Incomplete	4	17	0
Not performed	6	2	3

Table 8. Activity verification results.

The verification of “Eating” through glucose meter results in high confidence for both performed activities and activities that are not performed. In the other two activities, verification results do not have a high level of confidence for performed activities. Nevertheless, they return a high confidence level for “not performed activities”. The difference between verification among activities is because of the effect we chose for verification. When we analyzed the result, we observed that “making hot tea” affects humidity more than temperature. Also, body composition values are sensitively influenced by other factors such as person or floor status. Glucose meter shows consistent results. Therefore, the appropriate choice of which effect to measure is important for verifying the activities.

CONCLUSION

Accurate AR is very important for many practical or safety-critical applications. Nevertheless, current approaches to AR do not offer decisive information for determining whether an activity has or has not actually been performed.

To solve this problem, we introduced a method that can verify a recognized activity. In our approach, we distinguished between activity procedure recognition and activity effect recognition. Activity verification is performed by recognizing the effect of an activity. Our experiment results demonstrated that activity verification increases accuracy consistently for each activity considered.

REFERENCES

1. Helal, S., King, J., Zabadani, H., Kaddourah, Y. The Gator Tech Smart House: An Assistive Environment for Successful Aging. In *Advanced Intelligent Environments*, H. Hagrass, Editor, Springer Verlag (2008)
2. Mann, W., Helal, S.: Smart Technology A Bright Future for Independent Living. In *The Society for Certified Senior Advisors Journal*, vol. 21, pp. 15-20 (2003)
3. Kim, E.J., Helal, S., Cook, D. Human Activity Recognition and Pattern Discovery. In *IEEE Pervasive Computing* vol. 9, n. 1, pp. 48-52 (2010)
4. Kim, E.J., Helal, S.: Revisiting Human Activity Frameworks In 2nd International ICST Conference on Wireless Sensor Network Systems and Software (2010).
5. Kim, E.J., Practical and Robust Activity Modeling and Recognition In 8th international conference on wearable micro and nano technologies for personalized health (2011)
6. Altman, D.G., Bland J.M. 1994. Statistics notes: diagnostic tests 1: sensitivity and specificity. Br. Med. J. 308:1552, (1994)
7. Alvarez, S. A., "An exact analytical relation among recall, precision, and classification accuracy in information retrieval," in Technical Report BCCS-02-01, Computer Science Dept. Boston College, 2002.
8. Kuutti, K. Activity theory as a potential framework for human-computer interaction research. In B.A Nardi (eds.), *Context and consciousness: Activity theory and human-computer interaction*. Cambridge, MA MIT Press (1996)
9. Davydov, V. V., Zinchenko, V. P., Talyzina, N. F. The Problem of Activity. In *the Works of A. N.Leontjev*, Soviet Psychology, vol. 21, n. 4, pp. 31-42 (1983)
10. Constantine, L., Human Activity Modeling: Toward A Pragmatic Integration of Activity Theory and Usage-Centered Design. In *Human-Centered Software Engineering*, p24-50 (2009).
11. Pentney, W., et al. Sensor-Based Understanding of Daily Life via Large-Scale Use of Common Sense. In *Proc. AAAI '06*, Boston, MA, USA (2006)
12. Wallach. H. M., Conditional random fields: An introduction, Technical Report MS-CIS-04-21, University of Pennsylvania CIS. (2004)
13. Kasteren, T. Noulas, A. Englebienne, G., Krose, B. Accurate Activity Recognition in a Home Setting. In *Proc. of the Tenth International Conference on Ubiquitous Computing*, Korea, pp. 1-9 (2008)
14. Dey AK, Abowd GD. Towards a better understanding of context and context-awareness. CHI'2000 Workshop on the What, Who, Where, When, and How of Context-Awareness. (2000)
15. Kim, M. and Kim, M. A Formal Definition of Situation towards Situation-Aware Computing. In Proc WSKS (1). 553-563. (2009)
16. Endsley, M.R., Theoretical underpinnings of situation awareness: a critical review. In: Endsley, M.R., Garland, D.J. (Eds.), Situation Awareness Analysis and Measurement. Lawrence Erlbaum, Mahwah, NJ. (2000)
17. Chang, C.K., Jiang, H., Ming, H., and Oyama, K. Situ: A Situation-Theoretic Approach to Context-Aware Service Evolution. In Proc of IEEE T. Services Computing. 2009, 261-275.
18. Yau, S.S., Huang, D., Gong, H., Seth, S., Development and runtime support for situation-aware application software in ubiquitous computing environments. In 28th Annual International Computer Software and Application Conference (COMPSAC), Hong Kong 452–457 (2004)
19. Dousson, C., Gaborit, P., and Ghallab, M. Situation recognition: Representation and algorithms. In Proc. the Thirteenth International Joint Conference on Artificial Intelligence (IJCAI-93). Chambery, France, pp. 166–172. (1993)
20. Dahlbom, A., Niklasson, L., Falkman, G. and Loutfi, A., Towards template-based situation recognition. In Intelligent Sensing, Situation Management, Impact Assessment, and Cyber-Sensing, vol. 7352. SPIE. (2009)
21. J.M. Nigro and M. Rombaut. Idres: A Rule-Based Systemfor Driving Situation Recognition with Uncertainty Management, Information Fusion, Vol 4. (2003)
22. National Diabetes Information Clearinghouse (NDIC) http://diabetes.niddk.nih.gov/dm/pubs/hypoglycemia/
23. A service of the U.S. National Library of Medicine From the National Institutes of Health (NIH) http://www.nlm.nih.gov/medlineplus/ency/article/003400.htm
24. Ceyon, Korea, http://www.ceyon.co.kr/
25. Phidget, USA, http://www.phidget.com
26. Tanita, Japan, http://www.tanita.com/en/bc590bt/
27. Myglucohealth, USA, www.myglucohealth.net