Universidad de Granada

Departamento de Ciencias de la Computación
e Inteligencia Artificial

Recognizing human behaviour using information from Smart Environments

Tesis Doctoral

María Ros Izquierdo

Granada, Mayo de 2012

Editor: Editorial de la Universidad de Granada
Autor: María Ros Izquierdo
D.L.: GR 205-2013
ISBN: 978-84-9028-284-7

Universidad de Granada

Departamento de Ciencias de la Computación
e Inteligencia Artificial

MEMORIA QUE PRESENTA

María Ros Izquierdo

PARA OPTAR AL GRADO DE DOCTOR EN INFORMÁTICA

Mayo de 2012

DIRECTORES

Dr. Miguel Delgado Calvo-Flores y Dra. M. Amparo Vila Miranda

Departamento de Ciencias de la Computación
e Inteligencia Artificial

La memoria titulada "*Recognizing human activity using information from Smart Environments* ", que presenta Dña. María Ros Izquierdo para optar al grado de doctor, ha sido realizada dentro del Máster Oficial de Doctorado "*Soft Computing y Sistemas Inteligentes*" y del Programa Oficial de Posgrado "*Ciencias de la Computación y Tecnología Informática*" del Departamento de Ciencias de la Computación e Inteligencia Artificial de la Universidad de Granada bajo la dirección de los doctores D. Miguel Delgado Calvo-Flores y Dña M. Amparo Vila Miranda.

Granada, Mayo de 2012

El Doctorando

Los Directores

Fdo: María Ros Izquierdo

Fdo: Dr. Miguel Delgado Calvo-Flores

Fdo: Dra. M. Amparo Vila Miranda

Esta tésis doctoral ha sido subvencionada bajo el Programa de Becas de Formació de Profesorado Universitario, en la Resolución del 5 de Julio de 2008, bajo la referencia AP2007-03578. También ha sido parcialmente financiada con los fondos asociados al proyecto TIN2009-14538-C02-01, de la convocatoria de Proyectos del Plan Nacional, Convocatoria 2009, del Ministerio de Ciencia e Innovación.

Agradecimientos

Aunque parezca increíble, esta parte siendo una de las más difíciles de escribir. Tengo tanta gente a la que agradecer su ayuda, su apoyo y su amistad durante estos años que, sinceramente, no sé por donde empezar.

Es lógico comenzar agradeciendo a esas maravillosas personas que me dieron la oportunidad de disfrutar de estos cuatro años. Gracias Miguel y Amparo. Sin vosotros nunca podría haber realizado este camino. Gracias por tener siempre las palabras adecuadas, gracias por decirme las verdades cuando las necesitaba, gracias por ser guías pero nunca impositores, etc. Gracias, muchas gracias, por todo.

Asimismo, también quiero agradecer a todo el departamento de Ciencias de la Computación e Inteligencia Artificial, mi departamento. En él, he encontrado personas a las que admirar, de las que aprender y personas que me marcarán para el resto de mi vida.

A título personal, me gustaría agradecer a Nacho, que me enseñó que sigue habiendo profesores para los que sus alumnos siguen siendo lo más importante (por mucho que él rechiste); a María José, por mostrarme otra parte de la investigación y por permitirme formar parte de ella; y a Dani, quien me dio uno de los mejores consejos de mi vida, enseñándome a asistir a los congresos.

No me puedo olvidar de Rosa, esa persona que es capaz de apagar cualquier fuego por sofocante que pueda parecer. Muchas gracias por hacernos la vida tan fácil. En este mismo sentido, quería agradecer a Miguel, nuestro secretario, una de las personas más eficientes y amables que he tenido el gusto de conocer. Es fácil lidiar con "papeles", cuando son ellos los que te los solucionan.

En el departamento he encontrado grandes amigos, a lo que siempre tendré que agradecer que me permitiesen formar parte de sus vidas. A Javi y Mariola, esa perfecta pareja que me ha ayudado en lo inimaginable; a Aída, mi amiga desde los inicios; a Manolo, compañero de fatigas y algunos enfados, un amigo que me enseñó otra forma de trabajar y de ver las cosas; a Nacho y a Migue compañeros y amigos desde hace años (y espero por muchos más); a Fernando Bobillo, mi incansable compañero de viajes; a todos mis compañeros de la quinta planta, las orquídeas y el CITIC, muchas gracias.

A mis compañeros de ATC, amigos ya en su mayoría, quería agradecerles los buenos ratos que hemos pasado juntos; pero sobre todo, el permitirme ser una "acoplada más": esos cafés de escalera, esas cenas de becarios, etc.

Por otra parte, no puedo dejar pasar la oportunidad de agradecer a Jim y Ruth, mi familia americana. Ellos hicieron que cuatro meses en el centro de nowhere, fueran una experiencia inolvidable. Me cuidaron, mimaron y ayudaron en todo y siempre habrá un gran hueco en mi corazón para ellos. Acordarme también en esta ocasión de Derek, que me recordó lo agradable que es trabajar con alguien con el que poder discutir ideas. Durante mi estancia recibí muchos regalos (incluso mi nombre pirata), pero sobre todo gané grandes amigos que me ayudan cada día.

Agradecer también a Hani su ayuda y apoyo durante mi estancia en Essex. De esa estancia, me quedan grandes amigos que siempre estarán conmigo: Georgia, Javier, James, Nur, Bob, Eduard, Dario, etc. Pero sobre todo agradecer a mis dos salvadoras: Aysenur y Summer. Ellas hicieron mi estancia fácil, agradable y sobre todo, memorable. Desde el punto de vista profesional, fue una grandísima experiencia; pero en ningún caso comparable a la experiencia personal.

Sin duda también quiero agradecer a *mis niñas* toda la ayuda, el apoyo y las regañinas que me han dado durante estos años. Estudiar la carrera nos unió y cada día doy gracias por ello. Ellas han sabido apoyarme, y sobre todo, aguantarme durante estos nueve años. Por supuesto, también agradecer a mi *siempre compañera de piso*. Ella, casi más que nadie, me ha soportado durante todos estos años. Gracias por recorrer todo este camino conmigo. Tampoco puedo olvidarme de *mis musas*, ellas me ayudaron a cambiar y evolucionar como persona y eso siempre quedará conmigo.

Además, quería agradecer a toda mi familia su apoyo. A mi "*bueli*" por reírse de mi por estar siempre con mi "amigo", a mi hermano por reñirme y apoyarme a su manera, a mi prima por entender el término "en otra ocasión", a mi primo por ser siempre un referente al que admirar, a mis tíos, por estar siempre ahí.

Como último punto, pero casi el más importante, quiero agradecer con todo mi corazón a mis padres todo su apoyo, ayuda y esfuerzo de estos años. Sin ellos nunca podría haber superado ninguno de los retos que me he encontrado en la vida. Ellos son las personas que han hecho posible que haya llegado a este punto y nunca podré agradecerles lo suficiente los esfuerzos que ha realizado durante todos estos años. Por ellos y para ellos, es esta tesis.

Table of Contents

I. PhD dissertation **1**

1. Introduction . . . 1
   - 1.1. Justification . . . 1
   - 1.2. Background . . . 2
     - 1.2.1. Human Activity Learning and Recognition . . . 3
     - 1.2.2. Adapting the learnt behaviour . . . 4
     - 1.2.3. Smart homes . . . 4
     - 1.2.4. Background Review . . . 5
   - 1.3. Objectives . . . 7
   - 1.4. Document organization . . . 8
2. Joint Discussion of Results . . . 11
   - 2.1. Detecting and recognizing behaviour patterns from *Stream Data*. Apriori approach. . . . 12
   - 2.2. Processing temporal information to manage the uncertainty of behaviour. . . . 14
   - 2.3. Adapting the learnt behaviour patterns on time. A Learning Automata based approach . . . 16
     - 2.3.1. A behaviour formal representation. The behaviour model . . . 17
     - 2.3.2. Learning a behaviour model from user activities . . . 19
     - 2.3.3. Recognizing and adapting human behaviour model. . . . 21
     - 2.3.4. Experimental process in a Real Ambient Intelligence Environment. *iSpace* . . . 24
   - 2.4. Understanding change in human behaviour. A summarization tool . . . 27
   - 2.5. Generalizing the behaviours. Routine detection . . . 30
3. Concluding Remarks . . . 35
4. Future Work . . . 37

II. Publications **39**

1. Detecting and recognizing behaviour patterns from *Stream Data*. Apriori approach. . . 39

2. Processing temporal information to manage the uncertainty of behaviour 49
3. Adapting the learnt behaviour patterns on time. A Learning Automata based approach 65
3.1. Detection of abnormal human activities by means of Learning Automata . . . 65
3.2. Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows 81
3.3. A Fuzzy Logic Approach for Learning Daily Human Activities in an Ambient Intelligent Environment 99
4. Understanding change in human behavior. A summarization tool 109
4.1. Similarity measure for anomaly detection and comparing human behaviors . 109
4.2. Linguistic summarization of long-term trends for understanding change in human behavior 141

Bibliografía **153**

Table of Figures

1. Example. An elderly woman goes to bed 11
2. Frequent Itemset based-model diagram 14
3. Fuzzy Temporal Window over an ODB 15
4. "Go to sleep" behaviour model 18
5. Learning scheme in a multi-teacher environment. 21
6. System architecture with behaviour models in parallel 24
7. *iSpace* rooms 26
8. Example. Output of Wake up behaviour evolving for a year. Annotations show what happened in this year. 28
9. Fuzzy sets for the angle of the estimated linear equation. 29
10. Fuzzy sets for the mse in the user specified time window. 30
11. Type-2 Fuzzy system structure 32

Part I. PhD dissertation

1. Introduction

1.1. Justification

In recent years, successful ageing has become one of the most important problems in our society. According with the last study of National Statistical Institute (INE), the number of Spanish people over 65 years old has grown around 20,8 % from 1998 to 2011. Besides, in 2049 it would be doubled representing more than 30 % of the population [2]. To handle this problem, more resources are required to be able to maintain quality and style of life. However, the number of existing nursing home is not such high. Besides, most of elders do not want to move out, leave all their life behind and loose part of their independence.

For this reason, in medicine field has emerged a new philosophy: ageing in place. According with [1], it is defined as the ability of living in one's own home and community safely, independently, and comfortably, regardless of age, income, or ability level. This alternative arises as the perfect alternative to fulfil everyone's needs.

Technologies, and specially, computer sciences, can support the concept of ageing in place. For example, using monitoring systems, vulnerabilities of elderly people that live alone would be reduced. Detection their activities could help to react to dangerous situations quicker. Adverse events generally occur over short time periods (seconds or minutes), so, the reaction time is critical for those situations. In general, elderly people strongly reject to be supervised continuously. They like to keep their privacy, independence and freedom. At this point, the Ambient Intelligence concept can help. Under this paradigm, devices are embedded at the environment, and, therefore they are transparent for the users. People do not change their routines; simply, they continue doing daily activities as usual, while the system is working. In addition, the quality of their live could experience significant enhancements due to the increased support received from the environment: environmental control, communications, monitoring, etc. Such systems could also alert caregivers or family about any significant changes in the resident's behaviour, health, etc..

The potential means of this new tendency was taken notice by the European Commission. In 2002, IST Programme Advisory Group (ISTAG) vision statements for *Framework Programme 5* defines the concept of Ambient Intelligence (AmI) as new paradigm where users are enhanced through an environment that is aware of their presence and context; as well as, is able to be sensitive, adaptive and responsive the users' needs [22]. Thus, AmI environments are capable of anticipating user needs and behaviour, learning users' personal requirements and preferences, and

interacting with people in a user-friendly way [57]. AmI has potential applications in many areas of life [57]: home, office, transport, industry, entertainment, recommender systems, safety systems, ehealth, etc.

Ambient Intelligence builds on three recent key technologies: Ubiquitous Computing, Ubiquitous Communications and Intelligent User Interfaces [22]. Users are surrounded by intelligent interfaces, embedded in everyday objects such as furniture, clothes or mobile devices; which allow the interaction with the environment. Additionally, according with Computing Ubiquitous principles [65], users do not notice those interfaces or devices are around them. Devices are blended with the environment and therefore, disappear from human perception, making their use transparent and unobtrusive. In turn, the interactions are supported by developed networking technologies everywhere [22].

Computing favour applications were to acquire the context is essential to develop a useful, correct and reliable application. Context is characterized by detecting the situation provided by sensors, understanding their global common meaning and acting consequently (i.e., triggering actions). Therefore, using context-awareness in applications supports the detection of everyday living activities [57].

Focusing on understanding the context, an outstanding challenge is to exploit sensory information to recognize human activities. Systems usually detect which actions are correct and which are not, learn that differences and recognize user current activities to detect any wrong occurrences.

Our goal is to develop tools to solve the problem of ageing in place, increasing the autonomy and supporting the independent living of disabled and elderly people. They are capable of managing on one's own and being safe. With that aim in mind, we propose a system to recognize human behaviour in Smart Environments. Concretely, we model human activities in function of information collected from specific known-environments. That information is used to develop algorithms and structures that enable us to follow current user activities and detect anomaly or abnormal behaviour. We focus on the fact that a specific type of behaviour is characterized by a set of common actions, which we assume that they are the most frequent actions performed during the behaviour execution time. However, as we are dealing with time information, we ascertain that behaviour is not precise or constant, so that, we develop methods to manage the uncertainty and use it to improve the quality of the knowledge. Additionally, we generalize the problem from two points of view: on one hand, we adapt the knowledge to new circumstances, for example, user starts waking up later or besides making coffee, s/he starts having an orange juice every morning. These adaptations provide a tendency, which is studied to extract understandable information. On the other hand, we generalize the concept of behaviour to manage sets of behaviours, named routines.

1.2. Background

The rapid growth of the elderly population is a complex problem in our society. Life expectancy has improved, and with it, the resources required to maintain quality and style of life. Emerging applications combining *technology* and *medicine* seems to be a possible solutions. In concrete, applications including AmI concepts emerge as the perfect solution for these problems, supporting the ability of elders to maintain their independence and style of life in their *own homes*. Next, we present a brief summary of the most remarkable projects and works that would support our work.

1.2.1. Human Activity Learning and Recognition

As stated previously, modelling, recognition and prediction of Human activities has emerged as a key research issue for the successful realization of intelligent environments [14], thanks to its support to many different fields of study such as *medicine* [44], *human-computer interaction* [33], or *sociology* [15]. An Activity recognition system requires to deal with high-dimensional, multi-modal streams of data that are characterized by a large variability (e.g. due to changes in the user's behaviour or as a result of noise). Therefore, processing and managing that variability is one of the aim of any recognition proposal.

In Activity Recognition field, Activities of Daily Living (ADL)[37] [43] are becoming more and more important [21] due to its relation with elders and its security [42][46]. An ADL is a sequence of atomic actions within user's daily routine, such as toileting, making a meal or leaving home. In [43],a new paradigm for ADL inference is defined. It is based on radio-frequency-identification (RFID) technology to identify objects used by people. Besides, the authors propose a probabilistic inference engine for the recognition of complex activities. Recent projects generalize the ADL concept to any daily normal activities, such as waking up, having breakfast or leaving home[47].

Probabilistic and uncertainty models are the most common choice in the literature to model an ADL, represented as probabilistic sequences of objects touched by users [43]. Probabilistic methods offer tools both to model different types of behaviour performances and to handle different scenarios. Benefits and drawbacks of using probabilistic and statistical methods are studied in [34].

Several authors present the use of Hidden Markov Models (HMMs) as the best alternative to solve the human activity recognition [9][39][67]. However, HMM are stationary and cannot be applied over dynamic environments. Some authors propose alternative to the simple HMM, such as, Markov Decision Process [4], hybrid models of HMM and Linear Discriminant Analysis[64], Multiple Behavioural Hidden Markov Models [41].

Other stochastic methods have been proposed to recognize daily activities. Projects, such as [63][61], use Naive Bayes classifiers as tool to distinguish the behaviours, obtaining good accuracy results for abnormal behaviour detection. Processing large amounts of data is the issue to beat for these classifiers. On the other hand, Lee et al [35] combine Bayesian networks with fuzzy logic, with the aim of ignoring users' unintended behaviour.

Fuzzy logic is also a new applied technology to study normal and abnormal behaviours in an AmI. Acampora et al. [4] propose an AmI Fuzzy Computing system based on multi-agents, controlled by fuzzy control techniques. Fuzzy Logic improves the recognition process and enriches the knowledge about the user and environment. Other example of combination between multi-agent architecture and fuzzy logic is presented by Doctor et al. in [20]. They propose an unsupervised, data-driven, fuzzy, technique for extracting the fuzzy membership functions and rules that represent and model the users. The user's learnt behaviours can then be adapted online in a lifelong mode to satisfy the different user and system objectives. This system was tested in an experimental testbed for intelligent environments called the Essex intelligent Dormitory (iDorm).

With the recognition aim in mind, an explicit or implicit reference to where and when the meaningful events occur is necessary [16]. Activities recognition system has to be aware of actions performed by users in any moment and during certain period of time. Some authors focus their work on processing temporal information as support to extract correct patterns. For instance, Jakkula et al. [30][29] propose a method to identify the temporal relationships among behaviour's actions inside CASAS project[47]. They assure those temporal relationships are key in prediction and anomaly detection for ambient environments [31]. Augusto and Nugent [8] also use temporal reasoning together with a rule-based system to detect hazardous situations, take the environment

control and change it to a safe state.

1.2.2. Adapting the learnt behaviour

A specific behaviour model cannot be learnt just once, since users' routines might change with time: the user can change his/her preferences, new actions could appear or, even, the learnt knowledge needs to be refined. These require to perform an adaptive process to maintain the learnt knowledge reliability. Moreover, adapting a behaviour model could help the caregiver to understand common behaviours and to detect unhealthy habits. This trend is a new research issue emerged from the Activity Recognition Field. For this reason, not many proposals could be found in the literature.

Adaptation is a controversial field in Activity Recognition. According with [11] [12], adaptability implies the use of data analysis techniques to allow the system to adjust its performance to the behavioural patterns of the elder. They propose a system called *Necessity*, which incorporates a complete abstract behaviour model, which is complete with a classifier to provide it with the required knowledge. Later, they discuss about how to adapt the model to new knowledge and the convenience of adapting the classifier's parameters.

An adaptive agent architecture was proposed by Hagras and Doctor in [26]. This architecture is based on type-2 FLSs . It was shown that type-2 FLSs are able to handle the different sources of uncertainty and imprecision encountered in a pervasive space such as the iDorm to give a very good response. Different experiments were carried out by different users in the iDorm over a period of one year in which the type-2 agent has learnt and adapted to the user behaviour and the long term environmental variations while handling the short and long term uncertainties to give a very good performance that outperforms the type-1 fuzzy agents while using smaller rule bases. Using Fuzzy Logic, Acampora et al. [4] also propose a fuzzy computing system based on multi-agents that update the membership and rules regarding users' daily activities.

Mavhome project [18] propose a Markov Decision Process (MDP) associated with a reward/penalty system to refine its learnt knowledge. Negative reward is received when the user undoes the automatic action performed by the user [17]. In turn, Neural networks is the tool chosen by Mozer et al. [40] in the Neural Network House. They aim to develop an adaptive control system that controls lifestyle and energy consumption of its inhabitants. They base their approach in a feed-forward neural network to anticipate and predict the occupancy of each zone in the house.

1.2.3. Smart homes

Much of this work is focused on Smart Environment research, such as, Smart homes. A single environment is outfitted with sensors and designed to improve the experience of the resident in the environment[28]. A Smart Home [18] is mainly a house that has been equipped with special structured wiring to provide occupants with different applications to control their own environment. For instance, remotely controlling, interacting with any devices, or monitoring their activities and offers guidelines to improve them, etc..

In the literature, there are many works focus on hardware aspects of those systems. It is clear that these systems depend on the technologies, any advances could favour their development, integrity, reliability and robustness. Sensors and actuators are the main general components in the network, and provide the sensor information that later will be used to improve the user life. However, a smart home also depends on the quality of the systems implemented on them. Those applications

will allow the user to perform all the applications that the smart homes are designed for.

From Artificial Intelligence point of view, a Smart Home environment is a home dotted with difference devices to monitor users' activities and movements. It is capable of detecting dangerous situations, such us, unlocked doors or turned-on cook without supervision and take some decisions to solve it [57]. Smart Home equipment lets developers monitor the activities, and therefore, it should include a set of sensors and actuators.

Smart Home can be especially useful for elderly and disabled persons who wish to live independently. Smart Homes concept envisage systems where users can maintain their behaviours as normal, adding the required supervision thanks to monitorization of residents' or users' activities. That monitorization is matched with some previous information collected by the user. Smart Homes may help or guide residents to keep safely: depending on the former information, it may give advices regarding safety, health, medication, etc.. One such project is the *TigerPlace* facility [46]. This is a large interdisciplinary collaboration between electrical and computer engineers, gerontological nurses, social workers, physical therapist and others at the University of Missouri. They include technology to assist residents with .ageing in place"[1] as independently as possible. Their research is focused on the study of the data acquired from a huge number of motion, pressure, and sound sensors or IR-cameras. This information is used to extract conclusions about the user: heart pulse, breathing, fall detection, etc.

Another example is the Aware Home at the Georgia Institute of Technology [3] [32], focused on Context-Aware Computing. This home is composed by two identical but independent living spaces, each one with two bathrooms, two bedrooms, one office, one kitchen, one dining room, one living room and a laundry room. The project goal is to perform a multidisciplinary exploration of the new technologies to be adapted to a home environment. Among the technologies used in the environment, the most important ones were the ultrasonic sensors for human position tracking, the floor sensors and vision techniques for recognizing users' activities. The application uses sensor information to learn about the resident behaviours and adapt its services to her opinions and preferences. Other examples of Smart Homes are MavHome [18], the CASAS [47], the Gator Tech Smart House [27], the Neural Network House [40] and the Intelligent Home [36]. Further information regarding this area is summarized in [21].

1.2.4. Background Review

Smart home and recognition human applications are in general completely biased by the environment where they will be installed and by the set of sensors used as inputs. They are developed in function of the number and type of actuators or sensors the AmIE contains. These influences make extremely difficult to perform a real global and generic system able to be transferred to different environments and keep working.

Focusing on the different techniques presented in the literature to recognize human activities indicates that there are different varieties that could be used, depending on the needs of users and applications. However, in most of them we concluded that the selected behaviour representation is just the outcome produced by the method proposed. The behaviour model has not any influence on the rest of the systems.

This union is very common in project using *Neural Networks*. In spite of NNs are capable of managing complex data and models, they are a complete black box. The behaviour representation is gathered in that, represented by the own structure of a NN. That means that is no easy to understand their internal structure, in order to obtain explanation of the decisions or read the

knowledge learnt about the user. Other typical example is a *Rule based system*. The behaviour representation is a set of rules that need to relate actions across several rules. In general, rules relate a specific situation to an action easily, however, they are not good when is required relating actions across several rules, giving no sense of sequence to the actions. Besides, no relating actions among them will generate an excessive number of generated rules with very little meaning.

Other controversial point is the use of temporal information, as a tool to learn the human behaviour. Some proposals start managing temporal information to define order relationship between actions learning the time passed between them. From our point of view, learning the temporal of each action transforms the system in inflexible structure, where the user cannot feel independently.

In contrast to those drawbacks, we proposed a method that

- **is completely independent on the environment**. Our focus is the actions performed by the user, and not the network or sensors used to identify those actions. Therefore, we propose to abstract the concept of action, being aware of their meaning and not of the sensors used to monitorize it. This idea is supported by the fact that there is not only one sensor that could identify each action. For instance, when a user opens the door, what really matters is that s/he is opening the door; because that has a meaning inside our system and not if that action was detected through a door latch or a door sensor. In addition, this abstraction makes easier to transfer our system to different Ambient Intelligent Environments.

- **focuses its attention on the behaviour representation**. From our point of view, the behaviour concept is the important part of the system, and requires being the main developing point. Despite our first approach did not get completely the separation, we finally come up with a behaviour representation independent on the method used to learn, or recognize it. We propose a method based on probability distributions and in levels, to deal with the order relationships. Consequently, the learning and recognizing procedures are independent from the computer behaviour representation, which allows the use of different learning techniques. Other advantages of this separation are that the outcome system is generally more robust and reliable in cases of behaviour changes.

- **manages temporal information at behaviour level**. We introduce temporal processing, learning when approximately the behaviour will be performed. The temporal information will define which actions may belong to the behaviour with a specific membership degree, to improve the quality of the knowledge. The learning of order relationships is apart.

- **provides understandable explanation about what is happening with the user**. As aforementioned, those systems should be able to take decisions, and also, to provide explanations about that decision. We aim to make our system more understandable using a linguistic summarization method over tendency followed by the behaviour model during the time and sending alarms to whom may concerned in common language.

1.3. Objectives

The main goal of this thesis is to develop a system to model, recognize and adapt human behaviours, as well as, to send alarms when some abnormal actions within online user activities are detected. To achieve this aim, we set the following objectives:

1. **To develop a model to learn relevant patterns from sensor information**

   Nowadays, we are surrounded by sensors. They collect every change or event that happens at the environment. For example, we could know the state of a specific light: if it is on or off, the brightness level, etc.. Collection of all those events means a huge amount of information, where some groups of events have a common meaning and are repeated every certain time. Our objective consists of developing an algorithm to extract automatically those relevant patterns with common final meaning.

2. **To manage the time to deal with the uncertainty of human behaviour performances**

   In general, each person follows a schedule: s/he starts working at a specific time, and therefore, wakes up earlier, etc.. However, some days the person could wake up later because of s/he did not setting up the alarm, or could be late at work due to traffic jam. Our goal is to develop a tool to handle time information in order to manage this uncertainty.

3. **To design a human behaviour representation**

   Generally speaking, a specific type *Human Behaviour* challenges a specific issue. However, to achieve it, user needs to perform a certain sequence of actions with a known order relationship among actions. We will design a model to store those restrictions: set of actions and order relationship between them.

4. **To design an algorithm to recognize the learnt human behaviour patterns**

   Once behaviour model will be defined, our next aim will consist of developing an algorithm to online recognize user activities. Our proposal will determine if a behaviour performance is correct, or if there would have been some anomalies: some missed actions, delayed actions, etc.

5. **To send warning or alarms when the behaviour is incorrect**

   A detected incorrect behaviour performance means that something wrong has happened. In those cases, the system will warn about the situation. The alarms or warning could be sent to different place, depending on the resident situation: to themselves, to caregivers, to their family, etc..

6. **To evolve the model to an adaptive system**

   As indicated before, our system will learn from the user activity directly. However, users are not precise performing activities in the same way every day, station or year. Our model should analyse those changes in order to maintain the integrity and reliability on the represented knowledge.

7. **To study the evolution of human behaviour performances during long periods of time**

   When an activity is changing, it could be interesting to know how they evolve during some specific time, in order to find out reasons of those changes. We will study how similar two performances are, comparing to each other.

8. **To develop a prototype based on mobile technologies mobile**

   We will design and implement a prototype of our system to test the feasibility of the proposal. Our prototype will be independent of the chosen input sensor, as well as, it will use the mobile technologies to send alarms.

9. **To generalize the model to detect sequence of behaviour performances as a whole**

   Equivalently a sequence of actions could be understand as group, we could organize behaviour performances into ordered sequences with a common final end. Those sequences could be assumed as routines.

1.4. Document organization

This dissertation is organized in two parts. The first part introduces the objective and motivations of this work, and also presents a brief summary of it, emphasizing the most relevant obtained results; whereas the second part presents the publication that supports our contributions. We detail more deeply the contents of each section afterwards.

Part I. PhD Dissertation

Part I introduces this PhD work, presenting the main issues we focused on and a summary of the most important works in the field. Additionally, we summarize the work proposed in this dissertation, which is supported by the publications included in Part II.

Section 1 introduces the objectives of our work. We motivate the problem of recognizing human behaviour in Smart Environments: population ageing, advanced technologies, AmI application developments, etc. Furthermore, we have defined our framework and specified the general goals of this work.

Section 2 summarizes all work performed in this thesis. Briefly, our objective is to recognize human activities in Smart Environments. After analysing the problem, we assume that the relevant actions of a specific type of behaviour are these actions that are more likely performed. In that way, we proposed a method based on frequent itemsets to extract behaviour patterns where control the relevant actions and the order relationships among them. In addition, we required a method to match the current user activity with those behaviour patterns. Patterns coudl be considered as regular expressions, so we proposed to use Regular Grammars to recognize the current activities(see subsections 2.1).

Analysing the Stream Data, we realize that in general there are some specific types of behaviour which are performed roughly at the same moment. From this assumption, we introduce temporal processing to limit periods of time to study a specific type of behaviour (sees subsections 2.2).

On the other hand, we also detected that behaviour performances are not precise and constant; users might change his/her behaviour adding new actions or delaying some of them, etc. To handle this, we required to adapt our knowledge dynamically to encompass those changes. we tried to change our system to be adaptive, however, our structure was not flexible enough to accept the adaptation process. So, we introduced a new representation in layers (see subsection 2.3.1). Additionally, we developed a LA-based algorithm to learn the structure. It starts from the detected patterns and learns the behaviour models. These proposals are presented in subsection 2.3.2.

The recognition process takes advanteges of LA techniques to recognize sequences of actions, as well as, update the behaviour model thought updating the probabilities of LA (see subsection 2.3.3).

In order to test our proposal, we carried out two different tests. On one side, we run experiments in our laboratory and simulated scenarios. On the other side, we performed experiments at the iSpace, a real Ambient Intelligent Environment situated at the University of Essex, Colchester (see subsection 2.3.4).

On the other hand, during the research stay at the University of Missouri, we thought that it may be interesting to study how similar two types of behaviour or two performances of the same behaviour are; and using that information to monitor the evolution of a specific activity continuously. We also developed a linguistic summarization method to make easier to understand the extracted information. This process is summarize in subsection 2.4.

During the research stay at the University of Essex, we took advantages of the host centre experience in Type-2 fuzzy logic systems, and proposed the idea of organizing the learnt behaviour models into a hierarchy to be able to detect human routines (see subsection 2.5).

To end up Part I, Section 3 remarks the most important results obtained during this thesis period and Section 4 suggests some new open research paths to continue with the work performed in this dissertation.

Part II. Publications

Part II presents the publications that support our work. Those are included maintaining the format of the journal or conference where they were published or submitted. First of all, Section 1 presents the paper entitled *Correct behaviour identification system in a Tagged World* [19] It explains a data-mining based technique to extract behaviour patterns from sensor information. An algorithm based on frequent itemsets is proposed to find the relevant actions, whereas a Regular Grammar based algorithm is proposed to recognize the current user activity. In addition, a implementation using mobile technologies is presented.

The paper titled *Fuzzy method to disclose behaviour patterns in a Tagged World* [52] is presented in section 2. This paper presents the work related to the temporal information processing. We specify *Fuzzy Temporal Window* concept, in a formal way and adapt the method proposed in [19] to manage behaviour uncertainty.

Papers associated with our second approach are collected in section 3 with the contributions of three papers:

- *Detection of abnormal human activities by means of Learning Automata* [55]. In this paper, we formally present the main concepts of the problem, together with the proposed behaviour model. Besides, we propose and analyse a Learning Automata based algorithm to learn the behaviour model and recognize it in function of user current behaviour (See section 3.1)
- *Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows* [48]. Here, we extend the behaviour model concept, taking into account the temporal information. Processing that information makes it possible to analyse the changes in the user behaviour. An adaptive recognition algorithm was also proposed to adapt the behaviour model to the current user activities (See subsection 3.2).
- *A Fuzzy Logic Approach for Learning Daily Human Activities in an Ambient Intelligent En-*

vironment [53]. We analyse the system operation in a real Ambient Intelligent Environment testbed. We test our proposal by monitoring three participants' activities for three specific behaviours. In the article, we explain the experiments and summarize the most important results (See subsection 3.2).

Section 4 includes the paper entitled *Similarity measure for anomaly detection and comparing human behaviour* [7] associated with the similarity measure for comparing human behaviour models based on soft partitions and temporal probabilistic order relationships proposed in this dissertation (See subsection 4.1). Additionally, we present the paper *Linguistic summarization of long-term trends for understanding change in human behaviour* [54], where we summarize the work done for designing the linguistic summarization procedure for describing long-term trends of change in human behaviour. We define methods that provide information to elders, caregivers, social workers or even family in an understandable language (See subsection 4.2).

To end up, we present the bibliographic references used in this document.

2. Joint Discussion of Results

As stated before, our problem is situated at a controlled environment, known as *Ambient Intelligent Environment*, where every object may be identified by a unique id number (commonly stored in an id-tag). When a user uses this kind of environment, all their activity could be monitored: the system records any event happened in the environment. Thus, the system produces a continuous sequence of data (*Stream data*) collecting all that activity.

Our aforementioned objective consists of developing a system to learn the *user daily activity* (*behaviour performances*), in order to be able to recognize the activities online and detect if they are correct or incorrect. For those detected incorrect activities, the system sends alarms indicating why it is incorrect: the user has forgotten to perform some relevant actions, or s/he started it but never finished, etc.. A possible running example is shown in 2.1:

EXAMPLE **2.1 (Elderly woman goes to bed)** *Let us suppose that our system is installed in an elderly home where one resident is being monitored. Let us also suppose that we know the actions that the resident performs every day before going to bed (see figure 1): s/he sits on his/her bed, takes his/her slippers off, takes the pills and turns the light off. Each one of those actions is important in order to finish correctly the behaviour. We propose a system that is able to detect which actions are being performed, realizes that the resident is going to bed and reminds actions when they have been forgotten by sending alarms.*

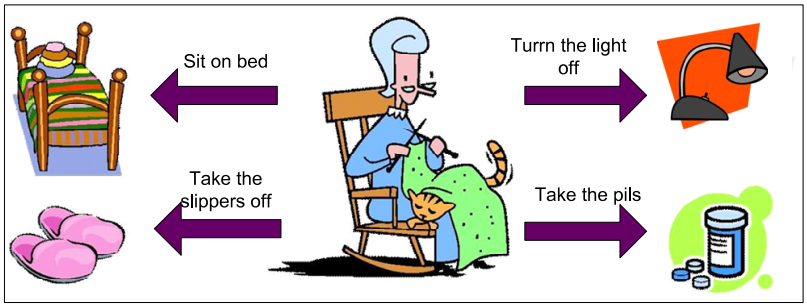


Figure 1: Example. An elderly woman goes to bed

Therefore, from a global point of view, our system should:

- **Extract which are the relevant actions for a specific type of behaviour.** Every day a person performs a huge set of actions, most of them are related to each other with a common final objective. For example, every morning when a person wants to *have breakfast*, there are some actions that has to carry out, such as, *to turn the coffee machine on/off* or *to make some toasts*. However, meanwhile the breakfast is being prepared, s/he could also do the washing-up. Our system learns for each type of behaviour, which are the *key actions* discarding those actions that do not provide any knowledge to the global meaning.
- **Recognize on real-time whether the sequence of actions performed by the user is correct or not.** Once the system knows which are the relevant actions for each type of behaviour, that knowledge is used in order to determine if a current sequence of actions is correct or not. Additionally, in those cases when the sequence of actions is incorrect, an

explanation of that incorrectness: the user has not performed some actions, or s/he has made an error during the activity performance, etc..

- **Generate and send alarms as an incorrect answer.** It is quite relevant to study if a person has performed correctly or incorrectly the activity, if s/he has lasted more than expected or if s/he has not concluded completely the activity. Nevertheless, this knowledge is useless if someone is not informed about it. Thus, from a practical point of view, one of the objective of our system will be to send alarms or warning, to who may concern, informing about what is happening.

In the following subsections, we summarize the different proposals presented in this dissertation and a brief discussion about the obtained results. In every section, we also indicate some publications that support our research.

2.1. Detecting and recognizing behaviour patterns from *Stream Data*. Apriori approach.

First of all, we set the problem out from the point of view of learning. We wanted to develop a method to learn human behaviour. With that aim in mind, our first step was to define what we would understand as behaviour. Mainly, we defined a type of behaviour as a sequence of relevant actions that has a common meaning, i.e., for leaving home, we need to take the keys and open, close and lock the door. The user may perform more actions, but those actions the *relevant* ones: missing any action means that s/he could not complete the behaviour correctly.

Therefore, for each type of behaviour, there will be always a set of actions that must be done; so that, a logic conclusion is that those actions will be the most frequent actions performed by the user. Under these circumstances, we propose to use *Frequent Itemset* concept to detect the *relevant* actions.

As stated before, we assume that a specific type of behaviour is a sequence of known actions with a common objective. Formally, we define the concept of *User behaviour* as:

DEFINITION **2.1 (User behaviour)** *Let* $A = \{ a_1, a_2, ...a_n, \}$ *be the set of possible user's actions in some situation or domain. A user behaviour is a finite set of actions:*

$$\beta = \{\alpha_1, \alpha_2, , ...\alpha_{p(\beta)}\}$$

with $\alpha_j \in A \forall j$ *, and where* α_j *is performed before* α_k *iif* $j \leq k$*.*

Our system receives as inputs the *Stream data* that collects the user activity for different days and it gathers in a database, called *Observation Base (ODB)*. This database could be represented as a *Transaction database* with a tuple for each day. Then, we use the *Apriori algorithm* to extract the *frequent itemsets*. This algorithm, introduced in [5], studies a set of transactions and extracts an itemset where the items are frequent in those transactions. However, not all collected information is relevant for a specific behaviour, hence only a subset of the daily transaction should be studied. To detect those subsets, we turn to an expert to limit the transactions to look for the frequent actions. Later works presents other alternatives.

Using *Apriori algorithm* , we extract the set of relevant actions for each type of behaviour. However, as indicated in the definition 2.1, a *user behaviour* is a sequence of actions with a specific

order relationship among them; what means that there are some sequences that should not ever happen since they are incorrect. Accordingly, a post-processing is required in order to guarantee the reliability of the extracted sequences. The algorithm 1 shows that process.

Algorithm 1 Algorithm to check the order relationship among the actions.

```
for b ∈ behaviour sequence do
  for t ∈ Transactional Database do
    for each two elements a_s and a_r in the sequence b do
      if a_s and a_r do not fulfil the order relationships of the actions in t then
        b is marked as incorrect
        Stop loop
      end if
    end for
  end for
  if i is marked as incorrect then
    Delete b from set of sequences
  end if
end for
```

As output, we obtained a set of behaviour correct sequence of actions, named as *behaviour patterns*. We gather all the *behaviour patterns* in a *Behaviour Base* that is the input to the *Recognizing System*. This module analyses the current user activity and determine if it is correct or not.

We support the *Recognizing System* using a structure known as *Behaviour Tree*, where an action is a node, and the specific behaviour is a leaf. A branch represents a correct sequence of actions for a specific type of behaviour. In addition to the structure, we propose an algorithm to cover across the tree. During every algorithm step, we know a current node, which represents the last node visited during the walk, and a set of accessible nodes (current node's and root's children and grand-children). When the system detects an input, it analyses if that input matches with any node in the set of accessible nodes. In that case, it moves to that node; in other case, it stays in the same node and stores the action. When a leaf is reached, the system produces an output indicating the behaviour in the leaf and the tree walk. For more details, please see the following reference [19][50], where we formally present the algorithm. Analysing the walk, we are aware of how the user has performed the specific behaviour. Our proposal studies that walk and sends alarms to a contact point, specifically a *mobile phone*, if the user has performed the behaviour incorrectly.

To sum up, we have proposed a system based on *Frequent itemsets* and *Regular Grammars* to detect behaviour performances [19]. The system is divided in two different modules: a *Inductive Learning* and a *Reasoning* Module. The first module is in charge of learning which correct behaviour performances are, while the second one uses that knowledge to determine if it is correct or not (Recognizing System). Figure 2 shows a diagram of the complete system.

In order to check the system, we implement this proposal using RFID (Radio-Frequency Identification) and mobile technologies in a laboratory [49]. RFID is a technology based on radio-frequency electromagnetic fields. It transfers data form a tag to a specific reader using those fields, therefore, it is mainly used in automatic identification and tracking.

Our experiment was performed in a laboratory at the ESTII Informática y Telecomunicaciones. Inside the laboratory, we simulated three rooms: a kitchen, a hall and a bedroom. We added

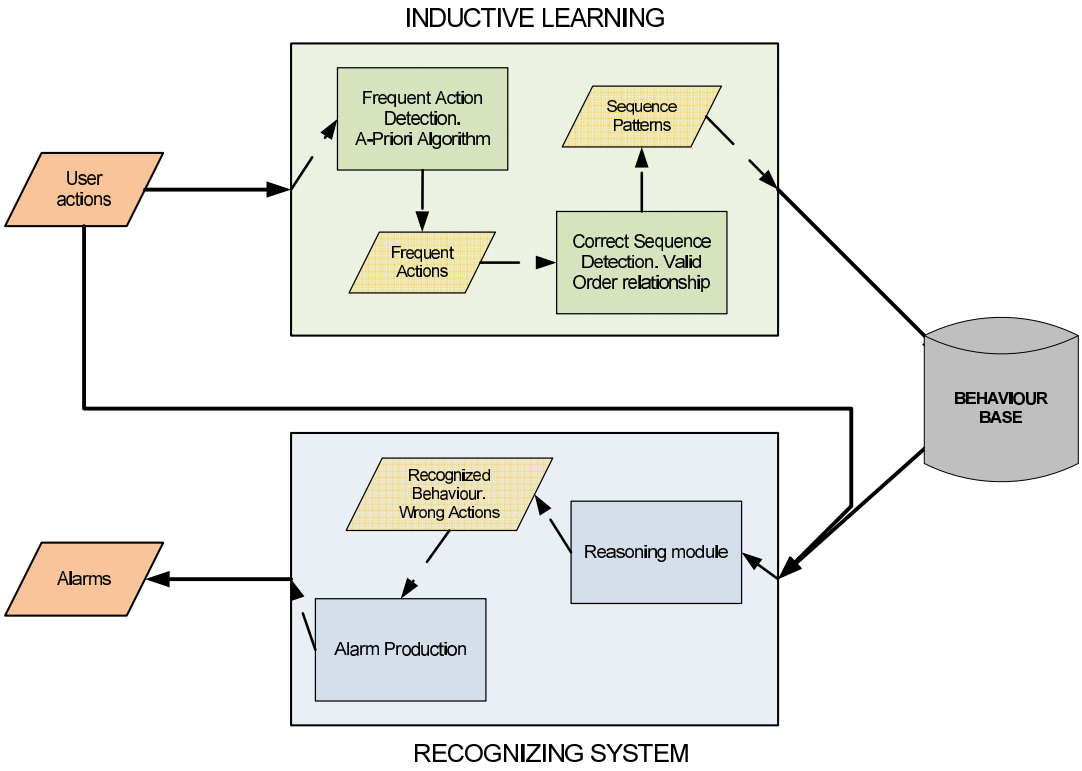


Figure 2: Frequent Itemset based-model diagram

tags to different objects (mobile phone, keys, bag, lamp, etc.) in the simulated environments and connected a RFID reader to a PDA (Personal Digital Assistant). We learnt and recognized two different behaviours in our experiment: *leave home* and *go to bed*.

We support this part with the following publication:

- Delgado, M., Ros, M. and Vila, A. Correct behaviour identification system in a Tagged World Expert Systems with Applications, 2009, 36, 9899-9906.

2.2. Processing temporal information to manage the uncertainty of behaviour.

Once we tested the method presented above, we realize that not all the information collected by sensors is relevant for a specific behaviour. Sensors gather information during the whole day; what means that we need to determine in which section of the collected stream data search for the behaviour patterns. In this point our challenge was to develop a method to be capable of dismissing the irrelevant information, getting just the useful information.

To manage these problems, we assume that a daily behaviour is usually performed around a known time [51][52].

EXAMPLE **2.2** *Every weekday, Rose leaves home at 8:30 a.m. to go to work.*

Nevertheless, the precision of the temporal information is something very difficult to deal with, since it is continuously drawing on. For this reason, the temporal information in general provide a temporal interval when the event will occur. Consequently, using that information we establish an

adjusted interval in which the behaviour should be performed. Formally, we define the concept of *Temporal Window* for that interval[52][56].

However, studying the defined window, we understood that not all actions of the interval are equally important for the behaviour's final performance. Those actions closer to the specific time will more surely belong to the behaviour than the rest. In order to solve this problem, in [52], we proposed a method entitled *Fuzzy Temporal Window (FTW)*. Formally presented in Definition 2.2, a FTW is a temporal interval associated with a specific behaviour performance and a fuzzy set. Every action in the interval has a specific degree of importance belonging to the fuzzy set, hence to the behaviour performance (see figure 3).

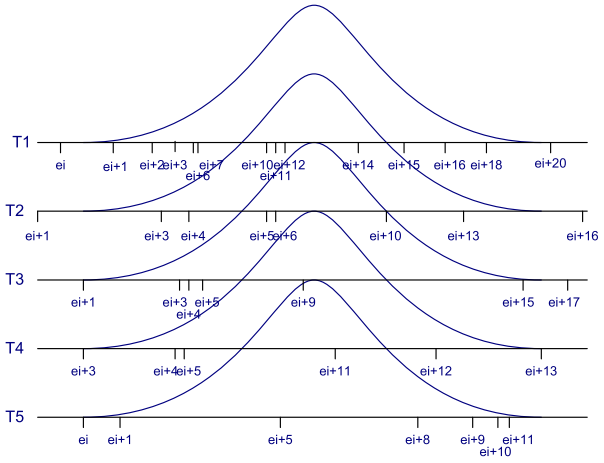


Figure 3: Fuzzy Temporal Window over an ODB

Definition **2.2 (Fuzzy Temporal Window, FTW)** *Let I be an interval from the temporal line τ, ODB the sensor information collected and $t \in ODB$ a tuple from ODB. Let i_j be an action $i_j = (h_j, l_j, d_j)$ in the interval I. Let fs be a fuzzy set over τ, then a* fuzzy temporal window *FTW is defined as the set of actions temporal occurred in I, where*

$$\forall i_j \in W(t)\ \mu_{FW}(i_j) = \mu_{fs}(l_j)$$

Applying a *Fuzzy Temporal Window* over the collected dataset ODB (Observation Database), we obtain a subset of ODB $O\tilde{D}B$ [52].

Definition **2.3 (Fuzzy ODB, $O\tilde{D}B$)** *Let ODB be an Observation Database and W a Fuzzy Temporal Window, it defines $O\tilde{D}B$ as a Fuzzy Observation Data Base constructs as W(ODB) such as*

$$\forall t \in ODB, W(t) \in O\tilde{D}B$$

$O\tilde{D}B$ is a representation of ODB where every action included in it has a membership degree for a specific FTW, so, we extract as many $O\tilde{D}B$ as number of behaviour we study.

In this point, we have limited the dataset in order to extract which action are relevant or not for a specific behaviour. The following step consists of finding which actions could represent the behaviour performance and which are the order constraints among them. To do that, we need to

change the method proposed in [19] in order to manage the recent introduced fuzzy information. The proposed changes are presented in [52][56]. Herein, we present a briefly summary of the method.

Suppose we want to find out which are the *behaviour patterns* for the behaviour B, which has a related FTW FTW_B and a known $O\tilde{D}B$. Firstly, we apply the α-cut concept over a $O\tilde{D}B$, we would have a new crisp image of $O\tilde{D}B$, ODB^{α} where every value is in $\{0, 1\}$. For each ODB^{α} extracted for behaviour B, we obtain its relevant actions and its order relationships using the method presented in [19].

Additionally, as we extract frequent itemsets and behaviour patterns for different α-cut values, we need to represent both sets as a unique fuzzy set. To do that, we assume the following *consistent restriction*:

$$If \quad \alpha_1 > \alpha_2 \quad then \quad I^{\alpha_1} \in I^{\alpha_2}$$

Thus, we have to ensure *consistent restriction* between every α frequent itemset I^{α_i}.

PROPOSITION **2.1** *Let $\tilde{T}$ be a fuzzy transactional database, T^{α_1} and T^{α_2} two α-cuts of $\tilde{T}$, where $\alpha_1 \geqslant \alpha_2$. Let I^{α_1} and I^{α_2} be the set of extracted frequent itemsets from T^{α_1} and T^{α_2} respectively. Then, $I^{\alpha_1} \subseteq I^{\alpha_2}$, i.e., every itemset which is frequent to level α_1, it is also frequent to level α_2*

For sequence patterns representation is the same.

PROPOSITION **2.2** *Let $\tilde{T}$ be a fuzzy transactional database, T^{α_1} and T^{α_2} two α-cuts of $\tilde{T}$, where $\alpha_1 \geqslant \alpha_2$. Let P^{α_1} and P^{α_2} be the set of extracted frequent patterns from T^{α_1} and T^{α_2} respectively. Then, $P^{\alpha_1} \subseteq P^{\alpha_2}$, i.e., every itemset which is frequent to level α_1, it is also frequent to level α_2*

The proof of these statements can be consulted in [52][56].

We study this new approach over some simulated database. Those contain the different possibilities that could be found in a real environment: uncompleted behaviour, incorrect behaviours and correct behaviours. After experiments, we conclude two important results:

- The fuzzy method improves the quality of the extracted knowledge and reduce the user interaction in the process.
- The fuzzy method outcomes depends on the selected *Fuzzy temporal Windows*. For this reason, we should select carefully the FTW in order to extract the best of the results.

The associated article to this part is:

- Ros, M.; Delgado, M. and Vila, A. Fuzzy method to disclose behaviour patterns in a Tagged World Expert Systems with Applications, 2011, 38, 3600-3612

2.3. Adapting the learnt behaviour patterns on time. A Learning Automata based approach

As already mentioned, Human behaviour is not something static and precise. Every person performs them in a different way and different moments. Besides, the own person could change the

performance of the behaviours, both time and relevant actions. For instance, during the winter, a user should not forget to take the coat. Even though, when the summer arrives, s/he changes his/her behaviour because the coat is not still needed. So, the user temporal patterns change. Another situation may be that the user gets another job and starts waking up at different time. Both examples highlight the need of adapt the human behaviours during the system operation in order to refine the knowledge extracted during the learning process.

In earlier approaches, we have proposed a method to learn and recognize human behaviours based on *Frequent Itemsets*. However, this technique is not adaptive. It would require starting from scratch the learning process to be able to absorb the new information. For this reason, we focused on finding a method that would be able to learn the human behaviours and, at the same, would be capable of adapting online the learned knowledge. On the other hand, we had to change the way the behaviour were represented. *Behaviour patterns* are sequences of actions with a specific order relationship. Nevertheless, that relationship was fixed for that representation; what means that could not be adapted during the recognition process, unless we repeated the learning process adding to the original database the new stream data. Therefore, we decided to introduce a new representation that fulfil the following requirement: a group of actions can have different or have no order relationships among them and each action would have a probability of being performed before or after others. We named the structure as *Behaviour model*.

In the following subsections, we introduce the *Behaviour model* and the learning and adapting algorithm used to manage it. And as conclusion, we present a summary of the experiments performed at *iSpace*, a real Ambient Intelligence Environment situated at the University of Essex.

2.3.1. A behaviour formal representation. The behaviour model

The *Behaviour Model* stores all the information associated with a behaviour concept. Concretely, we will contain actions, order relationships and temporal information about when the behaviour should occur. Therefore, the behaviour model could be divided in two modules: a *behaviour action sequence*, which take care of which actions introduces in the behaviour and the order relationship between them; and a *behaviour time* that contains the *fuzzy temporal windows* associated with the behaviour. We will start focusing on the *behaviour action sequence*.

As former indicated, a human behaviour is a sequence of human actions that could (or not) have temporal relationships constraints between them. Formally, we define the two concepts that support our proposal:

DEFINITION **2.4 (Human Action)** *A human action h_i is a triple $h_i =< l_i, s_i, t_i >$, where l_i is a label used to identify the action, s_i is a sensor event and t_i is a time stamp in which the action occurs. In brief, a human action labelled l_i is an activity (fact) that happens over a specific object in a time instant t_i, and it may be uniquely determined by mean of a sensor event s_i. Let's take the following instance as an example of a correct definition of human action "open the front door" could be $<$"OpenFrontDoor", "FrontDoorSensor#1234 = ON", "8 : 30 : 47a.m."$>$.*

DEFINITION **2.5 (Behaviour)** *A human behaviour B^i is a tuple $< H, R^i, C^i, \prec >$, where $H = \{h_1 h_2, ..., h_n\}$ is a set of available human tasks, $R^i = \{r_1^i, r_2^i, ..., r_{|R^i|}^i\} \subseteq H$ is the subset of available human tasks that are required to complete the behaviour, and C^i is a set of temporal constraints between the tasks in R^i, defined over a partial order relationship operator $\prec$. The partial order*

relationship $\prec$ defines the temporal dependency requirements over the actions of a behaviour, and its formal definition is described in equation I.1. In brief, an action r_a^i precedes r_b^i in time in a behaviour B^i, and it is written $r_a^i \prec r_b^i$, if the time stamp t_a of r_a^i is lower than the time stamp t_b of r_b^i, for all the possible instances of the behaviour.

$$r_a^i \prec r_b^i \leftrightarrow t_a < t_b \tag{I.1}$$

We illustrate the definition of behaviour with an example.

EXAMPLE **2.3 (Go to sleep)** *Let us define the behaviour Go to sleep. The set H is the set of all the available tasks that the user could carry out at home, and R contains the relevant tasks for the behaviour, as for instance the set { "To sit on the bed", "To take the slipper off", "To take the pills", and "To turn off the light"}. Let us assume that the behaviour is normal if the user always sits on the bed before to take the pills, and the last action s/he does is to turn off the light. Then the set of temporal constraints between the human tasks is C ={ "To take the slipper off"$\prec$"To turn off the light", "To take the pills"$\prec$"To turn off the light", and "To sit on the bed"$\prec$"To take the pills"}.*

These order relationships may be modelled graphically as a set of AND gates connected in cascade. Each human tasks is matched with an input line to an AND gate. Its value is *false* if the action has not been executed and *true* otherwise. Each AND gate could be seen as an *execution level*: all the actions assigned to level n should be performed before tasks in level $n+1$. Figure 4 shows an example of a possible behaviour model for the example *Go to sleep*.

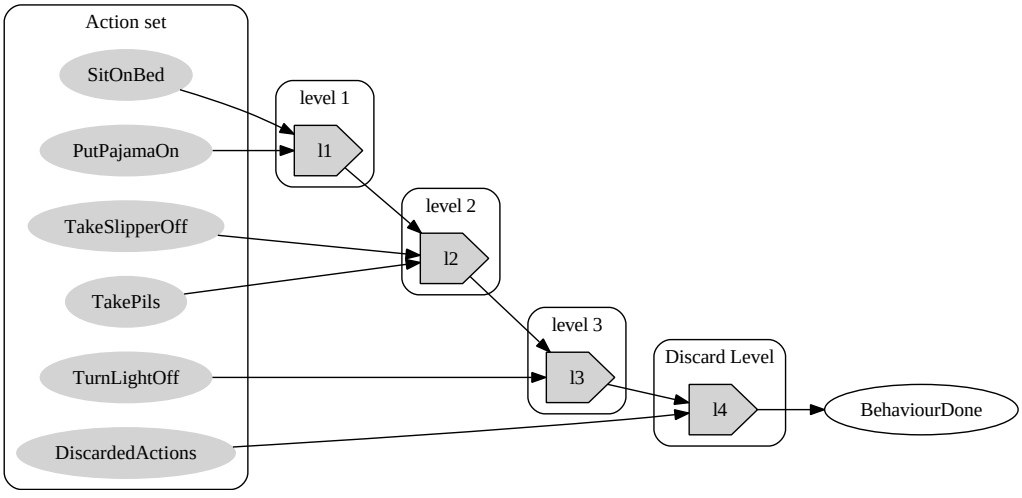


Figure 4: "Go to sleep" behaviour model

Nevertheless, this design is static and does not fit with the new behaviour view. The new representation should allow the behaviour model to be executed in different ways. For instance, the task *To take the slippers off* may be executed either in level 1 or 2, since it just has to be performed before turning the light off; but it has no other type of constraints. Instead of assigning

an action to a specific level, we study the probability of an action being assigned to each level in the behaviour model. Additionally, we include a last level, the *discard level*, where those actions that are not relevant for the behaviour are connected. Table I.1 shows a possible probability distribution for the behaviour model represented in Figure 4.

Table I.1: Example matrix to represent the behaviour *"Go to sleep"* (see Figure 4)

Tasks \ Levels	**1**	**2**	**3**	**4**	**U**
"To sit on the bed"	1	0	0	0	0
"To put pyjama on"	0.7	0.3	0	0	0
"To take the slipper off"	0.3	0.7	0	0	0
"To take the pills"	0	0.9	0.1	0	0
"To turn off the light"	0	0	1	0	0
"To open kitchen's tap"	0	0	0	0	1

Formally, the underlying representation of a behaviour use of a probability distribution for each action in the set H, so that an action h_i may be executed in each of the r possible levels of the cascade with probability $P^i = [p_1^i, p_2^i, ..., p_r^i]^t$. This probability distribution represents that each action can be executed within a cascade level with a concrete probability value. The representation as a sequence of ANDs in cascade is the most likely representation of behaviour performance, assigning each action to its most probable level in the cascade. Therefore, a human task h_i is located at the cascade level l if, and only if, $l = min\{j : p_j^i(k) = max_j\{p_j^i(k)\}\}$, i.e., l is the first level with a maximum probability value in the matrix.

On the other hand, regarding the *behaviour time* module, we stores a *Fuzzy Temporal Windows* associated with each behaviour in the system. As already explained, the *Fuzzy Temporal Window* comes with a *fuzzy set* that controls the temporal uncertainty of actions.

2.3.2. Learning a behaviour model from user activities

Once we had defined the behaviour model, we required a method able to learn that structure from some inputs (either *ODB* or *behaviour patterns*). The algorithm to develop should be capable of learning the probability distributions of each action according with each behaviour. For this purpose, we took advantages of *Learning Automata* and their way of operation.

A learning automaton [45] is a decision-making stochastic machine defined by the tuple $\langle \alpha, Q, R, T \rangle$, where α is a set of available action for the automaton, $Q$is the internal state, R is the set of available reinforcement values of the automaton and T is a reinforcement learning scheme. The running of a *Learning Automaton* is cyclical. At every iteration k, the *Learning automaton* selects an action $a(k)$ from the set of its available actions $\alpha(k)$. It selects the action taking into account Q, the internal state. In order to get the reinforcement value, the action $a(k)$ is evaluated in a random environment using a fitness function $f : Dx\alpha \longrightarrow R$. The obtained reward or penalty reinforcement value $\beta(k) = f(D, a(k))$, where D is the expectation to obtain a reward. This reinforcement value is used to update the internal state of the automaton, $Q(k+1) = T(Q(k), a(k), \beta(k))$. This cyclical process finishes when a predefined condition, such as the convergence to the learned optimal action, is fulfilled.

Although in the learning process we only use FALAs, during this dissertation we use three different types of *Learning Automata*. Herein, we introduce the main details of them[62]:

- *Finite Action-set Learning Automata (FALA).*- the set of available actions $\alpha = \{\alpha_1, \alpha_2, ..., \alpha_n\}$ is discrete and finite, and the internal state of the automaton is modelled as a probability distribution of the action selection
- *Variable Action-set Learning Automata (VALA).*- its action-set is finite as FALA, but the availability of action to the automaton depends on external variables such us time. Its state is modelled with a matrix, where each column is the action probability distribution corresponding to each time, $Q(k) = \left[P(k,1)^t, P(k,2)^t, ..., P(k,t)^t\right]$.
- *Continuous Action-set Learning Automata (CALA).*- this automaton handle continuous action spaces. Its probability distribution is defined by a Normal probability distribution $N(\mu(k), \sigma(k))$.

During the learning process, we separated between the *behaviour action-sequence* and the *behaviour-time* module. Let us start for the *behaviour action-sequence* part. As indicated previously, the learning process of this part consists of finding the probability distribution of each human task H in a given behaviour. For this purpose, we use a team of FALA, composed by the same number of human tasks in H. Thus, each FALA uniquely matches with a human task instance. We assume that the learning automaton LA^i is matched with the task H_i. The team members do not share information, what eases the inclusion or removal of an automaton when a new task is considered as part of H. For each FALA, the set of available actions is $M+1$, where M stands for the maximum number of levels of the behaviour model. This maximum value is provided as a parameter of the learning algorithm. When an automaton LA^i selects the j-th action, for $1 \leqslant j \leqslant M$, it is indicating that the human task h_i is require to complete the behaviour. Besides, that task is assigned to the j-th level in the behaviour model. If a task is assigned to $M+1$-th level means that it is useless for the studied behaviour. Therefore, the state of each FALA is represented as a probability distribution $Q^i(k) = \left[p_1^i(k), p_2^i(k), ..., p_{M+1}^i(k), \right[$, where p_j^i is the probability of selection of the j-th automaton action.

Our learning algorithm uses a *multi-teacher* [45] (see figure 5) scheme for the automaton environment. It is composed by S sample of correct behaviours (*teachers*) collected in the *Observation Database*. Specifically, at every iteration, the *Learning Automata* team obtains a possible behaviour model. This is validated by the *Validator module* (figure 5)). It checks if the proposed behaviour model has not intermediate unconnected levels and if the model starts at the first level. Once it is accepted, the proposed behaviour model is evaluated in the environment for each teacher in the environment. Each one returns a Q- model reinforcement value [45]. These values are accumulated in the overall score, $f(k)$. Later, we transform this value $f(k)$ into a P-model reinforcement value [45] as indicated in I.2. The Fuser module (see figure 5) is in charge of this conversion.

$$\beta(k) = \begin{cases} 1 & iff(k) \geqslant f(k-1) \\ 0 & otherwise \end{cases} \tag{I.2}$$

The learning algorithm is shown in algorithm 2. The algorithm details and the different reinforcement learning rules are deeply presented in [55].

Regarding the *behaviour time* module, we use the *Continuous Action-set Learning Automata* to learn the *fuzzy temporal windows* associated to the behaviour. Since the CALA probability

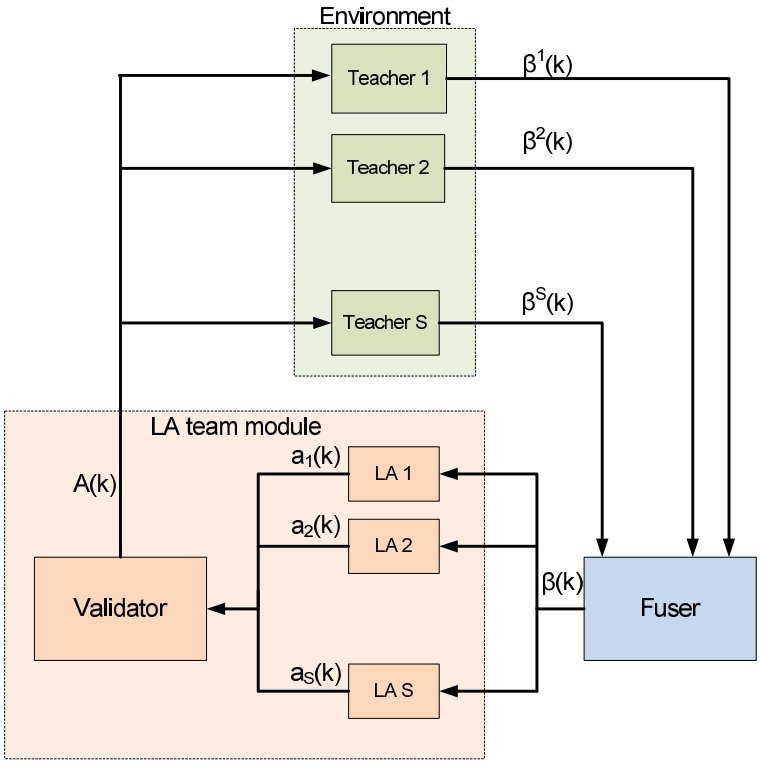


Figure 5: Learning scheme in a multi-teacher environment.

distribution uses a Normal distribution to determine its selection, we will use the traditional learning algorithms[45][62] to train the CALA with the values of the FTW.

In [55], we present a depth study about the influence of the different parameters and the learning rules in the quality of the behaviour model.

2.3.3. Recognizing and adapting human behaviour model.

Once we have learned the behaviour model, the next step was to develop an algorithm to recognize as behaviours the activity performed by the user. First of all, we proposed a method to recognize each possible type of behaviour independently, and later, we generalize the recognition problem in order to synchronize the recognition of different behaviour simultaneously.

In brief, the recognition process analyses every new human task received in the system and checks if all the relevant tasks in behaviour has been performed or not. Our recognition process has two main steps: firstly, we check if the user is performing the behaviour inside the temporal constraints; and secondly, we analyse one by one the relevant tasks determining which tasks has been performed correctly, which ones in a unexpected order, etc. Mainly, we study three different cases:

- **Normal behaviours**.- A behaviour performance is considered *normal* when all the actions fit in its associated temporal windows and no order relationship among their action has been broken.
- **Uncompleted behaviours** .- A behaviour performance is considered as *uncompleted* when some of its action are done out of its FTW. In other words, the user has forgotten some

Algorithm 2 Learning algorithm for the team

Require: $\triangle \in (0, 1]$ the learning rate

Require: n the number of automata in the team, typically $n = |H|$

Require: M the maximum number of levels allowed in a behaviour model, typically $M = |H|$

1: Initialize $k = 1, Q^i(k) : p^i_j(k) = 1/(M+1), \forall i = 1..n, \forall j = 1..M+1$
2: Initialize the best behaviour model found $V(k)$ with $v^i(k) = M+1, \forall i = 1..n$
3: **while** stopping criterion not satisfied **do**
4: **for** every automaton $i = 1..n$ **do**
5: The i-th automaton selects an action $a_j(k)$ according to the action probability distribution $Q^i(k)$
6: **end for**
7: $A(k) = Validate(a_1(k), a_2(k), ..., a_n(k))$
8: **for** every teacher $s = 1..S$ **do**
9: Retrieve reinforcement value $\beta^s(k)$ for the behaviour model $A(k)$
10: **end for**
11: Compute $[\beta(k), f(k)] = Fuser(\beta^1(k), \beta^2(k), ..., \beta^S(k))$
12: **if** $f(k) > max_{j=1..k-1}\{f(j)\}$ **then**
13: Update $V(k) = A(k)$
14: **end if**
15: **for** every automaton $i = 1..n$ **do**
16: Compute $Q^i(k+1) = T(Q^i(k), a_i(k), \beta(k), v_i(k))$
17: **end for**
18: k=k+1
19: **end while**
20: **return** $V(k), Q^i(k) \forall i \in \{1..n\}$

relevant actions during the execution time of the behaviour.

- **Violated of task relationship constraint behaviour**.- A behaviour performance is considered *violated*, when although all the relevant actions has been performed inside of the *FTW*, some order relationship constraints has been broken.

For the last two type of behaviour, an alarm or warning is sent to the user. In both cases, the user could be performing the behaviour deliberately in an incorrect way or not.

According with our behaviour model, a user can carry out some action in different levels. Our recognition process online adapts the *behaviour action-sequence* regarding the past activities to control this uncertainty. Every time the user performs a task h_i, the system detects the level L in which the action matches with the *behaviour action-sequence*:

- If h_i cannot be executed in level L, and h_i belongs to a deeper level and the probability of performing the rest of actions in level L is over a threshold, we affirm that a temporal relationship constraint has been violated, and, therefore, we detect a *Violated of task relationship constraint behaviour*.

- If h_i does not correspond to level L, but it could be performed in deeper levels and the probability of performing the rest of action in level L is under a threshold, then we adapt the sequence of ANDs to assign h_i to the level of maximum probability after L.
- If the time indicated by the FTW has expired, and the user has not performed the resting actions, then we recognize an uncompleted behaviour or a violated behaviour, depending if the algorithm has moved actions to different levels or not.
- If h_i does not match with any level in the behaviour model, we label it as *discarded*.

When all the tasks required to complete the behaviour has been carried out, all the AND gates return true and the behaviour is recognized. Up to now, we are dealing with the *uncompleted and violated behaviours* as *anomalies*. However, if those *anomalies* start being more common than the former structure, the *behaviour model* should absorb this information as normal. For this reason, we propose to complete the recognition process with a process of adaptation after a specific type of behaviour is recognized. The adaptation process improves the knowledge about the user and adjusts the *behaviour model* to the user features.

The *behaviour action-sequence* is updated depending on the level l in which each action was carried out and considering the no executed and discarded actions. The adaptation process consists of adapting the probabilities of the *behaviour model*. For this purpose, we adapt the probabilities of the automata regarding the discrete rule shown in Equation I.3.

$$p_j^i(k+1,1) = \begin{cases} max\left(p_j^i(k,1) - \Delta, 0\right) & if j \neq l \\ 1 - \sum_{z=1, i\neq l}^{S+1} p_z^i(k+1,1) & if j = l \end{cases} \tag{I.3}$$

where the Δ determines the learning rate of the user habits. Higher values of Δ, faster adaptation to user habits. Regarding the *behaviour time*, the adaptation process consists of updating the FTW values: $\mu_{FTW}(k)$ and $\sigma_{FTW}(k)$. As these values are associated with CALA, we use the adaptation process of this LA to adapt the probability distribution. These parameters are only updated if all the actions are included in the FTW. The update rules for the parameters are presented in Equations I.4 and I.5.

$$\mu(k+1) = \mu(k) + \lambda_{CALA}(t_{mean} - \mu(k)) \quad if \nexists j : \mu_{FTW}(t_j) < \mu_{min} \tag{I.4}$$

$$\sigma(k+1) = \begin{cases} \sigma(k) - \lambda_{CALA}\sigma(k) & if \nexists j : \mu_{FTW}(t_j) < \mu_{min} \\ \sigma(k) + \lambda_{CALA}\sigma(k) & otherwise \end{cases} \tag{I.5}$$

where $b_j = (a_j, t_j)$ are the relevant human tasks at the time instant associated with it, μ_{min} is the minimum membership value to an action be considered inside the FTW, t_{mean} is the average time of the executed actions and λ_{CALA} is the automaton learning rate.

So far, we have focused on studying a single behaviour. Generalizing the problem to m possible behaviours, our proposal architecture is shown in 6. We parallelize different behaviour models studied by the system. Each type of behaviour is considered as an autonomous module, therefore, the simultaneous processing of the behaviours is more accessible. We include an additional module, the *Behaviour Manager* to control the possible interactions among the behaviours:

- schedules and process the behaviours obtained in the recognition process,
- control the performed actions, classifying them as member or not of the behaviour model,

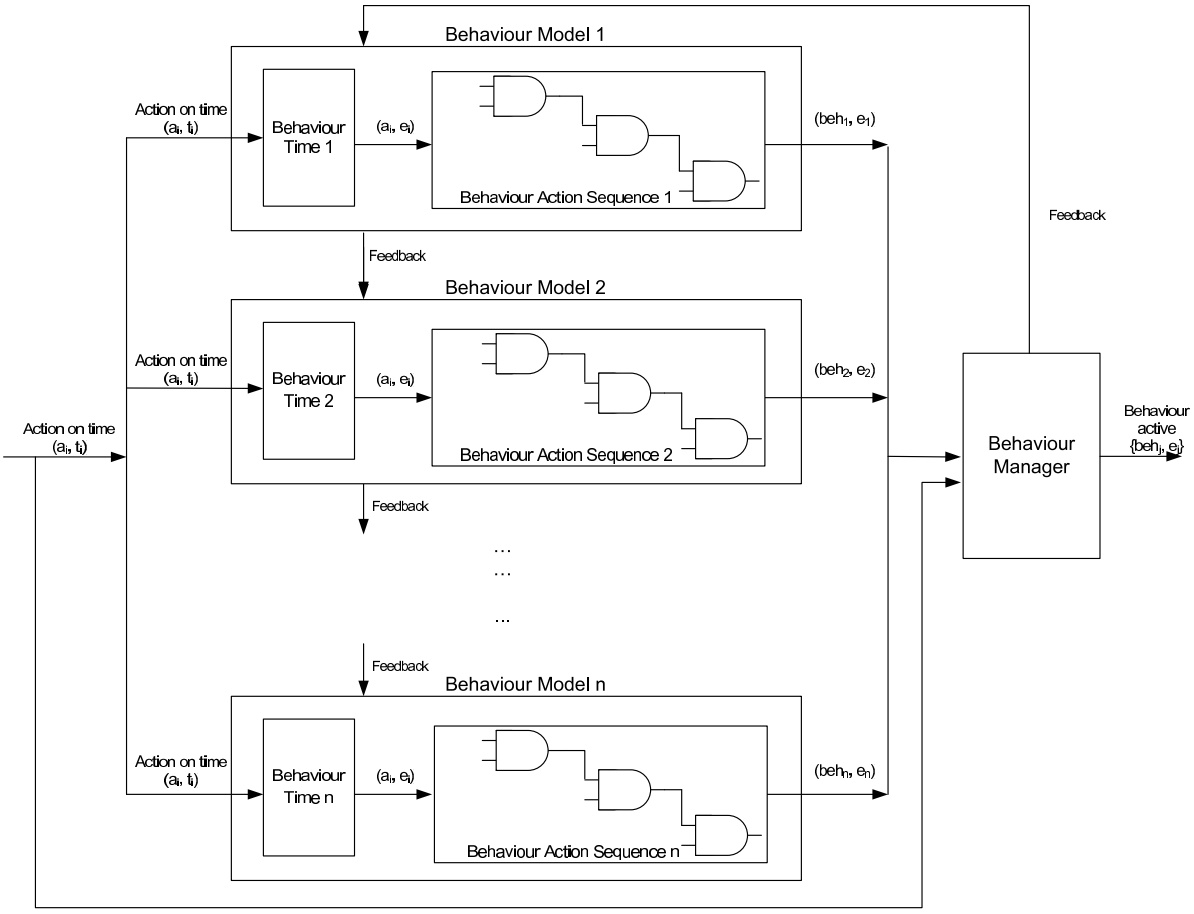


Figure 6: System architecture with behaviour models in parallel

- generate the feedback to adapt the system for environmental or behaviour changes,
- inform the user about the process, and send warnings or alarms when required.

Regarding the testing proces, in [55], the individual adaptation process is analysed. We test how the system is able to incorporate new actions and change the order relationship between the tasks. Additionally, we present a comparison between our approach and a HMM based one [59]. A global study is presented in [48], where we examine the recognition and adaptation process with different behaviours and different databases.

2.3.4. Experimental process in a Real Ambient Intelligence Environment. *iSpace*

Even though, we run experiments both simulated data and inside a small laboratory, we had the opportunity to test our system in a real Ambient Intelligence Environment, known as the iSpace, at the *University of Essex* as a result of a three-month research stay. We performed several unique experiments within the *iSpace* (see Figure 7). The *iSpace* is a two-bedroomed apartment composed of a bedroom, a study, an entrance hall, a fully equipped kitchen, a living room and a bathroom. The apartment is organized in four labelled areas: kitchen, sofa, bedroom and hall.

Each room contains a set of devices to interact and manage the environment, which are connected using the Universal Plug&Play (UPnP) architecture. The communication with the devices

is carried out using an API programmed in JAVA, which provides of the services belonging to a range of devices and also interaction via sending messages over the network.

For the experiment, we monitored sensors and actuators that provide a real image of the current *iSpace* state. Namely, we monitored two inputs sensors: door sensor and Ubisense Tracking Sensor. The door sensor indicates if the door is opened or closed. Regarding the Ubisense Tracking Sensor, it detects when the user either enters into or exits a zone. Additionally, we control the state of 20 actuators consisting of light-switches, curtain-controllers, TV remote control, door remote lock and switch controllers. The switch controllers manage whether some specific electrical appliance is connected to the electrical network or not. In concrete, those devices are: a coffee machine, a little bedroom lamp, an alarm clock, a toaster and a radio. To manage all the actuators, we have developed an interface to send the specific commands to the actuators, such as, turn them on and off (see figure 7f).

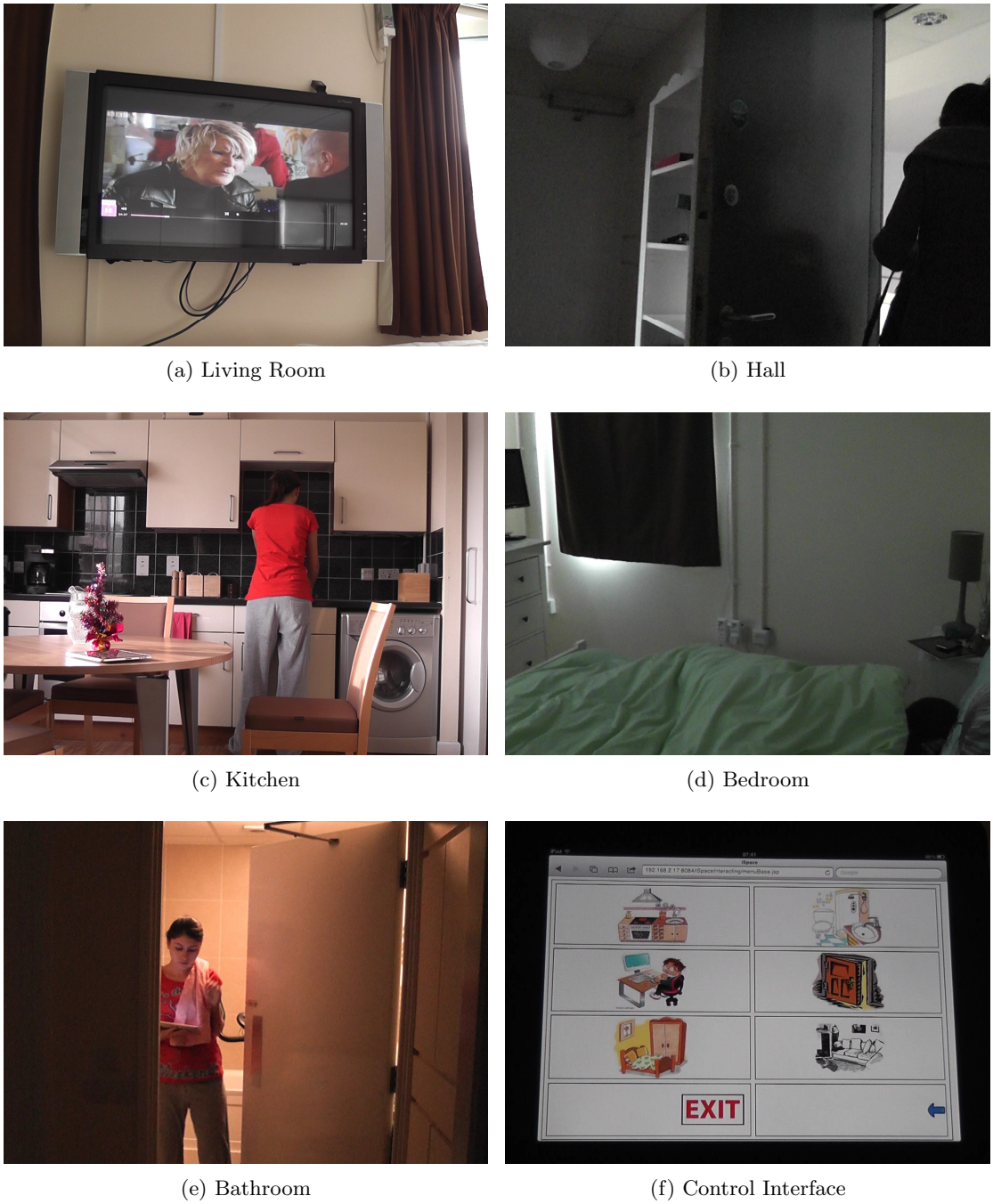
The goal of our experiment is to check that our system is capable of learning different behaviour models for different types of behaviours, recognizing the user current activities and online adapting the behaviour model in order to evolve the behaviour model with the user. For this purpose, we studied *three* participants and *three* different behaviours: *Wake up*, *Have breakfast* and *Leave home*. Each participant was requested to perform *independently* these behaviours in the *iSpace* in a natural way, i.e., performing exactly the *habitual* sequence of actions that s/he usually carries out at his/her own home. Additionally, we requested the participants to provide some expert knowledge about his/her way of performing the behaviours: the *temporal interval*, the *sequence of actions to carry out* and the *order constraints among them*.

At the beginning of each experiment session, the system applied a specific mode in *iSpace*, named as *Sleepy mode*, consisting of the following specific actions: closing all curtains, turning all the devices off and turning all the lights off. Each participant wore a tag of *Ubisense Tracking Sensor* to identify exactly their position at the environment. Each experiment session started when the alarm clock went off at a specific time provided by the participants. From that moment on, the system recorded all the user interaction (events) with the environment, received either form the sensors or through the interface. For each event, the *time* and the *activated sensor/actuator* were stored. Each experiment session finished when the participant left the *iSpace*.

The experiment results are partially presented in [53] . Herein, we briefly summarize the most important ones. We have concluded that our system is able to extract behaviour models very similar to the correct ones, comparing the learned behaviour models to the one generated from the user knowledge. We have also demonstrated that the online adaptation algorithm evolves the behaviour model in real time, being capable of changing the structure of the behaviour model, as the user executes tasks. Moreover, our system is capable of learning not only those actions which are considered as basics by the user, but also the ones that are unconsciously performed yet they belong to the action set of the behaviour. In general, the detection of these actions could be very useful to notice small changes in the behaviour performances that are imperceptible in other cases, such as, in degenerative disease supervision applications.

The associated article to this part are:

- Ros, M.; Pegalajar, M.; Delgado, M.; Vila, A. and Hagras, H. Detection of abnormal human activities by means of Learning Automata Transactions on Systems, Man and Cybernetics – Part A: Systems and Humans, 2011, Submitted to.
- Ros, M.; Cuéllar, M.; Delgado, M. and Vila, A. Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows Information Sciences, 2012, In press.



(a) Living Room (b) Hall

(c) Kitchen (d) Bedroom

(e) Bathroom (f) Control Interface

Figure 7: *iSpace* rooms

- Ros, M.; Delgado, M.; Vila, A.; Hagras, H. and Bilgin, A. A Fuzzy Logic Approach for Learning Daily Human Activities in an Ambient Intelligent Environment IEEE International Conference on Fuzzy Systems, 2012, In press

2.4. Understanding change in human behaviour. A summarization tool

The adaptation process provides us with information to monitor the user about how s/he behaves. However, we realize that a method to evaluate that evolution could be a great support in the resident monitoring for caregiver, nurses, etc.. So, during the research stay at the *Computational Intelligence Research Laboratory* in the *Electrical and Computer Engineering Department* at the *University of Missouri-Columbia* (MU), we took advantages of the huge experience of he host centre in Ambient Assisted Living and Linguistic Summarization process, to develop a method to measure how similar are two types of behaviour or two performances of the same behaviour.

In our previous works [55][48], we have presented a representation for human behaviour, the *Behaviour Model*, and a set of algorithm to learn, recognize and online adapt the model. Focusing on the adaptation process over the time, we propose a tool to identify changes in patterns of behaviour over longer periods of time (days, weeks, months or years). Until up, our work has been focused on detecting the incorrect behaviour performances on real time. With this new branch, we will be able to help predict cognitive and/or functional decline that affects overall quality of life [60].

First of all, in order to study those changes in the behaviour, a *Similarity Measure* is introduced for anomaly detection and for comparing pairs of behaviour models [6][54]. This *Similarity Measure* was designed based on the different relevant aspects of the *Behaviour Model*[48]. Basically, the measure includes three components that study all the important aspects of the behaviour model:

- Comparing soft partitions: determines the similarity between pairs of behaviours according to the strength of actions grouping into similar levels.

$$s_{cp}(U,V) = \frac{s(U,V)}{norm_1(U,V)}. \tag{I.6}$$

  where $norm_1(U,V) = maximum(s(U,U), s(V,V))$ and $s(U,V)$ is the strength of actions grouping into similar levels. It is calculated using the comparing soft partitions technique proposed in [7], viewing tasks as data points and levels as clusters. The normalization value, $norm_1(U,V) = maximum(s(U,U), s(V,V))$, is introduced to produce a more human interpretable value when two pair of behaviours are compared. We extends this normalization technique for comparing the *same* kind of behaviour adapted by a system over time, obtaining a comparing soft partitions part as:

$$s_{cp_2}(U,V) = \frac{s(U,V)}{maximum(norm_1(U,V_1), ..., norm_n(U,V_n)))}. \tag{I.7}$$

- Subsethood of Non-Discarded Actions: two behaviour models may share many actions in the discarded level. This component minimizes the influence of those actions in the measure.

$$s_{sub}(U,V) = \frac{|R_U \bigwedge R_V|}{|R_U \bigvee R_V|} \tag{I.8}$$

  where R_U and R_V are the relevant set of actions for behaviour U and V respectively, $\bigwedge$ is intersection, $\bigvee$ is union and $||$ is cardinality.

- Similarity of Partial Order Relations: determines if the order of levels is consistent. Two behaviour models may share a set of non-discarded actions and have a strong similarity of actions grouping into levels (clusters), but the order of levels is not consistent, the measure has to factor its effect.

$$s_{por}(U,V) = 1 - \frac{\sum_{(i,j)\in I'} \emptyset(E_U(i,j), E_V(i,j))}{|I'|} \tag{I.9}$$

where

$$\emptyset(E_U(i,j), E_V(i,j)) = \begin{cases} 0 & if E_U(i,j) = E_V(i,j) \\ minimum(P_{(U,i)}, P_{(V,i)}) & otherwise \end{cases} \quad \text{(I.10)}$$

where E_U is a $|H|x|H|$ matrix that represents the order relationships (before, same, after) between two actions and $P_{(U',i)}$ measures the ability to decide that action i belongs to some particular level.

Therefore, the final measure for comparing human behaviours is

$$s_{final}(U,V) = s_{cp_2}(U,V) \times s_{sub}(U,V) \times s_{por}(U,V) \quad \text{(I.11)}$$

With this *similarity measure*, we are able to detect longer time periods changes in the *same* type of behaviour (days, weeks, months, etc..), comparing each day the evolved behaviour model to an initial base refined behaviour (the base-line behaviour). Figure 8 shows an example of the evolution of *Wake up behaviour* for a specific resident for a year. If we observe just the graphic representation, this is not the most understandable by nurses, social workers, families or, even, residents. However, it is possible to summarize the information presented in the graphic in a more usual vocabulary (see annotations in Figure 8). Our objective consist of defining a linguistic summarization method to analyse the time series obtained after using the *similarity measure*.

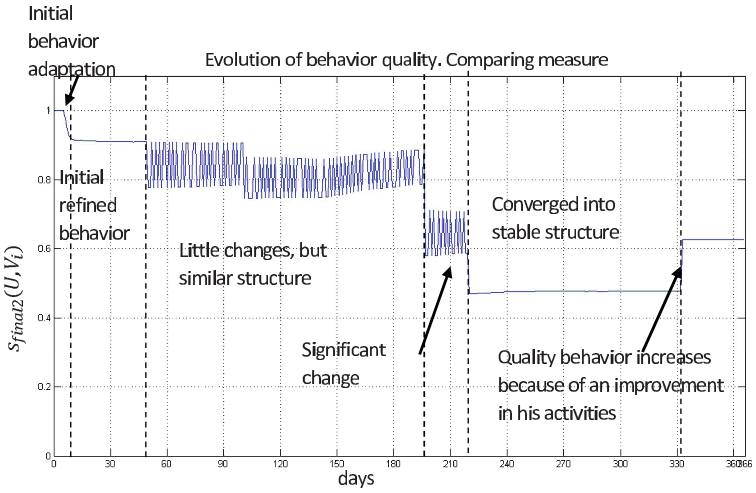


Figure 8: Example. Output of Wake up behaviour evolving for a year. Annotations show what happened in this year.

We are interested in detecting gradual changes in the elders' behaviours as soon as possible. We focus on partial trends concerning a smaller time span (e.g. a window). This defined time window has a considerable influence on the linguistic summarizations produced. For each time span, we applied the well-known method of linear regression in order to numerically summarize a partial trend time window. We calculate the slope of the estimated linear equation, which resides in [-90º, 90º]. This value provides us with the angle of the line segment that characterizes the rate of change of the time series. Additionally, we compute the mean squared error (mse), which varies in [0, 1]. From this value, we get the variability of the samples to the estimated linear equation.

Herein, linguistic summarization of change in human behaviour over long time periods consists of the generation of understandable sentences of the form:

Behaviour [X] is [angle] with [mse] in a time period equal to [time window]

We leverage the concept of *dynamics of change* (or speed of change) identified by Kacprzyc et al []. The authors propose linguistic terms corresponding to various inclinations of a trend line: *quickly decreasing*, *decreasing*, *slowly decreasing*, *constant*, *slowly increasing*, *increasing* and *quickly decreasing* (figure 9 shows the sets graphically).

- *Quickly decreasing* [-90 -90 -80 -60]
- *Decreasing* [-75 -60 -40 -25]
- *Slowly decreasing* [-40 -30 -20 -10]
- *Constant* [-20 -10 10 20]
- *Slowly increasing* [10 20 30 40]
- *Increasing* [25 40 60 75]
- *Quickly increasing* [60 80 90 90].

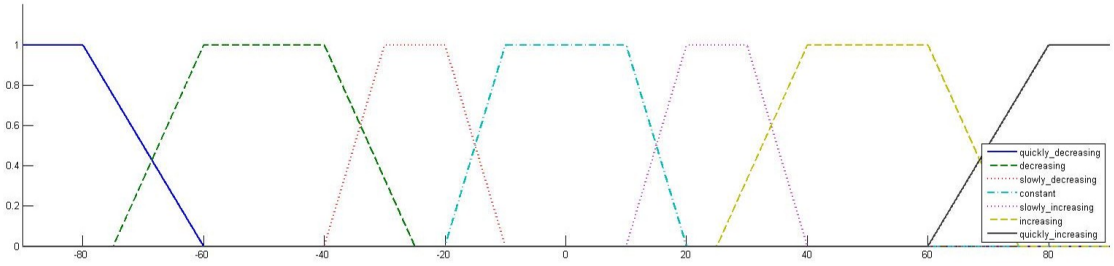


Figure 9: Fuzzy sets for the angle of the estimated linear equation.

Regarding the concept instability of a time series, we define a linguistic variable with following terms, defined in possible levels of instability:*no instability*, *low instability*, *medium instability*, *high instability* and *extreme instability* (figure 10 shows the sets graphically).

- *no instability* [0 0 0.0001 0.0012]
- *low instability* [0.0001 0.0005 0.001 0.0014]
- *medium instability* [0.0001 2 0.002 0.004 0.0048]
- *high instability* [0.004 0.009 0.04 0.07]
- *extreme instability* [0.05 0.071 1 1].

In paper [54], we present a set of experiments performed to prove the running of the summarization process in depth. Briefly, we emphasize that our method is able to detect both small and catastrophic changes. In the experiment, we show how the methods detects, in one hand, a degenerative disease; and on the other hand, a huge event that make the user changes completely their behaviour and the later rehabilitation process.

The associated papers to this part are:

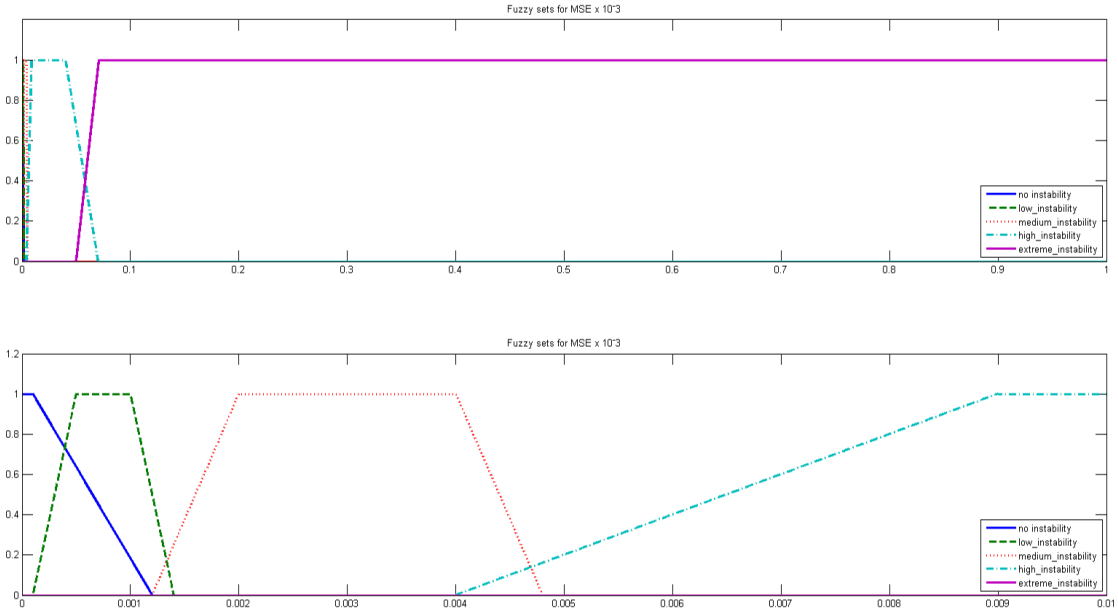


Figure 10: Fuzzy sets for the mse in the user specified time window.

- Derek T. Anderson and María Ros and James M. Keller and Manuel P.Cuéllar and Mihial Popescu and Miguel Delgado and Amparo Vila. Similarity measure for anomaly detection and comparing human behaviours. Submitted to International Journal of Intelligent Systems.
- Ros, M.; Pegalajar, M.; Delgado, M.; Vila, A.; Anderson, D.T.; Keller, J.M.; Popescu, M.; , "Linguistic summarization of long-term trends for understanding change in human behaviour," *Fuzzy Systems (FUZZ), 2011 IEEE International Conference on* , vol., no., pp.2080-2087, 27-30 June 2011

2.5. Generalizing the behaviours. Routine detection

After summarizing behaviour evolution knowledge, we realize that this mechanism could help to reduce the over-monitored process associated with this type of systems. Therefore, we required a process to summarize recognized behaviour: as a sequence of actions can be grouped as a specific behaviour, similarly, a set of behaviour may be also combined as an only one object with a common meaning: a routine. For example, let us think about the actions performed every morning. Every person usually *Wakes up*, *Haves breakfast* and *Leaves Home*. We could characterize this behaviour as an only one set: *Morning Routine*. Therefore, those three behaviours should be performed to be ready every morning.

This issue emerged during a research stay at the *Computational Intelligence Centre* at the *School of Computer Science* at the *University of Essex*, Colchester, UK. This centre has extensive experience in Type-2 Fuzzy Logic [38]. During this period, we took advantages of their experience and propose a Type-2 Fuzzy System to detect different behaviour performances and generalize it as a routine. The proposal follows the fuzzy control system principles, but including a small difference in the rule format.

Type-2 Fuzzy Logic was introduce by Zadeh in [66]. In brief, a type-2 fuzzy set is characterized by a fuzzy membership function, i.e., the membership value for each element of this set is a fuzzy set in [0, 1]. Membership functions of type-2 fuzzy sets are three dimensional, including a *Footprint Of*

Algorithm 3 Pseudocode of the recognition process

```
1: Wait until an action a is received
2: Execute the behaviour detection sub-system with a as an input
3: if The behaviour b is detected then
4:    Generate the temporal fuzzy set related to b, which gathers all the action belong to b
5:    Store behaviour b in the detected_behaviour_vector
6:    Execute the Type-2 FLS with the information collected in detected_behaviour_vector
7:    if Any rule is fired then
8:       return Consequent of the fired rule
9:       Clear the detected_behaviour_vector
10:   end if
11: end if
12: Go to 1
```

Uncertainty (FOU). The third dimension, together with the FOU, supply the additional degrees of freedom required to handle the uncertainties [38]. Type-2 Fuzzy Sets are very useful in applications where determining complete membership functions is complicated due to instability of the problem. Ambient Intelligent Environments (AIEs) is a perfect example of that uncertainty. AIEs are aware of userś context and respond to usersńeeds in an unobtrusive manner [22], what means that the users are who determine how the environment behaves and evolves. It is clear to understand that those applications are very uncertain since they are dealing with different users, with different needs, preferences, etc.

In this part of the work, our objective was to organize into a hierarchy the different learnt behaviours to recognize more general behaviour, named as routines. Our proposal is divided into two steps: firstly, we used our aforementioned method to recognized simple human types of behaviour. Those detected types will be the input to the second part of the system: *Routine detection sub-system.* This analyses the recognized behaviour and checks if any routine has been performed. The behaviour is stored in the system once it is detected. Both systems are independent of each other and exchange the information asynchronously. Therefore, the can be executed in parallel, allowing each system to be replaced by another, with the same objective, without detriment to the smooth running of the system. This process is summarized in the following pseudocode (see algorithm 3):

Routine detection sub-system has the structure of a Type-2 Fuzzy system, so that, it will composed by a fuzzification process, a rule base, an inference engine, a type reduction and fuzzification process (see figure 11). The method followed to develop this system is presented in [38]. Herein, we present the most relevant changes introduced to that proposal:

Fuzzification

Fuzzification comprises the process of transforming crisp values into grades of membership for linguistic terms of fuzzy sets. As indicated before, the major difference between a singleton FLS and a non-singleton FLS is in the fuzzification part [58].

In our proposal, we use *non-singleton Type-2 Fuzzy System.* When a type of behaviour is detected, we could know the temporal instant when it was recognized. This instant corresponds with the last detected action belonging to the behaviour. However, that moment cannot be assumed as the

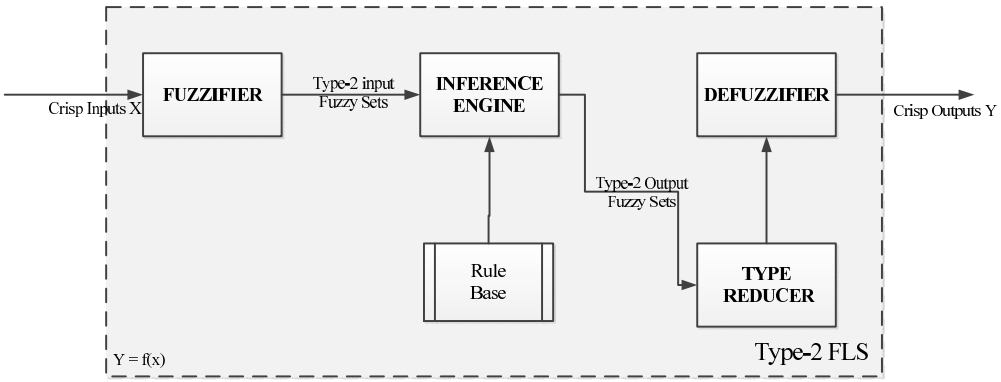


Figure 11: Type-2 Fuzzy system structure

identification point for the behaviour, since it is determined by a set of actions and not by only one action. Therefore, the incoming input of our *Routine detection subsystem* is a temporal fuzzy set that represents the times of all detected actions. This fuzzy set collects the uncertainty associated with the performance of a specific behaviour. With these conditions, we use a *non-singleton type-2 FLS* to be able to handle the uncertainty of the own problem.

Rule Base

Traditionally, rules have the same structure in type-1 FLS than in type-2 FLS. The difference is in the membership function of its elements. For a type-2 FLS, they are represented by interval type-2 fuzzy sets [38]. Considering a FLS having p inputs $x_1 \in X_1, x_2 \in X_2, \ldots, x_p \in X_p$ and c outputs $y_1 \in Y_1, \ldots, y_c \in Y_c$. Then, the *th*-rule in a FLS can be written as follows [25]:

$$\text{IF } x_1 \text{ is } \tilde{F}_1^i \text{ and } \ldots \text{and } x_p \text{ is } \tilde{F}_p^i \text{ THEN } y_1 \text{ is } \tilde{G}_1^i \ldots y_c \text{ is } \tilde{G}_c^i$$
$$i = 1 \ldots M \tag{I.12}$$

where M is the number of rules in the rule base.

In our proposal, we change this pre-defined rule structure, adding information about behaviour performance temporal information, as well as, information about the order relationships between behaviours.

In this way, each antecedent contains information about performed behaviours and their temporal order relationship. In concrete, the order relationship is specified by a temporal operator. On the other hand, each behaviour is characterized by a *Type-2 membership function*, which indicates when the user should perform the behaviour.

Then, considering having p inputs (types of behaviours) $b_1 \in B_1, b_2 \in B_2, \ldots, b_p \in B_p$ and output $r_1 \in R$, a *th*-rule in a *Type-2 Routine FLS* is written as follows:

$$\text{IF } B_1 \text{ is } F\tilde{T}W_1^i \text{ OP } B_2 \text{ is } F\tilde{T}W_2^i \text{ and } \ldots \text{and } B_{p-1} \text{ is } F\tilde{T}W_{p-1}^i \text{ OP } B_p \text{ is } F\tilde{T}W_p^i \text{ THEN } R \text{ is } \tilde{G}^i$$
$$i = 1 \ldots M \tag{I.13}$$

where M is the number of rules in the rule base and OP is a Temporal Operator.

The temporal operators represent operations defined over the temporal line, and indicate how tow temporal interval are related each other. Specifically, we use the Temporal Operator proposed in [23]. These operators work over fuzzy sets that, in our case, will be the aforementioned FTW.

Let FTW_1 and FTW_2 be two Fuzzy Temporal Windows, defined as a trapezoidal function $FTW_i = (\gamma_{FTW_i}, \alpha_{FTW_i}, \beta_{FTW_i}, \delta_{FTW_i})$, therefore we can applied the time operators defined in [23]. For the purpose of clarify, we will rename FTW_1 and FTW_2 as F_1 and F_2 respectively.

- Before $BEF(F_1, F_2)$ [23]: This operator computes the matching degree of F_1 with the complement of F_2 on the left side (the moment before F_2):

$$BEF(F_1, F_2) = \begin{cases} 1 & if \beta_{F_1} \leq \alpha_{F_2} \\ \frac{\alpha_{F_1} - \beta_{F_2}}{(\alpha_{F_2} - \beta_{F_2}) - (\beta_{F_1} - \alpha_{F_1})} & if \beta_{F_1} > \alpha_{F_2} and \alpha_{F_1} < \beta_{F_2} \\ 0 & otherwise \end{cases} \tag{I.14}$$

- After $AFTER(F_1, F_2)$ [23]: This operator computes the matching degree of F_1 with the complement of F_2 on the right side (the moment after F_2):

$$AFT(F_1, F_2) = \begin{cases} 1 & if \gamma_{F_1} \geq \delta_{F_2} \\ \frac{\delta_{F_1} - \gamma_{F_2}}{(\delta_{F_2} - \gamma_{F_2}) - (\gamma_{F_1} - \delta_{F_1})} & if \gamma_{F_1} > \delta_{F_2} and \delta_{F_1} < \gamma_{F_2} \\ 0 & otherwise \end{cases} \tag{I.15}$$

- Overlaps $OVL(F_1, F_2)$ [23]: This operator calculates if at least an instant of time belongs to both F.

$$OVL(F_1, F_2) = Sup_x\{Min\{\mu_{F_1}(x)\mu_{F_2}(x)\}\} \tag{I.16}$$

Those operators are defined for *Type-1 fuzzy membership* functions. We have extended them to be able to handle *Interval Type-2 Fuzzy sets*. We used the operators defined above, on *upper* and *lower* function ($\overline{op}$ and $\underline{op}$) independently.

Inference Engine

Inference Engine follows the method presented in [38]. However, as our rule are different from the traditional rule, we had to introduce a new parameter in the calculation of the firing strength interval. After fuzzification process, we have as input a Type-2 fuzzy set. To evaluate it over our rules, we need to take into consideration not only if the input matches with the membership function, but also the temporal operator evaluation. The temporal operator evaluates if the order relationship between the behaviour performances are correct or not. Therefore, if the temporal operation does not fulfil, the rule will not be fired.

We applied the singleton Type-2 FLS approach to the non-singleton type-2 FLS to obtain the upper and lower membership values of the fuzzification outputs for each input. Additionally, we have to include the condition presented above. Therefore, the firing strength of each rule is $F^l = op * [\underline{f^l}, \overline{f^l}$, where

$$\underline{f^l} = T_{k=1...2m, s=1...m} \underline{\mu^l_{s,k}(x_k)} * \underline{\mu^l_{s,k+1}(x_{k+1})} * op_s$$

$$\overline{f^l} = T_{k=1...2m, s=1...m}\ overline{\mu^l_{s,k}(x_k)} *\ overline{\mu^l_{s,k+1}(x_{k+1})} * op_s$$

where m is the number of antecedents and op_i is the result of evaluating the temporal operation included in the rule.

Type-Reduction

Once we have already obtained the lower and upper firing strengths for each rule, we use the center of sets type-reducer method to get the type-reduced sets.

Defuzzification

The final defuzzified crisp output is obtained by taking the average of the type-reduced set [38].

Experiments and results

Our proposal has been tested in the iSpace at the University of Essex. We have performed unique experiments with 5 different participants during 10 days. Each one was requested to perform *Morning Routine*, as they usually do at their own homes. We analyse their routines with our Type-2 method, and compare the results with a Type-1 Fuzzy System.

During the experiment, the most important detail was how to build the membership function of the rules. For the type-1 model, we choose 5 different membership function, obtained from user performances. Those membership functions are used to evaluate the rest of user performances. In comparison, we aggregate those 5 membership function in only one, accumulating all their knowledge for obtaining the type-2 fuzzy membership function.

As conclusions, we affirm that using Type-2 FLS in our problem, we are able to deal with different users and ways to perform the behaviour without requiring learning their temporal information from scratch. For this reason, our system is more generic reducing the effort of adapting the knowledge or re-learning when a new user starts using the system.

3. Concluding Remarks

The following section briefly summarize the obtained results and present several conclusions.

- We have developed a method for extracting which are the relevant actions for a specific type of behaviour from sensor information. We proposed an algorithm based on *Data Mining* techniques, specifically, *Frequent Itemsets* to extract those actions. Together, we designed a method to control the correct order relationship among the actions. Those patterns are collected in a *Behaviour Base*, which is the input to the *Recognising System*. This analyses current user activities and matches them with the learnt *behaviour patterns*. For this purpose, we proposed an original structure, named as *Behaviour Tree* and a *Walk Algorithm* to recognize the behaviours patterns.

- We have proposed a method to handle the temporal uncertainty of the problem. We assume that a specific behaviour is usually performed around known time. Using this information, we establish a *temporal window* over the temporal line, indicating when is more likely that a specific type of behaviour will be performed. Besides, we fuzzified the concept to provide more importance to those actions close to the interval centre. Finally, we have modified the extracted pattern method to handle with the new fuzzy information.

- We have generalized the *behaviour pattern* concept to an adaptive and probabilistic representation. Our new representation controls two aspects: the *behaviour sequence-actions* and the *behaviour time*. The first one contains a probability matrix to represent the actions and the order relationship among them. The second one stores the *Fuzzy Temporal Window* associated with the specific behaviour.

- We have designed a set of algorithms to learn and recognize the *new behaviour representation* based on *Learning Automata* technique. The learning behaviour uses either the *ODB* or the extracted patterns to create the environment where the *Learning Automata* evaluate their solutions. On the other hand, the recognising algorithm is able to adapt the *behaviour model* in two moments of the recognition process:

  - During the recognition process: The *Behaviour Model* is adapted dynamically to allow an action to be recognized in different levels depending on its probability matrix.
  - Once a behaviour performances has been recognized: The algorithm adapts the probabilities of the matrix depending on the detected actions, levels and behaviours.

- We have tested our algorithms in both simulation and a real ambient intelligence laboratory. Firstly, we studied the robustness and the reliability of the proposed algorithms. Additionally, we compared our results with a HMM method [59] obtaining better results in several cases. Secondly, we tested our method at the *iSpace*, an *Ambient Intelligence Environment* placed at the University of Essex, Colchester.

- We have proposed a tool to identify changes in patterns of behaviour over longer periods of time (days, weeks, months or years). To support this process, we designed a *Similarity Measure* that compares two different *Behaviour Models*. Basically, the measure includes three components: *Comparing Soft Partitions*, *Subsethood of Non-Discarded Actions* and *Similarity of Partial Order Relations*. With this *Similarity Measure*, we compare each day evolved behaviour model to an initial base refined behaviour (the base-line behaviour) and generate representation of the behaviour evolution. Next, we required to provide a more understandable mechanism to analyse that evolution. We proposed a summarization process that studies

the evolution in different periods of time, analysing the slope of the evaluation and the mean square error of the data.

- We have generalized the problem of recognizing human behaviour to analyse sets of behaviour performances with a common meaning: routine. We have developed a non-singleton Type-2 Fuzzy Logic System system that receives as inputs the FTW of aforementioned method. Both systems can work in parallel, through an asynchronous communication. As conclusions, we achieve the fact that using Type-2 Fuzzy System we are able to deal with different users and ways of performing specific behaviours, without requiring to start the learning process from scratch.

4. Future Work

This PhD work presents an original system to learn, recognize and adapt human behaviours in *Ambient Intelligence Environments*. We have proposed different methods to learn and recognize human current activities, developed an adaptive structure to encompass all the user activities changes, compared with other methods proposed in the literature, tested in real environment and summarized long-trend activities. However, activity recognition is an emerging research line in AmI [14], and therefore, it is in continuous evolution and change. In this section, we present some future open research lines raised from the proposals made in this memory.

Multi-User Social Behaviours

Human beings tend to live in social environments where they can interact with other individuals. This interaction is part of our daily routine and influences (or, even, conditions) our own behaviours. For example, in a house different members of a family live together sharing activities, such as, having dinner, where everyone collaborates and influences in each other. For elderly people or people with special needs, this interaction are even more common and deciding.

We will enhance our method to control the social interactions, defining profiles both individual and social ones. Those will contain the preferences of each user aggregated as a unique set in order to accomplish a specific user goal. Additionally, the profiles will provide users with fitted services regarding their current activities.

Furthermore, externally gathered information will be considered in order to enrich the collected data. Those externally gathered information may be extremely useful for understanding social behaviour, the like medical background of patients, preferences of the user specified in advance by the user or calendar information.

Ambient Intelligence and Human Behaviour Ontology

Among AmI aims, the context-awareness sets it apart from the rest. Having the defined context-awareness allows to know how the services should be provided in the environment, when, where and why it could happen, etc.. Projects as *COBRA-ONT* propose an ontology to represent Context-Aware Pervasive Environments [13]. An ontology [24] formally is a description of the concepts and relationships between them. Using ontologies, we will be able to specify the components in AmI environments, including a descriptive representation of the common system vocabulary.

Additionally, as stated before, the activity recognition is emerging as a key research challenge in Ambient Intelligence. The combination between the context-awareness and the activity recognition process will improve the characteristics of the Ambient Intelligent providing more fitted services to the end-users.

Long-Trend Summarization Expert Evaluation. Refining the method

The *Long-Trend Summarization Method* obtains a summary of how a behaviour performance evolves with the time (different days, months or years). However, the extracted sentences has been defined without supervision of the real users. The main objective of this process is to provide the nurses or caregiver with more understandable information about the real resident behaviour

evolution. For this purpose, we want to obtain the expert opinion to evaluate the method and improve it by adding more attributes to provide a more adapted knowledge.

Furthermore, focusing on the proposed method techniques, we would like to generalize the method to avoid having to define the time window to summarize. We would like to define a method that finds the changes studying the own data and its evolution.

Finally, we would like to propose a method to learn both the membership functions and the methods of producing the linguistic summarization.

Sensor Failure. Possibilities and consequences in the model and recognition process

The sensor are devices embedded into the Real Environment that collects the user activities on them. However, as they are hardware devices, they are subject to common fails: energy cuts, communication loss, etc.. We will investigate the effects of sensor failures on the activity inference processing. Besides, we would like to develop a probabilistic model for fault detection and identification for sensors. Our model capabilities will include the detection and identification of bias, drift, and noise in sensor readings. We will also extrapolate the method to multi-sensor networks to control the failures in big environments.

Knowledge transfer between different environments

Knowledge transfer is a quite known tool in organizational development and learning. Its goal consist in transferring knowledge learnt in different organization to another one. This process is equal to the human learning process, looking for patterns to repeat in different situations and scenarios[10]. This concept may be extrapolated to different fields, such as, machine learning or psychology. In concrete, we will use this concept in Activity Recognition, in order to transfer the learnt users' activities to different places.

Repetitive temporal patterns. Other application fields.

Studying repetitive and temporal patterns is a challenge that can be found in different application field, including psychology, psychiatry, ethology, traffic flow, industry, computer science, etc..

Inside our research group, there are some project to develop tools to improve building energy consumption. Our objective is to study how the users' behaviour could influence on the energy consumption of the building. A prediction model will be developed in order to detect events and conditions in specific environments, in order to optimize the user comfort minimizing the energy consumption. Among the studied sources, the users' daily activities inside the building would influence completely the obtained results. Additionally, we will study weather forecast, simulated building structures, etc..

Part II. Publications

1. Detecting and recognizing behaviour patterns from *Stream Data*. Apriori approach.

The journal paper associated to this part is:

- Delgado, M., Ros, M. and Vila, A. Correct behavior identification system in a Tagged World Expert Systems with Applications, 2009, 36, 9899-9906.
  - Status: **Published**.
  - Impact Factor (JCR 2009): 2.908.
  - Subject Category: Computer Science, Artificial Intelligence. Ranking 15 / 103 (Q1).
  - Subject Category: Engineering, Electrical & Electronic. Ranking 16 / 246 (Q1).
  - Subject Category: Operations Research & Management Science. Ranking 3 / 73 (Q1).

Expert Systems with Applications 36 (2009) 9899–9906

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Correct behavior identification system in a Tagged World

Miguel Delgado, María Ros *, M. Amparo Vila

Dept. Computer Science and Artificial Intelligence, ETSIIT – University of Granada, 18071 Granada, Spain

ARTICLE INFO

Keywords:
Tagged World
User behavior
RFID
Frequent itemsets
Regular Grammar

ABSTRACT

This paper presents a system that is able to process the information provided by a Tagged World to identify user's behavior and to produce alarms in dangerous situations. The system inputs are signals from sensors, which are used to recognize correct behavior (action sequences) by Inductive Learning, using Data Mining techniques. The inference engine is a reasoning device that is implemented by means of Regular Grammars. It permits us to control user's behavior. As output, the system produces and sends alarms when a user action sequence is wrong, indicating the erroneous actions, forgotten future, and so on. To test our system, the Tagged World is supposed to be at a house, where we have used RFID technology to control the objects inside it.

1. Introduction

In the recent years, the computer technology has suffered a wonderful evolution to the small computer. This evolution was predicted by Weiser (1993) where he defined the ubiquitous computing as *the method of enhancing computer use by making many computers available throughout the physical environment, but making them effectively invisible to the user*. Following this principle, the devices are made tinier each time. Inside these devices, the hand-held ones and the sensors are emphasized upon.

Many traditional environmental applications base their operations in sensors, for this reason sensors are becoming more and more common in daily life.

There exists different types of sensors to collect information about very different magnitudes: temperature, electromagnetic waves, mechanical effects, some chemical products, radio-frequency signals, etc. Today, systems based on radio-frequency identification (RFID) technology are a key tool in helping to move Weiser's vision closer to reality (Want, 2004a, 2004b). A RFID System is always made up of two components (Finckenzeller, 2003):

- A tag (small silicon chip that contain identifying data and sometimes other information), which is located on the object to be identified.
- A reader which, depending upon the design and the technology used, may be a read or write/read device.

This type of sensors makes the creation of a Tagged World easy .

Definition 1 Koyama, Nakagawa, and Shimakawa (2006). A Tagged World is defined as a smart area that serves to recognize user's behavior by using information about their daily activity. This information is collected by sensors placed (embedded) in the environment.

In particular, RFID sensors allow us to collect information specifically about the different objects that user touches. This permits to develop a system that is able to detect the user actions from sensors information with the goal of finding mechanisms to identify the different actions with the specific activity carried out by the user.

It is obvious that there are a wide range of possibilities in designing this type of system, which result into many projects that propose different alternatives. For example, in Yamahara, Takada, and Shimakawa (2006) a Tagged World project is presented which uses RFID tags that are attached with all objects around the environment. A user brings a wearable computer equipped with a RFID reader to record the access logs to these tags. They use Bayesian networks methodology for the probability statistics in the recognition and the reasoning to provide personalized services to make a more safe and easy life by recognizing and reasoning about the human behavior.

In contrast, in Nakauchi, Noguchi, Sonwong, Matsubar, and Namatame (2003), a method that uses clustering techniques for the classification of the behavior is proposed, through the algorithm ID4.

The clustering techniques are also used in conjunction with a rule based system based (Isoda et al., 2004). A representation model that expresses both the spatial and the temporal relationships is proposed. Moreover, the system provides the user support by coupling the result with the aim of the model consisting of a set of states.

* Corresponding author. Tel.: +34 958244019; fax: +34 958243317.
E-mail addresses: mdelgado@ugr.es (M. Delgado), marosiz@decsai.ugr.es (M. Ros), vila@decsai.ugr.es (M. Amparo Vila).

0957-4174/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2009.01.077

In Philipose et al. (2004), a system to infer Activities of Daily Living (ADL) is proposed. A new paradigm for ADL inferencing leverages radio-frequency identification technology, data mining and probabilistic inference engine is presented to recognize ADLs based on the objects that people use. They use two kinds of sensors. The basic information is obtained by RFID technology, but other sensors are also introduced to fill in the gaps of information that RFID tags could generate. The system represents activities as linear sequences of activity stages, and annotates each stage with both the involved objects and the probability of their involvement.

In this paper, we propose a system to overcome some shortcomings (basically, the inability of generalization and adaptation to change) found in the earlier proposals. We use Data Mining techniques to identify correct behavior, and Regular Grammars to control the implementation of a sequence of actions.

The remainder of the paper is organized as follows: Section 2 is devoted to develop the theoretical basis of our proposal, introducing also the different tools that have been used. In Section 3, we specify the inductive learning process, and in Section 4 the Reasoning System is presented. We conclude by explaining the current architecture of our system and showing some examples about its operation. Finally, some consider the developed implementation and future proposals.

2. Our approach. Theoretical background

In this section, we present some concepts about Tagged World and related constructs which are necessary to know and to understand our approach. In addition, we present a system overview where we present the system structure.

As said before our system tries to solve various problems arising in the task of to recognize user's behavior in Tagged Worlds. We have mentioned the word "behavior" several times before, but without stating precisely its meaning, which is needed as this word has a very general and context dependent semantics. To do this we chose the formal representation introduced in Koyama et al. (2006): *a behavior is composed by actions*. Therefore, we will understand a behavior as an action sequence ordered on time

Definition 2. Let $A = \{a_1, a_2, \ldots, a_n\}$ be the set of possible user's actions in some situation or domain. A user behavior is a finite set of actions:

$$\beta = \{\alpha_1, \alpha_2, \ldots, \alpha_{p(\beta)}\}$$

with $\alpha_j \in A \forall j$, where α_j is performed before α_k if, and only if $j \leqslant k$.

Let us observe that this definition of action does not consider the concrete instant time in which an action is performed. However there are problems in which the information about concrete time is to be relevant. For them a definition of action is needed to consider that time. As we indicated previously we will assume that RFID tags are the only sensors to collect the user information. This allows us to associate each action from the above definition with some specific object. So, we will denote each action with the corresponding object name.

Example 3. If we want to control the behavior `to go out home`, the user has the following possible actions: `to take the mobile phone`, `to take the keys` and `to close the door`. Then, these actions will be represented in our system as: `mobile phone keys door`.

The set of possible actions is always finite but its length depends on the context or domain. Similarly different behaviors may involve a different number of actions.

According to our objectives the system we propose is to be designed to carry out the following tasks:

- *To detect the "key" actions that correspond to a normal behavior.* For this only the actions that are common in any performance of a specific behavior are to be considered. For example, when we study the behavior `to go out home`, we know the user, normally, `to take the keys and to close the door`. However, if he takes a bag or a scarf, it will not be a common action in this behavior. These common actions are to be considered "key" actions and they specifies a behavior, that is, the definition of a behavior is a skeleton one, which is only composed of key actions.
- *To determine whether a sequence of actions as performed by the user is correct or not.* When a sequence of actions arrives to the system, it should indicate whether the sequence is right or wrong with regard to the previously studied and controlled behaviors. Thus, if the user is `to go out home`, and in general, `to take the phone`, `to take the keys`, `to close the door` and `to call the elevator`, the system has to detect any forgotten or wrong action: forgetfulness of the keys or the mobile phone, not closing the door...
- *To generate and send alarms when the sequence of actions is incorrect.* While from a theoretical point of view the main aim of the system is to identify user behavior, from a practical point of view it is very useful to provide a service able to generate and send alarms for the user when the sequence of actions is incorrect.

Besides these tasks, there exists another interesting goal: the Knowledge Mobilization. The aim of this idea is to allow the use of hand-held devices to communicate the alarms to the user (after processing signals on a server to identify the correct behavior).

When we talk about behavior, the user concept is not the same as the concept of the research. This different perception of the behavior could be a problem if we want the system would be used by *anybody*. For this reason, we think that it is convenient to use a simple and obvious behavior concept that would be controlled by the system.

Our objective is to design and implement a system that is able to generate and send alarms when a user omits some key action in a behavior. Thus, "right/correct behaviors" are to be identified, but to do this task we need to take into account different aspects of user's conducts:

- *A user does not carry out the behavior actions in the same order every time*. When a user goes out home, we know that he has to take the mobile phone, to take the keys, to close the door; but it is possible that he has to take the keys, to take the mobile phone, to close the door too.
- *To identify a behavior it is needed for the user to make all the behavior actions.* When a user takes the mobile phone, he could be carrying out various behaviors: to go out home, to speak, to tidy up, etc.
- *A user may carry out intermediate (non key) actions that have not been considered in the skeletal definition of a behavior.* As we have commented before the definition of a behavior is a skeleton one, which is only composed of key actions. However a user may perform some additional non key actions keeping the rightness of the behavior. When a user goes out home, we know that he has to take the mobile phone, to take the keys, to close the door. However, when the user takes the mobile phone, it is possible that he takes the bag before the keys.
- *A user may start a behavior without finishing a previous one.* When a user takes the key and the mobile phone, we could believe that he is going out home. Nevertheless, he could realize that he forgot to make the bed.

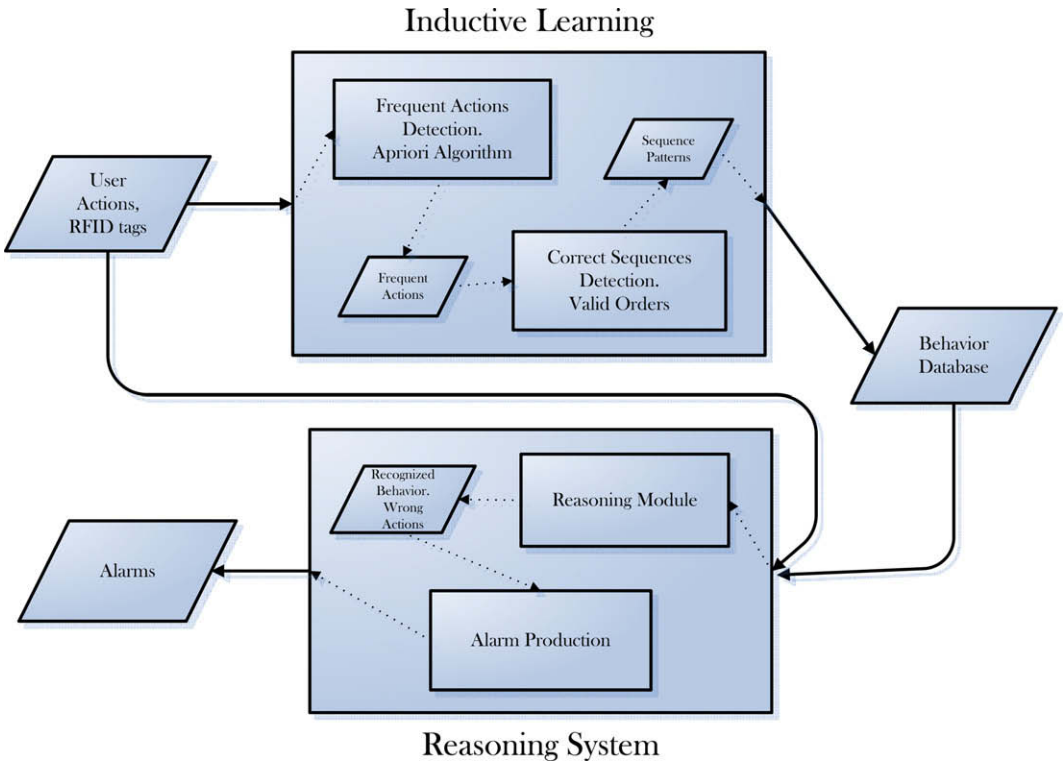


Fig. 1. Designed system diagram.

To summarize the main objective of our approach is to design and implement a system that is able to

- Read the signals from RFID tags.
- Learn the user behavior.
- Identify some concrete user activity sequence as a user known behavior.
- Send an alarm to warn the user in case of detecting a wrong or incomplete activity sequence.

To get this, we need to implement some learning method as well as a reasoning device. At the same time the system would be simple and understandable to the user. So, we need an adaptable system. Therefore, we have designed a system with two main parts: an inductive learning mechanism, whose final output will be a Behavior Database and a system to the recognition of sequences performed by the users, which will be built on the Behavior Database obtained in the previous process. In Fig. 1, we show a system structure diagram.

To process the RFID information, and consequently to construct the Behavior Database, we have used Data Mining techniques, since this information can be represented as strings and continuous data. On the other hand, the Reasoning System is based on Regular Grammar. We use an automaton to represent the user behavior, and its route allows to know if the user activity corresponds with a known user behavior.

In the next two sections, we present the details of these two parts of the system.

3. Inductive learning method

As we have mentioned before, we want to create a Behavior Database from which we could identify a user current sequence of actions to be a specific behavior. Here, we propose an automatic mechanism based on inductive learning techniques from the knowledge we have from user, which has been collected by the sensors in the environment. However let us mention that the Behavior Database could be from other non-automatic or semi-automatic ways: from an expert, studying the environment, from the user experience, etc.

The user activity is collected by using RFID tags and thus it is the identification number that we have assigned to a specific object in our Tagged World. So, we can collect the user activity as a sequence of identification numbers or a sequence of identified objects. This information is stored into an Observation Data Base that can be modelled as a Transactional Database. Then the key actions for each behavior can be identified with the concept of frequent itemset which can be extracted from the Transactional Database. That is, the key actions will be defined as the sequence of events that occur more often in the observed knowledge. Therefore each frequent itemset corresponds to a particular common behavior.

At first glance, to disclose frequent itemsets from the Observation Data Base we can use some of the versions of the Apriori Algorithm. This algorithm was introduced by Agrawal and Srikant (1994) and this allows to obtain the frequent itemsets in a Transactional Database. However the use of Apriori algorithm on our Observation Data Base has two problems that are necessary to solve (the first one, in fact, is less specific than the second one):

- *Assessing the support threshold*. This value will determine those actions that will be in a frequent itemset and those that will not, i.e. which are to be considered "key" (common) and which are not. Thus the support value determines the quality of the final Behavior Database.
- *Loss of sequentiality*. The algorithm Apriori returns itemsets that do not keep any order among its components. However, this is a problem for the system, since all actions of conduct preserve a certain temporal order among them. For example, the behavior to `leave home` is often the sequence of actions `door key`

`mobile_phone`, but `mobile_phone keys door` or `keys mobile_phone door` are also possible and right. However closing the door before without taking hold of the mobile phone or the keys is not correct, and therefore should not be considered.

To solve this problem the next step in the learning process is to find the valid orders of itemset's actions. The idea is to generate all the possible permutations of each frequent itemset and to compare each permutation with the sequences we have in the Observation Data Base. Then, we construct a Behavior set where we add the sequences whose ordering coincides with one of some real observation in the Observation Data Base. The set of valid sequences of each behavior constitutes the final Behavior Data Base. The algorithm is presented in Table 1.

Several researches have proposed Apriori algorithm modification to obtain directly a sequence from the data and not only itemsets, as in Agrawal and Srikant (1995). In our case, this is not a good possibility, so this kind of algorithm is prone to remove much interesting sequences due to the fact that they do not have a high frequency. We mean, in a sequence it is possible that the items that compose it are frequent. However, not all possible orders are frequents, but we want to collect them as an alternative.

4. Reasoning System. Alarm generation

The second part of the system is the reasoning model which does the behavior identification. It has the Behavior Database as base to do the identification of the actions. So, the reasoning consists in a matching mechanism between the user current actions and the Behavior Database. Using the representation of a behavior as a sequence of actions, we can represent the DB as a Regular Grammar, and can use it to recognize that the action has been realized by the user.

The Behavior Databases are considered as the set of words built with the formal grammar, where:

- The Alphabet is the identifiers of embedded sensors in the environment.
- Word of the alphabet (or actions behavioral X) is a management symbol of the alphabet.

Therefore, the identification of the sequence of actions consists primarily in the recognition of a word from the defined formal grammar.

The Regular Grammar allows to define an automaton (Hopcroft, Motwani, & Ullman, 2002) to make the studying word process. We define a finite automaton for each behavior the DB stores. The usual implementation to automaton is using a Graph. However, this representation induces several problems:

(1) *Lost the actions temporality.* Graph's nodes have not got any order between them, in other words, when we are in a node we do not know from which node we come,[1] we are not unaware of the word before. A transaction between nodes implies a read symbol, but not a temporality between them.
(2) *There could exist some valid transactions to some behavior, but not to other.* The transaction representation in a Graph means that between node `a` and node `b`, there exists a transaction, independently the behavior implies the transaction. However, it could be possible that there exists a behavior with the action `a` but not with action `b`. This means that the transactions between `a` and `b` have not to appear. This is a contradiction caused by Graph structure.

[1] We are speaking about traditional automaton.

Table 1
Algorithm to obtain the valid sequences.

```
For each i ∈ behavior sequence
  For each t ∈ Observation Data Base transactions
    Check if all elements in the sequence i respect the t elements order
      Get the position of two t elements
      Check the positions
        If these positions are incorrect in the sequence i
        Get other two t elements
If the sequence is not found in neither of Observation DB transactions, delete
the sequence.
```

(3) *Final state.* The final state indicates whether a word is accepted or not. This has some complications, like point 2. It could occur that a node will be a final state to a behavior but not to other. How do we determine when the sequence is valid and when it is not?
(4) *Several Graphs.* When we observe a person we may control many behavior, and not all of them are connected between them. This implies that we could have several graphs without connection.

Although the use of graph provides more disadvantages than advantages in the behaviors identification, it is possible to use its philosophy to design a support structure that allows the identification of behaviors. The alternative proposal will be a *Behavior Tree*, where each of the branches will represent a user behavior. The tree leaf nodes are identified with the behavior that has to be carried out, while the intermediate nodes symbolize the actions that consists a behavior.

The choice of the tree structure is justified, since this offers the following advantages over the graph:

(1) *Nodes Hierarchy.* It maintains the temporality of the behavior actions by means of the hierarchy of tree nodes. Thus, an action that is in a higher level of the tree will be an action prior to another action that will be in the lower level.

Table 2
Algorithm to Reasoning System.

```
Initialization
  Event ei
  Current node nc
  Second node ns
Process
If the current node nc has a son node nh to where can access reading ei
  If nh is a leaf node
    Output: nh determines the current behavior
    Initialize the current node nc at root
  Else
    If nh has not got sons
      Output: nh determines the next actions and the behavior
      Initialize the current node nc at root
Else
  If the second node nc has a son nh
    If nh is a leaf node
      Output: nh determines the current behavior
      Initialize the current node nc at root
    Else
      If nh has not got sons
        Output: nh determines the next actions and the behavior
        Initialize the current node nc at root
      Else
        If ei is an omitted action
          Output: current branch, indicating omitting actions and the behavior
      Else
        If ei is the first node in other branch
          Assign ns to nc
End
```

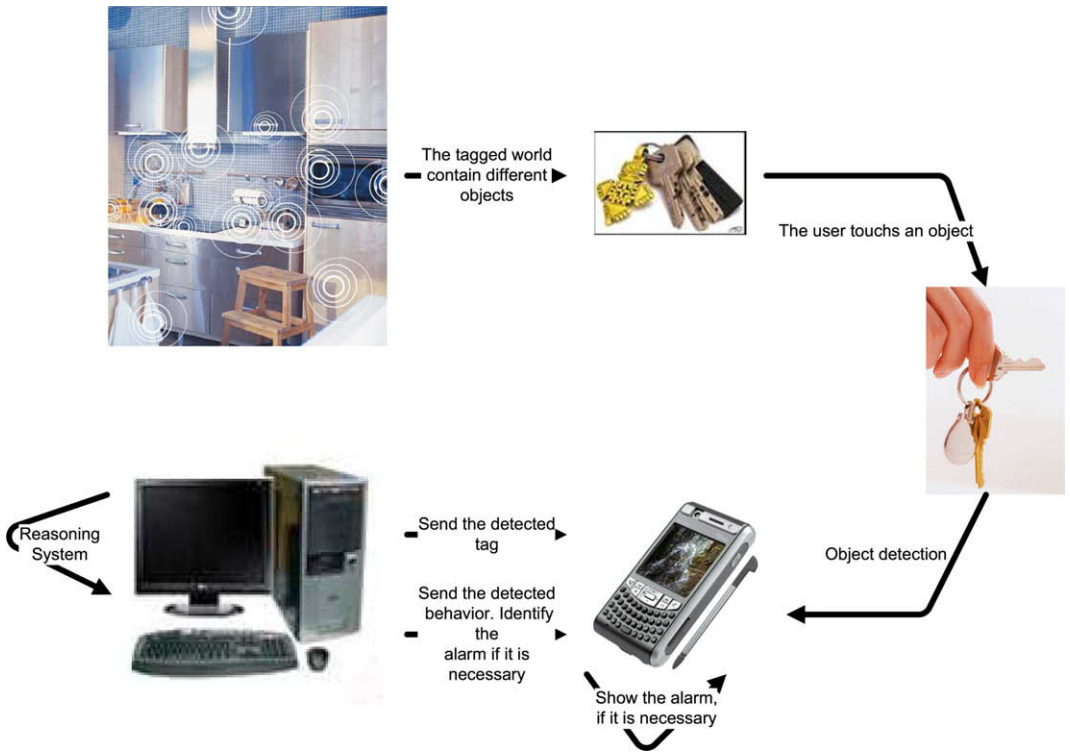


Fig. 2. Correct behavior identification system in a Tagged World.

(2) *Ramification.* It makes a ramification of concrete actions. To reach a leaf node it is necessary to move in advance by all higher levels of this group, and therefore, for all actions that define the behavior.
(3) *Common root.* The tree will have a common root to all behaviors, which allows the Reasoning System only to work with one structure.
(4) *Restricted search.* The tree scan allows us to limit the possibilities and to restrict the search in the Behavior Database.

This structure will not be enough to recognize a sequence of actions, but we should implement an algorithm to browse the tree and to get the correct behavior.

The system of reasoning is based on two key aspects: the browse of the support structure and a matching mechanism between the user current sequence and the *Behavior Tree*. We have designed an algorithm that try to have in account all possible situations. We do a preorder browse with some modifications as backward steps or changes of branches. This algorithm is presented in Table 2. The algorithm output will be of four types:

(1) Nothing.
(2) Only the identified behavior.
(3) The identified behavior and the recommended future actions using the identified behavior as reference.
(4) The identified behavior and the actions have been omitted by the user.

These outputs are transmitted to the Production Alarm Module, which will create an alarm when the situation demands it.

Alarms monitor provides information to the user about what happened in the system. Thus we will generate an alarm in the following situations:

- When the algorithm output recommended future actions, we would generate an alarm that informs the user: "According to the behavior X, you should not forget to do the actions Y", where the actions Y are the future actions which are being identified by the algorithm.
- When the algorithm output omitted actions, we would generate an alarm that informs the user: "According to the behavior X, you have forgotten to do the actions Y", where the actions Y are the omitted actions that have been identified by the algorithm.

Table 3
Transactions to the Go out home behavior.

Transaction	Objects
t1	Shoes MobilePhone Bag Keys OutDoor Elevator
t2	Shoes Keys MobilePhone OutDoor Elevator
t3	Bag Keys OutDoor Elevator
t4	MobilePhone Keys Bag Elevator OutDoor
t5	Keys Bag MobilePhone
t6	MobilePhone Keys OutDoor
t7	Shoes Bag Keys OutDoor
t8	Bag Keys Elevator OutDoor
t9	Keys MobilePhone OutDoor
t10	MobilePhone Keys OutDoor

Table 4
Possible permutations.

s_1	{Keys, OutDoor, MobilePhone}	Deleted
s_2	{Keys, MobilePhone, OutDoor}	
s_3	{OutDoor, Keys, MobilePhone}	Deleted
s_4	{OutDoor, MobilePhone, Keys}	Deleted
s_5	{MobilePhone, OutDoor, Keys}	Deleted
s_6	{MobilePhone, Keys, OutDoor}	

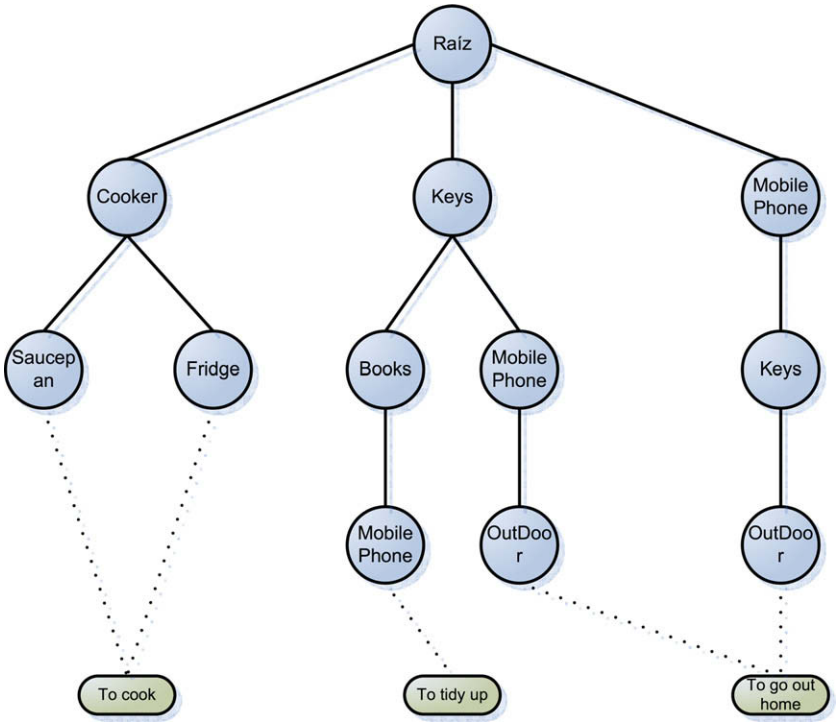


Fig. 3. Behavior tree. Example.

5. Architecture

The implementation of these systems must follow a basic principle: its implementation must not affect the user's daily life. In other words, the user may not be aware of the system, since the system should be invisible to him and his habits.

As we have indicated in the introduction, the Tagged World are environments that have been equipped with a sensor package whose signals are collected and processed to produce a service to the user. This indicates two needs for interaction with such systems: sensors, which are the input of the system and a computer, which will collect signals from input modules and process it to produce a service to the user.

Also, we mentioned above that we need to know when a user touches an object. We resolve this task using a RFID HF System. We have selected this frequency because we need to know that the reader range is not large, and the system will interact with the user continuously. A RFID System consists of RFID tags, RFID readers and a base computer. We have not designed a specific middleware, but we have used the fabric middleware from the RFID readers. In that implementation, we have used a PDA with a SD RFID Reader to do the tags readers. A system diagram is presented in Fig. 2.[2] The system architecture is an architecture Client/Server, where the client is implemented in the mobile device and the server is a web server and it is set up in the computer. This will achieve one of the basic aims of the Tagged World: knowledge mobilization.

The client has been generated as a Client mobile, where it serves only as an intermediary between the Reasoning System (setting up on the server) and the user. It is responsible for reading the passive tags, for sending the read tags to the server, for receiving the results and for sending, where appropriate, the alarms.

On the other hand, the server implements the Reasoning System. It receives two inputs: the user activity and the Behavior Database. The user activity is collected by the client and transmitted to it. Whether the system is learning or not, these inputs will be used.

- If the system is learning, the system has to execute the inductive learning process.
- If the system is reasoning, the system has to check if the read tag is in the system and if it is valid, then it has to reason with the identification number.

The server has to keep an object database, where the system controls all tags that are situated in the environment. It stores as well as the identification number, the identified object and another object information.

The system has been implemented using JAVA technology (Eckel, 2006), JAVA2 and JAVA Web Service for the server and JAVAME for the Client mobile.

6. A real example

For our experiments, we have situated the system in a specific environment: a house. This was a small experiment, so we have only provided with RFID tags a set of objects: the bath, the tap, the door, the mobile phone, the bag, the key, the bar of soap, the computer, the bed or the chair. Each object has a RFID tag, so the object database has to know that object.

We start with the inductive learning operation, thus we present the sequence pattern extraction to "Go out home" Behavior. We begin from the set of transactions in Table 3. If we apply the Apriori algorithm with a supp of 0.8 over that data set, we will get the following itemsets:

Keys OutDoor MobilePhone

From this itemset, we have to obtain the sequence patterns. For the first time, we obtain all possible permutations and we take out only the valid sequences. The solutions are shown in Table 4.

[2] Some figures has been taken from Want (2004a).

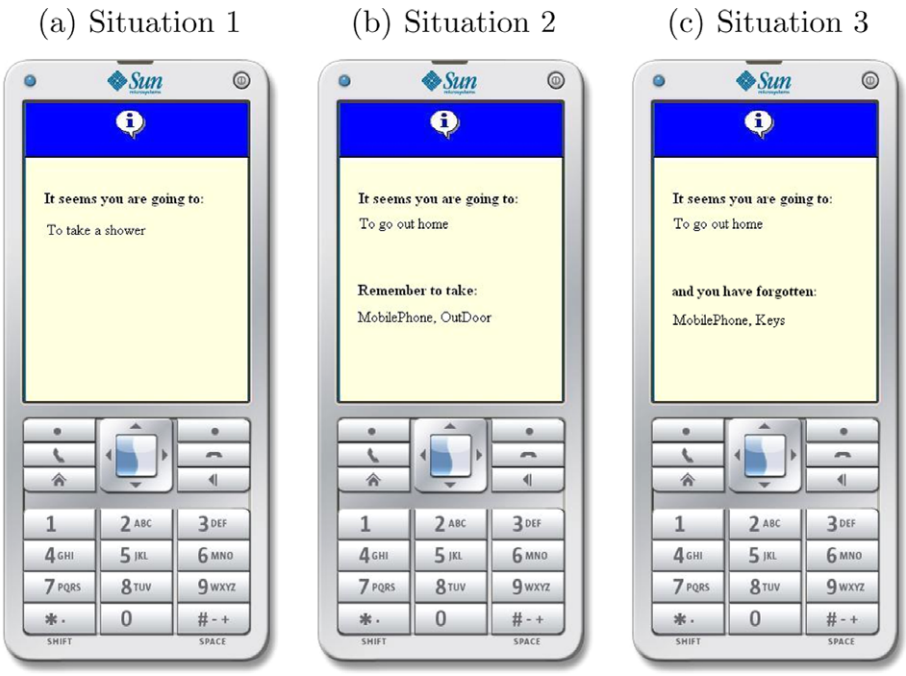


Fig. 4. Output situation.

This method should be repeated with each behavior to control obtaining the Behavior Database.

When we have learned the correct actions to every behavior, we could reason about the user activity. The information received from sensors is sent to the Reasoning System, which generates the "Behavior Tree" to help the reasoning the first time that it is used.

We have extracted the following sequence patterns from the Behavior Database to show the defined "Behavior Tree" (see Fig. 3).

(1) Cooker Saucepan → To cook.
(2) Cooker Fridge → To cook.
(3) Keys Books MobilePhone → To tidy up.
(4) Keys MobilePhone OutDoor → To go out home.
(5) MobilePhone Keys OutDoor → To go out home.

Finally, we present the final implemented application. For this, we propose some different examples above "House environment" to demonstrate the operation system. We are going to pose some different situations and the system response. For each situation, we have shown the output with the Sun Java Wireless Toolkit 2.5 for CLDC (Yuan, 2004).

- *Situation 1.* We suppose the user actives the sensor the bath tag. With this object, the system could recognize two possible behaviors:
  - bath tap → to take a bath.
  - bath soap → to wash his hands. So we need to wait for another object to identify the behavior. If we received the tap, we would say that the user will take a bath (see Fig. 4a).
- *Situation 2.* We suppose that the user takes keys. With this object, the system only could recognize one behavior: to go out home. So, we could recommend to the user some actions that he should not forget: to take the mobile phone and to close the door (see Fig. 4b).
- *Situation 3.* We suppose that the user tries to close the door. This implies that he is going out home. However, he has got neither keys nor mobile phone. The system reminds him that he has forgotten these objects. This is the principal aim of this system (see Fig. 4c).

7. Conclusions

This article presents a solution to a new problem, and booming now, the Tagged Worlds. The solution uses techniques of Data Mining and Knowledge Mobilization allowing a simple generalization to any field of study.

The use of data mining techniques, in particular mining sequence for inductive learning, and of Regular Grammars for recognizing the actions proposes a new theoretical solution to this problem. And, moreover, the use of technologies Mobile provides a solution from a practical point of view, by facilitating the implementation of these systems in the real world and for real users.

In future works, we will include the uncertainty concept over the RFID reader. The RFID system generates uncertainty in its readers, so the reader could fail or the RFID tags could break down. So we have two types of uncertainties we have to process: probabilistic uncertainty and possibility uncertainty. This will affect the inductive learning, because it may assume the uncertainty of the user behavior and manage it.

References

Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules. IBM Research Report RJ9839, Junio.

Agrawal, R., & Srikant, R. (1995). Mining sequential patterns. In *Proceedings of the international conference on data engineering*.

Eckel, B. (2006). *Thinking in Java*. Upper Saddle River, NJ: Prentice Hall. 1482 p.; 24 cm.

Finckenzeller, K. (2003). *RFID handbook. Fundamental and applications in contactless smart cards and identification* (2nd ed.). Munich/FRG: Carl Hanser Verlag.

Hopcroft, J. E., Motwani, R., & Ullman, J. D. (2002). *Introduccion a la Teora de Automatas, Lenguajes y Computacin* (Segunda ed.). Addison Wesley.

Isoda, Y., Kurakake, S., & Nakano H. (2004). Ubiquitous sensors based human behaviour modelling and recognition using a spatio-temporal representation of user States. In *Proceedings of the 18th international conference on advanced information networking and application*.

Koyama, K., Nakagawa, K., & Shimakawa, H. (2006). Embedded action detector to enhance freedom from care. *HCII 2006*.

Nakauchi, Y., Noguchi, K., Sonwong, P., Matsubar, T., & Namatame, A. (2003). Vivid room: Human intention detection and activity support environment for ubiquitous autonomy. In *Proceedings of 2003 IEEE/RSJ conference on intelligent robots and systems*.

Philipose, M., Fishkin, K. P., Perkowitz, M., Patterson, D. J., Fox, D., Kautz, H., et al. (2004). Inferring activities from interactions with objects. *Context-Aware Computing. Pervasive Computing, 3*(50–57).

Want, R. (2004a). Enabling ubiquitous sensing with RFID. *IEEE Computer, 37*(4), 84–86.

Want, R. (2004b). *RFID: A key to automating everything* (Vol. 10). Scientific American Magazine.

Weiser, M. (1993). Ubiquitous computing. *Perspectives Article for ACM Interactions*.

Yamahara, H., Takada, H., & Shimakawa, H. (2006). Rapid adaptation of behavioural to individual lifestyle. In *Proceedings of the 20th international conference on advanced information networking and application*.

Yuan, M. J. (2004). *Enterprise J2ME: Developing mobile Java applications*. Upper Saddle River, NJ: Prentice Hall PTR.

2. Processing temporal information to manage the uncertainty of behaviour

- Ros, M.; Delgado, M. and Vila, A. Fuzzy method to disclose behaviour patterns in a Tagged World Expert Systems with Applications, 2011, 38, 3600-3612
  - Status: **Published**.
  - Impact Factor (JCR 2010): 1.926.
  - Subject Category: Computer Science, Artificial Intelligence. Ranking 34 / 108 (Q2).
  - Subject Category: Engineering, Electrical & Electronic. Ranking 50 / 247 (Q1).
  - Subject Category: Operations Research & Management Science. Ranking 15 / 75 (Q1).

Expert Systems with Applications 38 (2011) 3600–3612

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Fuzzy method to disclose behaviour patterns in a Tagged World

María Ros *, Miguel Delgado, Amparo Vila

Department of Computer Science and Artificial Intelligence, ETSIIT – University of Granada, Granada 18071, Spain

ARTICLE INFO

Keywords:
Tagged World
Frequent itemsets
Sequence patterns
α-cut
Behaviour
Fuzzy temporal window

ABSTRACT

The main objective of the Ubiquitous computing is to include technology on the user life without modifying their daily routine. A lot of kinds of applications base their operations on using sensors situated on a Tagged World, which is a smart area that serves to recognize user's behaviour using information about their daily activity. It manages information collected by sensors to identify the user behaviour and to provide some services according to inferred behaviours. In this paper, we present a specific method to extract *behaviour patterns on time* from the collected sensor signals. The method is based on frequent itemsets that represent the common actions of the user. After, we obtain sequence patterns from these extracted itemsets. Our method should pay attention to the user activities because these have non-random imprecision by definition. Thus, we have designed a method to handle this imprecision, through establishing a temporal constraint that is called *Fuzzy Temporal Window*.

1. Introduction

In recent years, Ubiquitous systems are a reefs in the current technologies and researchers. Since Mark Weiser introduced the *Ubiquitous Computing* term (Weiser, 2008), many applications have arisen within this field. The main idea consists of including computers on daily life without disturbing normal user activities. Nowadays, the current trend in *Ubiquitous Computing* is to provide user environment with different mechanisms to collect information. This information is used to help users.

Our proposal develops a system to provide services in specific situations. To understand the problem, we will focus on these people that need some help, since their mobility or some physical abilities are reduced. Somebody should take care them. We propose a mechanism to reduce the dependence of these people on their carer.

Our objective is to study user daily activities and to provide some service when they have done something wrong or have forgotten important things, without altering their daily activities, in the sense of the Example 1.1 shows.

Example 1.1 (*An elderly woman is going to bed*). We suppose an elderly woman is going to bed. When she is in the bedroom, she sits down on her bed, she takes her slipper off and she turns the light off. However, she has forgotten to put the burner out and take her pills. The System is sending alarms to inform her about her forgotten actions (see Fig. 1).

As we have indicated before, we need a mechanism to gather the user information: *sensors*. Sensors are devices that measures what is happening in the environment. In this work, we are considering RFID sensors that let us identify objects in the environment with an unequivocal number. RFID systems's (Sheng, Li, & Zeadally, 2008) main function is automatic identification of objects or persons using radio waves. A typical RFID system consists of three elements (Want, 2004): a tag attached to the objects or persons, a reader that creates an RFID field for detecting radio waves and a computer network to connect the readers. RFID technology has been aimed at a limited number of applications (Domdouzis, Kumar, & Anumba, 2007), i.e., access control and electronic toll collection (Blythe, 1999; Fennani & Hamam, 2008), to many new application areas, i.e., industrial and commercial (Jalelkis et al., 1995), entertainment (Rashid, Bamford, Coulton, Edwards, & Scheible, 2006) or healthcare and pharmaceutical (Wu, Kuo, & Liu, 2005). Nowadays other uses are arising such as intelligent spaces, where gather the human activity and develop a system to detect the user actions from this information. Therefore, the goal of such systems is to find a mechanism that would identify the different actions with the specific activity performed by the user.

By placing sensors in an environment, we build an intelligent space that is able to capture information about the user position, the behaviours, the surrounding and so on (Yamahara, Takada, & Shimakawa, 2007). Some examples can be found in Kidd et al. (1999) and Mori et al. (2005). However, these two examples are developed without reasoning and inferencing.

The development of Kidd et al. (1999) and Mori et al. (2005) is the Tagged World (Koyama, Nakagawa, & Shimakawa, 2005) concept. A Tagged World is defined as a smart area that serves to recognize user's behaviour using information about their daily

* Corresponding author. Tel.: +34 958244019; fax: +34 958243317.
E-mail addresses: marosiz@decsai.ugr.es (M. Ros), mdelgado@ugr.es (M. Delgado), vila@decsai.ugr.es (A. Vila).

0957-4174/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2010.09.014

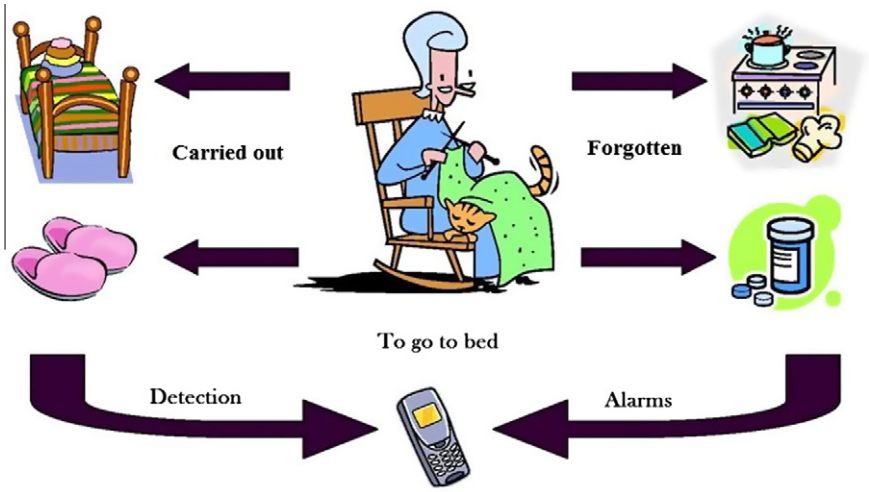


Fig. 1. Example 1. An elderly woman is going to bed.

activity. This information is collected by sensors placed in the environment. The Tagged World project (Koyama et al., 2005; Yamahara et al., 2007) was developed in the University of Ritsumeikan, Japan and consists of providing appropriate personalized services for each user, to make their life easier and safer by recognizing and reasoning the human behaviour. They use a wearable computer as a Pocket Assistant. This computer compares an access log with patterns to recognize human activities. The system is based on a Bayesian network and obtains the results with a probability value for every extracted behaviour.

Other alternative is the project proposed by Philipose et al. (2004). They propose a system to infer Activities of Daily Living in a known room from interactions with objects. They present a new paradigm for ADL inferencing leverage radio-frequency-identification technology, data mining and a probabilistic inference engine to recognize ADLs based on the objects people use. They basically use RFID technology with other sensor streams. The system represents activities as linear sequences of activity stages, and annotate each stage with the involved objects and the probability of their involvement.

In Ros, Delgado, and Vila (2009) Delgado, Ros, and Vila (2009) we propose a system to identify correct behaviour from the user activities using Data Mining Techniques on a Tagged World. The system is divided into two main parts: inductive learning mechanism, which produces a behaviour database and a reasoning system for the recognition of sequences that uses this database. The first stage uses Frequent Itemsets, while the second one does Regular Grammar.

The System uses the Frequent Itemset concept to process the sensor information, without keeping in mind the time if the action. However, the time is an important characteristic in this kind of systems and it should be studied in general. There are a lot of proposals about time constraints for sequence mining. Fiot, Laurent, and Teisseire (2007) present a soften temporal constraints used for generalized sequential pattern mining. They are established between the itemsets that are in the sequence patterns, pointing out the minimum and maximum distance among them. This proposal gets to establish this constraints using three values: two limits of time between two itemsets and a sliding window during which the records may be grouped into one itemsets. In our case, we set only one constraint, a temporal window, during which the actions may happen. In this version of our project, we have preferred to study the temporal relationship in a period of time, independently on the space among the actions if they were in the Window.

This paper is organized in six sections. In Section 2 we present the formal problem to solve. In the next section, the crisp model is presented explaining the method to obtain the knowledge for a specific behaviour. In Section 4 we present the Fuzzy Temporal Window concept and its generalization to adjust the window to the user activities. Section 5 presents the method to work with the uncertainty of the problem. As well, we present an illustrative example of operation model. To evaluate the method, in Section 6 we present some results we have obtained in our research, as well as, a statistical test which proves the method is better than the one presented in Ros et al. (2009) and Delgado et al. (2009). Finally, the conclusions and future works are reported in Section 7.

2. Formalization of the problem

This section defines the formalization of the problem: "*to obtain sequence patterns that identify the user behaviours in a specific domain and context from the information provided by a sensor network.*" In the following, we introduce the objectives of the system.

Definition 2.1 (*Action*). An action, γ in $\Gamma = \{\gamma_1, \ldots, \gamma_N\}$, is an activity (fact) that happens using a specific object.

The actions are usually relevant depending on the time and it is suitable to know when an action happens on time.

Definition 2.2 (*Action on time*). An action on time, γ_i^t, is an activity that happens over a specific object in a known time and it is denoted as a pattern $\gamma_i^t = (\gamma_i, l_j)$ $i,j \in N$, where $\gamma_i \in \Gamma = \{\gamma_1, \ldots, \gamma_N\}$ is an action and l_j is an established time when the action happens.

The associated time of an action, l_j, could be represented as specific moments or as a temporal label with the condition that we know an order relationship between their elements. These definitions give us the basic elements to introduce the main concept, which is the behaviour.

Definition 2.3 (*Behaviour*). Let $\Gamma = \{\gamma_1, \ldots, \gamma_N\}$ be set of possible user actions in some situations or domains. An "user behaviour" β is a finite set of actions:

$$\beta = \{\gamma_1, \gamma_2, \ldots, \gamma_{p(\beta)}\}$$

with $\gamma_j \in \Gamma$ $\forall j$, and where γ_j is performed before γ_k iif $j \leqslant k$.

In this definition, we do not pay attention on the time, although a behaviour happens in a specific moment on time.

Definition 2.4 (*Behaviour on time*). Let $\Gamma = \{\gamma_1, \ldots, \gamma_n\}$ be set of possible user actions in some situations or domains. Let τ be the temporal line, then an "user behaviour on time" β_{T_l} is a finite set of pair such as:

$$\beta_{T_l} = \left\{(\gamma_{T_1}), \ldots, \left(\alpha_{T_{n(b)}}\right)\right\} = \left\{(\gamma_1, t_1), \ldots, \left(\gamma_{n(b)}, t_{n(b)}\right)\right\},$$

where $\gamma_j \in \Gamma \ \forall j$, where $gamma_j$ is performed before γ_k iif $j \leqslant k$, $t_j \in [0, \tau] \ \forall t_i < t_j$ if $i \geqslant j$.

Our system's input is a database composed of user observation, i.e., actions performed by the user. In the rest of this paper, we call this database as Observation Data Base (ODB). In our case, we collect an action when an user touches an object; therefore, every time the user touches an object, we will add an action on the database. Every row of this database represents the user activities in the entire day, i.e., the sequence of actions performed by an user all day long. The ODB is represented by a transactional database T, where every column represents the touched objects, i.e., a possible action from Γ set, $\Gamma = \{\gamma_1, \ldots, \gamma_N\}$.

As we have indicated before, in general, a behaviour has associated a moment in the day when it happens. However, this moment is not a fact: the user hardly ever makes the actions at the same moment every time. Therefore, when we introduce the time in the problem a new difficulty arises in it: the time is not something exact. The time has vagueness that should be studied to minimize its effects.

3. Model without time considerations (Crisp Model)

In this section, we present a first approach to the problem explained in Section 2. It extracts our goals, (the user behaviours), without obtaining a *stationary* model. Under this constraint, no vagueness is present and then we use the term "crisp" to qualify our model. This approach is explained in Ros et al. (2009) and Delgado et al. (2009) so here, we only present a brief summary.

First of all, we should study what the input information is and what kind of knowledge needs our system. As we have indicate before, the system is designed to gather the inputs from a collection of sensors. However, these sensors work for the entire day, so we need to limit these sequences. Not only does it let us reduce the input data but adapt the behaviour to the user. To make this distinction, we should ask the users or an expert instead about the habits. An expert could indicate how a "normal" behaviour should be developed, that is, between what objects we could find the correct sequences. In this version, we consult an expert about where we have to study the database to find a specific behaviour.

In short, firstly, our System has to obtain the actions that could be part of the behaviour, and next decides which sequences of these actions are correct. We develop these goals in two stages: obtaining common actions and obtaining the correct sequences.

3.1. Obtaining common actions

In this first step, we have to identify the input data with the knowledge our system needs. The model (Delgado et al., 2009) matches a behaviour with the *user common actions*: the actions that the people normally perform to make a specific behaviour. In the Example 3.1 we present an possible situation to understand what the user common actions are.

Example 3.1 (*Common actions*). Let us considerer the `to leave home` behaviour which contains always two actions, `to take the keys` and `to close the door`. However, if the user only `to take a bag` or `to take a scarf`, it will not be a common action in the studied behaviour of `to leave home`.

As we have mentioned above in this paper, we represent the information in the *ODB*, which is provided by the sensors in the Tagged World, as a Transactional Database T. This structure let us apply a great variety of techniques of Artificial Intelligence to extract information. In this case, we have to get the objects that are more representative in the database, that is, the items of the database that are frequent. Therefore, we have chosen the Frequent Itemsets, a Data Mining technique, to try to solve the problem. We identify the frequent itemsets with the common actions of a behaviour: every frequent itemset corresponds to a particular common behaviour (Delgado et al., 2009). The extraction of common actions is made by an algorithm derived from the Apriori Algorithm (Agrawal & Srikan, 1994). An itemset is a set and, like one, there is no order relationship among its elements. This is a trouble, because a behaviour should have a known order relationship among its items to be quite right. Thus, we have to supervise which sequences of these itemsets are valid.

3.2. Obtaining correct sequences

After obtaining the common actions, we have to extract the correct sequences for the behaviour, although, before we should know when we could affirm that an itemset maintains a valid order relationship. The Example 3.2 shows an situation where the order relationship between the actions is a key factor.

Example 3.2 (*Valid order*). Let us considerer a `to leave home` behaviour. The itemset defines this behaviour as `door key mobile_phone`. To these actions, it could think in two possible order relationships: `mobile_phone keys door` or `keys mobile_phone door`. Nevertheless, we do not accept any order relationship where the door were before the mobile_phone or the keys.

To find these correct order relationships between the elements from an itemset, we construct all permutations of each frequent itemset, thus obtain all possible sequences of actions performed by the user as identify the Tagged World. With this step, we extract all possible order relationships between the elements, even though not all of them are correct. We only accept those permutations whose actions are ordered as in the ODB.

Definition 3.1 (*Correct sequences*). Let T be a Transactional Database that represents an *ODB* and let I be an itemset obtained from *ODB* for a specific behaviour, denoted by b. Then a correct sequence p is a permutation of I where the order between its elements are defined in the *ODB*.

We have designed an algorithm (Delgado et al., 2009) "ad hoc" to detect whether a *correct sequence* is found. Those sequences that the algorithm considers they are correct will be included in a final database called as *Behaviour Database*. This database stores all valid sequences for the different behaviours that have been studied.

The algorithm compares a possible sequence (a permutation of the itemset) with the items of the ODB. If it finds the same order relationship in both worlds, the possible sequence is accepted as a part of the *Behaviour Database*. In any other case, the possible sequence will be ruled out. We present an example about this process in Example 3.3.

Example 3.3 (*Correct sequences*). Let us consider an example of `to leave home` behaviour. The ODB is showed in the Table 1, represented as a transactional database T. If we apply the Apriori Algorithm over T, with support = 0.9, we obtain the itemset {Keys, ODoor, MobilePhone}. From this itemset, we obtain the permutations showed in Table 2. However, all sequences are not

Table 1
Leave home ODB.

Trans	Elements
t_1	Clothes, Shoes, MobilePhone, Bag, Keys, ODoor, Elevator
t_2	Shoes, Bag, Keys, MobilePhone, ODoor, Keys2, ODoor2, Elevator
t_3	Tap, Towel, BathroomDoor, BedroomDoor, Clothes, Shoes, Bag, Keys, MobilePhone, ODoor, Keys2, ODoor2, Elevator
t_4	Shoes, Keys, MobilePhone, Bag, ODoor, Elevator, Keys2, ODoor2
t_5	Shoes, Bag, MobilePhone, Keys, ODoor, Keys2, ODoor2, Elevator

Table 2
Possible permutations.

s_1	{Keys, ODoor, MobilePhone}
s_2	{Keys, MobilePhone, ODoor}
s_3	{ODoor, Keys, MobilePhone}
s_4	{ODoor, MobilePhone, Keys}
s_5	{MobilePhone, ODoor, Keys}
s_6	{MobilePhone, Keys, ODoor}

valid, according to the order relationship of actions, in the ODB. For example, the sequences s_2 and s_6 are involved in the ODB, whereas the rest could never happen.

We would like to clarify that our objective is to find possible valid sequences and not frequent sequences. From this point of view, we use the Frequent Itemset technique to limit the elements that compose the behaviour, because we only need the common actions (or, in our case, the frequent actions). Following our method, we can face up to any user sequence of actions and not only the frequent situations.

This approach is completed with a Reasoning System (Delgado et al., 2009), which extracts the current user behaviour and provides a service when it will be necessary. This part uses the *Behaviour Database* as an input together with the current activity of the user: the objects that the user is touching in that moment. The System infers the current user behaviour and check if it is correct according to the *Behaviour Database*. The final point of our approach is to send alarms when the inferred behaviour is not realized correctly. In the Example 3.4, we show the alarms that should be sent for an specific situation.

Example 3.4 (*Reasoning process*). For the case described in the Example 3.3, we know our *Behaviour Database* only has s_2 and s_6 as the correct sequences for the `to leave home` behaviour.

Let us suppose that the user wants to leave home. Then, he/she tries to close the door, so the system will infer he/she is leaving home. However, he has got neither keys nor mobile phone. The system reminds him/her that he/she has forgotten both objects.

Up to now, we have not paid attention to the moment when the action happens. However, this information could be quite useful to establish relationships between the elements. This idea is developing in the next section.

4. Managing actions on time

In the previous section, we have presented a system to extract user behaviours from information collected by sensors. However, we have needed an expert to limit the part of ODB where we should look up in, because of we do not know when a behaviour starts or ends in the sequence of touched objects. To avoid this dependency, we need to design a system that use other characteristic of the problem to limit the ODB. In this point, we are aware of the fact that generally all behaviours are realized in a specific and repetitive time.

Example 4.1. Every day, Rose leaves home at 8:30 to go to work.

Behaviours are not something precise, that means, he/she usually makes it at the same time roughly. Therefore, we can establish an adjusted interval in which we are certain of the behaviour happens. This raises a new problem: how we establish the limits of the interval to detect a specific activity. They could be set through using the knowledge about the user or using a specific width or etc. But, actually, ranges should consider non-random imprecision of the situations.

Example 4.2. Let us suppose that `Rose leaves home at 8:30`, then we could supervise actions that happen "about 8:30", for instance, from 8:20 to 8:40.

The first option is to consider a "crisp" interval where every item would have the same probability to belong to the behaviour, that is, the actions realized in this period of time could always appear when we study the behaviour. This interval is called as *Temporal Window* and determines a subset for each ODB tuple (see Fig. 2).

Nevertheless, if every item had the same importance in the interval, some actions which in fact do not belong to the behaviour could be included in the correct sequences. In the Example 4.3 we show a situation where some actions are more important than others.

Example 4.3. Let us suppose we want to extract the correct sequences to `to leave home` behaviour. Following with the Example 4.2, we are supervising an interval focused on 8:30, and situated from 8:20 to 8:40. Let us suppose we know the following actions belonged to the interval.

- 8:21 Making the bed.
- 8:29 Catching the mobile phone.
- 8:30 Catching the bag.
- 8:31 Closing the door.
- 8:32 Taking the elevator.
- 8:40 Driving the car.

Then, as every action in the interval has the same importance, we could affirm that "making the bed" or "Driving the car" is as important as "Catching the mobile phone" or "Catching the bag" in the behaviour "Leave home". But, actually, this is not true. We have to design a method to provide some actions with more importance than other kind of actions, that is, a method to soft the interval to adjust these differences.

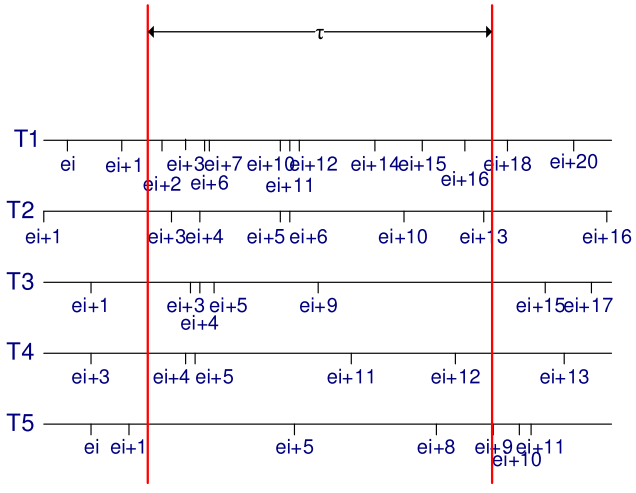


Fig. 2. Temporal Window over an ODB.

The Example 4.3 clearly shows that every actions have not the same importance inside the interval. Furthermore, we can conclude that actions are near the interval centre are more important than actions far away from the interval centre. So, we can attach importance to each action depending on its situation in the interval. To do it, we use the Fuzzy Logic Techniques, assigning a Fuzzy Set to every interval on the temporal line. This interval is called as *Fuzzy Temporal Window* (see Fig. 3). In this case, every action in the studied interval has related an importance degree established by the membership degree to the Fuzzy Set.

4.1. Model formulation

Throughout this subsection, we present the formal representation to *Temporal Window* and *Fuzzy Temporal Window* (Ros, Delgado, & Vila, 2008), as well as an explanatory example to check the process realized by the Windows over the ODB.

Definition 4.1 (*Temporal Window, W*). Let i be an instant of time related to a specific behaviour β and let I be the interval of time defined around t according to the semantic knowledge about β. Let *ODB* be the Observation Data Base and $t \in ODB$ a tuple from ODB. Let $\gamma_{T_j} \in t \forall j$ be an *action on time* $\gamma_{T_j} = (\gamma_j, l_j)$. We define $\Omega_j = (\gamma_{T_j}, \delta_j)$, associated with t, where δ_j is the membership degree of l_j to the interval I.

Then, we define a Temporal Window, *TW*, as the interval I where $m_j = 1\ \forall \Omega_j \in TW$.

Example 4.4. Let us suppose the System have collected the database represented on Table 3. In this table, we have identified every action with the object that the user has touched to make the action. For instance, the action "Putting the shoes on" is represented on the Table 3 as "Shoes", and so on.

Our objective consists of "extracting the correct sequence of actions related to a specific behaviour", in this case, we want to study the following behaviour.

Rose leaves home at 8 : 30.

First of all, we should extract the set of actions we have to study. According to the knowledge about the behaviour, we establish the Temporal Window *TW* between 8:25 and 8:35, so we attempt importance to every action on time. The result of applying *TW* over this ODB is showed in Table 4.[1] As we observe, the actions whose time label is collected in the *TW* has a membership degree equal to 1, while for those are not in the *TW* their membership degree is equal to 0. Like this, we distinguish the actions we have to study for a specific behaviour.

Definition 4.2 (*Fuzzy Temporal Window, FW*). Let i be an instant of time related to a specific behaviour β. Let f_s be a fuzzy set defined around i according to the semantic knowledge about β. Let *ODB* be the Observation Data Base and $t \in ODB$ a tuple from ODB. Let $\gamma_{T_j} \in t \forall j$ be an *action on time* $\gamma_{Tj} = (\gamma_j, l_j)$. We define $\Omega_j = (\gamma_{T_j}, \delta_j)$ where δ_j is the membership degree of l_j to f_s. Then, we define a Fuzzy Temporal Window, *TW*, as an temporal interval I associated with f_s and where $\forall \delta_j \in \Omega_j \Rightarrow \delta_j = \mu_{f_s}(l_j)$.

Example 4.5. For the case described in the Example 4.4, now, we apply a Fuzzy Temporal Window *FW* defined as the fuzzy set f_s. In this example, we define the fuzzy set f_s as a trapezoidal function represented as a $T = (a, b, c, d)$, where a, b are the downer limits of the function, whereas b, c are the upper limits. To be precise, our trapezoidal function T is defined as $T = (8{:}25, 8{:}28, 8{:}32, 8{:}35)$. The result of applying *FW* over this ODB is showed in Table 5.[2] Using this Window, we attach different importance degree to every action, distinguishing between the importance of the actions in the specific behaviour. For instance, the importance of the action "Putting the shoes on" (a_2) should not be the same of the action "Catching the keys" (a_5).

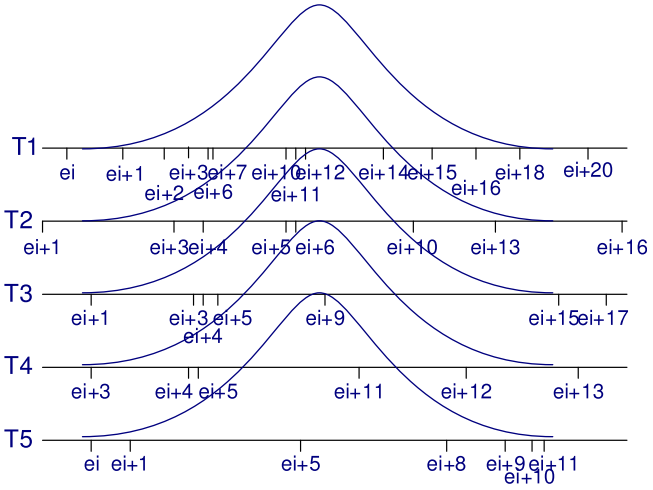


Fig. 3. Fuzzy Temporal Window over an ODB.

Once we have defined the Fuzzy Temporal Window, the next step consists of obtaining the representation of ODB for every behaviour to study. If we apply a Fuzzy Temporal Window over the ODB, every item of this database will have a membership degree associated to its importance in the *FW*. Therefore, we will obtain a new representation of ODB that will be Fuzzy, a Fuzzy ODB.

Definition 4.3 (*Fuzzy ODB applying W*). Let *ODB* be an Observation DataBase and *FW* a Fuzzy Temporal Window. Then, we define $\tilde{ODB}$ as a Fuzzy Observation Data Base where every item has an associated membership degree related to its importance in FW.

As we indicated in the Section 2, we represent the *ODB* as a Transactional Database. Consequently, the Fuzzy ODB will be represented as a Transactional Database too, where for every row and column we will have the membership degree according to the importance of the item in the Fuzzy Temporal Window.

4.2. Generalizing the Fuzzy Temporal Windows

Let us point that in the development before it is implicitly assured that the knowledge used to characterize the FW is completely true. However this assumption is no always correct, as the following Example 4.6 shows.

Example 4.6. Let us suppose that we want to check the behaviour `Rose leaves home at 8:30`. However, we know that Rose sometimes arrives late at work. So, we can affirm that `Rose leaves home at time almost always`.

To manage this kind of statement we will use a semantic approach based on evaluation of quantified sentences (Zadeh, 1983). A quantified sentence is an expression of the form "Q of F are G", where F and G are two fuzzy subsets of a finite set X, and Q is a relative fuzzy quantifier. Some examples could be (The most times, Rose leaves home, 8:30), (Almost never, Rose leaves home, on time), (Almost all times, Rose leaves home, 8:30).

[1] To manage easily the elements on the Table 3, we use the enumeration marked associated to every item.

[2] IDEM.

Table 3
A piece of a real database.

Time	t_1	t_2	t_3	t_4	t_5
8:20	Clothes (a_1)				Shoes (e_1)
8:21					Bag (e_2)
8:23			Tap (c_1)		MobilePhone (e_3)
8:24					Keys (e_4)
8:25	Shoes (a_2)	Shoes (b_1)	Towel (c_2)	Shoes (d_1)	
8:26	MobilePhone (a_3)		BedroomDoor (c_3)		ODoor (e_5)
8:27				Keys (d_2)	
8:28	Bag (a_4)			MobilePhone (d_3)	
8:29	Keys (a_5)		Clothes (c_4)	Bag (d_4)	Keys2 (e_6)
8:30	ODoor (a_6)	Bag (b_2)		ODoor (d_5)	ODoor2 (e_7)
8:31	Elevator (a_7)	Keys (b_3)	Shoes (c_5)	Elevator (d_6)	Elevator (e_8)
8:31		MobilePhone (b_4)	Bag (c_6)	Keys2 (d_7)	
8:32		ODoor (b_5)		ODoor2 (d_8)	
8:33		Keys2 (b_6)	Keys (c_7)		
8:33		ODoor2 (b_7)	MobilePhone (c_8)		
8:34		Elevator (b_8)	ODoor (c_9)		
8:35			Keys2 (c_{10})		
8:36			ODoor2 (c_{11})		
8:40			Elevator (c_{12})		

Table 4
Result of applying *TW* over Table 3.

t_1	$(a_1,0)\,(a_2,1)\,(a_3,1)\,(a_4,1)\,(a_5,1)\,(a_6,1)\,(a_7,1)$
t_2	$(b_1,1)\,(b_2,1)\,(b_3,1)\,(b_4,1)\,(b_5,1)\,(b_6,1)\,(b_7,1)\,(b_8,1)$
t_3	$(c_1,0)\,(c_2,1)\,(c_3,1)\,(c_4,1)\,(c_5,1)\,(c_6,1)\,(c_7,1)\,(c_8,1)\,(c_9,1)\,(c_{10},1)\,(c_{11},0)\,(c_{12},0)$
t_4	$(d_1,1)\,(d_2,1)\,(d_3,1)\,(d_4,1)\,(d_5,1)\,(d_6,1)\,(d_7,1)\,(d_8,1)$
t_5	$(e_1,0)\,(e_2,0)\,(e_3,0)\,(e_{14},0)\,(e_5,1)\,(e_6,1)\,(e_7,1)\,(e_8,1)$

Table 5
Result of applying *FW* over Table 3.

t_1	$(a_1,0)\,(a_2,0)\,(a_3,0.33)\,(a_4,1)\,(a_5,1)\,(a_6,1)\,(a_7,1)$
t_2	$(b_1,0)\,(b_2,1)\,(b_3,1)\,(b_4,1)\,(b_5,1)\,(b_6,0.66)\,(b_7,0.66)\,(b_8,0.33)$
t_3	$(c_1,0)\,(c_2,0)\,(c_3,0.33)\,(c_4,1)\,(c_5,1)\,(c_6,1)\,(c_7,0.66)\,(c_8,0.66)\,(c_9,0.33)\,(c_{10},0)\,(c_{11},0)\,(c_{12},0)$
t_4	$(d_1,0)\,(d_2,0.66)\,(d_3,1)\,(d_4,1)\,(d_5,1)\,(d_6,1)\,(d_7,1)\,(d_8,1)$
t_5	$(e_1,0)\,(e_2,0)\,(e_3,0)\,(e_{14},0)\,(e_5,0.33)\,(e_6,1)\,(e_7,1)\,(e_8,1)$

This knowledge is used to modify the Fuzzy Temporal Window *FW* to obtain another Window better adjusted to the problem, which is the case of Example 4.6. Concretely, we use the method presented in Gonzlez, Pons, and Vila Miranda (1999). This method transforms the fuzzy set, which is associated to the *FW* for a specific behaviour, according to the knowledge expressed by a quantified sentence. This method uses the following definition of a transformation function (González, Marín, Pons, & Vila, in press):

Definition 4.4. Let $A \in \tau$ such that

$$A = \{(m_1, m_2, a, b), h(A)\},$$

where m_1, m_2, a, b are the values that defines a trapezoidal fuzzy number and $H(A)$ is the height of A.

Let $\alpha \in (0,1]$ be. We will denote $\triangle = \frac{\alpha - h(A)}{\alpha h(A)}$ and define

$$T_\alpha(A) = \left\{\left(m_1, m_2, a + \frac{\triangle}{k}, b + \frac{\triangle}{k}\right), \alpha\right\}, \tag{1}$$

for those α in which the transformation makes sense (notice that some values of α lower than $H(A)$ could produces negative spreads).

To use the transformation operation contained into the Definition 4.4, we need to define two basic parameters:

- k-scale parameter that will be 1 in this case.
- α-value for doing the transformation.

From these values, we can define a process to obtain a new Window *FW'* in three steps:

1. Calculate α-value using the quantified sentence. We have to evaluate the quantified sentence defined according to the knowledge about the behaviour. This evaluation finds out the α-value. There are a lot of ways to evaluate a quantified sentence.
2. Truncate the fuzzy number according to α (Gonzlez et al., 1999). After this operation, we obtain a non normalized fuzzy set A^α.
3. Normalize the fuzzy set (Gonzlez et al., 1999). The authors assume that uncertainty is being translated into imprecision under the condition of the amount of information provided by the fuzzy number remains equal before and after normalization process. Fig. 4 represents the followed process over a trapezoidal window to make the transformation.

The Example 4.7 shows the defined process. We use a trapezoidal Fuzzy Temporal Window for the sake of clarity.

Example 4.7. Let us suppose that we want to check the behaviour `Rose hardly ever leaves home at 8:30`. Thus, our Fuzzy Temporal Windows, represented as a lr-number as $F = (m_1, m_2, a, b)$ where m_1, m_2 is the higher points of a trapezoidal fuzzy function, and a, b are the time to establish the lower points.

$$T = (8:25, 8:35, 0:02, 0:02). \tag{2}$$

Here we have selected a basic way to evaluate the quantified sentence, but not the best: *Zadeh Method*. The expression of the Zadeh Cardinal is $\alpha_A = \frac{P(A)}{|X|}$ where $P(A) = sum_{x \in X} A(x)$ and $|X|$ is the X set cardinal.

So, $\alpha_A = \frac{15}{20} = 0.75$. The evaluation of the quantified sentence is $Z_Q(A) = Q(\alpha_A) = 0.53$.

Next, we do the transformation of our windows using α value.

$$F = \{(8:25, 8:35, 0:05, 0:05), 1\} \rightarrow \tag{3}$$

$$F^\alpha = \left\{\begin{pmatrix} 8:25 - 0:05(1-\alpha), \\ 8:35 - 0:05(1-\alpha), \\ 0:05\alpha, 0:05\alpha \end{pmatrix}, \alpha\right\} \rightarrow \tag{4}$$

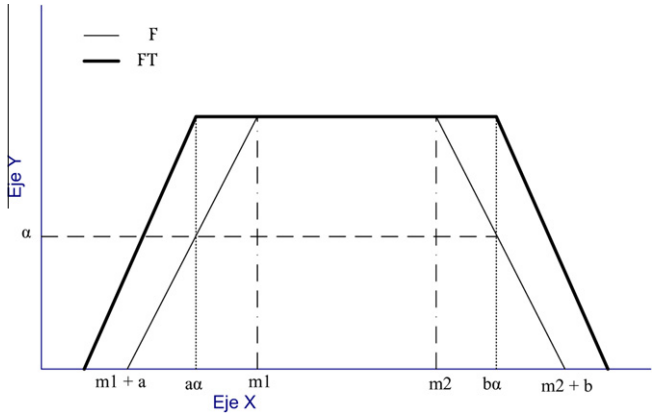


Fig. 4. Transformations over a Trapezoidal Fuzzy Temporal Windows.

$$F^T = \left\{ \begin{pmatrix} 8:25 - 0:05(1-\alpha), \\ 8:35 - 0:05(1-\alpha), \\ 0:05\alpha - \frac{1-\alpha}{k\alpha}, 0:05\alpha - \frac{1-\alpha}{k\alpha} \end{pmatrix}, 1 \right\}. \quad (5)$$

Replacing α value with $Z_Q(A)$, i.e., $\alpha = 0.53$ we expand our Fuzzy Temporal Windows to

$$F = (8:23, 8:37, 0:08, 0:08). \quad (6)$$

5. Obtaining sequence patterns by alpha-cuts. A method to extract fuzzy behaviour patterns

Up to now, we only have worked with the temporal constraints to limit the piece of the ODB where find the correct sequence of actions. However, actually, we only have applied the knowledge about the user, although without solving our problem: extracting fuzzy sequence patterns that represent the user's behaviour. Therefore, the next step consists of adapting the system to find a correct solution for our problem.

More concretely, the problem is to "extract the sequence patterns to specific behaviour when we have a Fuzzy ODB, represented as a fuzzy transactional database and a Fuzzy Temporal Window W defined from the user knowledge".

In this point, we have studied two alternatives to extract the sequence patterns:

1. to use a typical algorithm to extract fuzzy frequent itemsets;
2. to adapt the model presented in Section 3 to manage this new sort of data.

There are several algorithm to extract fuzzy frequent itemsets, such as, the one proposed by Delgado Calvo-Flores, Snchez Fernndez, Marn Ruiz, & Vila Miranda (2003). In this article, the authors propose a new algorithm to extract fuzzy association rules using a new measures for the frequency of the items. We have studied this option, although we have ruled out it because our system tries to find the objects more frequent and this algorithm is clearly addressed to extract fuzzy association rules. The use of these new measures avoids obtaining the support value that is really necessary for the next steps of the process.

Therefore, we prefer to redesign the method to the new circumstances. We can not apply the method explained in Ros et al. (2009) & Delgado et al. (2009) directly, because we have to manage membership degree corresponded to the Fuzzy Temporal Window, so we should find a way to transform the fuzzy data in crisp data.

Our proposal consists of transforming the fuzzy problem to a crisp problem with the aim of applying the method explained in the Section 2. To do that, we use the α-cut concept $\forall \alpha \in (0,1]$ Nguyen & Walker (2006). Using the α-cuts over a fuzzy transactional database, we can obtain a new transactional database where those items whose membership degree was bigger or equal than the α-cut value are studied (the membership degree will be 1), whereas the ones whose membership degree was lower than it are avoided (the membership degree will be 0).

Concretely in our proposal, we start with the ODB transactional database, which we represent as $\widetilde{T}$. By the α-cut, we would have a crisp new representation of $\widetilde{T}$, T^α for every value of α we want to study. Then, we apply the presented method to T^α, obtaining frequent itemsets and sequence patterns to specific α value (I^α and P^α, respectively). After extracting all sequence patterns, we create a fuzzy set **P** constituted by the sequences patterns which have obtained to each α-value applying the Representation Theorem (Nguyen & Walker, 2006). In the next section we show a complete example about this process.

We should take into account that the application of this theorem has an important restriction, the *consistent restriction*:

If $\alpha_1 > \alpha_2$ then $A^{\alpha_1} \subseteq A^{\alpha_2}$.

As we use frequent itemsets, we need to represent extracted frequent itemsets as an unique fuzzy set. Thus, we have to ensure that *consistent restriction* is accomplished between every α frequent itemset I^{α_i}.

Proposition 5.1. *Let $\widetilde{T}$ be a fuzzy transactional data base, T^{α_1} and T^{α_2}, α-cuts of $\widetilde{T}$, $\alpha_1 \geqslant \alpha_2$. And let I^{α_1}, I^{α_2} be the set of extracted frequent itemsets from T^{α_1}, T^{α_2}, respectively. Then, $I^{\alpha_1} \subseteq I^{\alpha_2}$, i.e., every item set which is frequent to level α_1, it is also frequent to level α_2.*

Proof 1. We suppose we have $\widetilde{T}$ a fuzzy transactional data base, where each $t \in \widetilde{T}$ represents a fuzzy set of T. If we apply the α-cut on $\widetilde{T}$, we obtain T^α. Then, if we use the values $\alpha_1, \alpha_2 \in (0,1]$ with $\alpha_1 \geqslant \alpha_2$ we get T^{α_1}, T^{α_2}, respectively. Using the *consistent restriction*, $T^{\alpha_1} \subseteq T^{\alpha_2}$.

Let I^{α_1}, I^{α_2} be the set of frequent itemsets from T^{α_1}, T^{α_2}, respectively. Then $\forall i \in I^{\alpha_1}$, $\exists t \in T^{\alpha_1} / i \in t$; with $T^{\alpha_1} \subseteq T^{\alpha_2} \Rightarrow t \in T^{\alpha_2}$.

Let I^{α_1} be the set of frequent itemsets for $T^{\alpha_1} \Rightarrow \forall i \in I^{\alpha_1}$, $supp(i) \geqslant minsup$ for $T^{\alpha_1} \Rightarrow supp(i) \geqslant minsup$ to T^{α_2}, i.e., $i \in I^{\alpha_2} \Rightarrow I^{\alpha_1} \subseteq I^{\alpha_2}$. □

In our method, we use the obtained frequent itemsets to create the sequence patterns. However, now we have the frequent itemsets associated to an specific α value, and we get sequence patterns to this specific α value. To represent the sequence patterns as a fuzzy set, we use the Representation Theorem, so we need to ensure the *consistent restriction* for sequence patterns too.

Proposition 5.2. *Let P^{α_i}, P^{α_j} be sets of sequence patterns where $\alpha_i \geqslant \alpha_j$ then $P^{\alpha_i} \subseteq P^{\alpha_j}$.*

Proof 2. Let I^{α_k}, I^{α_j} be the set of obtained frequent itemsets from T^{α_k} and T^{α_j}, respectively, with $\alpha_k \geqslant \alpha_j$. Let $P^{I^{\alpha_k}}$, $P^{I^{\alpha_j}}$ be the pattern valid set of I^{α_k} and I^{α_j}, respectively.

We define the set $perm(i)$ such that:

$$perm(i) = \{p / p \text{ is a permutation of } i \text{ and } p \in T\}, \quad (7)$$

where T is a transactional data base and i is a frequent itemset from T.

Then $\forall i \in I^{\alpha_k}$ $perm(I) \in P^{I^{\alpha_k}}$. As $I^{\alpha_k} \subseteq I^{\alpha_j} \Rightarrow i \in I^{\alpha_j}$. So $perm(i) \in P^{I^{\alpha_k}} \Rightarrow P^{I^{\alpha_k}} \subseteq P^{I^{\alpha_j}}$. □

Table 6
A piece of database to extract the behaviour "Leave home".

t_1	(MobilePhone, 08:15:00); (Bag, 08:20:50); (Keys, 08:21:00); (ODoor, 08:32:30); (Elevator, 08:38:20); (BedDoor, 08:44:10);
t_2	(Keys, 08:15:00); (Bag, 08:20:50); (MobilePhone, 08:26:40); (ODoor, 08:32:30); (Elevator, 08:38:20); (BathDoor, 08:44:10);
t_3	(Briefcase, 08:15:00); (BedDoor, 08:17:30); (BathDoor, 08:20:00); (Chair, 08:22:30); (Tap, 08:25:00); (Toothbrush, 08:27:30); (Trousers, 08:30:00); (Shoes, 08:32:30); (Bag, 08:35:00);
t_4	(MobilePhone, 08:37:30); (Keys, 08:40:00); (ODoor, 08:42:30); (Elevator, 08:45:00); (BathDoor, 08:47:30);
t_5	(Towel, 08:15:00); (Trousers, 08:17:11); (Chair, 08:19:22); (Trousers, 08:21:33); (Trousers, 08:23:45); (BedDoor, 08:25:56); (Bag, 08:28:07); (MobilePhone, 08:30:18); (Keys, 08:32:30);
t_6	(ODoor, 08:34:41); (Elevator, 08:36:52); (Toothbrush, 08:39:03); (Shoes, 08:41:15); (Briefcase, 08:43:26); (Chair, 08:45:37); (Shirt, 08:47:48);
⋮	
t_{148}	(BathDoor, 08:15:00); (Tap, 08:19:22); (Briefcase, 08:23:45); (MobilePhone, 08:28:07); (Bag, 08:32:30); (Keys, 08:36:52); (ODoor, 08:41:15); (Elevator, 08:45:37);
t_{149}	(Bag, 08:15:00); (MobilePhone, 08:18:53); (Keys, 08:22:46); (ODoor, 08:26:39); (Elevator, 08:30:33); (Toothbrush, 08:34:26); (Toothpaste, 08:38:19); (BathDoor, 08:42:13); (Table, 08:46:06);
t_{150}	(Trousers, 08:15:00); (Bag, 08:17:20); (Keys, 08:19:40); (MobilePhone, 08:22:00); (ODoor, 08:24:20); (Elevator, 08:26:40); (Briefcase, 08:29:00); (Toothpaste, 08:31:20); (Chair, 08:33:40); (Towel, 08:36:00); (Table, 08:38:20); (Towel, 08:40:40); (Table, 08:43:00); (Towel, 08:45:20); (Shoes, 08:47:40);

Table 7
A piece of database after applying the Fuzzy Temporal Window W.

t_1	(MobilePhone, 0.1); (Bag, 0.2); (Keys, 1); (ODoor, 1); (Elevator, 0.64); (BedDoor, 0);
t_2	(Keys, 0); (Bag, 0.1); (MobilePhone, 1); (ODoor, 1); (Elevator, 0.64); (BathDoor, 0);
t_3	(Briefcase, 0); (BedDoor, 0); (BathDoor, 0); (Chair, 0.46); (Tap, 1); (Toothbrush, 1); (Trousers, 1); (Shoes, 1); (Bag, 1);
t_4	(MobilePhone, 0.46); (Keys, 0); (ODoor, 0); (Elevator, 0); (BathDoor, 0);
t_5	(Towel, 0); (Trousers, 0); (Chair, 0); (Trousers, 0.266); (Trousers, 0.69); (BedDoor, 1); (Bag, 1); (MobilePhone, 1); (Keys, 1);
t_6	(ODoor, 1); (Elevator, 0); (Toothbrush, 0); (Shoes, 0); (Briefcase, 0); (Chair, 0); (Shirt, 0);
⋮	
t_{148}	(BathDoor, 0); (Tap, 0); (Briefcase, 0.69); (MobilePhone, 1); (Bag, 1); (Keys, 0.304); (ODoor, 0); (Elevator, 0);
t_{149}	(Bag, 0); (MobilePhone, 0); (Keys, 0.492); (ODoor, 1); (Elevator, 1); (Toothbrush, 1); (Toothpaste, 0.638); (BathDoor, 0); (Table, 0);
t_{150}	(Trousers, 0); (Bag, 0); (Keys, 0); (MobilePhone, 0.4); (ODoor, 0.84); (Elevator, 1); (Briefcase, 1); (Toothpaste, 1); (Chair, 1); (Towel, 0.2); (Table, 0.64); (Towel, 0); (Table, 0); (Towel, 0); (Shoes, 0);

Once we have presented the method, the following step consists of understanding completely how it works by means of a case studying. So, we show an Example 5.1 where we would like to clarify the most important aspects of the designed process. We are following the different examples showed throughout this Section 4.

Example 5.1 (*An illustrative example*). Let us suppose that we want to know what actions characterize the behaviour of "Leave home" for Rose. We have been observing her for a long period of time. We have collected the database show in Table 6[3]:

Let FW a Fuzzy Temporal Window defines by the following fuzzy set over temporal line τ:

$$W = \left\{ \begin{array}{c} \frac{0}{8:20}, \frac{0.2}{8:21}, \frac{0.4}{8:22}, \frac{0.6}{8:23}, \frac{0.8}{8:24}, \frac{1}{8:25}, \frac{1}{8:26} \\ \frac{1}{8:27}, \frac{1}{8:28}, \frac{1}{8:29}, \frac{1}{8:30}, \frac{1}{8:31}, \frac{1}{8:32}, \frac{1}{8:33}, \frac{1}{8:34} \\ \frac{1}{8:35}, \frac{0.8}{8:36}, \frac{0.6}{8:37}, \frac{0.4}{8:38}, \frac{0.2}{8:39}, \frac{0}{8:40}. \end{array} \right\} \quad (8)$$

Once we have the information from the user and the Fuzzy Temporal Window, we apply the process explained in the Section 4, to extract the representation of the database with the values in the Fuzzy Temporal Window (see Table 7).

Over this representation, we apply four different values of α-value: $\alpha_1 = 0.4$, $\alpha_2 = 0.6$, $\alpha_3 = 0.8$, $\alpha_4 = 1.0$. We obtain for each α values the T^{α} as the result of apply the α-cut in $\widetilde{T}$. In Table 8, we show the T^{α} for α_2.

After obtaining all T^{α}, we have transformed the fuzzy problem to a crisp problem. Now, we extract frequent itemsets and sequence patterns with the model explained in Ros et al. (2009) & Delgado et al. (2009).

We have executed the Apriori Algorithm with two support values: minsup = 0.8 and minsup = 0.9. The results are showed in Tables 9 and 10.

[3] In this paper, we do not show the whole Data Base due to the lack of space.

Since we have the common actions to specific behaviour, the next stage consists of obtaining the valid sequences patterns and the final expression as a fuzzy set applying the Identity Principle. The valid patterns are showed in Tables 11 and 12. And the final pattern expression in Eq. (9) to minsup = 0.8 and Eq. (10) to minsup = 0.9.

$$\widetilde{P} = \left\{ \frac{p_1}{\alpha_4}, \frac{p_2}{\alpha_4}, \frac{p_3}{\alpha_4}, \frac{p_4}{\alpha_2}, \frac{p_5}{\alpha_2} \right\} = \left\{ \frac{p_1}{1}, \frac{p_2}{0.4}, \frac{p_3}{1}, \frac{p_4}{1}, \frac{p_5}{0.6} \right\}, \quad (9)$$

$$\widetilde{P} = \left\{ \frac{p_1}{\alpha_2}, \frac{p_2}{\alpha_2}, \frac{p_3}{\alpha_3}, \frac{p_4}{\alpha_4} \right\} = \left\{ \frac{p_1}{0.6}, \frac{p_2}{0.6}, \frac{p_3}{0.8}, \frac{p_4}{1} \right\}. \quad (10)$$

These results let us confirm that the method fulfil the Propositions 5.1 and 5.2, where we affirm that "*every item set which is frequent to level α_1, it is also frequent to level α_2*". Therefore, we would be always able to extract a set of patterns with a value of membership to the window, using a general value provided by the α-value.

On the other hand, it is interesting to compare the results obtained for each value of minimum support, because it is clear that there is a strong dependence between the method and these values. We should not forget that we are using frequent itemsets in our process. So, we should be very carefully to choose these values. We should find a balance between the minimum support value and the value of α we would like to have as membership of the patter.

6. Performance evaluation

In this section, our objective is to compare the results obtained with the temporal processing (fuzzy model) and the model without considerations about the time (crisp model) presented in Ros et al. (2009) & Delgado et al. (2009). On the other hand, we also studied the advantages of modifying the windows using quantified sentences.

Table 8
A piece of T^{α} for $\alpha_2 = 0.6$.

t_1	(MobilePhone, 0); (Bag, 0); (Keys, 1); (ODoor, 1); (Elevator, 1); (BedDoor, 0);
t_2	(Keys, 0); (Bag, 0); (MobilePhone, 1); (ODoor, 1); (Elevator, 1); (BathDoor, 0);
t_3	(Briefcase, 0); (BedDoor, 0); (BathDoor, 0); (Chair, 0); (Tap, 1); (Toothbrush, 1); (Trousers, 1); (Shoes, 1); (Bag, 1);
t_4	(MobilePhone, 0); (Keys, 0); (ODoor, 0); (Elevator, 0); (BathDoor, 0);
t_5	(Towel, 0); (Trousers, 0); (Chair, 0); (Trousers, 0); (Trousers, 1); (BedDoor, 1); (Bag, 1); (MobilePhone, 1); (Keys, 1);
t_6	(ODoor, 1); (Elevator, 0); (Toothbrush, 0); (Shoes, 0); (Briefcase, 0); (Chair, 0); (Shirt, 0);
⋮	
t_{148}	(BathDoor, 0); (Tap, 0); (Briefcase, 1); (MobilePhone, 1); (Bag, 1); (Keys, 0); (ODoor, 0); (Elevator, 0);
t_{149}	(Bag, 0); (MobilePhone, 0); (Keys, 0); (ODoor, 1); (Elevator, 1); (Toothbrush, 1); (Toothpaste, 1); (BathDoor, 0); (Table, 0);
t_{150}	(Trousers, 0); (Bag, 0); (Keys, 0); (MobilePhone, 0); (ODoor, 1); (Elevator, 1); (Briefcase, 1); (Toothpaste, 1); (Chair, 1); (Towel, 0); (Table, 1); (Towel, 0); (Table, 0); (Towel, 0); (Shoes, 0);

Table 9
Frequent itemset to minsup = 0.8.

I^{α_1}	{Shoes, Bag, Keys, ODoor, MobilePhone} {Keys, Keys2, ODoor, ODoor2, MobilePhone}
I^{α_2}	{Shoes, Bag, Keys, ODoor, MobilePhone} {Keys, Keys2, ODoor, ODoor2, MobilePhone}
I^{α_3}	{Shoes, Bag, Keys, ODoor, MobilePhone}
I^{α_4}	{Shoes, Bag, Keys, ODoor, MobilePhone}

Table 10
Frequent itemset to minsup = 0.9.

I^{α_1}	{Keys, ODoor, MobilePhone}
I^{α_2}	{Keys, ODoor, MobilePhone}
I^{α_3}	{Keys, ODoor}
I^{α_4}	{ODoor}

Table 11
Frequent itemset to minsup = 0.9.

P^{α_1}	{Shoes, Bag, Keys, MobilePhone, ODoor}	p_1
	{Shoes, Keys, MobilePhone, Bag, ODoor}	p_2
	{Shoes, MobilePhone, Bag, Keys, ODoor}	p_3
	{Keys, MobilePhone, ODoor, Keys2, ODoor2}	p_4
	{MobilePhone, Keys, ODoor, Keys2, ODoor2}	p_5
P^{α_2}	IDEM	
P^{α_3}	{Shoes, Bag, Keys, MobilePhone, ODoor}	p_1
	{Shoes, Keys, MobilePhone, Bag, ODoor}	p_2
	{Shoes, MobilePhone, Bag, Keys, ODoor}	p_3
P^{α_4}	IDEM	

Table 12
Valid patterns to minsup = 0.9.

P^{α_1}	{Keys, MobilePhone, ODoor}	p_1
	{MobilePhone, Keys, ODoor}	p_2
P^{α_2}	IDEM	
P^{α_3}	{Keys, ODoor}	p_3
P^{α_4}	{ODoor}	p_4

To realize this comparison, we need a database from which extract the correct behaviour and a way to evaluate the obtained results.

6.1. Database generation

First of all, we had to gather the user information in a database (ODB). As we have indicated before in this paper, this database represents the user actions, therefore, it is composed of the different object that the user has touched.

For the evaluation of our system, we have preferred designing a synthetic database where we could represent all possible situations we should studied. Therefore, although every transaction of an ODB represents the actions realized by an user in the entire day, in this experimentation we have generated the data to study a specific behaviour, to avoid an unnecessary computational cost.

As we did not want to use only a database, but several databases, we should have designed a method to generate automatically these databases. To generate the data, we know:

- What the correct sequence of action is.
- What objects can appear in the sequence.
- Which order relationship rules there are among objects. These rules should have been indicated by an expert, since they were used to generate the database in an automatic way.

Thus, the generation of the database is realized in two phases: object and time generation.

1. Object generation.
   In this stage, we generate the objects that user touches for a specific behaviour. Let remember that a behaviour implies a sequence of actions with some order relationship among them. Thus, we generate a random sequence that fulfils the indicated order relationship and that contains unusual objects (rubbish).
   In this point, we distinguish between two types of generated sequence: those sequences that contain the whole correct sequence or those that only contain a piece of it. In general, the second type of generated sequence makes a better representation of the user daily activities than the other, because in general it is very common that an user usually forgets something when he/she is doing a behaviour. Therefore, it is easy to think that the obtained results with this second type are more reliable than the results from the first type of generation.
   With respect to the unusual objects or rubbish, they are introduced to check if the model is capable of extracting the key actions and omitting the incorrect actions (rubbish), because hardly ever an user makes some extra actions during the process of a behaviour and we have to be prepared for them.
2. Time generation.
   In the second stage, we have to generate moments on time. Our objective is that the moments were as independent as possible, thus we can get a database that was a better representation of real daily activities.
   We have designed three ways to generate the time:
   - Sequential generation.
     We generate moments with the same distance among them, i.e., the interval between two actions is always the same.

- Sequential generation with a random initial point.
  We use the same method to generate time but we establish a random initial point. For example, we use this generation to simulate when a user is late, or has forgotten something, etc.
- Non-sequential generation.
  We generate random intervals between actions, using a known initial point, which changes with the new generated point, and a generated one.

6.2. Fitness measure

Moreover, to compare the fuzzy and crisp approaches, we need a fitness measure to analyse the obtained outcomes. We designed a measure to evaluate the method that works with a sequence of words. So, we have to find a way to transform these sequences in a numerical way. It is obvious that our fitness measure ought to assess any sequence of words by numerical value such that synthesizes all important aspects of the approach performance: correction, rubbish, α-values, minimum support of every sequence, time, etc.:

$$c = w_1 correction + w_2 alpha + w_3 \frac{1}{time}, \tag{11}$$

where

- Correction stand for similarity measure of every pattern to the correct pattern. We define a function that associates a numeric value with sequences that a higher value indicates greater similarity.
- Alpha is the average of the alpha-values used for the α-cut for every pattern.
- Time is the amount of millisecond that the method inverts in obtaining results.

Fitness measure is inversely proportional to the time raises to the power of minus one and directly proportional to the correction and confidence measure.

We use specific values for w_i. We have chosen these values because we believe they are rougher to the reality. We use: w_1 and w_2 equals 0.3 and w_3 equals 0.4.

6.3. Experiments and results

We were evaluating our system with three different membership functions for the Fuzzy Temporal Windows:

- Trapezoidal (8:20, 8:25, 8:35, 8:40)

$$\mu_T(x) = \begin{cases} 0, & \text{si}(x < 8:20), \\ \frac{x-a}{b-a}, & \text{si}(8:20 \leqslant x \leqslant 8:25), \\ 1, & \text{si}(8:25 \leqslant x \leqslant 8:35), \\ \frac{d-x}{d-c}, & \text{si}(8:35 \leqslant x \leqslant 8:40), \\ 0, & \text{si}(x > 8:40). \end{cases}$$

- Gauss bell shaped with mean equals to 8:30 and a standard deviation equals as 40 min. N (8:30, 40).
- Gauss bell shaped with mean equals to 8:30 and a standard deviation equals as 60 min. N (8:30, 60).

To sum up, we had six different types of ODB: two types for the object generation and three for the time generation. This variety of ODB let us evaluate different aspects:

- If the method can extract the key actions when there are rubbish between the actions.
- If the method can extract the key actions independently of the window we use to extract them.
- The computational cost of the whole process.
- The values of the parameters, such as minimum support or α-value, used to extract the sequences.
- If the method extracts the correct sequence of actions for a specific behaviour.

The outcomes of the evaluation are shown in a graphic, where the X axes represents the possible values for minimum support whereas the Y axes represents the values of the Fitness Measure. As we have indicated before, the Fitness Measure is inversely proportional to the time and directly proportional to the rest of parameters. So, in the graphic we have to find the sort of window that obtains the highest results without forgetting the minimum support, since we use the frequent itemsets to extract our key actions.

From Figs. 5–10, six graphics show fitness measure for the different shapes of window for each ODB, as well as, the results of the crisp model to make a comparison.

If we observe the results, we extract some conclusions:

- In general, the Fuzzy model obtains better results than the Crisp one. The Fuzzy method includes a way to attach importance to some actions that are key but are irregular, whereas, the Crisp method consider all actions at the same importance.
- As well as, the Crisp method process all the data, while the Fuzzy method can reduce the size of transactions minimizing the computational cost. This reduction could be very interesting when we have a forgetful user, because we would have a database with many mistakes that we should have to avoid. Moreover, this size reduction of the database makes to find whether the order of the sequences are valid easier.

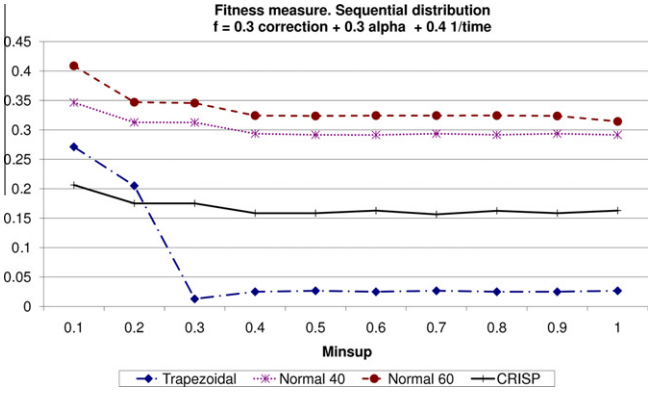


Fig. 5. Experiment results. Sequential distribution. Type A.

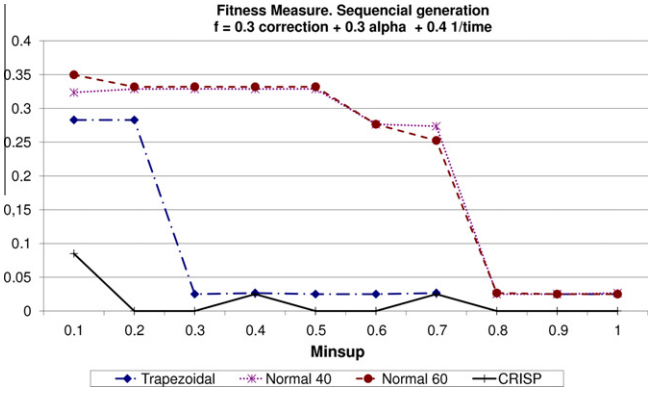


Fig. 6. Experiment results. Sequential distribution. Type B.

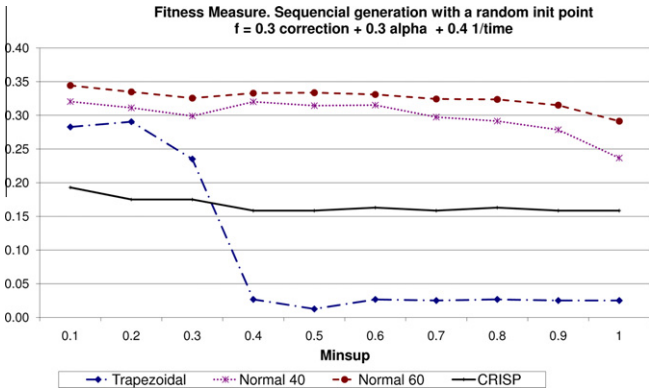


Fig. 7. Experiment results. Random initial sequential distribution. Type A.

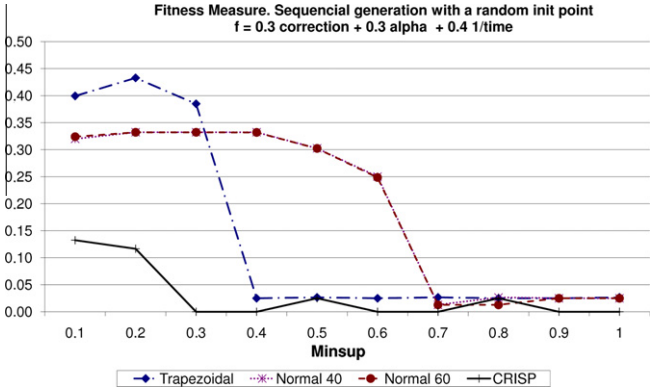


Fig. 8. Experiment results. Random initial sequential distribution. Type B.

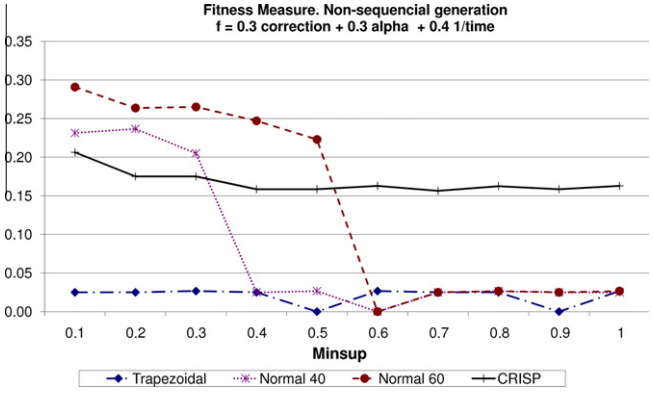


Fig. 9. Experiment results. Non-sequential distribution. Type A.

- It is interesting to observe that the differences between the crisp and fuzzy model are more significant when we uses the second type of database (those databases where the correct sequence of actions is not always complete). The fuzzy model is more much better than the crisp one. If we observe the graphics, we can affirm that the use of a Window always improve the results over the crisp one, independently of the type of used Windows. This point is extremely important and representative, because we remember that these databases represent better the daily activities of the user.
- Among the Fuzzy Temporal Windows, the best results are obtained with the Gauss bell shaped ones. The explication of this fact we could find it in the difference between the membership degree of each function. With a Gauss bell shaped one, we obtain a more relaxed degree of membership than using a trapezoidal function or the crisp model. In this way, when we apply the α-cut depending on the membership function we would extract different alternatives. Thus, with a relaxing function we obtain more information in every cut, and therefore, more opportunities to extract the correct sequence.

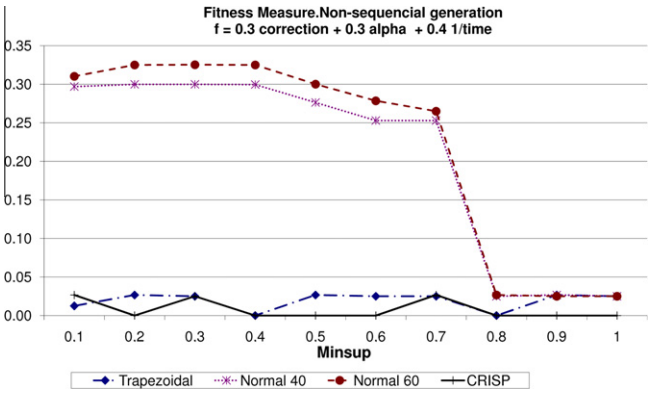


Fig. 10. Experiment results. Non-sequential distribution. Type B.

6.4. Statistical analysis

From the results in the previous section, it seems that the performance of the fuzzy approach depends on two factors: the type of generation of the database and the used method. In order to realize a deep analysis, we are going to do our statistical study to contrast them.

We studied our results with an ANOVA test. ANOVA is a general technique that can be used to test the hypothesis that the means among two or more groups are equal, under the assumption that the sampled populations are normally distributed. We used a 2-way ANOVA with the two null hypotheses:

H_0 : Results are dependent on the type of generation of the database.

H_1 : Results are dependent on the used method.

The results of the ANOVA test are shown in Table 13. These demonstrate that H_0 hypothesis is rejected (the results are independent of the generation of the database) while H_1 is accepted (the results are dependent on the method used for analysis).

Then, we can compare the crisp and fuzzy method directly, observing the graphics presented in Section 6.3 and we can affirm that in general the fuzzy method performs better results than the Crisp one, next we studied which of the fuzzy methods gets better results. We set the following assumptions for the study of the results:

H_2 : Results are dependent on the type of generation of the database.

H_3 : Results are dependent on the fuzzy used window.

The results of the ANOVA test are shown in Table 14. From this table, we confirm again that the method is independent of

Table 13
Results of the ANOVA test to hypotheses H_0 and H_1.

Source	F	Signification
Generation	1.222	0.436
Method	77.984	0.003

Table 14
Results of the ANOVA test to hypotheses H_2 and H_3.

Source	F	Signification
Generation	2.483	0.101
Fuzzy windows	9.767	0.000

generation of the database (H_2 hypothesis rejected). However, it is totally dependent on the *used fuzzy window* (accepted hypothesis H_3).

Therefore, we conclude that outcomes depend on the type of used window. So, we have to select the most appropriate window for every situation. However, after our experiments we affirm that, in general, the $N(\bar{x}, \sigma)$ is usually the best shape for the Sliding Window.

6.5. Analysis of process of extending the Fuzzy Temporal Window

Once we have checked our fuzzy method, we analysed the process of extending the Fuzzy Temporal Window from a knowledge expressed as a quantified sentence. We applied the Trapezoidal Temporal Window represented in Eq. (8) over the two databases used in the previous section, and the new window obtained after expanding the window. Fitness measure is plotted on Figs. 11 and 12.

From these graphics, we observe that we usually extract results with the second window where the first window do not. This event happens because with the second window we have expanded the amount of data we studied to obtain the results; although, the use of a Windows continue being convenient because we studied less data than the Crisp Method. Therefore, we expanded the window to get more adjusted results than original window, without studying the whole interval.

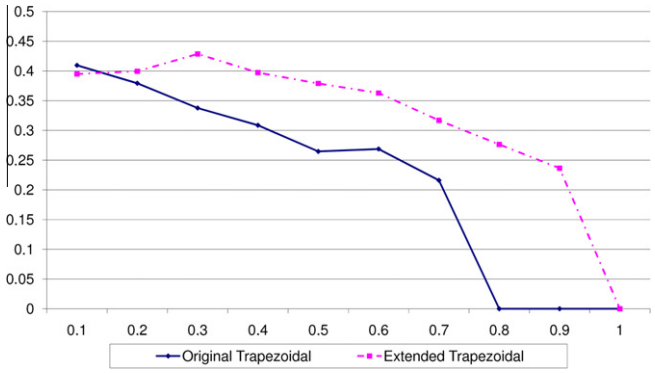


Fig. 11. Original vs. extended trapezoidal. Database a.

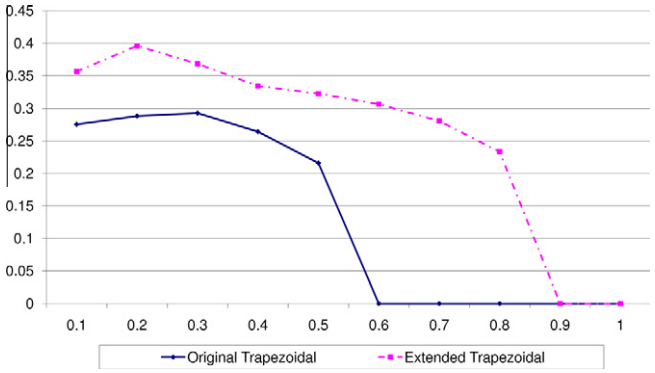


Fig. 12. Original vs. extended trapezoidal. Database b.

7. Conclusions

In this paper, we have presented a method to detect the user behaviour from the information provided by a Tagged World. As we have indicated, this problem is affected bye non-random imprecision, because we do not precisely know the interval of the time in which the action happens. The imprecision appears because a behaviour is not stationary on a context or domain and is specific for every user. Thus, we have designed a method which uses fuzzy logic to limit the events in a specific behaviour using a Fuzzy Temporal Window. From it, we can assign a membership degree for every event for the studied behaviour. In addition, we extend the Fuzzy Temporal Window concept, to adjust the interval using the knowledge that we have. This adjustment reduces the effect produced by a change of user habits.

The system results are the fuzzy pattern sequences which define the behaviour. To manage the non-random imprecision, we use the α-cuts and the Representation Theorem to obtain a fuzzy result applying the crisp method. The method obtain a fuzzy pattern sequence set. Moreover, we have demonstrated our method is better than crisp method. We check that the fuzzy method obtains better results than crisp one, concluding that results depend on the type of window we use. Finally, we demonstrate that the results are independent database is from.

In future work, we want to distinguish between three problems of our system:

- Non-random imprecision, which has been dealt with the Fuzzy Temporal Window.
- Random imprecision, because an error could occur in the process of the sensor reading. There are many reasons that could cause the mistakes of sensors in reading, for example, the sensors or the reader could fail, the environment affects to the sensors, sensors damage, so on.
- Uncertainty, because the behaviours are not always performed in the same way.

As well, we will try to develop a system which uses the knowledge organized in that stage to take decision and to send the correct alarms.

References

Agrawal, R., & Srikan, R. (1994). Fast algorithms for mining association rules. IBM Research Report RJ9839.

Blythe, P. (1999). RFID for road tolling, road-use pricing and vehicle access control. *IEEE Colloquium on RFID Technology*.

Delgado Calvo-Flores, M., Snchez Fernndez, D., Marn Ruiz, N., & Vila Miranda, M. A. (2003). Fuzzy association rules: General model and applications. *IEEE Transactions on Fuzzy Systems, 11*, 214–225.

Delgado, M., Ros, M., Vila, M. A., (2009). Correct behavior identification system in a Tagged World. *Expert Systems with Applications*, doi:10.1016/j.eswa.2009.01.077.

Domdouzis, K., Kumar, B., & Anumba, C. (2007). Radio-frequency identification (RFID) applications: A brief introduction. *Advanced Engineering Informatics, 21*(4), 350–355.

Fennani, A., & Hamam, H. (2008). An optimized RFID-based academic library. In *Proceedings of the 2008 second international conference on sensor technologies and applications – Vol. 00 (August 25–31, 2008) SENSORCOMM* (pp. 44–48). Washington, DC: IEEE Computer Societyhttp://dx.doi.org/10.1109/SENSORCOMM.2008.63.

Fiot, C., Laurent, A., & Teisseire, M. (2007). Extended time constraints for sequence mining. In *14th international symposium on temporal representation and reasoning TIME'07*.

González, A., Marín, N., Pons, O., & Vila, M. (2007). Qualification of fuzzy statements under fuzzy certainty. In P. Melin, O. Castillo, L. Aguilar, J. Kacprzyk, & W. Pedrycz (Eds.), *Foundations of fuzzy logic and soft computing. Lecture notes in computer science* (Vol. 4529, pp. 162–170). Berlin/Heidelberg: Springer.

Gonzlez, A., Pons, O., & Vila Miranda, M. A. (1999). Dealing with uncertainty and imprecision by means of fuzzy numbers. *International Journal of Approximate Reasoning, 21*(3), 233–256.

Jalelkis, E. J. et al. (1995). Radio-frequency identification applications in construction industry. *Journal of Construction Engineering Management*(June), 183–191. ASCE Website.

Kidd, C. D., Orr, R., Abowd, G., Atkeson, C. G., Essa, Irfan A., MacIntyre, B., Mynatt, E. D., Starner, T., & Newstetter, W. (1999). The aware home: A living laboratory for ubiquitous computing research. In *CoBuild'99: Proceedings of the second international workshop on cooperative buildings, integrating information, organization, and architecture* (pp. 191–198). Available from: http://portal.acm.org/citation.cfm?id=674887.

Koyama, K., Nakagawa, K., & Shimakawa, H. (2005). Embedded action detector to enhance freedom from care. In *Proceedings of 11th international conference on human–computer interaction, Las Vegas* (8 p.).

Mori, T., Noguchi, H., Takada, A., & Sato, T. (2005). Sensing room: Distributed sensor environment from measurement of human daily behaviour. In *IEEE/RSJ international conference on intelligent robots and systems (IROS 2005)* (Vol. 2, pp. 1703–1709).

Nguyen, H. T., & Walker, E. A. (2006). *A first course in fuzzy logic*. Chapman & Hall/CRC Taoylr & Frances Group.

Philipose, M., Fishkin, K. P., Perkowitz, M., Patterson, D. J., Fox, D., Kautz, H., et al. (2004). Inferring activities from interactions with objects. *Context-Aware Computing. Pervasive Computing, 3*(50–57).

Rashid, O., Bamford, W., Coulton, P., Edwards, R., & Scheible, J. (2006). PAC-LAN: Mixed-reality gaming with RFID-enabled mobile phones. *Computers in Entertainment, 4*(4), 4http://doi.acm.org/10.1145/1178418.1178425.

Ros, M., Delgado, M., & Vila, M. A. (2008). Obtaining fuzzy sequential patterns for ambient intelligence problem solving. In *XIV Congreso Español sobre Tecnologías y Lógica Fuzzy (ESTYLF 2008), 17–19 Septiembre 2008, Cuencas Mineras (Mieres–Langreo)* (pp. 377–384).

Ros, M., Delgado, M., & Vila, A. (2009). An ambient intelligent approach to control behaviours on a Tagged World. In S. Omatu, M. Rocha, J. Bravo, F. Fernández, E. Carchado, A. Bustillo, et al. (Eds.), *International conference on artificial neural networks (IWAAL09). Lecture notes in computer science* (Vol. 5518, pp. 764–771). Berlin/Heidelberg: Springer.

Sheng, Q. Z., Li, X., & Zeadally, S. (2008). Enabling next-generation RFID applications: Solutions and challenges. *IEEE Computer, 41*(9), 21–28.

Want, R. (2004). RFID: A key to automating everything. *Scientific American Magazine*(January), 10.

Weiser, M. (2008). *The computer for the twenty-first century*. Scientific American. p. 94. <http://nano.xerox.com/hypertext/weiser/SciAmDraft3.html> Accessed 02.11.08..

Wu, F., Kuo, F., & Liu, L. (2005). The application of RFID on drug safety of inpatient nursing healthcare. *Proceedings of the seventh international conference on electronic commerce (ICEC'05), Xi'an, China, August 15–17, 2005* (Vol. 113, pp. 85–92). New York, NY: ACMhttp://doi.acm.org/10.1145/1089551.1089571.

Yamahara, H., Takada, H., & Shimakawa, H. (2007). Detection of user mode shift in home. In *Proceedings of the UCS 2007. LNCS* (Vol. 4836, pp. 166–181).

Zadeh, L. A. (1983). A computational approach to fuzzy quantifiers in natural languages. *Computing and Mathematics with Applications*(9), 149–184.

3. Adapting the learnt behaviour patterns on time. A Learning Automata based approach

3.1. Detection of abnormal human activities by means of Learning Automata

The journal paper associated to this part is:

- Ros, M.; Pegalajar, M.; Delgado, M.; Vila, A. and Hagras, H. Detection of abnormal human activities by means of Learning Automata Transactions on Systems, Man and Cybernetics – Part A: Systems and Humans, 2011, Submitted to. .
  - Status: **Submitted**.
  - Impact Factor (JCR 2010): 2.093
  - Subject Category: Computer Science, Cybernetics. Ranking 5 / 19 (Q2).
  - Subject Category: Computer Science, Theory and Methods. Ranking 15 / 91 (Q1).

Detection of abnormal human activities by means of Learning Automata

M. Ros, M.P. Cuéllar, M. Delgado and A. Vila and Hani Hagras, Senior Member IEEE

***Abstract*—Ambient Assisted Living is a new paradigm that uses the advantages of Ambient Intelligence in a current new field: elderly people assistance at home. The aims are to increase elders' autonomy and their home staying. Among its issues, the monitoring of the human activities is becoming more prominent. The need to face unusual behaviours and to give support to the user when these abnormal situations exist is addressed in this type of systems. This problem encompasses tasks such as the learning of human habits, the inference of temporal relationship constraints between human actions, and the online adaptation to environmental and human behaviour changes. In this work, we propose a system based on Learning Automata to model human activities. The developed system combines advantages such as the visual modeling of the temporal constraints between actions, the capability of online adaptation to changes in the user habits, and modularity to ease the system upgrade. The system has been tested at the iSpace, a real Ambient Intelligent Environment testbed at the University of Essex. We will present the results obtained by monitoring three participants activities for three specific behaviours.**

***Index Terms*—learning automata, reinforcement learning, activities of daily living, ambient assisted living, real-time learning, ambient intelligence**

I. Introduction

A. Background

THE recognition of human activities is one of the most challenging issues in Ambient Assisted Living (AAL) [1]. The pursued goal is automatic people assistance at home with unobtrusive supervision, to avoid the user's rejection due to the loss of their daily independence [2]. Most of projects in this area focus in attendance of elders or people with special needs at smart homes, and more specifically in the modelling, recognition, and prediction of Activities of Daily Living (ADL) [3][4], as for instance the projects *TigerPlace* [2], *Cogknow* [5], and *CASAS* [6]. An ADL is a sequence of atomic actions within user's daily routine, such as toileting, making a meal, or leaving home. In addition, the actions of an ADL may have temporal constraints; for instance, the user should never open the front door before taking the keys when leaving home in a correct fashion.

There is a high interest in the identification of abnormal behaviours [1] for automatic home assistance. Previous works offer studies of a wide variety of aspects regarding AAL, ranging from the underlying sensor network for user data acquisition [3][7][8], to representation [3][9][10], behaviour modelling [11][12][13], and applications [2][5][6][14][15]. Probabilistic and uncertainty models are the most common choice in the literature to model an ADL, as probabilistic sequences of objects touched by the user [3], Hidden Markov Models (HMMs) [9], Markov Decision Processes [10], hybrid models of HMM and Linear Discriminant Analysis [16], agent and fuzzy systems [11][17], imaging techniques for data acquisition [7], video techniques to extract data features and Bayesian Networks to model the behaviour [8], and image feature extraction techniques combined with a Naive Bayes classifier to identify the human activities [18]. The benefits and drawbacks of different probabilistic and statistical techniques for human activities recognition are also studied in [19]. A complete taxonomy of ADL has not been proposed yet, but the literature offers some preliminary efforts to structure the actions within the ADL in respect to their relative time of occurrence and their duration [14][20].

Besides the ADL representation and modelling, we may find further challenges such as the behaviour mining and learning [6][9][10], the design of a system able to deal with environments containing several users simultaneously [21][22], the migration of learned ADLs to different smart spaces [15][23], and the ADL model adaptation to changes in user habits [13][24][25], between others. Unfortunately, nowadays there is no complete solution covering all these aspects in the literature. In respect to the later problem, a life-long fuzzy learning and adaptation technique for embedded agents in ubiquitous computing environments has been proposed in [12][13][26]. It was tested into *iDorm*, a test-bed bedroom equipped with light, temperature, humidity, pressure, and phone use sensors, and embedded technologies such as an intelligent fridge and a PDA to operate the system. Another similar approach has been developed in [27] with the aims of maximization of the user's comfort and to achieve adaptive features to changes in the human activities. A Naive-Bayes classifier with real-time adaptation has been proposed in [25] for activity changes detection. Lately, in [6][24] it is proposed the combination of two different algorithms: The first one is used to mine frequent activities with probabilistic methods, while the second finds relevant changes in the activities that are related to a habit change.

B. Proposal overview

The detection of human behaviour changes require the use of dynamic system modelling, so that the behaviour model can be adapted in real time to the new user circumstances or home

M. Ros, M.P. Cuéllar, M. Delgado and A. Vila Department of Computer Science and Artificial Intelligence, University of Granada, Granada, Spain. E-mail: (marosiz@decsai.ugr.es, manupc@decsai.ugr.es, mdelgado@ugr.es, vila@decsai.ugr.es). Hani Hagras is with The Computational Intelligence Centre, University of Essex, Colchester, United Kingdom. Email: hani@essex.ac.uk

space migration. In addition, the adaptation procedure must be efficient enough to minimize the false abnormal behaviour detection rate. This work addresses the problems of ADL inference and the real time adaptation of the learned model to changes in environment and user habits. In our approach, we separate the human behaviour model from the learning procedure. With this strategy, we allow the behaviour model to change dynamically in real time, according to the data learned and to the actions carried out by the user. Moreover, the learning procedure is independent from the computer behaviour representation, which allows the use of different learning techniques. The proposed behaviour representation is shown as a graphical model with the advantage of easy configuration by the user by means of a simple graphical interface. This feature will be useful in commercial developments, to allow the user to program new behaviours or to reconfigure the existing ones easily. In addition, the human monitoring of behaviours is easier when a simple graphical interface is provided.

With respect to the described literature, HMMs are stationary and cannot be applied over dynamic environments. In addition, they assume order relationship constraints between all the actions in a behaviour and show limitations when the actions can be executed in any order. These facts were highlighted in [28][19], where the authors suggested to use Skip Chain Conditional Random Fields [19] to overcome these problems, and multiple HMMs to consider every action order execution [28]. Additionally, first order assumption may become a problem for the detection of temporal constraints between actions if the user performs several tasks simultaneously. In our opinion, all this kind of problem can be solved thanks to definition of a flexible sequence of execution groups/levels of actions. Those actions that belong to lower levels must be executed before than the actions in the deepest ones. With this structure, we avoid the fact that an action only depends on the actions performed at the previous time. In addition, the behaviour model is capable of adapting to deal with the user own free will in the execution time, to deal with behaviours that may be executed in different ways.

On the other hand, proposals using knowledge-based systems [11][26][13] have proven a good adaptation to changes in user habits. However, the behaviour is encoded as sets of rules so that its representation is not intuitive for the final user and non-experts. The reconfiguration or creation of behaviours by the user requires a new learning in these cases, and makes difficult the migration to other smart spaces. In our approach these limitations are overcome by means of real time learning of the user behaviour changes, and the simple graphical model used to represent the ADLs allows the system reconfiguration by the user if needed. We offer a detailed description of the behaviour model in Section III.

To achieve the goal of learning of behaviour models and their adaptation to user habit changes, we select Learning Automata (LA) [29]. LA are simple stochastic learning machines based on reinforcement learning, which can be combined easily to build complex systems. The reference [30] offers a review on LA and recent applications are [34][35][36][37]. Learning Automata contribute to our approach as follows: From our point of view, a Learning Automaton is a *building block* that can be matched uniquely with a sensor event in a smart home. Thus, the goal of a Learning Automaton for the inference of behaviour models is to find the execution level in which its corresponding sensor event should be located in the model of a correct human activity. Each LA is autonomous and does not depend on the existence of other LAs, and this fact allows us to change, add or remove sensors in the smart space without need of system recalibration in respect to the remaining sensors. This feature also eases the migration of the smart space due to the high modularity provided by LA. On the other hand, another feature of interest regarding LA is their capability to learn online. In our work, this fact eases the real-time adaptation of the behaviour model to user-specific habit changes. Regarding human behaviour modelling, LA have been used to study the evolution of rumour diffusion in [31], and to design tutorial-like systems in [32]. To the best of our knowledge, the use of LA for ADL learning and modelling is new, since we have not found proposals in the literature to tackle this problem from a similar point of view. In this work, we use teams of LA [33] to discover models of human behaviours in AAL scenarios. We detail this contribution in Section IV.

The remainder of this paper is organized as follows. Section II contains an overview of Learning Automata to show the basis of our approach. After that, Section III describes the behaviour model representation, considering the temporal constraints between the user's actions. Section IV shows how a team of Learning Automata can generate a behaviour model representation and explains the learning process. In Section V, the system self-adaptation capability is studied in depth. In order to test the whole system, we have performed a set of experiments at iSpace. The experimental results are shown in Section VI, and finally, Section VII offers the conclusion.

II. Learning Automata overview

A learning automaton [29] is a decision-making stochastic machine defined by the tuple $< \alpha, Q, R, T >$, where α is a set of available actions for the automaton, Q is the internal state, R is the set of available reinforcement values for the automaton, and T is a reinforcement learning scheme. The operation of a learning automaton is described as follows (Figure 1): at each k time instant, the automaton selects an action $a(k) \in \alpha$ according to the automaton internal state $Q(k)$, and sends $a(k)$ to an unknown random environment. Then, the environment evaluates the action chosen by the automaton with a fitness function $f : D \times \alpha \rightarrow R$ and provides a reward or penalty reinforcement value $\beta(k) = f(D, a(k))$, where D is the expectation to obtain a reward. The internal state of the automaton is updated with the reinforcement scheme, $Q(k+1) = T(Q(k), a(k), \beta(k))$. This process is repeated until a predefined desired condition is fulfilled, as for example the convergence to the learned optimal action. There is a wide variety of LA depending on the design of α, R, and the environment. For the sake of brevity, we suggest reference [30] for a complete description of the classic LA schemes.

In this work, we use *Finite Action-set Learning Automata* (FALA) [30] to learn behaviour models from user monitoring.

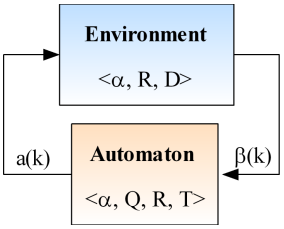


Fig. 1. Operation of a learning automaton

Here, the set of available actions $\alpha = \{\alpha_1, \alpha_2, ..., \alpha_n\}$ is discrete and finite, and the internal state of the automaton is modelled as a probability distribution for the action selection $Q(k) = P(k) = [p_1(k), p_2(k), ..., p_n(k)]^t$. At each k time instant, the automaton chooses the action $a(k) = \alpha_i$ according to this probability distribution, and applies it over the environment. Depending on the environment response, the environment model could be called in different ways. When the reward is a value between 0 or 1, $\beta(k) \in \{0, 1\}$, the environment is called *P-model*; if the reward is selected from a finite discrete response sets, $\beta(k) \in \{\beta_1, \beta_2, ..., \beta_m\}$, then it is called a $Q-model$; and finally, when the reward is included in the interval $\in [0, 1]$, $\beta(k) \in [0, 1]$, then the environment is a $S - model$.

The capabilities of a single learning automaton to solve complex problems are limited, since it is only able to learn a single optimal action. For this reason, LA-based systems use several learning automata that are organized to build the whole solution. The best known way to achieve this is to build teams of LA that play a common pay-off game to maximize the reward from the environment [33]. Nevertheless, the existing literature also offers other designs such as hierarchical systems of LA [38][39], networks of LA [35][40], other game theory-based approaches [32][41][42], or hybrid models [34][43][44]. In our work, the design of a system using teams of LA encompasses further advantages such as the ease of scalability. Moreover, the reinforcement learning capability is useful for real-time adaptation and the migration to similar smart spaces where the behaviour may take place.

III. The behaviour model

The behaviour model is a representation of a human behaviour. Some examples of behaviour models in the literature were described in the introduction, such as HMMs [9] and fuzzy rule-based control systems [12][13]. A human behaviour is a sequence of human actions that could (or not) have temporal relationship constraints between them. The formal descriptions that support our research are the following:

Definition 3.1 (Human Action): A human action h_i is a triple $h_i =< l_i, s_i, t_i >$, where l_i is a label to identify the action, s_i is a sensor event associated with the action, and t_i is a time stamp in which it occurs. An example instance of the human action *"open the front door"* could be $<$"$OpenFrontDoor$", "$FrontDoorSensor\#1234 = ON$", "$8:30:47a.m.$"$>$.

In order to distinguish between human and LA actions for reading clarity, we will rename the human action as human task from now on, without loss of meaning.

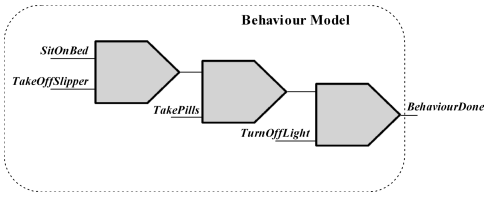


Fig. 2. behaviour model of the behaviour "To go to sleep"

Definition 3.2 (behaviour): A human behaviour B^i is a tuple $< H, R^i, C^i >$, where $H = \{h_1 h_2, ..., h_n\}$ is a set of available human tasks, $R^i = \{r_1^i, r_2^i, ..., r_{|R^i|}^i\} \subseteq H$ is the subset of human tasks that are required to complete the behaviour, and C^i is a set of temporal constraints between the tasks in R^i, defined over a partial order relationship operator $\prec$ (eq. 1). In brief, an action r_a^i precedes r_b^i in time in a behaviour B^i, and it is written $r_a^i \prec r_b^i$, if the time stamp t_a of r_a^i is lower than the time stamp t_b of r_b^i, for all the possible instances of the behaviour.

$$r_a^i \prec r_b^i \leftrightarrow t_a < t_b \tag{1}$$

We illustrate the definition of behaviour with the example *"To go to sleep"*. The set H is the set of all the available tasks that the user could carry out at home, and R contains the relevant tasks for the behaviour, as for instance, the set {*"To sit on the bed"*, *"To take the slipper off"*, *"To take the pills"*, and *"To turn off the light"*}. Let us assume that the behaviour is normal if the user always sits on the bed before taking the pills, and the last action he/she does is to turn off the light. Then, the set of temporal constraints between the human tasks is $C =$ { *"To take the slipper off"*$\prec$*"To turn off the light"*, *"To take the pills"*$\prec$*"To turn off the light"*, and *"To sit on the bed"*$\prec$*"To take the pills"*}.

The temporal constraints between tasks allow to build a behaviour model composed by execution levels (Figure 2), so that the tasks at level $l+1$ should be carried out only if all the tasks in the lower levels have been performed. This model can be easily implemented as a sequence of AND gates connected in cascade. However, this design is static and does not allow to model behaviours that can be executed in different ways. For instance, the task *"To take the slipper off"* could be executed either in level 1 or 2 in the previous example. To avoid this situation, and to model the stochastic nature of the user routine, we allow each task to be assigned to an execution level with a probability value. Thus, the underlying representation of a behaviour is a matrix whose rows are the available tasks for the user, and the columns are the probability of being performed at each execution level. We also include an additional probability value to identify if the task is relevant for the behaviour or not. Table I exemplifies a matrix with 4 levels maximum to represent the behaviour of Figure 2, where the level *U* means that the action is unused in the behaviour.

The relation between the graphical model and the underlying probability matrix is that the first one always contains the *most expected behaviour*, i.e. the assignment of a task to the level of maximum probability of execution. This allows the graphical model to evolve in time as the user executes tasks,

TABLE I
EXAMPLE MATRIX TO REPRESENT THE BEHAVIOUR *"To go to sleep"*

Tasks \ Levels	1	2	3	4	U
"To sit on the bed"	1	0	0	0	0
"To take the slipper off"	0.7	0.3	0	0	0
"To open kitchen's tap"	0	0	0	0	1
"To take the pills"	0	1	0	0	0
"To turn off the light"	0	0	1	0	0

in behaviours that can be executed in different ways. In the example, if the user sits on the bed (level 1) and takes his/her pills (level 2), the graphical model must change in real time to represent that the user must take the slipper off at level 2 with probability 1. Otherwise, a temporal constraint violation would be detected and the behaviour would be considered as abnormal.

IV. A TEAM OF LEARNING AUTOMATA TO DISCOVER HUMAN BEHAVIOURS

This section addresses the learning of behaviour models from user activity monitoring. The problem is stated as to find the probability distribution for each human task in a given behaviour, to match the task with their corresponding level in the behaviour model, or to discard the action if it is not relevant for the activity. In our approach, we have selected Finite Action-set Learning Automata (FALA) to solve the problem, since the stochastic nature of FALA is suitable to learn the best probability distribution of the task assignment in the behaviour model. In addition, it adapts this probability distribution in real time to user habit changes, or reinforces the learned behaviour models to a specific user. The use of LA eases the modularity of the system, since each LA is considered as an autonomous *building block*. This advantage is especially relevant when the system is upgraded with new sensors, or when the behaviour models must be migrated to other smart spaces, since these changes do not alter the system design and a new re-learning is not required.

A behaviour model is learned by means of a team of FALA. The team is composed by the same number of FALA as human tasks in H, although additional FALA could be included as team members to model behaviours where the same task may be carried out several times. Each FALA is uniquely matched with a human task instance. For simplicity, we assume that the learning automaton LA^i is matched with the task h_i. We also remark that there is no information sharing or communication between FALA. This fact eases the inclusion/removal of automata if we add/change the sensing elements at the smart space.

The cardinal of the set of available actions for each FALA is $M+1$, where M stands for a parameter with the maximum number of levels allowed in a behaviour model. The selection of the j-th action by automaton LA^i, for $1 \leq j \leq M$, means that human task h_i is required to complete the behaviour and it is assigned to the j-th level in the behaviour model, while the $M+1$-th action is to indicate that the task is unused. With these considerations, the state of each FALA is represented as the probability distribution $Q^i(k) = [p_1^i(k), p_2^i(k), ..., p_{M+1}^i(k)]$, where $p_j^i(k)$ is the probability of selection of the j-th automaton action.

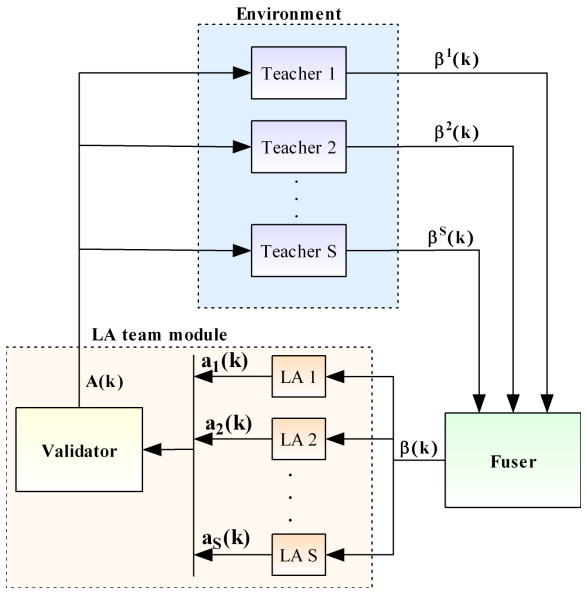


Fig. 3. Learning scheme in a multi-teacher environment

The learning goal is to find the optimal action probability distribution for each FALA. To achieve this, the team plays a common pay-off game to maximize the reward obtained by the environment. In this work, the environment is classified as *multi-teacher* [29], and it is composed by S behaviour correct samples (*teachers*) obtained from human monitoring. Figure 3 describes this design. At each time instant, the team provides the environment with a proposed behaviour model, which is evaluated by all the teachers. Each one returns a Q-model reinforcement value [29] $\beta^1(k), \beta^2(k), ..., \beta^S(k)$. These values are integer numbers containing the score of correct positioning of the tasks in the proposed behaviour model in respect to the others, for each sample. The overall score is obtained as $f(k) = \sum_{s=1}^{S} \beta^s(k)$, and the automata receive a P-model [29] reward $\beta(k) = 1$ if $f(k) \geq f(k-1)$, or a penalty $\beta(k) = 0$ otherwise. The *Fuser* module in Figure 3 performs the tasks of computing $f(k)$ and to transform the environment Q-model values $\beta^s(k)$ into the P-model reinforcement value $\beta(k)$ obtained by the team. To be more precise, the values $\beta^s(k)$ are computed for each sample as follows:

- Initially, a reinforcement value $\beta^s(k) = 0$ is assumed for the s-th behaviour sample.
- For all available automata LA^i in the team, if task h_i is unused in the behaviour proposed and it is not contained in the s-th sample, this is a success and reinforcement value $\beta^s(k)$ is increased in $+n$, where n is the number of automata in the team. In contrast, if the sample contains h_i then $\beta^s(k)$ is decreased in $-n$ units.
- Every pair of tasks assigned to different levels in the behaviour model increases the reward $+1$, if their temporal relationship constraints are not violated in the sample. Otherwise, the reinforcement is decreased in -1.
- Every pair of not penalized tasks assigned to the same

level is rewarded with $+1$ in order to minimize the number of used levels in the behaviour model.

Algorithm 1 Learning algorithm for the team

Require: $\triangle \in (0, 1]$ the learning rate
Require: n the number of automata in the team, typically $n = |H|$
Require: M the maximum number of levels allowed in a behaviour model, typically $M = |H|$
1: Initialize $k = 1, Q^i(k) : p_j^i(k) = 1/(M+1), \forall i = 1..n, \forall j = 1..M+1$
2: Initialize the best behaviour model found $V(k)$ with $v^i(k) = M+1, \forall i = 1..n$
3: **while** stopping criterion not satisfied **do**
4: **for** every automaton $i = 1..n$ **do**
5: The i-th automaton selects an action $a_j(k)$ according to the action probability distribution $Q^i(k)$
6: **end for**
7: $A(k) = Validate(a_1(k), a_2(k), ..., a_n(k))$
8: **for** every teacher $s = 1..S$ **do**
9: Retrieve reinforcement value $\beta^s(k)$ for the behaviour model $A(k)$
10: **end for**
11: Compute $[\beta(k), f(k)] = Fuser(\beta^1(k), \beta^2(k), ..., \beta^S(k))$
12: **if** $f(k) > max_{j=1..k-1}\{f(j)\}$ **then**
13: Update $V(k) = A(k)$
14: **end if**
15: **for** every automaton $i = 1..n$ **do**
16: Compute $Q^i(k+1) = T(Q^i(k), a_i(k), \beta(k), v_i(k))$
17: **end for**
18: k=k+1
19: **end while**
20: **return** $V(k), Q^i(k) \forall i \in \{1..n\}$

Common pay-off games of LA have been traditionally solved using estimator algorithms such as the *Pursuit* schemes [30], which also help to speed up the learning convergence rate. However, due to their memory overhead limitations because of the large number of LA in the team [45], these methods become ineffective in our work. We have developed a learning algorithm to overcome this situation, and it is described in depth in Algorithm 1. This algorithm starts by initializing all the automata to the pure chance automaton. In addition, vector $V(k) = [v^1(k), v^2(k), ..., v^n(k)]^t$ stores the best behaviour model found until the k-th iteration, and it is initialized to the empty behaviour model at line 2, i.e., no human tasks are considered in the behaviour. The learning process is controlled between lines 3 to 19: First, all the automata select their action to build a behaviour model, and this is sent to the *validator* module (lines 4-7). The *validator* ensures the robustness of the behaviour model proposed by the team so that there are no intermediate unconnected levels, and the model starts at the first level. After that, the behaviour samples in the environment evaluate the behaviour model and return the reinforcement values $\beta^s(k)$ (lines 8-10). The *fuser* receives values $\beta^s(k)$ from the environment and calculates the score $f(k)$ and the reward/penalty value $\beta(k)$ to be provided to the LA team. If $f(k)$ is the best (maximum) score found until the k-th iteration, then vector $V(k)$ is updated with the behaviour model previously proposed by the team. Finally, the automata internal states are updated according to a reinforcement learning rule in line 16. In the experimental section, we test which update scheme is the best choice for our application, considering the classic continuous rules *Continuous Pursuit Reward-Penalty* (CP_{RP}, equation 2) and *Reward-Inaction* (CP_{RI}, equation 3), and the discrete rules *Discrete Pursuit Reward-Penalty* (DP_{RP}, equation 4) and *Reward-Inaction* (DP_{RI}, equation 5).

$$p_j^i(k+1) = \begin{cases} (1-\triangle)p_j^i(k); \forall j \neq v^i(k) \\ (1-\triangle)p_j^i(k) + \triangle;\ if\ j = v^i(k) \end{cases} \quad (2)$$

$$p_j^i(k{+}1) = \begin{cases} p_j^i(k);\ if\ \beta(k) = 0 \\ (1-\triangle)p_j^i(k);\ if\ \beta(k) = 1 \wedge j \neq v^i(k) \\ (1-\triangle)p_j^i(k) + \triangle;\ if\ \beta(k) = 1 \wedge j = v^i(k) \end{cases} \quad (3)$$

$$p_j^i(k+1) = \begin{cases} max\{p_j^i(k) - \triangle, 0\}; \forall j \neq v^i(k) \\ 1 - \sum_{l \neq v_i(k)} p_l^i(k+1); if\ j = v^i(k) \end{cases} \quad (4)$$

$$p_j^i(k{+}1) = \begin{cases} p_j^i(k); if\ \beta(k) = 0 \\ max\{p_j^i(k) - \triangle, 0\};\ if\ \beta(k) = 1 \wedge j \neq v^i(k) \\ 1 - \sum_{l \neq v_i(k)} p_l^i(k+1);\ if\ \beta(k) = 1 \wedge j = v^i(k) \end{cases} \quad (5)$$

In our approach, we also develop parallelization of the LA team to speed up the convergence rate during learning [46]. The experimental section shows how this strategy improves the convergence rate meaningfully since it implements a better exploitation of the solution space. It has been designed as follows: the whole LA team module of Figure 3 is replicated as many times as the number of parallel LA teams which are required. This fact implies the need to adapt both teachers in the environment and Fuser module in order to deal with the new scheme. Let us label LA_j^i as the learning automaton matched with human task h_i at the j-th parallel module. The way the system works is as follows:

- Automata LA_j^i matched with the same task h_i share the same action probability distribution $P^i(k)$.
- Every automaton LA_j^i selects its action independently of the other automata in the same or different modules.
- Every parallel module j provides a validated behaviour model from the automata in the j-th team independently of the other modules.
- A teacher t in the environment receives the behaviour models proposed by all the modules and returns a reinforcement value $\beta_j^s(k)$ for every parallel module j.
- If $\beta(k) = 1$, then a single parallel module j that obtained the highest score $f^j(k)$ at iteration k will be allowed to update the action probability distribution. On the other hand, if all the received modules $\beta(k) = 0$ and the learning rule is reward-penalty, then the probability distributions are updated by a randomly selected module.

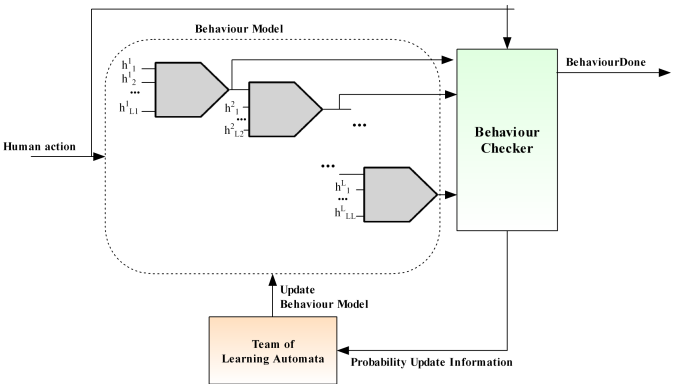


Fig. 4. Design of the system for online adaptation of the behaviour model

V. Human behaviour recognition and online adaptation

The recognition of an activity must consider that some user tasks can be executed at different levels of the behaviour model, depending on the way the behaviour is being carried out. To achieve this, we allow the behaviour graphical model to evolve in time as the user performs tasks, using the underlying task probability distributions for the behaviour. The scheme for the recognition and dynamism of the behaviour model processes is summarized in Figure 4. When a human task is carried out, it is sent to the behaviour model and to the *behaviour Checker* module, which manages the dynamic of the behaviour model for the activity being detected, the violation of temporal task relationship constraints, and the end of behaviour. In brief, the system operation is as follows:

1) Initially, the active behaviour model is the most expected behaviour, i.e. each automaton LA^i is assigned to the level j of maximum probability $\alpha_j = max_j\{p_j^i(0)\}$, and $L = 1$ stands for the current active level, i.e. the level of the first AND gate that returns false. In addition, current iteration is set to $k = 1$, and the *record of actions performed* is initialized to the empty set $B = \emptyset$.
2) Every time the user performs a task h_i, it is fetched by the system and sent to the behaviour model and the behaviour checker. For each received action, the behaviour checker module finds out the level in which the action was carried out, and saves the value in the *record of performed actions*. Here, we may encounter different situations:
   - A temporal relationship constraint is violated if h_i cannot be executed at level L and it belongs to a deeper one, and the probability $\sum_{x=1}^{L} p_x^j(k)$ to carry out the missed tasks h_j is over a threshold γ, i.e. the user did not perform some tasks and one or more cannot be executed in later levels. The behaviour checker sends an alert to the user, giving him the possibility either to respond with a corroboration of this fact, or to decide to rectify the sequence of actions.
   - On the other hand, if $\sum_{x=1}^{L} p_x^j(k) \leq \gamma$, the missed tasks h_j can be executed in deeper levels, and the graphical model evolves and sets h_j to the level of maximum probability after L.
   - If a predefined time window has finished and none of the resting relevant tasks for the behaviour has been executed, then it may be concluded that either the user was not carrying out the behaviour, a human habit change is taking place, or he/she forgot the tasks. This fact also requires user interaction to find out for sure if it was done deliberately.
3) When all the tasks required to complete the behaviour have been carried out, then all the AND gates return true and the system enables a $behaviourDone$ response which indicates that the behaviour has been recognized. The probabilities $P^i(k+1)$ are updated with Equation 6.

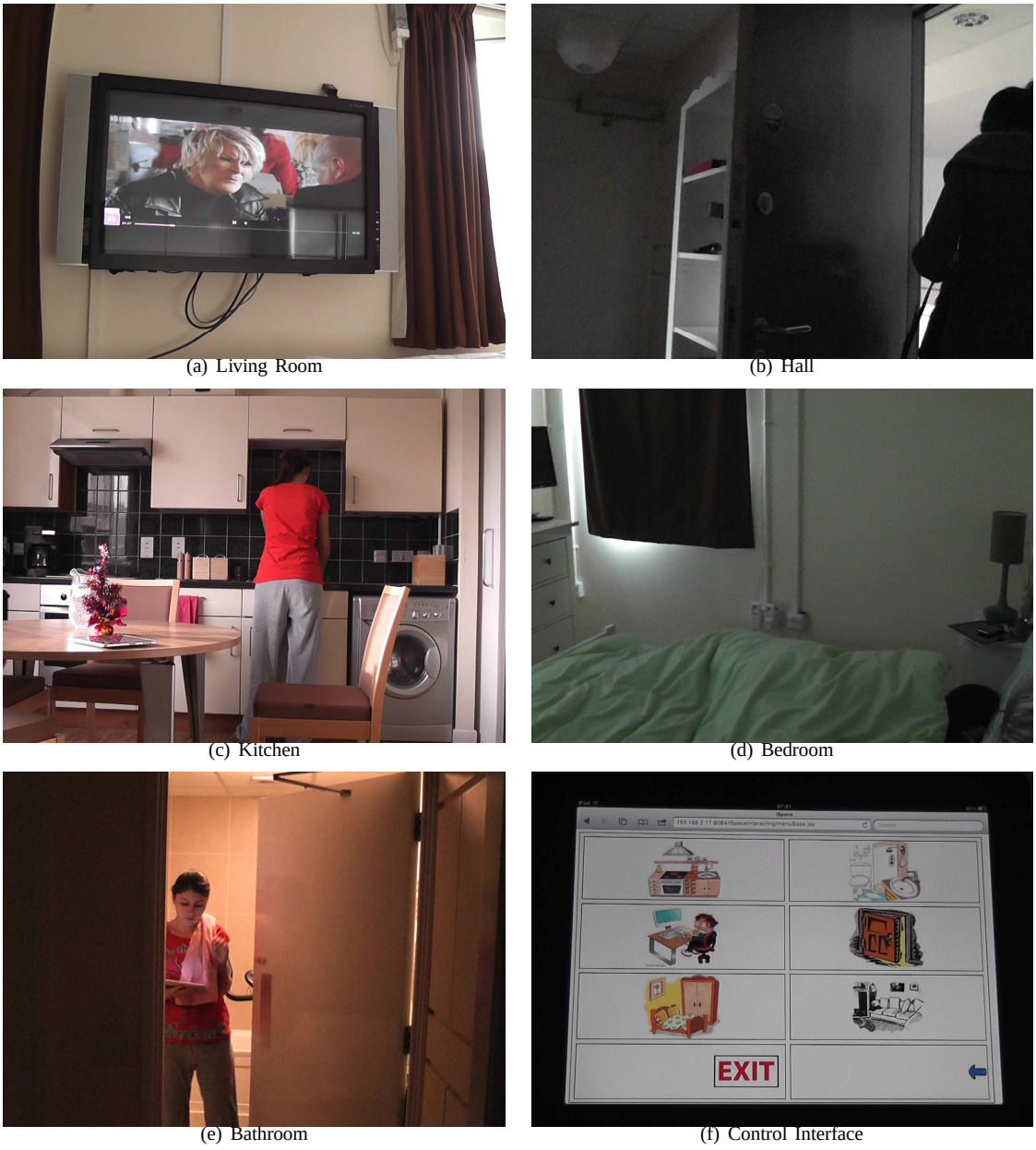
$$p_j^i(k+1) = \begin{cases} max(p_j^i(k) - \triangle, 0); j \neq l \\ 1 - \sum_{m=1, i \neq l}^{S+1} p_m^i(k+1, 1); j = l \end{cases} \quad (6)$$

The key aspect to achieve a suitable adaptation with equation 6 is to find the optimal $\triangle$ parameter. Small values of $\triangle$ may slow down the convergence to the adapted behaviours and to increase the alerts and user interaction, while high values may over train the system and to include other tasks that are not relevant for the activity and that they have been carried out by the user during the behaviour. In these cases, the false negative rate is also increased. We study this fact in the experimental section.

VI. Experiments and results

In order to prove the running of our proposal, we have performed several unique experiments within the iSpace at the University of Essex. The iSpace is a real Ambient Intelligent Environment established in a two-bedroomed apartment, composed by a bedroom, a study, an entrance hall, a fully equipped kitchen, a living room and a bathroom (see figure 5). Each room contains a set of devices to interact and manage the environment, which are connected using the Universal Plug&Play (UPnP) architecture. The communication with the devices is carried out using an API programmed in Java, that provides discovery of the services belonging to a range of devices and interaction through messages over the network.

Regarding the devices in our experiment, we monitored both sensors and actuators, to control their state in every moment. Namely, two input sensors are monitored: door sensor and Ubisense Tracking Sensor. iSpace is organized in four labelled areas as kitchen, sofa, bedroom and hall. Ubisense Tracking Sensor detects when the user either enters into or exits a zone. In addition, we check the state of 20 actuators consisting of light-switches, curtain-controllers, TV remote control and switch controllers. The last set of actuators (switch controllers) controls whether some specific electrical appliance is connected to the UPnP network or not. Those devices are: a coffee machine, a little bedroom lamp, an alarm clock, a toaster, a radio and the door lock. To manage the environment, we have developed an interface to send the specific commands to the actuators such as turn them on and off (See figure 5(f)).



(a) Living Room (b) Hall (c) Kitchen (d) Bedroom (e) Bathroom (f) Control Interface

Fig. 5. iSpace rooms

The goal of our experiment is to prove that our system is capable of learning the behaviour model for different types of behaviour, recognizing the user current activities and adapting online the behaviour model in order to evolve the behaviour model with the user. For this purpose, we collected data from our participant, during the performance of three different behaviours: Wake up, Have breakfast and Leave home. The participant was requested to perform these behaviours in the iSpace in a natural way, i.e., performing exactly the same sequence of actions that s/he usually carries out at his/her home own home.

At the beginning of each experiment session, the system applied a specific mode in iSpace, named as *Sleepy mode*, consisting of the following specific actions: closing all curtains, turning all the devices off and turning all the lights off. The participant wore a tag of Ubisense Tracking Sensor, which identified exactly where s/he is at the space at any moment. Each experiment session started when the alarm clock went off at a specific time provided by the participant. From that moment on, the system recorded all the user interactions (events) with the environment, received either from the sensors or through the interface. For each event, the time and the activated sensor/actuator were stored. Each experiment session finished when the participant left the iSpace.

The participant was requested to perform the experiment during 25 days. Briefly, we explain the general scenario that our user performed during the experiments, since the scenarios might be performed in different ways and times. Once the alarm clock had gone off, our participant turned it off and switched the bedroom lights on. She left the bedroom and went to the bathroom, where she turned the lights on. After some minutes, she exited it and went into the kitchen, through the

hall, turning some lights on/off. At the kitchen, she prepared her breakfast, using the coffee machine and the toaster. In some sessions, the participant watched the television at the living room meanwhile she was having breakfast. Straight afterwards, she left the kitchen and went into the bathroom again. Several minutes later, she entered into the bedroom, packed her stuff, went into the hall, unlocked the door, opened the door, stepped out and closed the door. The door was locked again automatically. During the first 5 days, the participant was asked to perform the behaviour as valid, since those data would be used to learn the behaviour model. From this moment, the participant had completely freedom to perform the behaviour both correct and incorrect. For the last 10 days, the participant kept doing the same behaviours, however, we changed the environment: the bedroom light could not be turned on, so she had to open the curtains; the toaster could not be used and the door did not require to be unlocked to open. The user had to change her behaviour, and, therefore, our behaviour model had to be adapted as well. When the experiment finished, the participant was requested to classify his own behaviours as normal or abnormal. The collected data was organized as 3 databases: DB_1 with only valid behaviours, DB_2 with normal and abnormal behaviours and DB_3 with normal behaviours in the adapted environment as simulation of habits change.

A. Study of learning rules

In this section we study which learning rule is the best choice for ADL inference and the benefits of parallelization. Concretely, we learned the behaviour models from the data in DB_1 using the algorithm explained in Section IV. In order to define an uniform comparison criterion, we maximized the overall score value $f(k)$ calculated by the *Fuser* module. This value provides a measure of the suitability of the behaviour model being learned with respect to the behaviour data patterns.

The algorithms were executed using the following parameters:

- Primary stopping criterion: to reach 20 iterations where the LA team returns the same behaviour model.
- Secondary stopping criterion: to reach 10000 iterations.
- Learning rate for the learning process: $\triangle = 0.001$ (continuous learning rules) and $\triangle = 0.0001$ (discrete learning rules).
- Number of parallel modules of teams of LA (only for the parallel approaches): 10.

The previous learning rates were obtained after the execution of preliminary experiments to optimize this parameter. In the preliminary results, higher values of the learning rates made the algorithms become trapped into local optima solutions, while lower values provided a slower convergence. All the experiments were replicated 30 times to carry out statistical analyses about the robustness of our proposal. Table II shows the best, worst, average overall score value, its standard deviation and the quartiles provided by each learning rule and its parallel version. The best results for each behaviour have been highlighted.

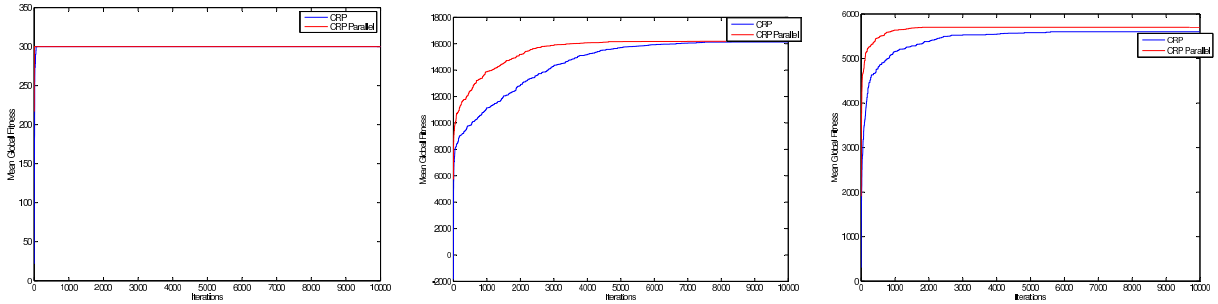
The learning rule that provides the best results in all the problems is the Continuous Reward-Penalty (CP_{RP}). It is also the most robust method, since the standard deviation is the lowest in 4 of the 6 studied problems. Moreover, Table II shows that the mean value of the overall score of CP_{RP} is the closest to the best solution found for each problem. From this fact, we may conclude that the expected performance of CP_{RP} provides optimal or close to optimal solutions on average. With respect to the other learning schemes, in Table II we observe that the learning rules based on Reward-Inaction reinforcement are also able to provide adequate solutions regarding the best overall score. However, due to the existence of absorbing states in the studied problems, these methods tend to become trapped into local optimal solutions. The existence of the absorbing states influences the robustness of these methods, as well. The high standard deviation that they provide in general aids to corroborate this assumption. This situation could be solved by decreasing the learning rate even more; however, the speed of convergence would be affected consequently slowing down the methods.

Table II also shows that the parallel approaches makes a better exploitation of the solution space and improve results of the normal ones. Non-parallel approaches make a single sample over the solution space and it is provided to the environment, while the parallel method makes several samples that are evaluated. Then, the probability distributions are updated using the best solution scored by the environment. The increase of the average and worst overall score value of these methods suggests that the expected quality of a solution is higher using parallelization. Furthermore, we observe a general increase in the robustness of the results for the parallel approaches: Their standard deviation is lower than in the non-parallel ones, which means that there is a higher guarantee to find near-optimal solutions with these algorithms. Another hot point of the parallel proposals is the increase in the convergence rate in respect to the non-parallel approaches. Figure 6 supports this assumption and shows the average evolution of the overall score during the learning of the behaviour model across the 30 algorithm executions. In the figure, we only focus in the CP_{RP} learning rule for clarity, since this rule provided the best experimental results in Table II. We may observe that the evolution of the overall score to the optimal solution is faster in the parallel proposal since its value is always over the non-parallel algorithm. In addition, the final environment value is also higher in the parallel proposal and this fact supports the results in Table II. We may conclude that parallelization is preferred to normal learning rules for a better mining of the behaviour model.

To support the previous analyses, we have applied a Students T-test with 95% of confidence level to the results obtained for each method, in order to compare if there are significant differences among the results obtained by the different approaches. The T-test was taken on pairs of algorithm, sorted in descending order of overall score. Table gathers the results of the test. In addition, we remark with (-) those algorithms whose test concludes that there are significant differences in the results and therefore, the method on the left (higher average overall score) is better. Regarding the results obtained for the

TABLE II
SUMMARY OF RESULTS. ROBUSTNESS AND PERFORMANCE OF THE LEARNING RULES.

		Normal							Parallel						
behaviour	**Algorithm**	**Best**	**Worst**	**Mean**	**S.d.**	**Q(1)**	**Median**	**Q(3)**	**Best**	**Worst**	**Mean**	**S.d.**	**Q(1)**	**Median**	**Q(3)**
$Wake\ up$	CP_{RP}	**300**	**300**	**300**	**0**	-	**300**	-	**300**	**300**	**300**	**0**	-	**300**	-
	CP_{RI}	300	300	300	0	-	300	-	300	300	300	0	-	300	-
	DP_{RP}	300	300	300	0	-	300	-	300	300	300	0	-	300	-
	DP_{RI}	300	300	300	0	-	300	-	300	300	300	0	-	300	-
$Have\ Breakfast$	CP_{RP}	**16400**	**15944**	**16132**	**169**	**15992**	**16088**	**16400**	**16400**	**15992**	**16186**	**172**	**15992**	**16196**	**16304**
	CP_{RI}	16400	15196	15880	363	15612	15972	16088	16400	15992	16134	163	15992	16088	16352
	DP_{RP}	15940	12420	14233	996	13504	14340	15008	16400	15196	15917	389	15388	15992	16304
	DP_{RI}	16304	13588	15146	743	14712	15200	15532	16400	15008	15960	337	15532	15992	16252
$Leave\ Home$	CP_{RP}	**5700**	**5400**	**5600**	**1438**	**5400**	**5700**	-	**5700**	**5700**	**5700**	**0**	-	**5700**	-
	CP_{RI}	5700	5400	5640	1221	5400	5700	-	5700	5700	5700	0	-	5700	-
	DP_{RP}	5700	3900	5150	4392	5000	5350	5400	5700	5400	5603.3	1402	5400	5700	-
	DP_{RI}	5700	5000	5410	2354	5050	5400	5700	5700	5400	5666.7	884	5450	5700	-



(a) behaviour $Wake\_up$ (b) behaviour $Have\_breakfast$ (c) behaviour $Leave\_home$

Fig. 6. Mean of the evolution of the overall score in the studied behaviours using parallel modules of teams of LA

behaviour $Wake\_up$, all the methods are statistically equivalent since all of them are able to get the best solution. However, the results from both $Have\_breakfast$ and $Leave\_home$ contains differences that should be emphasized. For instance, the CP_{RP} is in general the best choice since it is always able to extract the best alternative in any of its variants (it produces significant differences).

Another interesting result is that the continuous rules have provided a better performance than the discrete ones. In our experiments, the discretization of the probability distribution of the automata is not suitable. It may speed up the process, but it also may become trapped in local optima solutions more easily.

B. System evaluation

Once the learning process has been completed, the next step is to evaluate the recognition of normal and abnormal behaviours. We used the data collected in DB_2, which contains 7 normal and 3 abnormal behaviours according with the participant opinion. The behaviour data were tested over the LA team that obtained the best overall score using the Parallel CP_{RP} learning rule after the training in the previous section.

In order to reinforce our proposal, we compared our results with another method proposed in the literature [47]. This method is based on Hidden Markov Models (HMM), a common technique in ADL recognition. The authors propose to build a HMM for each behaviour, where the number of states is the number of available user activities. The prior and transition probabilities were learned from the training data, and the emission probability matrix was set to the identity matrix because our main assumption is that each human task is uniquely determined by a sensor event. For the recognition, a behaviour is correctly identified if the probability to generate the sequence of human tasks with the HMM is over a threshold. This threshold was set to the minimum probability of HMM sequence generation in the training patterns, and we use this value because it provided the most accurate results in our experiments. The HMM has been training under the same conditions that we trained our own proposal. In this way, we are in the correct condition to compare the results of both methods. Table IV summarizes the results.

The valid and abnormal test patterns were used for recognition, using a sliding window to bound the input tasks to the HMM. We classify the normal/abnormal behaviour samples that are correctly identified as true positive and negative patterns respectively, and the false positive samples are the abnormal behaviours being detected as normal. As shown in Table IV, the normal behaviour samples were correctly identified using both techniques. The values for behaviour *Have_breakfast* should be emphasized, since it is the HMM the one that obtains the better results for the true positive patterns. This fact is justified because of the large number of actions that belongs to that behaviour (about 10 different actions).

Regarding the abnormal behaviours, our proposal provides better results in comparison to HMM. The reason is the processing of the *rubbish tasks* in our system, i.e. those tasks that do not belong to the behaviour, but they are carried out by the user during its execution. In a HMM model, any action depends on the previous actions carried out. This means that, if the user performs a rubbish action, then the system must have it into consideration during recognition. It would be difficult to propose higher order Markov models to solve the ADL

TABLE III
RESULTS OF THE T-TEST FOR THE COMPARISON OF THE LEARNING RULES

Wake_up		Have_breakfast	Leave_Home
$CP_{RP}Parallel$ - CP_{RP}	1.0000 (-)	0.2227(-)	0.0003
CP_{RP} - CP_{RI}	1.0000(-)	0.0000	0.2502(-)
CP_{RI} - DP_{RI}	1.0000(-)	0.0000	0.0000
DP_{RI} - DP_{RP}	1.0000(-)	0.0000	0.0059

TABLE IV
RECOGNITION ACCURACY OF TEMPORAL RELATIONSHIP CONSTRAINTS

behaviour	Model	% true positive	% true negative	% false positive	% false negative	% Total
$Wake\_up$	behaviour model	100	100	0	0	100
	HMM	100	100	0	0	100
$Have\_breakfast$	behaviour model	85,72	100	0	14,28	92,86
	HMM	100	33,33	66,67	0	66,67
$Leave\_home$	behaviour model	100	100	0	0	100
	HMM	100	66,67	33,33	0	83,33

recognition problem in general terms, since we do not know a priori the number of rubbish tasks that a user could do, as for instance those related to interleaved behaviours. As far as we considered, this type of actions should not influence the recognition process, but the adaptation (since its execution in several behaviour instances may become a change in user habit). Our proposal only considers those actions that are relevant in the behaviour model for recognition. Therefore, the execution of rubbish tasks does not influences the correct behaviour detection.

C. Adaptation to habit changes

In this section we test the system's adaptation capability to small changes in the human habits. As stated previously, we changed the environment, blocking the control to some facilities at iSpace. This data are collected in DB_3. The initial configuration to test the adaptation feature contained the best solution reported by the Parallel CP_{RP} algorithm in section VI-A.

The adaptation is a continuous process. When a behaviour is detected as correct, the probabilities of the LA in its behaviour model are rewarded according with the equations 6; while if the behaviour is detected as incorrect, the probabilities are not rewarded unless the user certifies that the behaviour was done in a right manner. In this way, we reinforce or change the learned knowledge. We remark that we do not only reward the action that already belongs to the behaviour, but also those actions that might belong but that are considered rubbish at that moment. This considerations allow new tasks to become part of (or to be removed from) the behaviour model in the future.

To justify the adaptation process, we compare the results obtained using DB_3 in three different situations: without adaptation, with an adaptation rate of 0.1 and with an adaptation rate equals to 0.01. All the execution starts from the same behaviour model, the one obtained using the Parallel CP_{RP} learning rule after the training in section VI. Table V summarizes the results.

If we observe the results collected in Table V without adaptation (first row of each behaviour), we can affirm that an online adaptation is justified since none of the learned behaviours are recognized. When the environment changes, the user habit usually does it too; therefore, it is more than recommended to add an adaptation process to allow the system to deal with these new circumstances. Table V shows that, when the adaptation is running, the results are improved. Note that if we do not use the adaptation, the system is not able to detect the behaviours; what means that the adaptation data set does not match with any of the learned behaviour models. However, when we activate the adaptation, the recognition results are improved. A brief explanation of this fact is as follows: At first, the adaptation data set does not match with any of the learned behaviour models, therefore the system should adapt itself in order to improve the recognizing process. After the execution of some patterns, the system changes the structure of the behaviour model and the new action relationship constraints are found.

In addition, in these experiments we studied the effect of the learning rate for the adaptation capability. This learning rate manages the speed of the adaptation process. We found out that small values of $\triangle$ produced a slower adaptation, while large values speed up the process excessively. Therefore, this rate should be selected carefully. In general, we could think that large values should be better in situations as the ones presented in this paper: absorb the change of the environment as soon as possible. However, together with the changes in the environment, the model could accept as relevant other additional *rubbish* tasks that are not part of the behaviour, since the probability distribution of their corresponding LA changes drastically with the high learning rate. When a new valid pattern would be analysed and would not contain such tasks, the behaviour would not be recognized and the false negative rate might be increased.

Regarding the time consumption during the adaptation process, Table VI shows the results obtained for the adaptation process after processing DB_3 in seconds. We observe how the

TABLE V
RECOGNITION ACCURACY AFTER ADAPTATION

behaviour	Model	% true positive	% false positive
$Wake\_up$	behaviour model	0	100
	BM adapted $\triangle = 0.001$	40	60
	BM adapted $\triangle = 0.1$	70	30
$Have\_breakfast$	behaviour model	10	90
	BM adapted $\triangle = 0.001$	10	90
	BM adapted $\triangle = 0.1$	60	40
$Leave\_home$	behaviour model	0	100
	BM adapted $\triangle = 0.001$	20	80
	BM adapted $\triangle = 0.1$	90	10

TABLE VI
RECOGNITION ACCURACY OF TEMPORAL RELATIONSHIP CONSTRAINTS

behaviour	Time consumption (seconds)
$Wake\_up$	$5.74732 * 10^{-2}$
$Have\_breakfast$	$8.9309 * 10^{-2}$
$Leave\_home$	$1.21298 * 10^{-2}$

results for $Have\_Breakfast$ are larger than for the other two behaviours. The reason is that time consumption depends on the number of relevant actions that belong to the behaviour model, due to the number of elements to wait to be able to recognize a behaviour is larger.

VII. CONCLUSIONS

This work addressed the problem of learning and recognizing human behaviours. With regard to the existing literature, our method is novel since it separates the behaviour model itself from the learning process. This fact eases the real-time adaptation to new environmental situations and to possible changes of human habits. First of all, we propose a representation for a behaviour model that contains both the tasks that belong to a specific behaviour and the order relationship among them. The main advantages of this approach is that it is able to learn and mine temporal relations between actions in a behaviour, to recognize behaviours that can be executed in different ways, and is described as a graphical model that can be easily adapted manually by the user by means of a simple computer interface if needed. In addition, the learning method of human activities is independent of the behaviour model itself.

We developed a method based on Learning Automata that is able to infer behaviour models from raw data obtained from daily activities carried out by a user. The proposal uses a team of Learning Automata, where each automaton is in charge of learning the position of a human task in the model. The learning algorithm developed is suitable to train large teams of learning automata with large number of actions, therefore overcoming the traditional limitations of these models of a low convergence rate. In order to increase the speed and the convergence of the method, the system has been also parallelized. Different learning rules are studied to know which is the best learning strategy, in order to obtain the most suitable behaviour models. In our experiments, we found that a reinforcement learning rule based on *Continuous Reward-Penalty* is capable to overcome the limitations of discrete approaches, and its parallelization is able to improve the algorithm convergence rate.

The system capability to recognize normal and abnormal behaviours has been studied. In our case, we focus in the capability to evolve the behaviour model in real time as the user executes tasks, to model human activities that can be performed in any order. We found that the approach is able to learn and recognize all the temporal constraints imposed to the tasks of a behaviour. Indeed, the approach overcomes the limitations of traditional techniques such as the first order assumption of HMMs. In addition, the capability of the system to adapt itself to human habit changes also provides promising results. In the experiments, our approach has shown a fast convergence to learn the new behaviour changes and to update online the learned behaviour models.

Above all, we have performed several unique experiments in the iSpace at the University of Essex, a real Ambient Intelligent Environment equipped with ubiquitous devices (sensors and actuators) which were used to collect information about the user activities. The participant performed three behaviours: Wake up, Have Breakfast and Leave Home. From this data, we showed how the system is able to learn, recognize and adapt the behaviours.

As a future project, we would like to study the temporal information of the system, not only knowing the temporal constraints between the actions, but also studying the most likely interval when the behaviour should be performed. This fact could help both in the recognition and adaptation processes. On the other hand, the study of relations (composition, dependence, etc) between behaviours is raised as an interesting future work. To achieve these goals, we think that a more important study to categorize human tasks must be carried out.

ACKNOWLEDGEMENTS

This project has been partially supported by the project TIN2009-14538-C02-01, I+D+I national program from the Ministry of Ciencia y Tecnología, Government of Spain. We also thank the participant of our experiment for her dedication and the reviewers for their useful comments that helped to improve the manuscript quality.

REFERENCES

[1] R.-M. Droes, M. Mulvenna, C. Nugent, D. Finlay, M. Donnelly, M. Mikalsen, S. Walderhaug, T. v. Kasteren, B. Krose, S. Puglia, F. Scanu, M. O. Migliori, E. Ucar, C. Atlig, Y. Kilicaslan, O. Ucar, and J. Hou, "Healthcare systems and other applications," *IEEE Pervasive Computing*, vol. 6, no. 1, pp. 59–63, 2007.

[2] M. Rantz, M. Aud, G. Alexander, D. Oliver, D. Minner, M. Skubic, J. Keller, Z. He, M. Popescu, G. Demiris, , and S. Miller, "Tiger place: An innovative educational and research environment," in *AAAI in Eldercare: New Solutions to Old Problems*, 2008.

[3] M. Philipose, K. P. Fishkin, M. Perkowitz, D. J. Patterson, D. Fox, H. Kautz, and D. Hahnel, "Inferring activities from interactions with objects," *IEEE Pervasive Computing*, vol. 3, no. 4, pp. 50–57, 2004.

[4] L. Liao, D. J. Patterson, D. Fox, and H. Kautz, "Learning and inferring transportation routines," *Artif. Intell.*, vol. 171, no. 5-6, pp. 311–331, 2007.

[5] C. Nugent, M. D. Mulvenna, F. Moelaert, B. Bergvall-Kereborn, F. Meiland, D. Craig, R. Davies, A. Reinersmann, M. Hettinga, A.-L. Andersson, R.-M. Dries, and J. Bengtsson, "Home-based assistive technologies for people with mild dementia," in *Pervasive Computing for Quality of Life Enhancement*, vol. 4541, 2007, pp. 63–69.

[6] P. Rashidi and D. J. Cook, "Keeping the resident in the loop: adapting the smart home to the user," *Trans. Sys. Man Cyber. Part A*, vol. 39, no. 5, pp. 949–959, 2009.

[7] S. Park and H. Kautz, "Hierarchical recognition of activities of daily living using multi-scale, multi-perspective vision and rfid," in *Intelligent Environments, 2008 IET 4th International Conference on*, jul. 2008, pp. 1–4.

[8] ——, "Privacy-preserving recognition of activities in daily living from multi-view silhouettes and rfid-based training," in *AAAI Fall 2008 Symposium on AI in Eldercare: New Solutions to Old Problems, Washington, DC, November 7 - 9*, 2008.

[9] D. Wilson, D. Wyaat, and M. Philipose, "Using context history for data collection in the home," in *Pervasive*, vol. 3468, 2005.

[10] J. Boger, J. Hoey, P. Poupart, C. Boutilier, G. Fernie, and A. Mihailidis, "A planning system based on markov decision processes to guide people with dementia through activities of daily living," *IEEE Trans Inf Technol Biomed*, vol. 10, no. 2, pp. 323–333, 2006.

[11] G. Acampora and V. Loia, "Fuzzy control interoperability and scalability for adaptive domotic framework," *IEEE Transactions on Industrial Informatics*, vol. 1, no. 2, pp. 97 – 111, may. 2005.

[12] F. Doctor, H. Hagras, and V. Callaghan, "A fuzzy embedded agent-based approach for realizing ambient intelligence in intelligent inhabited environments," *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, vol. 35, no. 1, pp. 55–65, 2005.

[13] H. Hagras, V. Callaghan, M. Colley, G. Clarke, A. Pounds-Cornish, and H. Duman, "Creating an ambient-intelligence environment using embedded agents," *IEEE Intelligent Systems*, vol. 19, no. 6, pp. 12–20, 2004.

[14] V. R. Jakkula, A. S. Crandall, and D. J. Cook, "Enhancing anomaly detection using temporal pattern discovery," in *Advanced Intelligent Environments*, A. D. Kameas, V. Callagan, H. Hagras, M. Weber, and W. Minker, Eds. Springer US, 2009, pp. 175–194.

[15] P. Rashidi and D. Cook, "Home to home transfer learning," in *Proceedings of the AAAI Plan, Activity, and Intent Recognition Workshop*, 2010.

[16] J. A. Ward, P. Lukowicz, G. Troster, and T. E. Starner, "Activity recognition of assembly tasks using body-worn microphones and accelerometers," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 10, pp. 1553–1567, 2006.

[17] G. Acampora and V. Loia, "A proposal of an open ubiquitous fuzzy computing system for ambient intelligence," in *Computational Intelligence for Agent-based Systems*, ser. Studies in Computational Intelligence, R. Lee and V. Loia, Eds. Springer Berlin / Heidelberg, 2007, vol. 72, pp. 1–27.

[18] R. Messing, C. Pal, and H. Kautz, "Activity recognition using the velocity histories of tracked keypoints," in *Computer Vision, 2009 IEEE 12th International Conference on*, sep. 2009, pp. 104 –111.

[19] E. Kim, S. Helal, and D. Cook, "Human activity recognition and pattern discovery," *IEEE Pervasive Computing*, vol. 9, no. 1, pp. 48–53, 2010.

[20] V. Jakkula and D. Cook, "Enhancing smart home algorithms using temporal relations," in *Technology and Aging, IOS Press*, 2008.

[21] A. Crandall and D. J. Cook, *Learning Activity Models for Multiple Agents in a Smart Space*, 2010, p. 751.

[22] D. J. Cook, A. Crandall, G. Singla, and B. Thomas, "Detection of social interaction in smart spaces," *Cybernetics and Systems*, vol. 41, no. 2, pp. 90–104, 2010.

[23] P. Rashidi and D. Cook, "Multi home transfer learning for resident activity discovery and recognition," in *Proceedings of the International workshop on Knowledge Discovery from Sensor Data*, 2010.

[24] P. Rashidi, S. Member, D. J. Cook, L. B. Holder, and M. Schmitter-edgecombe, "Discovering activities to recognize and track in a smart environment," *IEEE Trans. on Knowledge and Data Engineering*, vol. to appear, p. 14, 2010.

[25] E. M. Tapia, S. S. Intille, and K. Larson, "Activity recognition in the home using simple and ubiquitous sensors," in *Pervasive Computing*, ser. Lecture Notes in Computer Science, A. Ferscha and F. Mattern, Eds. Springer Berlin / Heidelberg, 2004, vol. 3001, pp. 158–175–175. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-24646-6_10

[26] H. Hagras, M. Colley, V. Callaghan, G. Clarke, H. Duman, and A. Holmes, "A fuzzy incremental synchronous learning technique for embedded-agents learning and control in intelligent inhabited environments," in *Fuzzy Systems, 2002. FUZZ-IEEE'02. Proceedings of the 2002 IEEE International Conference on*, vol. 1, 2002, pp. 139 –144.

[27] G. Acampora, M. Gaeta, V. Loia, and A. V. Vasilakos, "Interoperable and adaptive fuzzy services for ambient intelligence applications," *ACM Trans. Auton. Adapt. Syst.*, vol. 5, no. 2, pp. 1–26, 2010.

[28] U. Naeem and J. Bigham, "A comparison of two hidden markov approaches to task identification in the home environment," in *Proceedings of the 2nd International Conference on Pervasive Computing and Applications*, 2007, pp. 624–634.

[29] A. S. Poznyak and K. Najim, *Learning Automata and Stochastic Optimization*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1997.

[30] M. A. L. Thathachar and P. S. Sastry, "Varieties of learning automata: an overview," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 32, no. 6, pp. 711–722, 2002.

[31] M. Meybodi and M. Taherkhani, "Application of cellular learning automata in modeling of rumor diffusion," in *Proceedings of Ninth Conference on Electrical Engineering, Power & Water Institute of Technology (ICEE 2001), Tehran, Iran*, pp. 102–110.

[32] B. J. Oommen and M. K. Hashem, "Modeling a student-classroom interaction in a tutorial-like system using learning automata," *Trans. Sys. Man Cyber. Part B*, vol. 40, no. 1, pp. 29–42, 2010.

[33] M. A. L. Thathachar and K. R. Ramakrishnan, "A cooperative game of a pair of learning automata," *Automatica*, vol. 20, no. 6, pp. 797 – 801, 1984.

[34] R. Forsati and M. R. Meybodi, "Effective page recommendation algorithms based on distributed learning automata and weighted association rules," *Expert Syst. Appl.*, vol. 37, no. 2, pp. 1316–1330, 2010.

[35] J. A. Torkestani and M. R. Meybodi, "Clustering the wireless ad hoc networks: A distributed learning automata approach," *Journal of Parallel and Distributed Computing*, vol. 70, no. 4, pp. 394 – 405, 2010.

[36] C. Unsal, P. Kachroo, and J. Bay, "Simulation study of learning automata games in automated highway systems," in *IEEE Conference on Intelligent Transportation System, ITSC '97*, nov 1997, pp. 936 –941.

[37] J. Torkestani and M. Meybodi, "Graph coloring problem based on learning automata," in *Information Management and Engineering, 2009. ICIME '09. International Conference on*, april 2009, pp. 718 –722.

[38] G. Papadimitriou, "Hierarchical discretized pursuit nonlinear learning automata with rapid convergence and high accuracy," *IEEE Transactions on Knowledge and Data Engineering*, vol. 6, no. 4, pp. 654 –659, aug 1994.

[39] N. Baba and Y. Mogami, "Learning behaviors of the hierarchical structure stochastic automata," in *Knowledge-Based Intelligent Information and Engineering Systems*, 2005, pp. 624–634.

[40] J. A. Torkestani and M. R. Meybodi, "Mobility-based multicast routing algorithm for wireless mobile ad-hoc networks: A learning automata approach," *Computer Communications*, vol. 33, no. 6, pp. 721 – 735, 2010.

[41] P. Vrancx, K. Verbeeck, and A. Nowé, "Networks of learning automata and limiting games," in *Adaptive Agents and Multi-Agents Systems*, 2007, pp. 224–238.

[42] M. Peeters, K. Verbeeck, and A. Nowé, "Solving multi-stage games with hierarchical learning automata that bootstrap," in *Adaptive Agents and Multi-Agent Systems III. Adaptation and Multi-Agent Learning*, ser. Lecture Notes in Computer Science. Springer, 2008, pp. 169–187.

[43] M. Howell, T. Gordon, and F. Brandao, "Genetic learning automata for function optimization," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 32, no. 6, pp. 804 – 815, dec 2002.

[44] H. Beigy and M. R. Meybodi, "Cellular learning automata with multiple learning automata in each cell and its applications," *Trans. Sys. Man Cyber. Part B*, vol. 40, no. 1, pp. 54–65, 2010.

[45] P. Sastry and M. Thathachar, "Learning automata algorithms for pattern classification," *Sadhana*, vol. 24, no. 4-5, pp. 261–292, 1999.

[46] M. A. L. Thathachar and M. T. Arvind, "Parallel algorithms for modules of learning automata," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 28, no. 1, pp. 24–33, 1998.

[47] G. Singla, D. J. Cook, and M. Schmitter-Edgecombe, "Tracking activities in complex settings using smart environment technologies," *International Journal of BioSciences, Psychiatry and Technology*, vol. 1, no. 1, 2009.

3.2. Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows

- Ros, M.; Cuéllar, M.; Delgado, M. and Vila, A. Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows. Information Sciences, 2012, In press.
  - Status: **In press**.
  - Impact Factor (JCR 2010): 2.836.
  - Subject Category: Computer Science, Artificial Intelligence. Ranking 10 / 128 (Q1).

ARTICLE IN PRESS

Information Sciences xxx (2011) xxx–xxx

Contents lists available at SciVerse ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows

M. Ros *, M.P. Cuéllar, M. Delgado, A. Vila

Department of Computer Science and Artificial Intelligence, University of Granada, E.T.S. Ingenierías Informática y de Telecomunicación, C/Pdta. Daniel Saucedo Aranda s.n., 18071 Granada, Spain

ARTICLE INFO

Article history:
Available online xxxx

Keywords:
Learning automata
Ambient assisted living
Online learning
Smart homes
Human behavior

ABSTRACT

Smart Homes are intelligent spaces that contain resources to collect information about user's activities and ease the assisted living. Abnormal behavior detection has been remarked as one of the most challenging application fields in this research area, due to its possibilities for assisting elders or people with special needs. These systems help to maintain people's independence, enhancing their personal comfort and safety and delaying the process of moving to a nursing home. In this paper, we describe a new approach for the behavior recognition problem based on Learning Automata and fuzzy temporal windows. Our proposal learns the normal behaviors, and uses that knowledge to recognise normal and abnormal human activities in real time. In addition, our proposal is able to adapt online to environmental variations, changes in human habits, and temporal information, defined as an interval of time when the behavior should be performed.

1. Introduction

In the past years, life expectancy has got longer, thanks, in most of cases, to advances in medicine and technology. This fact leads to the need for new resources in order to maintain this lifestyle. Technology raises as the most powerful tool to help in this issue. The advances in electronics and sensor technologies allow the development of new intelligent environments, where the combination of independence, continuous supervision, and assistance is taking place. One example of this combination is the project presented by Rantz et al. [34], at the University of Missouri. They develop an interdisciplinary research that involves disciplines, such as nursing, electrical and computer engineering, social work, or health informatics, among others. The project is developed in a real nursing home, where elderly people are able to live independently. Their research is focused on the study of the data acquired from a huge number of motion, pressure, and sound sensors, or IR-cameras. This information is used to extract conclusions about the user: heart pulse, breathing, fall detection, etc. Other outstanding study is *CARE* (*Context Awareness in Residence for elders*) [21], developed in the University of Amsterdam. *CARE* lets elders retain their independence by living longer at home. They use sensors such as switches or pressure mats, in order to monitor the inhabitants' behavior unobtrusively and provide benefits to both families and elders by means of network services. Focusing on the people with special needs, in [31,30], the authors proposed a project to help people suffering from mild dementia to navigate using contextual information and wireless and mobile device technologies, including location-based services. Related to location services, a flexible and dynamic access control model based on user-activity, Activity-Oriented Access Control (AOAC) is proposed in [22]. A user is allowed to perform an activity if he/she holds a number of satisfactory attributes (i.e. roles, assignments, etc.) under a certain environment constraint. Other examples in the literature

* Corresponding author.
E-mail addresses: marosiz@decsai.ugr.es (M. Ros), manupc@decsai.ugr.es (M.P. Cuéllar), mdelgado@ugr.es (M. Delgado), vila@decsai.ugr.es (A. Vila).

0020-0255/$ - see front matter © 2011 Elsevier Inc. All rights reserved.
doi:10.1016/j.ins.2011.10.005

are MavHome project [7], the iDorm [10], the CASAS project [35], and the Georgia Tech Aware Home [18]. Further projects regarding this research area are summarised in [11].

One of the most challenging issues regarding home assistance is the recognition of normal and abnormal human behaviors for elderly people, to enhance their personal comfort and safety, and delaying the process of moving to a nursing home. The literature offers a great variety of approaches in this way [4]. One interesting work is the one presented in [32]. It defines a new paradigm for Activities of Daily Living (ADL) [24,32] inference, using radio-frequency-identification (RFID) technology to identify objects used by people, and a probabilistic inference engine for the recognition of complex activities. They have developed the system *Proactive Activity Toolkit* (*PROACT*) to represent activities as a probabilistic sequence of used objects. It uses the RFID signals to extract the user context and then, mines probabilistic models of activity use from plain english descriptions of activities.

Recent projects generalise the problem of behavior recognition to usual activities of daily living, such as morning routine, having breakfast, or leaving home. To achieve this, the trend is the use of probabilistic techniques to supervise the current activities. The reason for resorting to stochastic methods is that they are able to model different scenarios and possible ways to finish the human tasks correctly. Naive Bayes classifiers have been used in some projects [17,39] obtaining good accuracy results for abnormal behavior detection, even when the system receives large amounts of data. Other similar works that model human activities with Hidden Markov Models (HMMs) are [5,46,27]. Naeem and Bigham present in [28] an approach based on Multiple Behavioral Hidden Markov Models (MBHMM). They propose the creation of multiple hidden Markov models for each variation of an action, so that the system is able to determine which tasks are currently active even if the user has not yet completed the activity. In [23], Lee et al. propose an algorithm that incorporates a modified fuzzy competitive learning structure with a Bayesian decision rule that is capable of ignoring unintended behavior of the user. An extensive study about the benefits and drawbacks of different probabilistic and statistical techniques for human activities recognition is presented in [19].

Lately, some projects that involve different techniques for the problem of recognising normal and abnormal behaviors have arisen. In [2], the authors propose an ambient intelligence approach, and develop a fuzzy computing system to learn and model human behaviors. Their work joins the advantages of a multiagent-based framework with fuzzy control techniques to improve the recognition process and continuously enrich the knowledge about the user and his/her environment. On the other hand, Rashidi et al. in [36] introduce an unsupervised method to discover activities in a smart environment, using data mining techniques to find activity patterns and a clustering process to group these patterns into activity definitions, and HMMs to represent the activities and their variations for their later detection and recognition. Other authors have included temporal information in order to extract correct patterns for each behavior. For instance, in [16,45], the authors propose a learning machine to identify temporal relations among daily activities in a smart home. These relations are used for prediction, decision making, and anomaly detection of ADL. Further related projects are summarised in [26,12,13,43].

Despite the efforts carried out to recognise human activities in smart home environments, the need for feedback information is required for the system adaptation to environmental or human habit changes. Stochastic systems are usually used to acquire reasonable behavior in environments characterized by uncertainty. In [20], Kitakoshi et al. propose an on-line reinforcement learning system that adapts to environmental changes using a mixture of Bayesian networks. They try to imitate how people adapt to changes: reusing experiences with suitable modifications to solve current problems. Every BN in the mixture represents the experiences in an specific environment. When environmental changes are observed, the system modifies the mixture to adapt to the changed environment.

Specifically, in Ambient Assisted Living, some of the most important projects consider this issue, such as the CASAS project [35,36]. It includes an adaptive activity mining component to find changes in activity patterns using a hierarchical behavior model with probabilistic methods and a feedback-based learning component in charge of finding relevant changes in the activities. Among the projects based on agent systems, in iDorm [10,15,14], the authors developed AOFIS, a life-long fuzzy learning and adaptation technique for embedded agents in ubiquitous computing environments. This technique is an unsupervised data-driven one-pass approach for extracting fuzzy rules and membership functions from user data modelling for behavior detection and online adaptation. Other examples are [1], where Acampora et al. improve the system proposed in [2] by means of the adaptation of both the framework and the fuzzy services. They use feature mining to detect changes in the human activities. Finally, in [39], the authors propose a system for recognising activities that are capable of adjusting the internal model in real-time as new examples of activities become available.

In previous works, we developed a method to mine frequent human activities using inductive learning, and provided a behavior model as a tree. The branches represent a way in which a behavior could be performed [9]. Lately, we studied the benefits of using a fuzzy temporal window to disclose the most relevant actions of an activity [37], in contrast to the classic crisp temporal sliding window. However, this approach is static and the need of adaptation to environment and habit changes require a new learning after sufficient user activity monitoring. The behavior representation was improved in [8], where we proposed to separate learning methods and behavior models for a better problem statement. This approach is able to learn temporal relationship constraints between actions in a behavior, and has the advantage of being a graphical model to ease the user monitoring. However, it lacks of time processing and detection of several behaviors simultaneously. In this article, we propose a system to recognise human behavior from current user activity. To be more concise, we focus on the learning of relevant actions of the activity, their order relations, and temporal data of behaviors. The proposal overcomes the limitations of HMMs regarding first order assumption [5,46,27], since behavior actions are structured in execution levels. We include temporal information in the developed system as a component of a behavior execution, and our approach is also

Please cite this article in press as: M. Ros et al., Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows, Inform. Sci. (2011), doi:10.1016/j.ins.2011.10.005

capable of online self-adaptation when a behavior or environmental change occurs, therefore adjusting the behavior knowledge to a specific user. For each human behavior, we distinguish between two types of constraints: temporal and order relationship constraints. The temporal constraints relate to the period of time when the behavior must be performed, whereas the order relationship constraint controls which actions must be executed before the others, i.e. the temporal order relationships between actions in a behavior. The proposed system is able to learn these constraints and it generates a representation of a behavior as a module. Key aspects of our approach are: (a) The separation of the behavior model from the behavior learning procedure, in order to avoid shrinking problems produced by the learning algorithms, as happens with HMMs; (b) the modularisation of the approach, which allows the migration of a learnt behavior to other contexts or user environments; (c) the online learning and adaptation capability, by means of learning automata and fuzzy temporal windows, to learn the changes produced either in the user habits or the environment. The proposal has been designed as an abstract software layer, so that it works independently on the underlying sensor network. Several technologies could be used to adjust the system to a specific scenario by means of matching user actions and sensor events that trigger such actions, such as RFID [32,28,25] of processed video streams [6,44].

The paper is organised as follows: In Section 2, we introduce basic theoretical aspects of Learning Automata which will be useful for our later developments. Next, Section 3 summarises the *behavior model* concept. The description of the model is presented in Section 4, where the relevant aspects of the system are remarked: the behavior time and the action order relationship constraints models. After that, in Section 5, we address the problems of online behavior recognition and their adaptation to environmental and human habit changes. Section 6 presents the outlines of simulating the system with different data sets and includes an in-depth analysis with the obtained results. Finally, the conclusions and future works are discussed in Section 7.

2. Learning Automata. Theoretical background

A learning automaton (LA) [33] is a decision-making stochastic machine defined by tuple $\langle \alpha, Q, R, T \rangle$, where α is a set of available actions for the automaton, Q is the internal state, R is a set of reinforcement values to the automaton, and T is a reinforcement learning scheme. The operation of a learning automaton (Fig. 1) is described as follows: at each time instant k, the automaton selects an action $a(k) \in \alpha$ according to the automaton internal state $Q(k)$, and returns $a(k)$ to an unknown random environment. The environment evaluates the action chosen by the automaton with a fitness function $f: D \times \alpha \to R$ and provides a reward or penalty reinforcement value $\beta(k) = f(D, a(k))$, where D is the expectation to obtain a reward reinforcement value. Finally, the internal state of the automaton is updated with $Q(k+1) = T(Q(k), a(k), \beta(k))$. This process is repeated until a predefined desired condition is fulfilled, as for example the convergence to the learnt optimal action. There is a wide variety of LA depending on the design of α, R and the environment. For the sake of brevity, we suggest reference [41] for a complete description of the classic LA schemes. Next, we describe the models that are useful for this work.

A Finite Action-Set Learning Automaton (FALA) [41] is a LA model with a finite action set, $\alpha = \{\alpha_1, \alpha_2, \ldots, \alpha_r\}$, where r stands for the number of available actions. The state $Q(k)$ usually contains a probability distribution over the action-set of automaton $Q(k) = P(k) = [p_1(k) p_2(k) \ldots p_r(k)]$, although other more complex state structures may be designed to improve the learning capabilities of the automaton, as happens with the estimator learning algorithms [3]. On the other hand, Continuous Action-Set Learning Automata (CALA) [41] can be used to handle continuous action spaces. The action probability distribution in a CALA, at the learning iteration k, is defined by a Normal probability distribution $N(\mu(k), \sigma(k))$. The update of its action probability distribution is executed by updating $\mu(k)$ and $\sigma(k)$ [33,41]. Finally, another LA model of interest to our work is the *variable action-set learning automaton* (VALA) [40,42,29]. Here, the action-set is finite as FALA, but the availability of actions to the automaton depends on external variables such as time. The state of the VALA may be easily modelled with a matrix, where each column is the action probability distribution corresponding to each time, $Q(k) = [P(k,1)^t, P(k,2)^t, \ldots, P(k,t)^t]$.

In this work, we use a team of FALA to online learn the temporal action relationship constraints of a human behavior, and a CALA team to learn the parameters of a fuzzy temporal window in which the behavior occurs. For the behavior recognition and online adaptation, the FALA team is implemented into a VALA team to achieve the dynamisation of the behavior model.

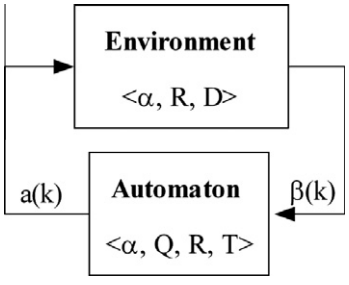


Fig. 1. Operation of a learning automaton.

Please cite this article in press as: M. Ros et al., Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows, Inform. Sci. (2011), doi:10.1016/j.ins.2011.10.005

This helps to model situations where a behavior has several ways to finish correctly and its actions can be executed in any order. The next sections explain these issues in depth.

3. Behavior model

The objective of the article is to present a system that is able to learn human behaviors online and detect when the user is carrying them out, either correctly or incorrectly. Therefore, the first step consists of defining what behaviors are and how the system deals with them. Example 1 introduces the problem to solve:

Example 1. Let us suppose that every night, when the user goes to bed at approximately 9.30 p.m., he/she opens the bedroom door, puts his/her pyjamas on, sits on his/her bed, takes his/her slippers off, takes the pills and turns off the light. Then, all these actions could be embraced as a unique routine: to go to bed. Let us suppose that a single day the user does not take his/her pills. This fact is considered as an abnormal behavior for this routine, and it should be controlled. However, if that oversight becomes more usual, the system should stop recognising it as abnormal, and accept it as normal.

Example 1 outlines the concepts of behavior and human action: the set of actions performed to get the routine *to go to bed* is a behavior, and the behavior is associated with a time in which it occurs. The relevant concepts are formally introduced in the next subsections.

3.1. Behavior action sequence

We assume that the tasks carried out by the user may be identified by means of sensor events. Our starting point are actions that have been defined over a set of sensor events to be independent of the sensor network design. The action and behavior concepts are as follows:

Definition 1 (*Human action*). A human action h_i is a triple $h_i = \langle l_i, s_i, t_i \rangle$, where l_i is a label to identify the action, s_i is a sensor event and t_i is a time stamp in which the action occurs. In brief, a human action labelled l_i is an activity (fact) that happens over a specific object in a time instant t_i, and it may be uniquely determined by means of a sensor event s_i.

Definition 2 (*Behavior action sequence*). A human behavior action sequence B^i is a tuple $\langle H, R^i, C^i \rangle$, where $H = \{h_1 h_2, \ldots, h_n\}$ is a set of available human tasks, $R^i = \left\{ r^i_1, r^i_2, \ldots, r^i_{|R^i|} \right\} \subseteq H$ is the subset of available human tasks that are required to complete the behavior, and C^i is a set of temporal constraints between the tasks in R^i, defined over a partial order relationship operator $\prec$. The operator $\prec$ defines the temporal dependency requirements over the actions of a behavior, and its formal definition is described in Eq. (1). In brief, an action r^i_a precedes r^i_b in time in a behavior B^i, and is written $r^i_a \prec r^i_b$, if the time stamp t_a of r^i_a is lower than the time stamp t_b of r^i_b, for all the possible instances of the behavior.

$$r^i_a \prec r^i_b \leftrightarrow t_a < t_b \tag{1}$$

A behavior action sequence is represented by a set of tasks that must fulfil (or not) temporal order relationship constraints. In addition, the behavior may be associated with a time interval in which it usually occurs. Because of these restrictions, we use the representation presented in [8] for the behavior action sequence. In this representation, the tasks are organized in execution levels connected in cascade. These cascade levels are implemented with AND gates, and the levels simulate the order relationships between the human tasks. By default, all input lines to AND gates are set to false, and their values change to true if the human actions that they represent have been performed. On the other hand, although all the tasks in set H may be taken into consideration for a specific behavior, only those actions that are included in set R are really outstanding for it. To handle these situations, the authors introduce an additional discard-level in which those tasks that are not relevant for the behavior are assigned.

A behavior does not have an only one way to be performed, since some tasks may be executed with different order constraints. Taking the level cascade model representation into consideration, this means that a task might be an input of different AND gates in the cascade, depending on the user execution context. For instance, in the Example 1, the task *take his/her pills* could be performed either in level 1 or in level 2 in the model that represents a normal *to go to bed* behavior. To deal with this disparity, in [8], the authors proposed the use of a probability distribution for each action in the set H, so that an action h_i may be executed in each of the r possible levels of the cascade with probability $P^i = [p^i_1, p^i_2, \ldots, p^i_r]^t$. This probability distribution represents that each action can be executed within a cascade level with a concrete probability value. Therefore, the representation as a sequence of ANDs in cascade is the most likely representation of a behavior, assigning each action to its most probable level in the cascade. Formally, a human task h_i is located at the cascade level l if, and only if, $l = min\left\{ j : p^i_j(k) = max_j \left\{ p^i_j(k) \right\} \right\}$, i.e. l is the first level with a maximum probability value in the action-set of the automaton. If the task is not executed at such level, the model is updated online to the next most probable execution level according to the behavior model probability distribution. This situation is illustrated with the Example 1. A possible probability distribution for the example would be described in Table 1, and Fig. 2 shows the most likely behavior execution:

Please cite this article in press as: M. Ros et al., Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows, Inform. Sci. (2011), doi:10.1016/j.ins.2011.10.005

Table 1
Example of probability distribution for the behavior of Fig. 2.

	$Level_1$	$Level_2$	$Level_3$	$Level_4$	$Level_{discard}$
Sit on bed	0.78	0.22	0	0	0
Put pyjamas on	0.76	0	0	0	0.24
Take slippers off	0.02	0.75	0.23	0	0
Take pills	0.34	0.46	0.2	0	0
Turn light off	0	0	0.65	0.45	0
Remaining tasks in H	0	0	0	0	1

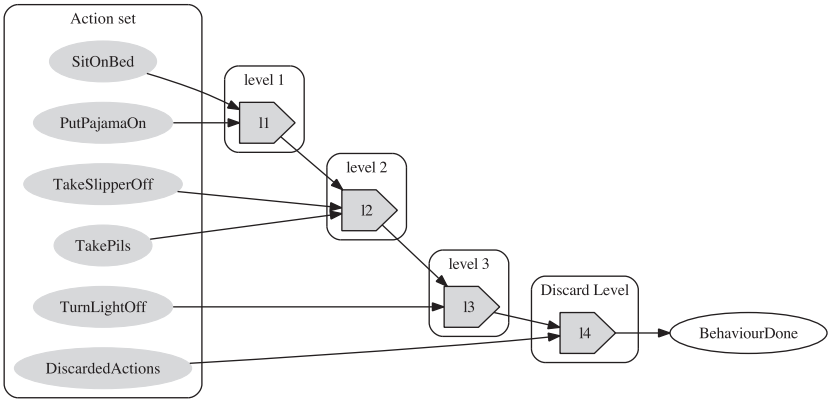


Fig. 2. Go to bed Behavior model. Example 1.

Among the advantages of this proposal, the separation of the described behavior model from the probability distribution learning procedure is presented as one of the most significant ones. This separation allows to achieve a better system modularisation and to use a higher number of learning methods for the same behavior model.

The system uses a set of FALA to learn each *behavior action sequence* from raw sequences of user actions. Every set is composed for $|H| + 1$ number of FALAs, where $|H|$ is the cardinal of the set H. Each learning automaton in the team is uniquely matched with a human task, and its action-set contains the level of the AND cascade in the behavior action sequence in which the task can be located. The goal of each automaton is to learn the optimal level in the sequence of gates in at which its corresponding task should be located. If the automaton action is associated with level $|H| + 1$, then it means that its matched task has been discarded for the behavior, and a selection of a level between 1 and H means that the matched task is located at such level. For a wider explanation about offline learning of behavior action sequences, we suggest to read [8].

3.2. Behavior time

The *behavior time* is in charge of checking if the time stamp of a human task corresponds to the time interval in which a behavior must occur. In our approach, this module stores a *Fuzzy Temporal Window* (*FTW*) for each behavior in the system [37]:

Definition 3 (*Temporal Window, W*). Let I be an interval from the temporal line τ, ODB the Observation Data Base of user tasks carried out during the behavior time, $t \in ODB$ a tuple from ODB, and $i_j = (h_j, l_j, d_j)$ a human task; then, a Temporal Window W for a specific behavior that occurs in the interval I of τ, is defined as a subset of I where

$$\forall i_j \in W(t) \text{ then } i_j \in t \text{ and } l_j \in I$$

Definition 4 (*Fuzzy Temporal Window, FTW*). Let I be an interval from the temporal line τ, ODB the Observation Data Base and $t \in ODB$ a tuple from ODB. Let i_j be an action $i_j = (h_j, l_j, d_j)$ and a Temporal Window W for a specific behavior. Let fs be a fuzzy set over τ, then it is defined a *fuzzy temporal window FTW* as a Temporal Window where

$$\forall i_j \in W(t) \mu_{FW}(i_j) = \mu_{fs}(l_j)$$

In this work, an FTW is represented as a Gaussian function whose mean value is the middle point of the interval associated with the behavior, and its standard deviation defines the amplitude of the execution time. The FTW is also used for obtaining a *reliability degree* of execution of each behavior. A membership degree can be calculated for each human task car-

Please cite this article in press as: M. Ros et al., Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows, Inform. Sci. (2011), doi:10.1016/j.ins.2011.10.005

ried out in the time interval of an FTW. It provides every task with the reliability of belonging to a specific behavior. To estimate that reliability degree of the overall behavior, we aggregate the membership degrees of all their executed tasks as just one value. We use the OWA operator as average of these values for simplicity, although any aggregation operator could be used. Example 2 summarises the explained process.

Example 2. The user has performed the sequence of tasks {Keys, 8:27 a.m.}, {Cellular, 8:30 a.m.}, {FrontDoor, 8:37 a.m.} for behavior *Leave home at 8:30 a.m.*. The FTW is defined with mean equals to 8:30 a.m. and standard deviation of 20 min. Then, the membership values of the actions to the FTW are 0.956, 1 and 0.783 respectively, and the average aggregation is 0.913, what means that this sequence of actions is considered as a normal behavior with a reliability degree of 0.9129.

It is required to develop a method that is able to learn the *FTW* associated to a specific behavior, defined by two real numbers μ_{FTW} and σ_{FTW}. In this work, we include two CALA [41] into a team to learn these values for each behavior. The main advantage of this selection is the capability of online self-adaptation to changes in the behavior temporal constraints by means of the feedback loop of the LA.

4. System description

Our proposal distinguishes two parts for each behavior (see Fig. 3): action processing and temporal processing. Each behavior is considered as an autonomous module, which makes the processing of several simultaneous behaviors easier. In addition, the control and interaction of all behaviors is managed by an additional module called *behavior manager*, that unifies the solutions and controls the simultaneousness of the behaviors. The *behavior action sequence* and *behavior time* are adapted as regards user routine and environment changes. These changes could cause modifications in the execution levels or in the actions involved in the behavior. The decision about the update of the internal state of each behavior model is taken by the *behavior manager* during the recognising process.

The goals of the *behavior manager* include to schedule and process the behaviors obtained in the recognition process; to control which actions have been performed, belonging or not to a behavior; to provide the necessary feedback to adapt the system for environmental or behavior changes; and to send warning alarms for the user when required. This module retrieves the action performed by the user and the results obtained during the recognition process. In short, it returns the identification of the detected behaviors, the reliability degree of the detected behavior, and if the behavior may be considered as normal or abnormal. To be more precise, the system is able to detect three types of behaviors: normal behaviors, uncompleted behaviors, and violation order constraints behaviors. These last two types are classified as abnormal behaviors:

- *Normal behaviors* are those action sequences which fit the temporal window associated with the behavior and whose actions have been executed according to the behavior action sequence model.

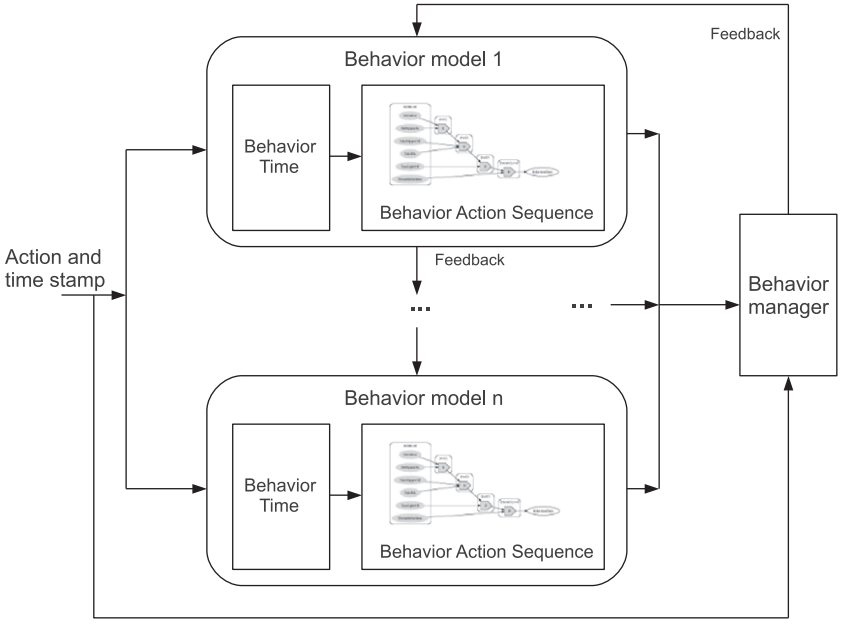


Fig. 3. Behavior models in parallel.

Please cite this article in press as: M. Ros et al., Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows, Inform. Sci. (2011), doi:10.1016/j.ins.2011.10.005

- *Uncompleted behaviors* are action sequences that have not finished when the FTW expires. This means that there are some relevant actions that have been missed by the user during the behavior execution. If a human activity is classified as an uncompleted behavior, it may be due to either deliberately action missing or a human error, and then, an alarm should be sent to the user.
- *Violation of task relationship constraints*: The patterns that are classified in this class fulfill the behavior temporal constraints, but they contain one or more violations in the *behavior action sequence*. If a pattern is matched with this class, then, an abnormal behavior has been detected or a human habit change might be taking place. An alarm must be sent to the user to warn about this abnormal situation.

The behavior manager is in charge of producing the signals to send the alarm by an existing higher level application when an abnormal behavior is detected, and of generating the feedback for behavior model online adaptation if a change in user routine is taking place. Therefore, it saves a record about the executed actions and the level in which the actions have been carried out in the behavior models.

5. Online behavior recognition and adaptation

During the recognition of a behavior, the temporal constraints of executed actions are checked using the *behavior time* module (see Fig. 3), which provides a membership degree of every performed action. Since a behavior may be performed in different ways, the *behavior action sequence* must be adapted online regarding the user past activity. *Variable Action-Set Learning Automata* (VALA) offers the best option to change the action relationship constraint model online. This type of learning automata changes its action probability distribution depending on an external parameters such as *time*. The internal state of an automata in the VALA team used in our system is represented with a matrix of size $n \times (|H|+1), Q^i_{VALA}(k) = (P^i(k,1), P^i(k,2), \ldots, P^i(k,|H|+1))^t$, where $P^i(k,1) = Q^i_{FALA}(k)$, i.e. the action probability distribution learnt with the FALA team. The remaining probability distributions $P^i(k,l) = \left[p^i_1(k,l), p^i_2(k,l), \ldots, p^i_{|H|+1}(k,l)\right]^t, 1 < l \leqslant |H|+1$ are calculated using Eq. (2). A value $p^i_j(k,l)$ stands for the *a priori* probability of human task h_i to be executed at the jth level in the behavior model when a human action assigned with level l has been executed.

$$p^i_j(k,l) = \begin{cases} 0; & j < l \\ \sum_{m=1}^{l} p^i_m(k,1); & j = l \\ p^i_j(k,1); & l < j < |H|+1 \end{cases} \tag{2}$$

During the recognition process, both *behavior time* and *behavior action sequence* update their internal state depending on the feedback received from the *behavior manager*. This adaptation process allows to improve the knowledge about the user, adjusting the *Behavior model* to the user features.

The *behavior action sequence* is updated depending on the level l in which each action was carried out, and considering those actions not executed and discarded. The adaptation process consists on updating the probabilities of the automata following a discrete rule, as shown in Eq. (3). The parameter Δ determines the learning rate of the user habits. High values of Δ mean a faster adaptation to the user habits, and therefore, a low system–human interaction is obtained.

$$p^i_j(k+1,1) = \begin{cases} max\left(p^i_j(k,1) - \Delta, 0\right); & j \neq l \\ 1 - \sum_{z=1, i\neq l}^{S+1} p^i_z(k+1,1); & j = l \end{cases} \tag{3}$$

As far as *behavior time* is concerned, the adaptation process consists of updating the parameters that define a CALA: $\mu_{CALA}(k)$ and $\sigma_{CALA}(k)$. The traditional CALA update rule may be found in [41]. However, this rule performed too slowly in our experiments and a large number of update iterations were required for a good system performance. This problem is well-known as the traditional limitation of the learning automata. To overcome this limitation, we changed the update rule and provided the best approach which we found for this work in Eqs. (4) and (5).

$$\mu(k+1) = \mu(k) + \lambda_{CALA}(t_{mean} - \mu(k)); \quad if \ \nexists j : \mu_{FTW}(t_j) < \mu_{min} \tag{4}$$

$$\sigma(k+1) = \begin{cases} \sigma(k) - \lambda_{CALA}\sigma(k); & if \ \nexists j : \mu_{FTW}(t_j) < \mu_{min} \\ \sigma(k) + \lambda_{CALA}\sigma(k); & otherwise \end{cases} \tag{5}$$

where $b_j = (a_j, t_j)$ are the relevant human tasks and the time instant in which they have been carried out respectively, μ_{min} is the minimum membership value allowed for an action to be considered inside the fuzzy temporal window of the behavior, t_{mean} is the average time of the executed actions, and λ_{CALA} is the learning rate for the automaton.

According to Eqs. (4) and (5), the parameter $\mu(k)$ is updated only in the case in which all the actions are included in the fuzzy temporal window. In this case, the value $\sigma(k)$ is also reduced to find the fuzzy temporal window of minimum size that

Please cite this article in press as: M. Ros et al., Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows, Inform. Sci. (2011), doi:10.1016/j.ins.2011.10.005

contains all the behavior actions. On the other hand, if an action was detected outside the window, then the value $\sigma(k)$ increases in order to cover the missing actions.

5.1. Algorithm for behavior recognition and system adaptation

When an action is received in the system, the time constraints of each behavior are checked in order to discard those behaviors which are out of time for the sake of efficiency. Once we know those behaviors whose instants of time match, the action is placed and compared in the *behavior action sequence*. The output of each *Behavior Model* is sent to the *behavior manager*, who analyses those outputs, detecting if the behavior has been finished and its type; and decides the next step: keeping on with the recognising process, warning the user, updating the behavior models, or reinforcing the models.

A scheme of the algorithm used for the recognition process is presented next. Notice that the main body of the algorithm (from step 2 to 7) is executed every predefined period of clock time. The algorithm assumes that the learning process has been performed previously and starts from a set of learnt *behavior models*.

1. **Initialisation.** In this step, the list of current active behaviors and the behavior action sequence buffer are initialised. A behavior is active if the evaluation of the current task is in its *fuzzy temporal window* (the membership degree of the task over a threshold). The list of current active behaviors contains those behaviors that are active at each time instant. A behavior is removed from the list when it is recognised, and a behavior action sequence buffer stores the tasks that have been carried out during the behavior time.
2. **Finding the active behaviors.** First, the current time t and the actions that have been executed at such time instant are fetched. Then, the membership value of time t to the *fuzzy temporal window* of each behavior is computed. Those behaviors that are active will be added to the list of current active behaviors within the *behavior manager*.
3. **Checking if active behaviors are uncompleted.** In this step, the algorithm checks if all the behaviors stored in the list of active behaviors remain active. It is possible that those behaviors that were included in the list in previous iterations are not active at the current moment. This fact implies that the fuzzy temporal constraints of the behavior have been violated and then, the behavior is tagged as unfinished. When these behaviors have been identified, the algorithm has to test if the behavior was effectively incomplete or the user did it deliberately by sending an alarm to the user. If this confirms the change in the behavior, the action sequence model of the behavior is updated to remove the missing actions.
4. **Current action processing.** The next step is the processing of the current action into its possible behavior models. First of all, the algorithm checks which behaviors contain the action executed by the human at the current time in the AND cascade model. If we find one or more behaviors that fulfill this condition, the action is then included in their *action sequence model* buffer. On the other hand, if the action is not relevant for any behavior, it is included in all the active behaviors, since it may mean a new user habit change that includes the new action.
5. **Updating the reliability degree.** The actions included in step 4 have a temporal membership value associated with its time of occurrence. These values are used to update the reliability degree of the behaviors.
6. **Checking violations of action order relationship constraint behaviors.** Once the algorithm has processed the temporal information, the next step is to know if any action order relationship constraints have been violated. If a violation is found, then the user's opinion is required in order to find out whether the actions have been done on purpose or if it was a human mistake.
7. **Checking finished behaviors.** Finally, if no violations were found, a normal behavior recognition is detected. The action sequence model and the behavior time are updated to reinforcement the correct human habits.
8. **Go to step 2.**

6. Experiments

In this section, we present the results obtained after a testing process. Firstly, in Section 6.1 we show the experimental settings. Then, Sections 6.2 and 6.3 describe the experiments for testing the system recognition and adaptation capabilities, respectively, and a final Section 6.4 is included to discuss further results and obtain conclusions about them.

6.1. Experimental settings

As previously indicated, our system input is user activity as sequence of human tasks $h_i = \langle l_i, s_i, t_i \rangle$. To develop a sequence of correct experiments, we define the experimental contextualization as follows: The environment is a simulated home equipped with 26 sensors able to distinguish between different human tasks. We focus on seven behaviors *To Get Up* (B_1), *To Have a Shower* (B_2), *To Have Breakfast* (B_3), *To Leave Home* (B_4), *To Cook* (B_5), *To Wash the Dishes* (B_6), and *To Go To Sleep* (B_7). These are defined by a set of temporal and task order constraints (see Table 2) that let us define sequence of actions to represent those behaviors confidently. In Table 2, column 1 describes the behaviors and the number of actions involved, column 2 stands for the action relationship constraints, and column 3 shows the behavior time. We remark that the number of actions in a behavior (column 1) may be higher than those appearing in column 2, since some actions may have no order constraints. We did not included these actions in the text for space limitations, since its inclusion is not key to assess the proposal in the next subsections. On the other hand, the reason for choosing simulation instead of real user monitoring is

Please cite this article in press as: M. Ros et al., Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows, Inform. Sci. (2011), doi:10.1016/j.ins.2011.10.005

Table 2
Action relationship constraints and time interval for the behaviors.

Behavior	Constraints	
	Order relationship	Temporal
B_1 (4 actions)	*TurnBedLightOn* $\prec$ *TakeCellular* *TakeCellular* $\prec$ *SitOnBed* {*TurnBedLightOn*, *TakeCellular*, *SitOnBed*} $\prec$ *PutSlipperOn*	[7:25–7:45]
B_2 (3 actions)	*UseBathTap* $\prec$ *TakeTowel* {*UseBathTap*, *TakeTowel*} $\prec$ *UseBath*	[7:45–7:55]
B_3 (6 actions)	*TakeMilk* $\prec$ *UseSaucepan* *UseSaucepan* $\prec$ *UseGas*	[7:55–8:05]
B_4 (5 actions)	*OpenFrontDoor* $\prec$ *TakeLift* {*TakeCellular*, *TakeKeys*, *TakeBag*} $\prec$ *OpenFrontDoor*	[8:20–8:40]
B_5 (5 actions)	{*TakeCereals*, *UseSaucepan*, *TakeFryingPan*} $\prec$ *UseGas*	[13:40–14:20]
B_6 (3 actions)	*TakeSoap* $\prec$ *CleanDishes* {*CleanDishes*, *TakeSoap*} $\prec$ *OpenKitchenTap*	[14:50–15:10]
B_7 (4 actions)	*PutSlipperOff* $\prec$ *TakePills* *TakePills* $\prec$ *SitOnBed* {*SitOnBed*, *TakePills*} $\prec$ *TurnBedLightOff*	[22:20–22:40]

that we found no differences between the synthetic user and to give guidelines to a real one about how to perform the behaviors, so that the system could be validated. In addition, the use of simulation helps us to force situations that could be dangerous for users without putting them in risk, to test the constraints violation detection rate.

We generated three databases with user action sequences for the experiments. Each database is a timetable where every row means a day (24 h) and every column is a second. A cell in the timetable contains the actions performed by the user at that time. The content and goals of each database is described as follows:

1. Database for learning process. It contains information about the execution of normal instances of behaviors B_1–B_7 during 365 days. For its generation, each action of a behavior was assigned to a random cell of the timetable during the behavior time. After that, actions that did not fulfill the relationship constraints were exchanged until the database contained no constraints violations. Finally, we added noise actions during each behavior time to simulate other activities that the user could perform during its execution. These *rubbish* tasks were 30% of each behavior.
2. Database to test recognition. This database contains 365 normal, uncomplete and constraint violations behaviors, and it was generated under the same conditions as the previous one. It is used to validate the approach for recognition of normal and abnormal activities in Section 6.2. It contains 50% of normal behaviors, and 25% of uncompleted and constraint violation samples, respectively.
3. Database to test adaptation. The adaptation is carried out when the user changes his/her routine. To simulate this situation, we changed deliberately the constraints and time information of the behaviors to those shown in Table 3. This database contains 365 normal instances of the modified behaviors B_1^*—B_7^*, and it is used in Section 6.3.

Before starting the recognition and adaptation experimentation, a learning process is required to establish a common start point. In this paper, we used the learning algorithm presented in [8] to mine relevant tasks of behaviors and their order relationship constraints. To finalize, Table 4 shows the values of the parameters studied for the online recognition and adaptation. Concretely, we will study three parameters: the minimum membership value for the fuzzy temporal window (μ_{min}), and the learning rates for the FALA (λ_{FALA}) and CALA (λ_{CALA}) teams. The value μ_{min} is the minimum value required for an action being considered into the fuzzy temporal windows, and those actions whose membership degree would be lower than this parameter would be discarded in the recognition/adaptation process. On the other hand, the learning rates are used to check the feasibility of the online adaptation in real time and the adaptation speed.

6.2. Analysis of online recognition

In this section, we analyse the capability of our model to recognise normal and abnormal behaviors. We run the recognition process using the test database for this purpose as input, with the parameter values defined in the *online recognition* column of Table 4. Fig. 4 summarises the results obtained for a recognition process with parameters $\mu_{min} = 0.001$ and $\lambda_{FALA} = 0.001$. These parameters provided the best score in our experiments, and an in-depth study of this selection is described in the next section. In general terms, our method obtains accurate results in the experimentation: All the behavior samples were correctly matched to their corresponding behavior, 96.05% of them were correctly detected and the remaining 3.95% were misclassified. The normal behavior detection rate is 92.50%, and the remaining 7.50% were detected as uncompleted of

Please cite this article in press as: M. Ros et al., Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows, Inform. Sci. (2011), doi:10.1016/j.ins.2011.10.005

Table 3
Action relationship and time interval constraints for the adapted behaviors.

Behavior	Constraints	
	Order relationship	Time
B_1^* (4 actions)	*TakeCellular* $\prec$ *TurnBedLightOn* *SitOnBed* $\prec$ *TakeCellular* {*TurnBedLightOn*, *TakeCellular*, *SitOnBed*} $\prec$ *PutSlippersOn*	[7:30–7:50]
B_2^* (3 actions)	*TakeTowel* $\prec$ *UseBathTap* {*UseBathTap*, *TakeTowel*} $\prec$ *UseBath*	[7:40–7:50]
B_3^* (6 actions)	*UseSaucepan* $\prec$ *TakeMilk* *UseSaucepan* $\prec$ *UseGas*	[8:00–8:20]
B_4^* (5 actions)	*TakeKeys* $\prec$ *TakeLift* *TakeBag* $\prec$ *TakeKeys* *OpenFrontDoor* $\prec$ *TakeLift* {*TakeCellular*, *TakeKeys*, *TakeBag*} $\prec$ *OpenFrontDoor*	[8:30–8:35]
B_5^* (5 actions)	*TakeCereals* $\prec$ *UseSaucepan* {*UseSaucepan*, *UseFridge*, *TakeFryingPan*} $\prec$ *UseGas*	[14:15–14:30]
B_6^* (3 actions)	*CleanDishes* $\prec$ *TakeSoap* {*CleanDishes*, *TakeSoap*} $\prec$ *OpenKitchenTap*	[14:50–15:00]
B_7^* (4 actions)	*TakePills* $\prec$ *TakeSlippersOff* *TakePills* $\prec$ *TurnBedLightOff* {*TakePills*, *TurnBedLightOff*} $\prec$ *SitOnBed*	[22:35–22:45]

Table 4
Values for the used parameters.

Parameter	Online recognition	Online adaptation
Temporal fuzzy window minimum membership (μ_{min})	{0.001, 0.01, 0.1, 0.2, 0.3}	{0.001, 0.01, 0.1, 0.2, 0.3}
FALA learning rate (λ_{FALA})	{0.001, 0.01, 0.1, 0.2, 0.9}	{0.2, 0.3, 0.5, 0.75, 0.9}
CALA learning rate (λ_{CALA})	0.001	{0.001, 0.01, 0.1, 0.5, 0.9}

constraint violation samples. We notice that none of uncompleted or constraint violation samples were classified as normal behaviors, so that unusual or dangerous situations for the user are never recognized as true behaviors, which is very important for the goals of our system. In addition, 100% of uncompleted samples and 99.36% of constraint violations were correctly classified. The false positive detection rate of uncompleted samples is 2.47%, which correspond to normal and constraint violation behaviors that were misclassified. However, the 11.99% of false positives regarding the detection of constraint violations means that some of the normal behavior are erroneously detected, because its constraints are not included in the knowledge learnt by the behavior models. An in-depth analysis of this situation is presented in Table 5. In this table, we may observe that most of learned models were able to recognize the temporal and order relationship constraints of the studied behaviors. However, some normal behavior samples of B_2, B_3 and B_6 were not correctly classified. We concluded that these detection errors are due to overtraining of the system: The models learned correctly the relationship constraints of each problem, but they also inferred further non-existent constraints including the remaining actions of the behavior. In the case of behavior B_6, actions that do not belong to the activity were also included in the behavior model because the simulated user performed them in high number of samples. This conclusion is supported by the false positive rate of constraint violations in Table 5, and the false positive rate of uncompleted behaviors in B_6. Despite of the overtraining, we solve this situation with the online adaptation process.

To validate the approach, we compare the results obtained with Hidden Markov Models (HMMs) since this technique is a baseline method widely used in AAL recognition. To achieve this, the design and learning of HMMs were done following the guidelines proposed in [38]. An HMM was trained for each behavior using the training data. The HMMs have as many states as available human actions. The prior and transition probabilities were learned from the training data, and the emission probability matrix was set to the identity matrix because our main assumption is that each human task is uniquely determined by a sensor event. The training and test data were the sequences of tasks carried out by the user, ordered in time. A time window was used to shrink the input data to those in the specific time of each behavior.

Once the models learned the training data, the recognition database was used as test so that we can compare results with our proposal. We run two types of experiments: Firstly, we are interested in knowing if, for a given a sequence of tasks of a behavior, it is possible to know which is the best HMM that can generate that sequence. To achieve this, we select the HMM that provides the maximum probability of HMM sequence generation for each test sample. On the other hand, in the second experiment we assume that the HMM for each sequence of tasks in the test data set has been correctly identified, and our goal is to know if the model is able to identify if the behavior sample is normal or abnormal. Since HMMs cannot distinguish between uncompleted and constraint violation samples, both behavior samples are considered as abnormal for the analysis.

Please cite this article in press as: M. Ros et al., Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows, Inform. Sci. (2011), doi:10.1016/j.ins.2011.10.005

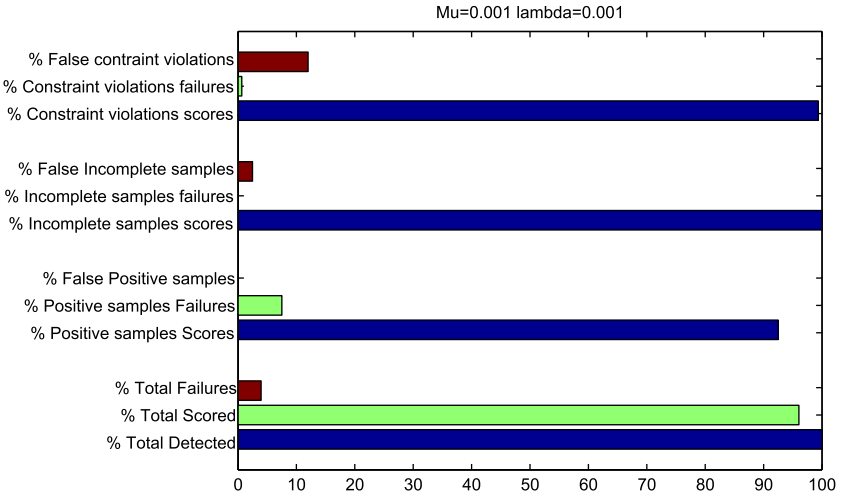


Fig. 4. Summary of online recognition results.

Table 5
Classification rate (%) for each specific behavior.

	B_1	B_2	B_3	B_4	B_5	B_6	B_7
% Success (normal)	100.00	66.27	85.42	100.00	100.00	93.81	100.00
% Failure (normal)	0	33.73	14.58	0	0	6.19	0
False positive (normal)	0	0	0	0	0	0	0
% Success (uncompleted)	100.00	100.00	100.00	100.00	100.00	100.00	100.00
% Failure (uncompleted)	0	0	0	0	0	0	0
False positive (uncompleted)	0	0	0	0	0	15.84	0
% Success (C. violations)	100.00	100.00	100.00	100.00	100.00	95.35	100.00
% Failure (C. violations)	0	0	0	0	0	4.65	0
False positive (C. violations)	0	39.58	25.45	0	0	0	0

To achieve this, a behavior is considered as normal if its probability of sequence generation using the corresponding HMM is over a threshold, otherwise it is considered abnormal. The threshold was set to the minimum probability of HMM sequence generation for all samples in the training data, because this value provided the best results in our experiments.

Table 6 describes the classification rate in both experiments for each behavior. We observe that, for the experiment 1, normal and abnormal behavior samples were identified by its corresponding HMM in 69.51% of instances, and 30.41% of instances could not be classified under any model. This last set of instances are samples with constraint violations that do not fulfill the behavior requirements, and valid and uncompleted samples with rubbish tasks that the models did not learn during the training. One positive aspect is that only 0.35% of all samples were misclassified in other behaviors than the reference class. Here, the HMM was not able to distinguish between tasks of the behavior and rubbish tasks associated to other HMM instead of B_6. On the other hand, if we focus on experiment 2 the true positive rate of normal behaviors correctly classified is 90.07%. However, the true positive rate of abnormal behaviors detection decreases to 52.62%. Thus, 47.38% of abnormal behaviors are classified as normal ones, which is a high security breach for user safety. If we compare the results obtained by HMM and our approach, we notice that the proposal in this work overcomes the limitation of HMM regarding first order assumption, and it is able to provide a high detection rate in both normal and abnormal behaviors. In addition, our behavior model is able to distinguish between relevant and non-relevant tasks for a behavior, which eases the recognition of different concurrent behaviors. Further important advantages of our proposal is the possibility to see the order relationship constraints between tasks graphically to ease the behavior management by experts if it would be necessary, and also the online adaptation capability studied in the next section.

6.3. Online adaptation to behavior changes and parameter influence analysis

The adaptation process is simultaneous to the recognition process. Thus, the behavior model is able control all changes in the behavior, as seasonal changes or fall repercussions in elders. However, the system needs user opinion to find out for sure if the behavior is changing or the user has carried out an abnormal behavior. In order to minimize the system-user interaction, the online adaptation process requires a fast online learning that depends on the system parameters, in contrast to the

Please cite this article in press as: M. Ros et al., Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows, Inform. Sci. (2011), doi:10.1016/j.ins.2011.10.005

recognition stage once a behavior is stable. In this section, the study is focused on parameters influence over final scores. Fig. 5 shows the evolution of percentages of correct classified behaviors in the 365 patterns of the recognition and adaptation databases, regarding μ_{min}, λ_{FALA} and λ_{CALA} parameters.

Firstly, Fig. 5(a) shows the relevance of parameters λ_{FALA} and μ_{min} regarding the recognition capability of the system. As the parameter λ_{FALA} increases, the percentage of correct detections decreases. This fact is due to the high adaptability rate that makes the behavior model lose its flexibility driving the *behavior model* to a just one available representation, increasing the number of failures. As far as μ_{min} concerned, larger ranges are more effective than shorter ones. As μ_{min} increments its value, it turns more restrictive, reducing the number of actions that could be part of the behavior, so that small changes in temporal constraints are allowed. On the other hand, as this value decreases, more elements are accepted to be analysed for the recognition process and large habit changes could be collected. However, if the assigned value is excessively small, the temporal constraints may turn less restrictive, provoking that in some cases, an out of time behavior could be recognised as normal instead of uncompleted.

Secondly, Fig. 5(b) and (c) describe the influence of parameters μ_{min}, λ_{CALA} and λ_{FALA} for adaptation to habit changes. As happened in the experiments for behavior recognition, larger values of μ_{min} are more restrictive to allow temporal violations and therefore, the adaptation rate is slower and the scores decrease as this parameter increases, independently of the learning rate for the FALA and CALA teams in the behavior models. In addition, it is shown that a fixed value of λ_{CALA} provides similar results independently of the value of μ_{min}. Smaller values of λ_{CALA} are more desirable for a better adaptation, while larger values may critically change the temporal window to areas in which the behavior cannot be executed. This fact makes the percentage of undetected behaviors to increase. However, large values for λ_{FALA} are more desirable when the system adaptation takes place in contrast to a recognition stage, since it allows the system to have a faster adaptation capability to the new action relationship constraints circumstances. This situation will be analysed in-depth in the discussion.

Considering the time required for a full adaptation to behavior changes, Fig. 6 shows the evolution through the time for adaptation from behavior *To leave home* (B_4) to behavior B_4^*, for different values of parameters λ_{FALA} and λ_{CALA}. Only this behavior is shown because its evolution is representative of all the remaining ones, and for space limitations. High values stand for a positive detection and low values mean that the sample was misclassified. This picture helps to support the previous assumption about the fact that large values of λ_{CALA} move the time window too much and then, the behaviors become even undetected. Thus, a large value of λ_{CALA} results in the increase of the accumulated undetected behaviors over the time, while the small values of this parameter make the undetected behaviors to be almost zero. Furthermore, this figure also explains that large values of λ_{FALA} may help for a quick adaptation, while lower ones decrease the adaptation rate. The time required for the full adaptation with the optimal values of the parameters is between 25 and 50 behavior detections, considering all the behaviors B_1^*—B_7^*. However, unexpected recognition failures may be found during the online recognition once the behaviors have been adapted, due to the existence of the noisy actions as explained in the previous section.

6.4. Discussion and final remarks

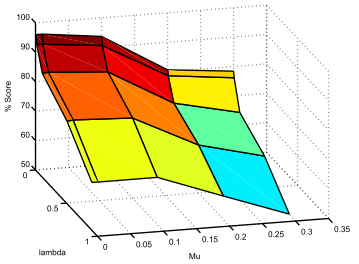
In previous sections, we concluded that parameter λ_{CALA} is not relevant for the task of behavior recognition, and also that small values of λ_{CALA} are more desirable for a better adaptation. On the other hand, we discovered that the recognition and adaptation capabilities are higher if the temporal constraints are more relaxed and the value of parameter μ_{min} is small. However, small values of parameter λ_{FALA} provide high detection scores in behavior recognition, while high values of λ_{FALA} are preferable for a faster adaptation. Thus, we find a conflict about which value should take λ_{FALA} to achieve high scores in both adaptation and recognition tasks.

To solve this situation, we propose to use two different values to update the action relationship model and the FALA teams within, λ^1_{FALA} and λ^2_{FALA}, with $\lambda^1_{FALA} < \lambda^2_{FALA}$. The first value λ^1_{FALA} would be used to update the models after a normal behavior recognition, so that the update would be carried out softly. On the other hand, the second one would be used after critic situations in which the system fails the recognition and detects a false positive abnormal behavior. Since the human–system interaction is required in these situations to find out if the abnormal behavior detected was executed deliberately by the user, we could use a larger value of FALA team adaptation λ^2_{FALA} to update the action models faster. With this strategy, we

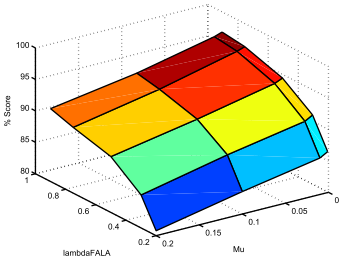
Table 6
Classification rate (%) of Hidden Markov Models.

	Experiment 1			Experiment 2	
	% Classified	% Misclassified	% Unknown	% Classified (normal)	% Classified (abnormal)
B_1	66.30	0	33.70	90.70	55.96
B_2	74.52	0	25.48	92.90	42.35
B_3	70.96	0	29.04	83.85	46.24
B_4	64.38	0	35.62	88.13	57.98
B_5	71.51	0	28.49	90.59	52.76
B_6	72.05	2.47	25.48	92.27	52.63
B_7	66.85	0	33.15	92.02	60.45
Average	69.51	0.35	30.14	90.07	52.62

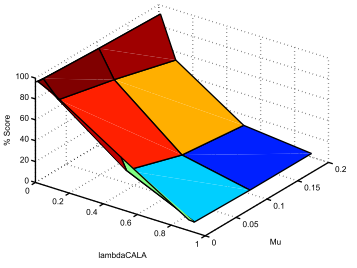
Please cite this article in press as: M. Ros et al., Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows, Inform. Sci. (2011), doi:10.1016/j.ins.2011.10.005



(a) Best scores obtained considering μ_{min} and λ_{FALA} for recognition



(b) Best scores obtained considering μ_{min} and λ_{FALA} for online adaptation



(c) Best scores obtained considering μ_{min} and λ_{CALA} for online adaptation

Fig. 5. System's score depending on parameters.

take advantage of both recognition and adaptation extreme situations, so that we are able to ensure the maximum score percentage.

As conclusion, the reliability degree obtained with the OWA operator should be studied. To be considered as a good reliability degree, it should only depend on the set of actions that the system has recognised as part of the behavior. Therefore, the value assigned to μ_{min} has to affect the quality of the final results. To prove this assumption, we calculated the average value of the evolution of the *reliability degree*, obtained with a constant $\lambda_{FALA} = 0.5$ and two different values of $\mu_{min} = 0.001$, 0.1. We obtained two data populations containing reliability degree values for each 365 normal behavior samples using these parameters. In our results, the reliability degree provides better (higher) results using the smallest μ_{min} value, for the determination of action sequences belonging to a behavior. These values are between 0.6 and 0.9, independently of the value of parameter μ_{min}. In order to test if both data populations are statistically similar, and therefore to know if the response of the OWA operator depends on μ_{min}, we applied a Student's *T*-test with a confidence level of 95% over the two average OWA data set. The test provided a probability value of 2.7250e−079, which concludes that there are significant differences between

Please cite this article in press as: M. Ros et al., Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows, Inform. Sci. (2011), doi:10.1016/j.ins.2011.10.005

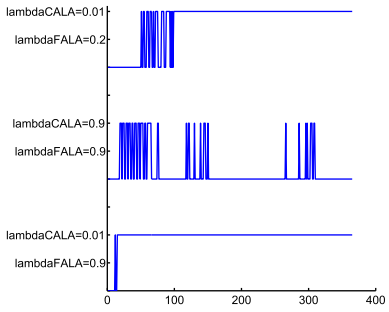


Fig. 6. System's scores and missing detections depending on parameters λ_{FALA} and λ_{CALA} during online adaptation.

both data populations. Therefore, this result supports that the reliability degree depends on the actions performed corresponding to the behavior, determined by the μ_{min} parameter.

7. Conclusions

In this work, we have developed a system which is able to recognise human behaviors from current user activity. The system uses a representation of the behaviors, known as *behavior model*, which contains all the information about a behavior: its temporal and order relationship constraints. To learn every *behavior model* and to recognise the current activity, we developed a method based on Learning Automata, taking advantages of the own feedback process for adapting itself when a behavior or environmental change occurs.

In order to deal with the two types of constraints which are typical of a behavior, we distinguish two modules in each *behavior model*: *behavior time* and *behavior action sequence*. The first one is in charge of studying the temporal constraints. To make this processing easier, we introduce the use of a *a fuzzy temporal window*, which allows to establish a defined interval where a behavior is probably performed. The FTW is learnt and analysed using a team of 2 continuous action-set learning automata (CALA). On the other hand, the order relationship constraints are responsibility of the *behavior action sequence*. This module is implemented using a team of finite action-set learning automata (FALA) to learn action order relationship constraints in a behavior model and a team of variable action-set learning automata (VALA) to recognising and adaptation process.

Our proposal has obtained suitable performances regarding both recognition and adaptation tasks. We have developed a complete experiment where we have studied the effect of the parameters over the results, concretely, the minimum value for belonging to a fuzzy temporal window and automata learning rates. Regarding the minimum membership degree, we have detected that using wider values for this parameter, the system is able to detect a bigger number of behaviors, but with the drawback that the reliability degree of the behavior decreases. On the other hand, we have found out that large values of the CALA learning rate may change drastically the area of the time windows, therefore making the adaptation difficult. In these cases, an extreme update of the parameters may locate the temporal window over time which does not correspond to the behaviors to be recognised, and then, the classification finds a larger number of undetected normal activities. Smaller values for the temporal learning rate are preferred, although the adaptation rate decreases. The opposite situation is found when the adaptation of human action order relationship constraints is considered. When the system is stable and just the behavior recognition is required, smaller values for the FALA learning rate are preferred to avoid the adaptation to noisy human actions carried out during the behavior time. However, when the behavior suffers from a change in the human habits, larger values of the learning rate are more suitable in order to learn the new action order relationship constraints in fewer iterations. These two situations may be controlled by means of the system–human interaction when a violation of an action relationship constraints is found: in these cases, a large learning rate is used, while a smaller one is applied during the system normal operation.

In this paper, we have studied time constraints between actions in a behavior and developed a complete model that also takes into consideration order relationship constraints. Up to now, the system controls behaviors individually and independently. However, the relations between behaviors, their integration and organisation to model other more complex ones or environmental situations should be addressed in future work. On the other hand, the development of a semantic representation of the environment, behaviors, and system begins to be essential in fact.

Acknowledgement

This work has been supported by the project TIN2009-14538-C02-01, I + D + I National Program from Spain Government

References

[1] G. Acampora, M. Gaeta, V. Loia, A.V. Vasilakos, Interoperable and adaptive fuzzy services for ambient intelligence applications, ACM Trans. Auton. Adapt. Syst. 5 (2010) 8:1–8:26.

Please cite this article in press as: M. Ros et al., Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows, Inform. Sci. (2011), doi:10.1016/j.ins.2011.10.005

[2] G. Acampora, V. Loia, Fuzzy control interoperability and scalability for adaptive domotic framework, IEEE Trans. Indus. Inform. 1 (2005) 97–111.
[3] M. Agache, B.J. Oommen, Generalized pursuit learning schemes: new families of continuous and discretized learning automata, IEEE Trans. Syst. Man Cybernet., Part B 32 (2002) 738–749.
[4] L. Atallah, G. Yang, The use of pervasive sensing for behaviour profiling – a survey, Perv. Mob. Comput. 5 (2009) 447–464.
[5] J. Boger, J. Hoey, P. Poupart, C. Boutilier, G. Fernie, A. Mihailidis, A planning system based on markov decision processes to guide people with dementia through activities of daily living, IEEE Trans. Inform. Technol. Biomed. 10 (2006) 323–333.
[6] J. Candamo, M. Shreve, D.B. Goldgof, D.B. Sapper, R. Kasturi, Understanding transit scenes: a survey on human behavior-recognition algorithms, IEEE Trans. Intell. Transp. Syst. 1 (2010) 206–224.
[7] D.J. Cook, M. Youngblood, E.O. Heierman III, K. Gopalratnam, S. Rao, A. Litvin, F. Khawaja, Mavhome: an agent-based smart home, in: PERCOM'03: Proceedings of the First IEEE International Conference on Pervasive Computing and Communications, IEEE Computer Society, Washington, DC, USA, 2003, p. 521.
[8] M. Cuéllar, M. Ros, M. Delgado, A. Vila, Detection of abnormal human activities by means of Learning Automata, Technical Report, Department of Computer Science and Artificial Intelligence, University of Granada, Department of Computer, 2010.
[9] M. Delgado, M. Ros, M. Amparo Vila, Correct behavior identification system in a tagged world, Expert Syst. Appl. 36 (2009) 9899–9906.
[10] F. Doctor, H. Hagras, V. Callaghan, A fuzzy embedded agent-based approach for realizing ambient intelligence in intelligent inhabited environments, IEEE Trans. Syst. Man Cybernet., Part A 35 (2005) 55–65.
[11] R.M. Droes, M. Mulvenna, C. Nugent, D. Finlay, M. Donnelly, M. Mikalsen, S. Walderhaug, T.v. Kasteren, B. Krose, S. Puglia, F. Scanu, M.O. Migliori, E. Ucar, C. Atlig, Y. Kilicaslan, O. Ucar, J. Hou, Healthcare systems and other applications, IEEE Perv. Comput. 6 (2007) 59–63.
[12] T. Gu, Z. Wu, X. Tao, H.K. Pung, J. Lu, Epsicar: an emerging patterns based approach to sequential, interleaved and concurrent activity recognition, in: IEEE International Conference on Pervasive Computing and Communications, vol. 0, 2009, pp. 1–9.
[13] V. Guralnik, K.Z. Haigh, Learning models of human behaviour with sequential patterns, in: Proceedings of the AAAI-02 Workshop "Automation as Caregiver", pp. 24–30.
[14] H. Hagras, V. Callaghan, M. Colley, G. Clarke, A. Pounds-Cornish, H. Duman, Creating an ambient-intelligence environment using embedded agents, IEEE Intell. Syst. 19 (2004) 12–20.
[15] H. Hagras, M. Colley, V. Callaghan, G. Clarke, H. Duman, A. Holmes, A fuzzy incremental synchronous learning technique for embedded-agents learning and control in intelligent inhabited environments, in: Proceedings of the 2002 IEEE International Conference on Fuzzy Systems, 2002. FUZZ-IEEE'02, vol. 1, pp. 139–144.
[16] V. Jakkula, D.J. Cook, Anomaly detection using temporal data mining in a smart home environment, Methods Inform. Med. (2008) 70–75.
[17] T. van Kasteren, B. Krose, Bayesian activity recognition in residence for elders, in: 3rd IET International Conference on Intelligent Environments, 2007, IE 07, pp. 209–212.
[18] J.A. Kientz, S.N. Patel, B. Jones, E. Price, E.D. Mynatt, G.D. Abowd, The georgia tech aware home, in: CHI'08: CHI'08 Extended Abstracts on Human Factors in Computing Systems, ACM, New York, NY, USA, 2008, pp. 3675–3680.
[19] E. Kim, S. Helal, D. Cook, Human activity recognition and pattern discovery, IEEE Perv. Comput. 9 (2010) 48–53.
[20] D. Kitakoshi, H. Shioya, R. Nakano, Empirical analysis of an on-line adaptive system using a mixture of bayesian networks, Inform. Sci. 180 (2010) 2856–2874.
[21] B. Krose, van T. Kasteren, C. Gibson, van den T. Dool, Care: context awareness in residences for elderly, in: Proceedings of ISG'08: The 6th International Conference of the International Society for Gerontechnology, pp. 101–105.
[22] X.H. Le, S. Lee, Y.K. Lee, H. Lee, M. Khalid, R. Sankar, Activity-oriented access control to ubiquitous hospital information and services, Inform. Sci. 180 (2010) 2979–2990.
[23] S.W. Lee, Y.S. Kim, K.H. Park, Z. Bien, Iterative bayesian fuzzy clustering toward flexible icon-based assistive software for the disabled, Inform. Sci. 180 (2010) 325–340.
[24] L. Liao, D.J. Patterson, D. Fox, H. Kautz, Learning and inferring transportation routines, Artif. Intell. 171 (2007) 311–331.
[25] H. Liu, H. Darabi, P. Banerjee, J. Liu, Survey of wireless indoor positioning techniques and systems, IEEE Trans. Syst. Man Cybernet., Part C 37 (2007) 1067–1080.
[26] S. Luhr, G. West, S. Venkatesh, Recognition of emergent human behaviour in a smart home: a data mining approach, Perv. Mobile Comput. 3 (2007) 95–116. Design and Use of Smart Environments.
[27] X. Meng, K.K. Lee, Y. Xu, Human driving behavior recognition based on hidden markov models, in: IEEE International Conference on Robotics and Biomimetics, vol. 0, 2006, pp. 274–279.
[28] U. Naeem, J. Bigham, A comparison of two hidden markov approaches to task identification in the home environment, in: 2nd International Conference on Pervasive Computing and Applications, IEEE, 2007, pp. 383–388.
[29] K. Najim, A. Poznyak, Multimodal searching technique based on learning automata with continuous input and changing number of actions, IEEE Trans. Syst. Man Cybernet., Part B: Cybernet. 26 (1996) 666–673.
[30] C. Nugent, M.D. Mulvenna, F. Moelaert, B. Bergvall-Koreborn, F. Meiland, D. Craig, R. Davies, A. Reinersmann, M. Hettinga, A.L. Andersson, R.M. Droes, J. Bengtsson, Cogknow: Helping people with mild dementia to navigate their day, <http://www.cogknow.eu>, 2005 (accessed 10.05.10).
[31] C. Nugent, M.D. Mulvenna, F. Moelaert, B. Bergvall-Koreborn, F. Meiland, D. Craig, R. Davies, A. Reinersmann, M. Hettinga, A.L. Andersson, R.M. Droes, J. Bengtsson, Home-based assistive technologies for people with mild dementia, in: Pervasive Computing for Quality of Life Enhancement, vol. 4541, pp. 63–69.
[32] M. Philipose, K.P. Fishkin, M. Perkowitz, D.J. Patterson, D. Fox, H. Kautz, D. Hahnel, Inferring activities from interactions with objects, IEEE Perv. Comput. 3 (2004) 50–57.
[33] A.S. Poznyak, K. Najim, Learning Automata and Stochastic Optimization, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1997.
[34] M. Rantz, M. Aud, G. Alexander, D. Oliver, D. Minner, M. Skubic, J. Keller, Z. He, M. Popescu, G. Demiris,, S. Miller, Tiger place: an innovative educational and research environment, in: AAAI in Eldercare: New Solutions to Old Problems.
[35] P. Rashidi, D. Cook, Keeping the resident in the loop: Adapting the smart home to the user, IEEE Trans. Syst. Man Cybernet., Part A: Syst. Humans 39 (2009) 949–959.
[36] P. Rashidi, D.J. Cook, L.B. Holder, M. Schmitter-Edgecombe, Discovering activities to recognize and track in a smart environment, IEEE Trans. Knowl. Data Eng. 99 (2010).
[37] M. Ros, M. Delgado, A. Vila, Fuzzy method to disclose behaviour patterns in a tagged world, Expert Systems with Applications 38 (2011) 3600–3612.
[38] G. Singla, D.J. Cook, M. Schmitter-Edgecombe, Tracking activities in complex settings using smart environment technologies, Int. J. BioSci., Psy. Technol. 1 (2009).
[39] E.M. Tapia, S.S. Intille, K. Larson, Activity recognition in the home using simple and ubiquitous sensors, in: A. Ferscha, F. Mattern (Eds.), Pervasive Computing, Lecture Notes in Computer Science, vol. 3001, Springer, Berlin/ Heidelberg, 2004, pp. 158–175.
[40] M.A.L. Thathachar, B.R. Harita, Learning automata with changing number of actions, IEEE Trans. Syst. Man Cybern. 17 (1987) 1095–1100.
[41] M.A.L. Thathachar, P.S. Sastry, Varieties of learning automata: an overview, IEEE Trans. Syst. Man Cybernet., Part B 32 (2002) 711–722.
[42] J. Torkestani, M. Meybodi, Graph coloring problem based on learning automata, in: ICIME'09. International Conference on Information Management and Engineering, 2009, pp. 718–722.
[43] A.C. Tran, S. Marsland, J. Dietrich, H.W. Guesgen, P. Lyons, Use cases for abnormal behaviour detection in smart homes, in: Aging Friendly Technology for Health and Independence. Proceedings of the 8th International Conference on Smart Homes and Health Telematics, ICOST 2010, Seoul, Korea, June 22–24, 2010, pp. 144–151.

Please cite this article in press as: M. Ros et al., Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows, Inform. Sci. (2011), doi:10.1016/j.ins.2011.10.005

[44] P.K. Turaga, R. Chellappa, V. Subrahmanian, O. Udrea, Machine recognition of human activities: a survey, IEEE Trans. Circuits Syst. Video Technol. 18 (2008) 1473–1488.
[45] D.J.C. Vikramaditya Jakkula, A. Crandall, Enhancing Anomaly Detection using Temporal Pattern Discovery, Advanced Intelligent Systems, Springer, 2008. pp. 175–194.
[46] D. Zhang, D. Gatica-Perez, S. Bengio, I. McCowan, G. Lathoud, Modeling individual and group actions in meetings with layered hmms, IEEE Trans. Multimedia 8 (2004) 509–520.

Please cite this article in press as: M. Ros et al., Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows, Inform. Sci. (2011), doi:10.1016/j.ins.2011.10.005

3.3. A Fuzzy Logic Approach for Learning Daily Human Activities in an Ambient Intelligent Environment

The conference paper associated is:

- Ros, M.; Delgado, M.; Vila, A.; Hagras, H. and Bilgin, A. A Fuzzy Logic Approach for Learning Daily Human Activities in an Ambient Intelligent Environment IEEE International Conference on Fuzzy Systems, 2012, In press
  - Status: **In press**.
  - Conference Category (CORE 2008): A

A Fuzzy Logic Approach for Learning Daily Human Activities in an Ambient Intelligent Environment

María Ros, Miguel Delgado, Amparo Vila
Department of Computer Science and Artificial Intelligent
University of Granada
Granada, Spain

Hani Hagras, Aysenur Bilgin
The Computational Intelligence Centre
University of Essex
Colchester, United Kingdom

***Abstract*— This paper addresses the problem of learning human behavior models from sensor information in a smart home environment. Any smart home is provided with many devices that can determine the state of the environment at any moment, as well as the user interaction with the environment. This information is used by our approach to learn a flexible and reliable human behavior representation, extracting the relevant actions and the order constraints among them.**

In order to test our learning approach, we have performed experiments in the iSpace at the University of Essex which is an Ambient Intelligent Environment (AIE) testbed. We will present the results obtained by monitoring three participants' activities for three specific behaviors. The learned behavior model is compared with the behavior model provided by the participants. The results show that our proposed system effectively learns the behavior models for any behavior, acquiring not only the actions the user considers as basic, but also those unconsciously performed yet important ones done by the user.

Keywords-component; behavior modelling, fuzzy logic, ambient intelligence, Learning Automata.

I. INTRODUCTION

Ambient intelligence (AmI) is a new vision for electronic environments that are sensitive and responsive to human presence [1]. Inside these environments, the electronic devices are virtually invisible and are able to respond to the user needs. The systems learn the user behavior using the information collected from users' daily activities, always following the principles of ubiquity, transparency and intelligence [1].

In the AmI paradigm, the modeling, recognition, and prediction of Activities of Daily Living (ADL) [2], [3] are becoming more important. An ADL is a sequence of atomic actions within user's daily routine, such as toileting, making a meal, or leaving home. Most of the projects are focused on elderly people, such as [4], [5], [6]. For example, [4] proposed a method based on temporal clustering to detect a gradual change as a result of a deteriorating condition.

Recently, there has been a high interest in the identification of behavior models [7] for assisted living. Previous works offered studies of a wide variety of aspects, ranging from the underlying sensor network for user data acquisition [2], [8], to representation [2], [9], [10], behavior modeling [11], [12], [13] and applications [4], [5], [6], [14]. Probabilistic and uncertainty models are the most common choice in the literature to model an ADL as probabilistic sequences of objects touched by the user [2]. Other approaches have also been employed such as Hidden Markov Models (HMMs) [9], [15], [16], Markov Decision Processes [10], hybrid models of HMM and Linear Discriminant Analysis [17], agents and fuzzy systems[11][18], etc.

Nevertheless, in most of the previous work, the behavior model is completely determined by the method proposed. The idea of a behavior model separated from the learning technique used is presented in [19], [20]. With this strategy, the learning procedure is independent from the formal behavior representation, which allows for the use of different learning techniques. In this work, we present an approach to learn these behavior models. Our approach is based on two main assumptions: any behavior has associated temporal information and a behavior model could be determined by its common actions and the order constraints among them. Following these assumptions, we proposed a method that primarily uses the behavior temporal information to find the important actions. Next, a new Frequent Itemsets-based algorithm finds the relevant actions and the order constraints among them. This information is used by a multi-agent system, implemented by Learning Automata, to learn the behavior model. In order to evaluate our approach, we have performed unique experiments in which our proposed method is tested, in the iSpace at the University of Essex which is a real world AIE testbed. In our experiments, we had three participants performing three behaviors (Wake up, Have Breakfast and Leave Home) during five days. We will use the Similarity Measure, proposed in [21], [22], to evaluate the quality of the results.

The rest of this paper is organized as follows: In Section II, the behavior model concept is introduced. Section III presents a description of the method designed to learn any behavior model. Section IV presents experiments and results. Section V provides the conclusions and future work.

II. MODELLING HUMAN BEHAVIOR

The behavior model is a representation of a human behavior. Some examples of behavior models in the literature include employing HMMs [9] and fuzzy rule-based control systems [12], [13]. In contrast, we define a human behavior as a sequence of human actions that could (or not) have temporal relationship constraints between them [20]. The formal descriptions that support our research are the following:

Definition 1. (Human Action)[19],[20]: A human action $h_i = < l_i, s_i, t_i >$ *h_i is a triplet $h_i = < l_i, s_i, t_i >$, where l_i is a*

label used to identify the action, s_i is a sensor event and t_i is a time stamp in which the action occurs. In brief, a human action labeled l_i is an activity (fact) that happens over a specific object in a time instant t_i and it may be uniquely determined by means of a sensor event s_i. An example instance of the human action "open the front door" could be <"OpenFrontDoor", "FrontDoorSensor\#1234 = 1", "8:30:47a.m.">.

In this work, we will deal with two types of actions: Human and learning automaton actions, as we will see in Section III. In order to distinguish between them for the purpose of clarity, we will rename the human action to be human task from now on, without any loss of meaning.

Definition 2. (Behavior) [19],[20]: A human behavior B_i is a quadruple $B_i = < H, R_i, C_i, \prec >$, where $H = \{h_1, h_2, \dots, h_n\}$ is a set of available human tasks, $R_i = \{r_1^i, r_2^i, \dots, r_{|R_i|}^i\} \subseteq H$ is the subset of available human tasks that are required to complete the behavior, and C_i is a set of temporal constraints between the tasks in R_i defined over a partial order relationship operator $\prec$. The partial order relationship $\prec$ defines the temporal dependency requirements over the actions of a behavior, and its formal definition is

$$r_a^i \prec r_b^i \leftrightarrow t_a \prec t_b. \quad (1)$$

In brief, an action r_a^i precedes r_b^i in time in a behaviour B_i, and it is written $r_a^i \prec r_b^i$, if the time stamp t_a of r_a^i is lower than the time stamp t_b of r_b^i, for all the possible instances of the behavior.

We illustrate the definition of behavior with an example. Let us define the behavior *Go to sleep*. The set *H* is the set of all the available tasks that the user could carry out at home, and *R* contains the relevant tasks for the behavior, as for instance the set *{To put the pajama on, To sit on the bed, To take the slippers off, To take the pills* and *To turn off the lights}*. Let us assume that the behavior is normal if the user always sits on the bed and puts his/her pajama on before taking the pills, and the last action s/he does is to turn off the lights. Then the set of temporal constraints between the human tasks is *C = {To put pajama on ≺ To take the pills, To sit on the bed ≺ To take the pills, To take the slippers off ≺ To turn off the light*, and *To take the pills ≺ To turn off the light}*.

These order relationships may be modeled graphically as a set of AND gates connected in cascade. Each human task is matched with an input line to an AND gate. Its value is *false* if the action has not been executed and *true* otherwise. In addition, each human task is assigned to a level in the cascade considering the temporal precedence in respect to the other tasks. An action *a* that precedes another action *b* is located in an upper level, while *b* is located in a lower one. Figure 1 shows an example of a possible behavior model for the previous example *Go to sleep*. In this example, the task *To take the pills* is located at the second cascade level since the task *To sit on the bed* precedes this one. In addition, the task *To turn off the light* is located at the deepest level since all the relevant tasks in the behavior precede this action.

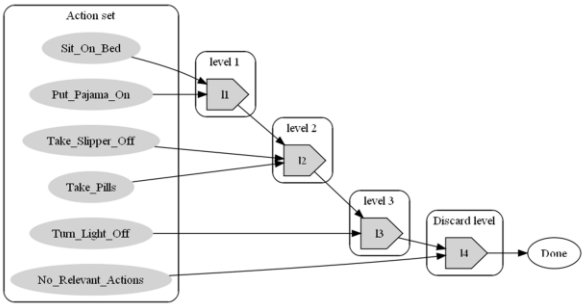


Figure 1 Behavior model for *to go to sleep*

TABLE I
EXAMPLE MATRIX TO REPRESENT THE BEHAVIOR TO GO TO SLEEP

Tasks	L1	L2	L3	L4	LU
To sit on the bed	1	0	0	0	0
To put pajama on	0.7	0.3	0	0	0
To take the slippers off	0.3	0.7	0	0	0
To take the pills	0	0.9	0.1	0	0
To turn off the light	0	0	1	0	0
To open kitchen's tap	0	0	0	0	1

However, this design is static and does not allow to model behaviors that can be executed in different ways. For instance, the task *To take the slippers off* could be executed either in level 1 or 2 in the previous example, since it just has to be performed before turning the light off; but, it does not have another type of constraint. To avoid this situation, we propose a method based on a distribution of each task in levels; so that, the levels represent the order constraints among the actions. Each task is assigned to an execution level with a probability value; what means that this task will be performed before and after the actions in the rest of levels with a specific probability. Additionally, we include a last level, *the discard level*, where the actions that are not relevant for the behavior are connected.

Formally, the underlying representation of a behavior makes use of a probability distribution for each action in the set *H*, so that an action h_i may be executed in each of the *r* possible levels of the cascade with probability $P^i = [p_1^i, p_2^i, \cdots, p_r^i]^t$. This probability distribution represents that each action can be executed within a cascade level with a concrete probability value. On the other hand, the representation as a sequence of ANDs in cascade is the most likely representation of behavior performance, assigning each action to its most probable level in the cascade. Therefore, a human task h_i is located at the cascaded level l if and only if, $l = \min\{j : p_j^i(k) = max_j\{p_j^i(k)\}\}$ i.e., l is the first level with a maximum probability value in the matrix. Table I exemplify a matrix with maximum 4 levels to represent the behavior of Figure 1, where the level *U* is the discard level. Thus, *To sit on bed* and *To put pajama on* are assigned to level 1, since the minimum level where their probabilities are maximum is the first level.

III. LEARNING THE BEHAVIOR MODEL

One of the main advantages of our system is the independence of the model approach and the method used to learn the user behavior. This allows us to achieve better system modularization and to use a larger variety of learning methods for the same activity model [20].

In this paper, we propose a method composed of two main steps: the extraction of the behavior patterns and the learning of that information to build the behavior model. The first step will infer a set of relevant actions and their order constraints, while the second step will provide a unique behavior model that represents the user activities. With the former step, we want to reinforce the correctness of the learning process of the behavior model, providing only the relevant actions.

A. Extracting behavior patterns

As indicated previously, our system learns the behavior model from sensor information collected by monitoring the user activity for several days. However, not all collected information is relevant for a specific behavior, hence only a subset of that information should be studied. The questions under focus are which subsets are to be extracted and how. To manage these problems, our proposal assumes that a daily behavior is usually performed around a known time [23], being able to establish an adjusted interval in which the behavior should be performed by the user. However, not all actions of the interval are equally important for the behavior's final performance. In order to solve this problem, in [23], we proposed a method entitled "Fuzzy Temporal Window (FTW)". Formally presented in Definition 3 a FTW is a temporal interval associated with a specific behavior performance and a fuzzy set. Every action in the interval has a specific degree of importance belonging to the fuzzy set, hence to the behavior performance.

Definition 3. (Fuzzy Temporal Window): Let I *be an interval from the temporal line* τ*, ODB is the sensor information collected and* $t \in ODB$*. Let* i_j *be an action* $i_j = (h_{,} l_j, d_j\}$ *and a Temporal Window* W *for a specific behavior. Let fs be a fuzzy set over* τ *, hence a* fuzzy temporal window FTW *is defined as a Temporal Window where*

$$\forall i_j \in W(t), \mu_{FW}(i_j) = \mu_{fs}(i_j) \quad (2)$$

After applying a Fuzzy Temporal Window over the collected dataset ODB (Observation Database), we obtain a subset of ODB $\widetilde{\text{ODB}}$ [23].

Definition 4. (Fuzzy ODB, $\widetilde{ODB}$ *): Let ODB be an Observation Database and W a Fuzzy Temporal Window, it defines* $\widetilde{ODB}$ *as a Fuzzy Observation Data Base constructs as W(ODB) such a*

$$\forall t \in ODB, W(t) \in \widetilde{ODB} \quad (3)$$

$\widetilde{\text{ODB}}$ is a representation of ODB where every action included has a degree of membership for a specific FTW, so, we extract as many $\widetilde{\text{ODB}}$ as number of behaviors we study.

At this point, we have limited the dataset in order to extract which actions are relevant (or not) for a specific behavior. The following step consists of finding which actions could represent the behavior performance and which order constraints could exist among them. For this purpose, we apply the method based on *Frequent Itemset* as presented in [23], [24].

Suppose we want to find out the behavior patterns for the behavior *B*, which has a related FTW, noted FTW_B, and a known $\widetilde{ODB}_B$. First of all, as we apply the α-cut concept over a $\widetilde{ODB}_B$, we have a new crisp image of $\widetilde{ODB}_B$, $\widetilde{ODB}_B^{\alpha}$ where every value is in *[0, 1]*. For each $\widetilde{ODB}_B^{\alpha}$ extracted for behavior *B*, we obtain its relevant actions using the *Apriori algorithm.* However, a behavior pattern is not only defined by its relevant actions, but also by the order constraints that must follow. In [24], we present an algorithm to find them using the user information that is already known.

Additionally, as we use *Frequent Itemsets*, we need to represent them as unique fuzzy set. Thus, we have to ensure the *consistent restriction* between every α *frequent itemset* I^{α}. For sequence patterns representation is the same. The proof of these statements is in [23].

B. Using the behavior patterns to learn the behavior model

Once the behavior patterns have been extracted, the next step is to learn the behavior model. With this aim in mind, we proposed a multi-agent system [19], [20], based on Learning Automata [26]. This method is able to infer behavior models from the sensor data obtained during the monitoring process of the user. However, instead of using the sensory data, we propose in this paper to make use of the pattern behaviors in order to improve the quality of the learned model. Next, we introduce the concept of Learning Automata and summarize briefly the method proposed in [19].

1) Learning Automata Overview: The multi-agent system we employ is based on Learning Automaton [27] in which a decision-making stochastic machine is defined by the tuple $\langle \alpha, Q, R, T \rangle$, where α is a set of available actions for the automaton, Q is the internal state, R is a set of input values to the automaton, and T is a reinforcement learning scheme. At each time instant *k*, the automaton selects an action $a(k) \in \alpha$ according to the automaton internal state $Q(k)$, and returns $a(k)$ to an unknown random environment. Then, the environment evaluates the action chosen by the automaton with a fitness evaluation function $f: D\ x\ \alpha \rightarrow R$ and provides a reward or penalty reinforcement value $\beta(k) = f(D,\ a(k)) \in R$, depending on the suitability of $a(k)$ for the environment internal fitness evaluation function, where *D* is the expectation to obtain a reward reinforcement value. The internal state of the automaton is updated with the reinforcement scheme, considering the action chosen by the automaton and the reinforcement value received, $Q(k+1) = T\big(Q(k), a(k), \beta(k)\big)$. This process is repeated until a predefined desired condition is fulfilled, as for example the convergence to the learnt optimal action.

In this work, we use Finite Action-set Learning Automata (FALA) [26] to learn behavior models from user monitoring. Here, the set of available actions $\alpha = \{\alpha_1, \alpha_2, \cdots, \alpha_n\}$ is discrete and finite, and the internal state of the automaton is modeled as a probability distribution for the action selection $Q(k) = P(k) = [p_1(k), p_2(k), \cdots, p_n(k)]^t$. At each *k* time instant, the

automaton chooses the action $\alpha = \alpha_i$ according to this probability distribution, and applies it over the environment.

2) A team of Learning Automata to build a Behavior model: Our proposal [18], [20] uses a team of FALA for learning the behavior model. Every team has the same number of FALA as the number of available human actions in *H*. The *set of available actions* for each learning automaton is $\alpha = \{\alpha_1, \alpha_2, \cdots, \alpha_{|H|+1}\}$, where $|H|$ is the cardinal of *H*. Each automaton is matched uniquely with a task. We write LA^i to name the learning automaton assigned to the human task h_i.

The selection of the action α_j by the automaton LA^i means that the human task h_i is required to complete the behaviour and it should be included in *R*, for $j \leq |H|$. In addition, h_i is assigned with the *j*-th cascade level in the behavior model. On the other hand, if the automaton selects the action $\alpha_{|H|+1}$, then it is assumed that the action is not required to complete the behaviour and it should not be included in *R*.

The state of the automaton LA^i at the *k*-th iteration is defined by

$$Q^i_{FALA}(k) = P(k) = \left[p^i_1(k), p^i_2(k), \cdots, p^i_{|H|+1}(k)\right]^t, \quad (4)$$

where $p^i_j(k)$ is the probability of LA^i to select the action α_j at the *k*-th iteration of the learning process, i.e. the probability to assign the human task h_i to the *j*-th level of the behavior model if $j \leq |H|$, or to select that the action is not required to complete the behavior correctly if $j = |H| + 1$.

The created behavior model is sent to the environment, which contains *S behavior patterns* obtained in the previous step. This is a big difference when compared with our previous approach [18] [20], where the LA environment included only the sensory data gathered from monitoring of the user actions. Using this pre-processed environment, we ensure that the learning process will be more effective and quicker than the one in previous approaches. The effectiveness comes from the fact that the environment only contains correct samples, i.e. unbroken order constraint in the relevant action-sets. On the other hand, the reduction of the environment size makes the process lively. The behavior model is evaluated in each sample, and the environment returns a reinforcement value for each behavior sample following a *Q*-model [27]. The reinforcement is applied over the internal state Q of every automaton in the team. This process will continue until the convergence criteria will be fulfilled.

IV. EXPERIMENTS AND RESULTS

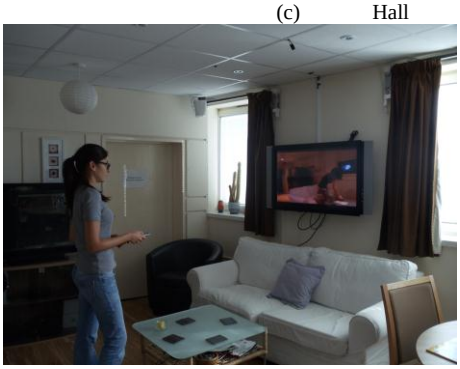
We have performed several unique experiments within the iSpace at the University of Essex (see Figure 2). The iSpace is a real Ambient Intelligent Environment equipped with all the facilities of a two-bedroomed apartment, together with the required devices to manage it. The apartment is composed of a bedroom, a study, an entrance hall, a fully equipped kitchen, a living room and a bathroom, which are organized in four labeled areas as kitchen, sofa, bedroom and hall (see Figure 2). On the other hand, each room contains a set of various devices to interact with, all of which use an interface based on the *Universal Plug&Play* (UPnP) architecture. The

(a) Bedroom

(b) Bathroom

(c) Hall

(d) Kitchen



(e) Living room

Figure 2 iSpace Rooms

TABLE II
3-PARTICIPANTS EXPERT KNOWLEDGE

Behavior	Participant	Mean	Deviation	Sequence of actions
Wake up	1	06:55:00	00:20:00	Alarm clock off ≺ Lamp on
	2	07:00:00	00:20:00	Open bedroom curtains ≺ Alarm clock off
	3	07:25:00	00:30:00	Alarm clock off ≺ Radio on
Have Breakfast	1	07:30:00	00:20:00	{Kitchen curtains opening, Coffee Machine on} ≺ Toaster on
	2	07:40:00	00:25:00	{Kitchen curtains opening, Coffee Machine on} ≺ Toaster on
	3	08:00:00	00:20:00	{Coffee machine on, Toaster on} ≺ {Coffee machine off, Toaster off}
Leave home	1	08:00:00	00:20:00	Hall light off ≺ Opening door Opening door ≺ Closing door
	2	08:15:00	00:25:00	Enter hall ≺ Exit hall
	3	08:30:00	00:20:00	{Enter the kitchen, Radio off} ≺ Enter the hall Enter the hall ≺ Exit the hall

communication with the devices is made through an API, programmed in Java language, which provides discovery of the services belonging to a range of devices and also interaction via sending messages over the network.

Regarding the devices in our experiment, two input sensors are monitored: doortrap sensor and *Ubisense Tracking Sensor*. The *Ubisense Tracking Sensor* detects when the user either enters into or exits a zone. In addition, we check the state of 18 actuators consisting of light-switches, curtain-controllers, TV remote control and switch controllers. The last set of actuators (switch controllers) controls whether some specific electrical appliance is connected to the UPnP network or not. Those devices are: a coffee machine, a little bedroom lamp, an alarm clock, a toaster and a radio. To manage the environment, we have developed an *interface* to send the specific commands to the actuators such as turn them on and off.

The goal of our experiment is to prove that our system is capable of learning the *behavior model* for different people and for different type of behaviors. For this purpose, we studied *three* participants and *three* different behaviors, which are *Wake up*, *Have breakfast* and *Leave home*. Each participant was requested to perform these behaviors in the *iSpace* in a *natural* way, i.e., perform the same sequence of actions that s/he usually carries out at his/her own home. Additionally, every participant was requested to provide some expert knowledge about his/her way of performing the behaviors: *the temporal interval*, *the sequence of actions to carry out* and *the order constraints among them*. Table II shows a summary of the aforementioned expert knowledge. The temporal interval, which is provided by the participants (see Table II), is represented as a fuzzy temporal window that uses a Gaussian function, having the mean equal to the value of the arithmetic average of the two end points of the interval where the deviation equals to the halved interval length. The order constraints are indicated using $\prec$ operator, where $a \prec b$ means that the task *a* must be done before task *b*. In Section IV.B, this expert knowledge will be used to test our proposed approach.

For each participant, we learn the behavior model for each type of behavior. In order to evaluate the effectiveness of our method according to each user behavior, we define a measure to determine the similarity between the learned behavior and the real user behavior by using *Similarity measure* [21], [22].

A. Similarity measure

In [21][22], the authors propose a Similarity measure for comparing pairs of behavior models. Basically, the measure includes three components that study all the important aspects of the behavior model explained in Section II:

a) Comparing soft partitions: determines the similarity between pairs of behaviors according to the strength of actions grouping into similar levels.

$$s_{cp}(U,V) = \frac{s(U,V)}{maximum(\, s(U,U)\, ,s(V,V)\,)}. \quad (5)$$

where $s(U,V)$ is the strength of actions grouping into similar levels. It is calculated using the comparing soft partitions technique proposed in [28], viewing tasks as data points and levels as clusters.

b) Subsethood of Non-Discarded Actions: two behavior models may share many actions in the discarded level. This component minimizes the influence of those actions in the measure.

$$s_{sub}(U,V) = \frac{|R_U \wedge R_V|}{|R_U \vee R_V|} \quad (6)$$

where R_U and R_V are the relevant set of actions for behavior U and V, respectively; $\wedge$ is intersection; $\vee$ is union, and $|\bullet|$ is cardinality.

c) Similarity of Partial Order Relations: determines if the order of levels is consistent. Two behavior models may share a set of non-discarded actions and have a strong similarity of actions grouping into levels (clusters), but the order of levels is not consistent, if the measure has to factor its effect as follows:

$$s_{por}(U,V) = 1 - \frac{(\sum_{(i,j)\in I'} \emptyset(\mathrm{E_U(i,j)}, \mathrm{E_V(i,j)}))}{|I'|}, \quad (7)$$

$$\emptyset\big(\mathrm{E_U(i,j)}, \mathrm{E_V(i,j)}\big) = \begin{cases} 0 & if\ \mathrm{E_U(i,j)} = \mathrm{E_V(i,j)} \\ \min(\mathrm{P_{U,i}}, \mathrm{P_{V,i}}) & else \end{cases} \quad (8)$$

where E_U is a $|H|\ x\ |H|$ matrix that represents the order relationships (before, same, after) between two actions and $P_{(U',i)}$ measures the ability to decide that action *i* belongs to some particular level.

Therefore, the final measure for comparing human behaviors is

$$s_{final}(U,V) = s_{cp}(U,V) \times s_{sub}(U,V) \times s_{por}(U,V) \quad (9)$$

B. Results and Discussion

As stated before, in our experiments, each participant was requested to perform *independently* the sequence of *habitual* actions that s/he performs to *wake up*, *have breakfast* and *leave home* in the *iSpace*. During 5 days, each user's activities were recorded using the sensors and actuators in the *iSpace*. At the beginning of each experiment session, the system applied a specific mode in *iSpace*, named as *Sleepy mode*, consisting of the following specific actions: closing all curtains, turning all the devices off and turning all the lights off. Each participant wore a tag of *Ubisense Tracking Sensor*, which identified exactly where s/he is at the space at any moment. Each experiment session started when the alarm clock went off at a specific time provided by the participants. From that moment on, the system recorded all the user interactions (events) with the environment, received either from the sensors or through the interface. For each event, the *time* and the *activated sensor/actuator* were stored. Each experiment session finished when the participant left the *iSpace*.

Next, we briefly summarize the participants' activities during the experiment. Note that this is just the general scenario they performed. At each experiment session, the scenarios might be performed in different ways and times:

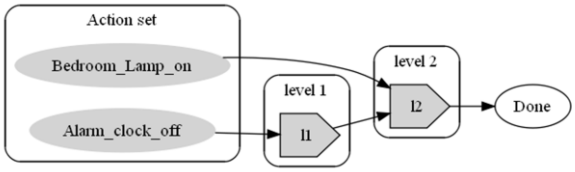
- Participant-1: Her alarm clock went off at 6:55. She turned it off and switched the bedroom lights on. She left the bedroom and went to the bathroom, where she turned the lights on. After some minutes, she exited it and went into the kitchen, through the hall, turning some lights on/off. At the kitchen, she prepared her breakfast, using the coffee machine and the toaster. In some sessions, the participant watched the television at the living room meanwhile she was having breakfast. Once she finished, she left the kitchen and went into the bathroom again. Several minutes later, she entered into the bedroom, packed her stuff, went into the hall, opened the door, stepped out and closed the door.
- Participant-2: The alarm was set to 7:00 A.M. The first thing she did was to open the curtains in the bedroom. After postponing the alarm for a couple of times, she got out of the bed and went to the bathroom for a few minutes. Then, she usually (in 4 of 5 sessions) came back to the bedroom and spent there around 25 minutes getting dressed. Then, she exited the bedroom, passed by the hall and went into the kitchen, where she always opened the curtains. She prepared her breakfast turning the coffee machine and the toaster on. Once the breakfast was ready, she had it while she was reading. Around 15 minutes later, she went to the bathroom for several minutes. Before she left home, she went into the living room to take her things, then entered into and finally exited the hall.
- Participant-3: Although her alarm clock went off at 7:25 AM, she repeatedly postponed it for 5 minutes. Then, she switched the radio on and went into the bathroom. After a while, she came back to the bedroom to get dressed. The next thing she did was to have breakfast. She left the bedroom, passed through the hall, and entered into the kitchen. While preparing her breakfast, she used the coffee machine, the fridge and the toaster. It took her around 15 minutes to have breakfast. After that, she went into the bathroom. When she was ready to leave home, she went into the living room to pick up her bag, went into the hall, opened the door, got out and closed the door.

TABLE III
EXAMPLE MATRIX TO REPRESENT THE BEHAVIOR TO GO TO SLEEP

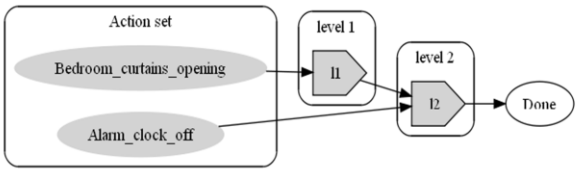
Behavior	Participant	Similarity measure
Wake up	1	0.8912
	2	0.9012
	3	0.8953
Have Breakfast	1	0.7716
	2	0.9012
	3	**0.4964**
Leave home	1	0.9481
	2	0.8819
	3	0.8883

Examining the scenarios, we can affirm that the same behavior is performed in different ways for different people: which actions are considered important, how and when they are performed, which sensors/actuators are activated to carry the actions out, etc. Even more, the behaviors are performed differently by the same user in different days. Our method is able to deal with this uncertainty involved in learning a *behavior model* that embraces all these changes.

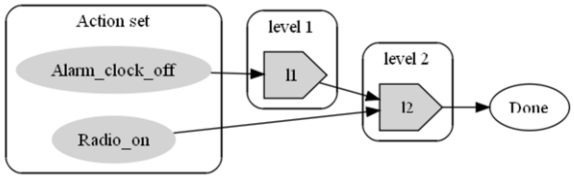
Figure 3 shows the most likely representation of the *learned Wake up behavior model* using the AND gates for each participant. Herein, it is shown that the system is capable of learning the correct *behavior model*, when the participant, the actions and the timetable are different. Moreover, analyzing the figures, we can see how the most probable actions match up with the most relevant actions provided by the participants (see Table II). Nevertheless, although we could analyze the results graphically, the calculation of the *similarity measure* will provide us a numerical way to compare the results.



(a) Participant-1



(b) Participant-2



(c) Participant-3

Figure 3 Participant-3 behavior models

Before applying the similarity measure over the learned behavior, we need to define a behavior model to be compared with. We build this behavior model using the expert knowledge indicated in Table II. For example, based on the information in Table II, the theoretical behavior models for Participant-3 are shown in Table IV.

Once we have defined the correct behavior model, we can calculate how similar the learned behavior model is with respect to the theoretical one. Table III shows the results of applying the similarity measure presented in Section IV-A. The results reveal that, in general, our method obtains very similar behavior models, reaching values around *0.9* in the similarity measure. However, there is a value that should be studied carefully. Observe that the similarity measure obtained for Participant-3 for Have Breakfast behavior (*0.4964*) is much lower than the rest of the values. Why is this value so "bad"?

Comparing the results presented in Table III with the theoretical behavior model shown in Table IV, the learned behavior model contains many actions that are not relevant according to the user's opinion for performing the *have breakfast behavior*. Nevertheless, when those so-called irrelevant (by the user) actions are examined carefully, we infer that they indeed should be performed at some moment by Participant-3 if she wants to have breakfast. For example, actions *Enter the kitchen* and/or *Exit the bedroom* should be performed to be able to have breakfast. That means that our system is capable of learning not only those actions which are considered as basics by the user, but also the ones that are unconsciously performed yet they belong to the action set of the behavior. In general, the detection of these actions could be very useful to notice small changes in the behavior performances that are imperceptible in other cases, such as, in degenerative disease supervision applications. During a degenerative disease, the capability of performing the desired behaviors usually decreases over time, introducing small changes in the usual performance, which are very difficult to detect without a continuous supervision. Hence, it can be seen that our method is capable of analyzing the user behavior over the time, just comparing the similarity measure values obtained from the learning process in different moments, reducing the required supervision and increasing the independence of the user.

As shown in the experiments, our proposal is able to handle different type of behaviors and different users. The algorithm is robust enough to converge in a correct solution independently of the inputs and the users. For more details, see [19][20], where we present an lengthy study about the converge of the algorithm for different behaviors and users.

TABLE IV
THEORETICAL BEHAVIOUR MODELS FOR PARTICIPANT 3

WAKE UP BEHAVIOR MODEL

Tasks	L1	L2	L3	L4	LU
Alarm clock off	1	0	0	0	0
Radio on	0	1	0	0	0
Rest of action	0	0	0	0	1

HAVE BREAKFAST BEHAVIOR MODEL

Tasks	L1	L2	L3	L4	LU
Coffee machine on	1	0	0	0	0
Toaster on	1	0	0	0	0
Coffee machine off	0	1	0	0	0
Toaster off	0	1	0	0	0
Rest of actions	0	0	0	0	1

LEAVE HOME BEHAVIOR MODEL

Tasks	L1	L2	L3	L4	LU
Enter the kitchen	1	0	0	0	0
Radio off	1	0	0	0	0
Enter the hall	0	1	0	0	0
Exit the hall	0	0	1	0	0
Rest of actions	0	0	0	0	1

V. CONCLUSIONS

In this paper, we have presented a novel method to learn human behavior model based on two main assumptions: a behavior performance could be determined by an ordered sequence of actions and any behavior has temporal information related to it.

First of all, we presented the concept of *behavior model*, a representation of any behavior regarding its relevant actions and the order constraints among them. Next, we have presented a method consisting of two steps: the extraction of common actions for a specific behavior and the learning process of the behavior model using the information extracted previously. The former is an algorithm based on Frequent Itemset concept, whereas the latter is a multi-agent system implemented using Learning Automata. Moreover, we apply the Fuzzy Temporal Window to manage the uncertainty of the related temporal information.

Above all, we have performed several unique experiments in the iSpace at the University of Essex, a real Ambient Intelligent Environment equipped with ubiquitous devices (sensors and actuators) which were used to collect information about the user activities. To evaluate the effectiveness of our method, we used a Similarity Measure that compares any pairs of behavior models.

For future work, we would like to improve the system to be able to recognize the user performance. Additionally, as stated before, a behavior performance is not static, and it changes along time according to multiple factors. We will also develop the adaptation system to manage these changes. As an improvement to the system, we will develop a Type-2 fuzzy based system to manage the uncertainty of those actions that can vary during a behavior performance.

ACKNOWLEDGMENT

This work has been supported by the project TIN2009-14538-C02-01, I+D+I national program from Spain Government. The work has been also supported in part by the King Abdel Aziz University within the scope of the Scale up project. We would like to acknowledge the input of Areej Malibari, Mohamed Al Haddad and Danniyal Al Ghazawwi.

REFERENCES

[1] K. Ducatel, M. Bogdanowicz, F. Scapolo, J. Leijten, and J. C. Burgelman, "Scenarios for Ambient Intelligence in 2010," IST Advisory Group, Tech. Rep., Feb. 2001. [Online]. Available: ftp://ftp.cordis.lu/pub/ist/docs/istagscenarios2010.pdf

[2] M. Philipose, K. P. Fishkin, M. Perkowitz, D. J. Patterson, D. Fox, H. Kautz, and D. Hahnel, "Inferring activities from interactions with objects," IEEE Pervasive Computing, vol. 3, no. 4, pp. 50–57, 2004.

[3] L. Liao, D. J. Patterson, D. Fox, and H. Kautz, "Learning and inferring transportation routines," Artificial. Intelligence, vol. 171, no. 5-6, pp. 311–331, 2007.

[4] M. Rantz, M. Aud, G. Alexander, D. Oliver, D. Minner, M. Skubic, J. Keller, Z. He, M. Popescu, G. Demiris, , and S. Miller, "Tiger place: An innovative educational and research environment," in AAAI in Eldercare: New Solutions to Old Problems, 2008.

[5] C. Nugent, M. D. Mulvenna, F. Moelaert, B. Bergvall-Kereborn, F. Meiland, D. Craig, R. Davies, A. Reinersmann, M. Hettinga, A.-L. Andersson, R.-M. Dries, and J. Bengtsson, "Home-based assistive technologies for people with mild dementia," in Pervasive Computing for Quality of Life Enhancement, vol. 4541, 2007, pp. 63–69.

[6] P. Rashidi and D. J. Cook, "Keeping the resident in the loop: adapting the smart home to the user," Trans. Sys. Man Cyber. Part A, vol. 39, no. 5, pp. 949–959, 2009.

[7] R.-M. Droes, M. Mulvenna, C. Nugent, D. Finlay, M. Donnelly, M. Mikalsen, S. Walderhaug, T. v. Kasteren, B. Krose, S. Puglia, F. Scanu, M. O. Migliori, E. Ucar, C. Atlig, Y. Kilicaslan, O. Ucar, and J. Hou, "Healthcare systems and other applications," IEEE Pervasive Computing, vol. 6, no. 1, pp. 59–63, 2007.

[8] S. Park and H. Kautz, "Privacy-preserving recognition of activities in daily living from multi-view silhouettes and rfid-based training," in AAAI Fall 2008 Symposium on AI in Eldercare: New Solutions to Old Problems, Washington, DC, November 7 - 9, 2008.

[9] D. Wilson, D. Wyaat, and M. Philipose, "Using context history for data collection in the home," in Pervasive, vol. 3468, 2005.

[10] J. Boger, J. Hoey, P. Poupart, C. Boutilier, G. Fernie, and A. Mihailidis, "A planning system based on markov decision processes to guide people with dementia through activities of daily living," IEEE Trans Inf Technol Biomed, vol. 10, no. 2, pp. 323–333, 2006.

[11] G. Acampora and V. Loia, "Fuzzy control interoperability and scalability for adaptive domotic framework," IEEE Transactions on Industrial Informatics, vol. 1, no. 2, pp. 97 – 111, may. 2005.

[12] F. Doctor, H. Hagras, and V. Callaghan, "A fuzzy embedded agent-based approach for realizing ambient intelligence in intelligent inhabited environments," IEEE Transactions on Systems, Man, and Cybernetics, Part A, vol. 35, no. 1, pp. 55–65, 2005.

[13] H. Hagras, V. Callaghan, M. Colley, G. Clarke, A. Pounds-Cornish, and H. Duman, "Creating an ambient-intelligence environment using embedded agents," IEEE Intelligent Systems, vol. 19, no. 6, pp. 12– 0, 2004.

[14] P. Rashidi and D. Cook, "Home to home transfer learning," in Proceedings of the AAAI Plan, Activity, and Intent Recognition Workshop, 2010.

[15] D. Zhang, D. Gatica-Perez, S. Bengio, I. McCowan, and G. Lathoud, "Modeling individual and group actions in meetings with layered hmms" IEEE Trans. on Multimedia, vol. 8, pp. 509–520, 2004.

[16] X. Meng, K. K. Lee, and Y. Xu, "Human driving behavior recognition based on hidden markov models," IEEE International Conference on Robotics and Biomimetics, vol. 0, pp. 274–279, 2006.

[17] J. A. Ward, P. Lukowicz, G. Troster, and T. E. Starner, "Activity recognition of assembly tasks using body-worn microphones and accelerometers," IEEE Trans. Pattern Anal. Mach. Intell., vol. 28, no. 10, pp. 1553–1567, 2006.

[18] G. Acampora, M. Gaeta, V. Loia, A.V.Vasilakos, "Interoperable and adpative fuzzy services for ambient intlligence applications". ACM Transactions on Autonomous and Adaptive Systems (TAAS) vol. 5 (2), pp 8:1-8:26, 2010.

[19] M.P. Cuéllar, M. Ros, M. Delgado, and A. Vila, "Detection of abnormal human activities by means of learning automata," IEEE Trans. on Systems, Man and Cybernetics, Part A., submitted for publication.

[20] M. Ros, M. Cuéllar, M. Delgado, and A. Vila, "Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows," Information Sciences, vol. -, no. 0, pp. –, 2011, in press. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0020025511005408

[21] D. T. Anderson, M. Ros, J. M. Keller, M. P.Cuéllar, M. Popescu, M. Delgado, and A. Vila, "Similarity measure for anomaly detection and comparing human behaviors," International Journal of Intelligent Systems, submitted for publication.

[22] M. Ros, M.P. Cuéllar, M. Delgado, A. Vila, D. T. Anderson, J. M. Keller, and M. Popescu, "Linguistic summarization of long-term trends for understanding change in human behavior," in Fuzzy Systems (FUZZ), 2011 IEEE International Conference on, June 2011, pp. 2080 – 2087.

[23] M. Ros, M. Delgado, and A. Vila, "Fuzzy method to disclose behavior patterns in a tagged world," Expert Syst. Appl., vol. 38, pp. 3600–3612, April 2011. [Online]. Available: http://dx.doi.org/10.1016/j.eswa.2010.09.014

[24] M. Delgado, M. Ros, and M. Amparo Vila, "Correct behavior identification system in a tagged world," Expert Syst. Appl., vol. 36, pp. 9899–9906, August 2009. [Online]. Available: http://dl.acm.org/citation.cfm?id=1528927.1529066

[25] P.-N. Tan, M. Steinbach, and V. Kumar, Introduction to Data Mining, (First Edition). Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2005.

[26] M. A. L. Thathachar and P. S. Sastry, "Varieties of learning automata: an overview," IEEE Transactions on Systems, Man, and Cybernetics, Part B, vol. 32, no. 6, pp. 711–722, 2002.

[27] A. S. Poznyak and K. Najim, Learning Automata and Stochastic Optimization. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1997.

[28] D. Anderson, J. Bezdek, M. Popescu, and J. Keller, "Comparing fuzzy, probabilistic, and possibilistic partitions," Fuzzy Systems, IEEE Transactions on, vol. 18, no. 5, pp. 906 –918, oct. 2010

4. Understanding change in human behavior. A summarization tool

4.1. Similarity measure for anomaly detection and comparing human behaviors

The journal paper associated to this part is:

- Derek T. Anderson and María Ros and James M. Keller and Manuel P.Cuéllar and Mihial Popescu and Miguel Delgado and Amparo Vila. Similarity measure for anomaly detection and comparing human behaviors. Submitted to International Journal of Intelligent Systems.
  - Status: **Accepted**.
  - Impact Factor (JCR 2010): 1.331
  - Subject Category: Computer Science, Artificial Intelligence. Ranking 57 / 108 (Q3).

International Journal of Intelligent Systems

Similarity measure for anomaly detection and comparing human behaviors

Derek T. Anderson, María Ros, James M. Keller, Manuel P. Cuéllar, Mihail Popescu, Miguel Delgado and Amparo Vila

Abstract

Herein, we put forth a new similarity measure for anomaly detection and for comparing human behaviors based on the theories of learning automata, comparison of soft partitions and temporal probabilistic order relations. In particular, focus is placed on monitoring individuals in a home setting for their own well-being. This work is a high-level investigation focused on the structure of human behavior. Examples demonstrate the utility of this approach for (1) understanding the similarity of pairs of behaviors for an individual (or alternatively between individuals) and (2) detecting significant change between changing behavior and a baseline model. In the context of eldercare, significant change in behavior can be a precursor to cognitive and/or functional health related problems. Simulated resident behavior is used to show different scenarios and the response of the proposed measure.

Index Terms

comparing soft partitions, learning automata, human behavior, anomaly detection.

Acknowledgements and Affiliations

This work is supported by NSF under grant EIA-0325641 and by TIN2009-14538-C02-01 in the I+D+I national program from the Spain Government. *D. T. Anderson is with the Electrical and Computer Engineering Department, Mississippi State University, Mississippi State, MS 39762 (phone number: 662-325-3912; fax: 662-325-2298; anderson@ece.msstate.edu). J. M. Keller is with the Electrical Engineering Department, University of Missouri, Columbia, MO 65211 USA (kellerj@missouri.edu). M. Popescu is with the Health Management and Informatics Department, University of Missouri, Columbia, MO (popescum@missouri.edu). M. Ros, M. Cuéllar, M. Delgado, and A. Vila are with the Ciencias de la Computación e Inteligencia Artificial, Universidad de Granada, Granada, Spain (marosiz@decsai.ugr.es, manupc@decsai.ugr.es, mdelgado@ugr.es and vila@decsai.ugr.es).*

I. Introduction

The goal of this research is the monitoring of individuals for their well-being. Elderly people living alone have a set of vulnerabilities that can be ameliorated by a monitoring system. A large concern for the elderly is adverse event detection, e.g., fall recognition [1-4]. Adverse events generally occur over short time periods (seconds or minutes). The method introduced here is designed to identify changes in patterns of behavior over longer periods of time (days, weeks or months). Identification of changes in behavior over longer time periods is significant in the sense that it can be used to help predict cognitive and/or functional decline that affects overall quality of life [5-7]. If significant change in human behavior can be recognized early, then caregivers can intervene and take action in order to help avoid or postpone catastrophic events.

No single sensor or algorithm solves all behavior analysis problems. While some researchers focus on the modality of sensing, i.e., wearable sensors [8], RFID tags [9], etc., others explore computational problems independent of a specific sensor, i.e., *probabilistic graphical models* (PGMs) [10], *learning automata* (LA) [11,12], linguistic summarization [1-3], etc. Once the sensors and computational tools are selected, focus can be placed on addressing higher level behavior questions. In this article, we focus on the high level task of comparing the structure of different human behaviors and also on looking for significant change in a behavior model that has been adapted over time according to elder activities.

This article is organized as follows. In section 2, we summarize prior research related to this work. In section 3, the new similarity measure for comparing human behavior is introduced. Subsequent sections are in-depth investigations into the utility of the proposed measure for behavior analysis with a focus on the elderly.

II. Related Work

II.A State of the Art

In recent years, multiple Eldercare projects have emerged that combine the benefits of Engineering and technology with healthcare. The topic of Eldercare is real and important given population sizes and the fact that modern medicine continues to extend lifespan. One such project is the *Center for Eldercare and Rehabilitation Technology* (CERT) [13] focused on the TigerPlace facility [14]. This effort is a large interdisciplinary collaboration between electrical and computer engineers, gerontological nurses, social workers, physical therapists and others at the University of Missouri. This group has installed multiple sensing technologies in the apartments of elders living at TigerPlace. One goal of these technologies includes assisting residents with living in place as independently as possible. Other efforts include the MavHome project [15], iDorm [16], CASAS [17] and the Georgia Tech Aware Home [18]. In addition, in [19] eight other relevant projects in this field are summarized.

While the field is working to solve the real sensor based problems, we can abstract the challenge and simultaneously address higher level behavior tasks. That is, instead of focusing on which particular sensors to use and how, rather we look to identify computational procedures capable of representing and recognizing different normal and abnormal behaviors. One such inquiry is that of Philipose et al [20]. They define a method based on *Activities of Daily Living (ADL)* inference [21] using a probabilistic inference engine for the recognition of complex activities and a sequence of used objects as representation. In the literature, we also find many works based on *hidden Markov models* (HHMs) [22,23]. The use of stochastic methods is motivated by their capacity of modeling different scenarios and possible ways to finish the human tasks correctly.

Naeem et al. present an approach based on *multiple behavioral HMMs* (MBHMMs) [24]. They generate a multiple HMMs for each variation of an action. Their system is able to determine the behavior performed even if it has not been completed yet. An extensive study of the benefits and drawbacks associated with different probabilistic techniques for human activity recognition is presented in [25]. In [26], the authors propose an AmI fuzzy computing system, which is a multilayer architecture to learn and model human behavior. They attempt to join the advantages of a multi-agent based framework with fuzzy control techniques to improve the recognition process and continuously enrich the knowledge about the user and environment.

On the other hand, temporal processing has recently emerged as an interesting starting point in behavior learning and recognition. For instance, in [27], the authors propose a learning machine to identify temporal relations among daily activities in a smart home. These relations are used for prediction, decision making, and anomaly detection of ADL. Other similar proposals are presented in [28,29]. These methods identify the frequent/normal patterns using a fuzzy temporal window and then the task switches to one of recognizing user activities considering this information. This method combines techniques of data mining to generate a representation of a behavior and regular grammars to recognize activity.

In the field of behavior recognition, many tend to assume that behavior is a fixed concept. This means that a resident cannot change their behavior over time, clearly not the case for elderly persons. Systems must be able to adapt their behavior knowledge to address new situations. For example, in [7,19] Rashidi et al. include an adaptive activity mining component to find changes in activity patterns using a hierarchical behavior model with probabilistic methods and a feedback-based learning component in charge of finding relevant changes in the activities. Among the projects based on agent systems, in iDorm [6, 20] the authors developed AOFIS, a

life-long fuzzy learning and adaptation technique. Mainly, AOFIS is an unsupervised data-driven one-pass approach for extracting fuzzy rules and membership functions from data. On the other hand, Ros et al. presented a self-adaptation method based on LA [12]. The key aspects of this approach are; the separation of the behavior model from the behavior learning procedure (to avoid shrinking problems produced by the learning algorithms, as happens with HMMs), the modularization of the approach (which allows the migration of a learnt behavior to other contexts or environments) and the online learning and adaptation capability by means of LA and fuzzy temporal windows to learn the changes produced either in the user habits or the environment.

II.B Modeling Human Behavior

Comparing behavior depends in part on the representation. A popular approach in human activity analysis is PGMs [10], e.g., HMMs [30], *hierarchical HMMs* (HHMMs) [31], etc. PGMs are probabilistic models in which a graph denotes conditional independence structure between random variables. In this article, focus is placed on a computational framework investigated by our research team in [11, 12]. The choice of representation is a sequence of cascaded AND gates with an underlying probability distribution that characterizes temporal dependency requirements over a set of actions for the different ways in which a user can perform a behavior. The work in [11, 12] is summarized in this subsection.

An abnormal situation occurs when a person does not perform the behavior as expected, i.e., forgetting things, performing actions in a dangerous order, etc. Example 1 demonstrates the role of detecting abnormal situations in the context of the goals of this work.

Example 1: Suppose that every morning a resident wakes up, goes to the bathroom, takes his pajamas off, takes a shower, washes his face, gets dressed and returns to the bedroom. All of

these actions could be embraced as a single behavior concept, "morning routine". However, suppose that for a period of time the person does not take a shower. This is considered as a change in the "morning routine" behavior. This abnormal situation needs to be identified and the behavior model needs to be updated accordingly.

In example 1, two important concepts are introduced, *action* and *behavior*. Informally, an action is the output of a sensor or a low-level algorithm interpreting sensor firings, e.g., the refrigerator door is closed, the front door is opened, a person is sitting on the couch, etc. Behavior is the set of actions and the order in which the actions need be performed. Formally, these concepts are characterized as follows.

Definition 1 (Human Action) *[11, 12]: A human action h_i is a triple $h_i = < l_i, s_i, t_i >$, where l_i is a label used to identify the action, s_i is a sensor event and t_i is a time stamp in which the action occurs.*

Definition 2 (Behavior) *[11, 12]: A human behavior B_i is a quadruple $B_i = < H, R_i, C_i, \prec >$, where $H = \{h_1, h_2, \dots, h_n\}$ is a set of available human tasks, $R_i = \{r_1^i, r_2^i, \dots, r_{|R_i|}^i\} \subseteq H$ is the subset of available human tasks that are required to complete the behavior, and C_i is a set of temporal constraints between the tasks in R_i defined over a partial order relationship operator $\prec$. The partial order relationship $\prec$ defines the temporal dependency requirements over the actions of a behavior, and its formal definition is*

$$r_a^i \prec r_b^i \leftrightarrow t_a \prec t_b.$$

Thus, a behavior is represented as a set of relevant actions, which maintain a specific order of performance between them. In [11,12], we demonstrate that a behavior can be represented as a

set of cascaded AND gates. An action in the system corresponds to an input of a specific AND gate. Each input will be true if the action has been performed, or false if not. On the other hand, although all the tasks in the H set may be taken into consideration for a specific behavior, only those actions that are included in the R set are really required for it. To handle this situation, the authors introduce a discard-level to which those actions that are irrelevant are assigned.

Besides relevant actions, this representation contains more knowledge about a specific behavior, such as the order relationship between diverse actions. As previously pointed out, a behavior is designed as a set of actions that maintains an order relationship between them. The level-representation for a behavior simulates the order relationship between the actions in such a way that those actions that are the input to an AND gate in Level_n have to be done previously to those that are the inputs to the following AND gate, in Level_{n+1}. Fig. 1 shows the sequence of AND gates obtained from example 1.

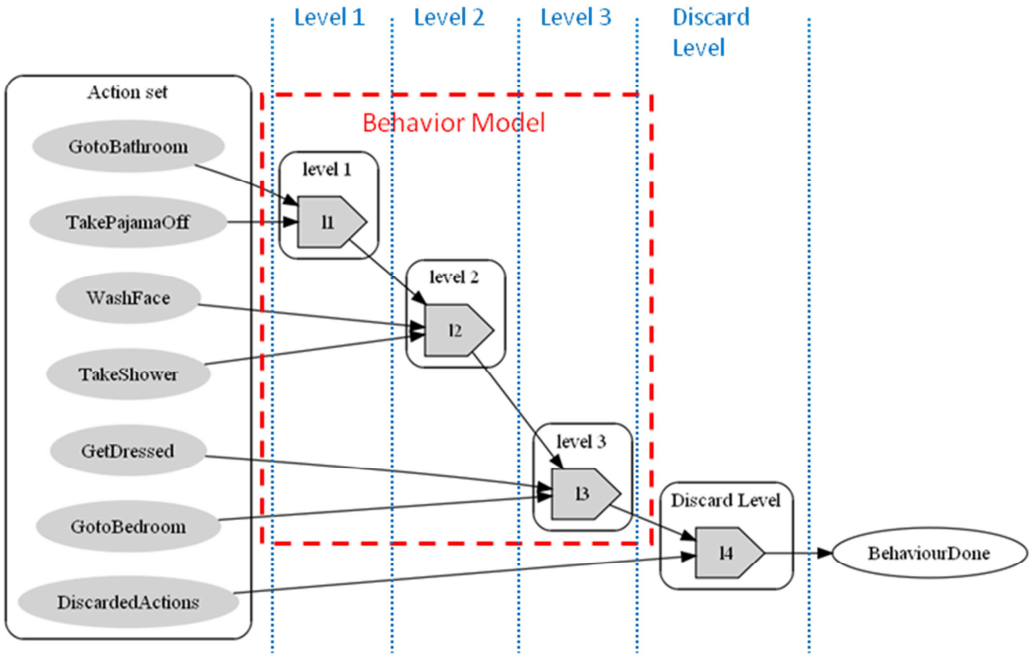


Fig. 1. Morning routine behavior model (corresponding to example 1.1).

Nevertheless, most behaviors have more than one reasonable method of performance. Hence, there may be several combinations of the same set of actions for a behavior, i.e., that some tasks may be performed at different levels of the structure without loss of meaning. For instance, in example 1, the action *wash face* could be performed at either level 2 or at level 3 and still constitute a legitimate Morning Routine. To deal with this disparity, in [11, 12] a probability matrix, $P(i,j)$, is proposed, whose rows, $1 \leq i \leq |H|$, are actions, the H set, and columns, $1 \leq j \leq |L|$, are available levels in the sequence of ANDs. Each element of P stands for the probability of an action might be done in a particular level. Therefore, the behavior model represents, for each action, its most expected level of activation. Next, keeping on with the prior example, a possible matrix is presented below.

Continuation of example 1: Suppose we want to study a "morning routine" behavior. In addition, suppose that we have the actions presented in example 1. Then, if the maximum number of levels was fixed to four, a possible P matrix to represent the model presented in Fig. 1 could be

$$\begin{bmatrix} & L_1 & L_2 & L_3 & L_4 & L_5(discard\ level) \\ go\ to\ bathroom & 0.7 & 0.2 & 0 & 0 & 0 \\ take\ pajama\ off & 0.6 & 0.4 & 0 & 0 & 0 \\ take\ shower & 0.2 & 0.7 & 0.1 & 0 & 0 \\ wash\ face & 0 & 0.5 & 0.3 & 0 & 0.2 \\ get\ dressed & 0 & 0.2 & 0.6 & 0.2 & 0 \\ go\ to\ bedroom & 0 & 0.1 & 0.5 & 0.3 & 0.1 \\ go\ to\ kitchen & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Among the advantages of our proposal, the independence on the method used for representing the behavior is suggested as the most significant one. This separation allows us to achieve better system modularization and to use a higher number of learning methods for the same activity model [11]. Concretely, the method used to learn the behavior model is not studied here. Our

objectives consist of extracting conclusions about what the behavior model represents, how it evolves amid activity changes and how it differs from others behaviors including past ones.

In [11, 12], we proposed a multi-agent LA approach to model human behavior. The outcomes of its different processes are the input to the method developed here. LAs may be considered as "building blocks" and be included in more complex designs. Therefore, a LA is an agent in a multi-agent environment, so that multi-agent theory can be used to build systems of LAs. For training, we use a team of LAs [32,33,34], specifically *finite action learning automata* (FALA) [33]. We propose another team of LAs, the *variable action learning automata* (VALA) [11], for the recognition task [12]. The use of VALA is justified because different action sequences can represent the same valid behavior, so that the probability matrix of matching actions and cascade levels evolves with time. VALA manages this feature by means of inclusion of time in the automaton action selection. Therefore, in the recognition stage we use the probability hypermatrix $P(t, i, j)$, i.e., the probability of action i being executed at level j when the t first cascaded ANDs return true.

In this paper, we assume that the team of LAs has been trained for specific behaviors. Both algorithms use a modification of classic LA reinforcement rules. Those rules will be delimited by a learning and adaptation rate respectively, which establishes the strength of processes. For the learning process, we propose a learning algorithm based on traditional pursuit schemes [11][12]. Using that learnt behavior model, we propose a recognition and adaptation process in function of the current user activity. The algorithm tries to detect when a behavior is performed incorrectly, determining both uncompleted behaviors and violated relationship constraints. In this case, the use of VALA lets a person perform actions in different levels without violation detection. Once

the recognition process has finished, the learnt probabilities of the behavior model are updated in function of the level in which the action has been detected.

II.C Comparing Soft Partitions

The next tool used in this investigation comes from the field of clustering. Let $O = \{o_1, \ldots, o_n\}$ denote n objects. When each object in *O* is represented by a (column) vector *x*, the set $X = \{x_1, \ldots, x_n\} \subset \Re^p$ is called an *object data representation* of *O*. When each object in $o_i \in O$ has a *physical label*, *O* is a set of *labeled data*; otherwise, *O* is unlabeled. Let integer *c* denote the number of groups (clusters), $1 < c < n$. Clustering in unlabeled data is the assignment of one of four types of labels to each object in *O*. The label vectors of the objects are the columns of *c*-partitions of *O*, which are sets of (cn) values $\{u_{ik}\}$ that can be conveniently arrayed as $(c \times n)$ matrices, $U = [u_{ik}]$. These labels can be crisp, probabilistic, fuzzy or possibilistic.

When clustering produces more than one candidate for partitioning of a finite set of objects *O*, there are two approaches to validation, i.e., selection of a "best" partition, and implicitly, a best value for *c*. First, one can use an internal index that evaluates each partition separately. Second, one can compare pairs of candidates to each other or to a reference partition that purports to represent the "true" cluster structure in the objects. In [35], we generalized many of the classical indices that have been used with outputs of crisp clustering algorithms so that they are applicable for candidate partitions of any type, i.e., crisp or soft, soft comprising the fuzzy, probabilistic and possibilistic cases. In particular, we concentrated on the Rand index and its modifications. We showed that our extension of the Rand index is $\boldsymbol{O}(n)$.

Let *U* and *V* be two crisp partitions of *O*. *U* and *V* need not possess the same number of clusters, $r \neq c$. The four classical combinations for pairs of objects from $O \times O$ in clusters of *U* and *V*

are: (a) paired in *U* and *V*; (b) not paired in *U* nor in *V*; (c) paired in *V* but not in *U*; and (d) paired in *U* but not in *V* [36]. The comparison of *U* to *V* with a similarity measure s begins with the $r \times c$ contingency matrix $N = UV^T$ that contains the counts of the number of occurrences of each of the four types over the $n(n-1)/2$ distinct, unordered pairs in $O \times O$. Entry n_{ij} is the number of objects common to subgroups U_i and V_j. The building blocks of many similarity measures for $s(U,V)$ are expressions (a)-(d) below. These four calculations count the number of occurrences amongst the $n(n-1)/2$ pairs of each of the four types of unordered pairs:

$$a = \frac{1}{2}\sum_{i=1}^{r}\sum_{j=1}^{c} n_{ij}(n_{ij}-1);$$

$$d = \frac{1}{2}\left(n^2 + \sum_{i=1}^{r}\sum_{j=1}^{c} n_{ij}^2 - \left(\sum_{i=1}^{r} n_{i\bullet}^2 + \sum_{j=1}^{c} n_{\bullet j}^2\right)\right);$$

$$b = \frac{1}{2}\left(\sum_{j=1}^{c} n_{\bullet j}^2 - \sum_{i=1}^{r}\sum_{j=1}^{c} n_{ij}^2\right);$$

$$c = \frac{1}{2}\left(\sum_{i=1}^{r} n_{i\bullet}^2 - \sum_{i=1}^{r}\sum_{j=1}^{c} n_{ij}^2\right).$$

The sums $(a+d)$ and $(b+c)$ are usually interpreted, respectively, as (the total number of) *agreements* and *disagreements* between *U* and *V*. In [35], we tabulate a (non-exhaustive) list of 14 coefficients that have been proposed for $s(U,V)$ based on functions of *a, b, c* and *d*; Sokal and Sneath list many others [37]. In addition, we also present a scaling value in the case of possibilistic labels. Here, we focus on the Rand, the classical form of which is

$$s_R(U,V) = (a + d) / (a + b + c + d).$$

The measure $s(U,V)$ can be used to address a variety of clustering problems. For example, it can be used to compare a clustering algorithm output to a reference partition that purports to represent the “true cluster structure”, measure the effect of noise or missing data with respect to a specific clustering algorithm, study the convergence of a clustering algorithm at successive iterations, perform prediction (regression), as well as comparing different clustering algorithms.

III. Proposed Similarity Measure

Using the theories discussed in section 2, a similarity measure is now introduced for anomaly detection and for comparing pairs of different behaviors from trained LAs. The measure includes three different components based on (1) comparing partitions, (2) subsethood of non-discarded actions, and the (3) similarity of partial order relations.

III.A Comparing Soft Partitions

The first component of this work is the determination of similarity between pairs of behaviors according to the strength of actions grouping into similar levels. The key is to think of actions as data points and levels as groups (clusters). Specifically, $s(U,V)$ is used, where U and V are behavior probability matrices acquired using the procedures described in section 2, i.e., $U(i,j)$ is the probability that action i belongs to level j.

It is important to note that our proposed $s(U,V)$ is a measure not a metric. Hullermeier et al. previously put forth a (similarity) pseudo-metric for the specific case of fuzzy partitions. However, their index is only a full metric on a (very) small subset of cases one might encounter in practice (the set of "normal partitions" - or partitions that have at least one 1 in each row of the partition) [38]. A comparison between our method and Hullermeier et al. is presented in [35,39].

Because our $s(U,V)$ is a measure and not a metric, some properties, such as $s(U,U) = 1$ are not guaranteed. While an algorithm can use $s(U,V)$ "as is", we scale $s(U,V)$ in order to produce a value in the interval [0,1] that is more appealing to human interpretation. The formula used to compare different behaviors is

$$s_{cp}(U,V) \;=\; s(U,V) \,/\, maximum(\, s(U,U)\,, s(V,V)\,).$$

III.B Subsethood of Non-Discarded Actions

While $s_{cp}(U,V)$ measures one notion of similarity between a pair of behaviors, it inflates (in comparison to what a human might desire) the value in part because *U* and *V* generally share many actions in the discard level (cluster). The discard level cannot just be simply removed or the resulting matrix is no longer a probabilistic partitioning. Thus, we introduce a scaling factor to reduce the effect of the discard level (cluster).

First, the most likely level, $L_{i,max} = argmax_j\, P(i,j)$, for each action is found. Next, the crisp sets, $R_U \subseteq H_U$ and $R_V \subseteq H_V$, of actions in which $L_{i,max}$ does not equal the discard level is found. The matrix P contains evidence about the importance of actions to a behavior and the criteria of picking all actions that are not most likely in the discard level is feature selection. The subsethood measure over R_U and R_V is

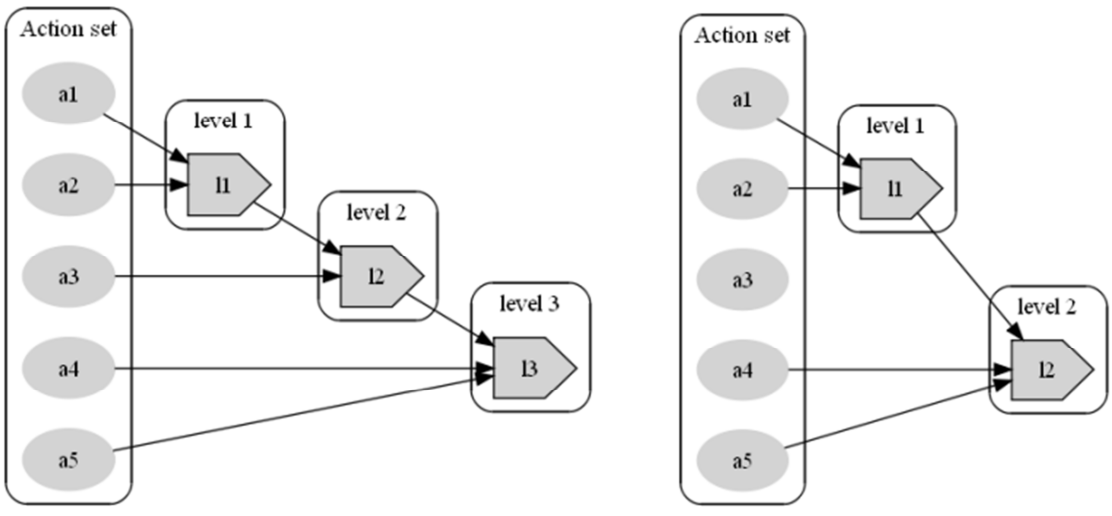
$$s_{sub}(U,V) \;=\; |R_U \wedge R_V| \,/\, |R_U \vee R_V|,$$

where $\wedge$ is intersection, $\vee$ is union and $|\bullet|$ is cardinality. The new similarity is

$$s_{cp\_sub}(U,V) \;=\; s_{cp}(U,V) \,\times\, s_{sub}(U,V).$$

III.C Similarity of Partial Order Relations

The above definition of $s_{cp\_sub}(U,V)$ does not take into account the similarity between the partial order relations. That is, two behaviors may share a set of non-discarded actions and have a strong similarity of actions grouping into levels (clusters), but the order of levels is not enforced. The value $s_{cp\_sub}(U,V)$ is desirable in one respect because it is flexible enough to allow a cluster of actions to *shift* around. For example, consider the scenario of adapting the behavior given in Fig. 2(a) into Fig. 2(b). Table 1 is the respective set of corresponding P matrices. Specifically, the constraint that action 3 be performed before actions 4 and 5 is removed. Therefore, actions 4 and 5 still "cluster" into a similar level, however the level has "shifted" around. Accordingly, the cluster does not have the same level identifier anymore, it was level 3 in Fig. 2(a) and it is level 2 in Fig. 2(b). Hence, to truly compare two behaviors, the partial order relations need to be factored into the measure. We resolve this by counting the number of disagreements between U and V and weight the disagreements by their individual importance.



(a) (b)

Fig. 2. Example hand crafted behaviors. Behavior (a) corresponds to tables 1(a) and 1(c) while behavior (b) corresponds to table 1(b).

Table 1. Hand crafted P matrices. Table 1(a) is referred to as $U_{(a)}$, 1(b) is $U_{(b)}$, and 1(c) is $U_{(c)}$ in the text.

Fig 2(a)	L1	L2	L3	Discard
A1	0.51	0.49	0	0
A2	1	0	0	0
A3	0	0.8	0	0.2
A4	0	0	0.9	0.1
A5	0	0	0.6	0.4

(a)

Fig 2(b)	L1	L2	L3	discard
A1	0.51	0.49	0	0
A2	1	0	0	0
A3	0	0	0	1
A4	0	1	0	0
A5	0	0.6	0	0.4

(b)

Fig 2(a)	L1	L2	L3	discard
A1	0.8	0.1	0.1	0
A2	1	0	0	0
A3	0	0.9	0	0.1
A4	0	0	1	0
A5	0	0	0.9	0.1

(c)

Table 2. Partial order relations for the behaviors specified in table 1. Here, S represents *Same Time*, B corresponds to *Before*, and A represents *After*.

Table 1(a)	A1	A2	A3	A4	A5
A1	S	S	B	B	B
A2	S	S	B	B	B
A3	A	A	S	B	B
A4	A	A	A	S	S
A5	A	A	A	S	S

(a)

Table 1(b)	A1	A2	A3	A4	A5
A1	S	S	Na	B	B
A2	S	S	Na	B	B
A3	Na	Na	Na	Na	Na
A4	A	A	Na	S	S
A5	A	A	Na	S	S

(b)

Table 1(c)	A1	A2	A3	A4	A5
A1	S	S	B	B	B
A2	S	S	B	B	B
A3	A	A	S	B	B
A4	A	A	A	S	S
A5	A	A	A	S	S

(c)

First, a $|H|x|H|$ matrix, $E_{U'}$, is created for behavior *U*'. For each action, the level with the greatest probability (most likely explanation) is found, i.e., $L_{i,max}$. Next, $E_{U'}(i,j)$ is assigned a label from the set $\{B, A, S\}$, where 'B' (before) means $L_{i,max} < L_{j,max}$, 'A' (after) means $L_{i,max} > L_{j,max}$, and 'S' (same) means $L_{i,max} = L_{j,max}$. Cell $E_{U'}(i,i) = 'S'$ and if $E_{U'}(i,j)$ is

'B' ('A') then $E_{U'}(j,i)$ is 'A' ('B'). Table 2 is an example relationship matrix for the example in Fig. 2 and P matrices in Table 1.

A simple method of measuring similarity of partial order relations between the two behaviors is to count the number of disagreements in $E_{U'}$ and to weight that number by the total number of partial order relations. However, this is a harsh (binary) method of penalizing disagreements. Remember, $L_{i,max}$is the most likely level that action i belongs to. If the associated probability is 1, it is certain that action i belongs to $L_{i,max}$, and so, a disagreement is significant. However, if the value is smaller, e.g., 0.51, and the next largest probability is close in value, e.g., 0.49, then a disagreement is less significant, meaning it was never very certain to which level the action belongs. We characterize this uncertainty by calculating an importance value per action. For action i, $P_{U',i}$ measures the ability to decide that action i belongs to some particular level,

$$P_{U',i} = \max_{j} P_{U'}(i,j) - \max_{\substack{k \\ j \neq k}} P_{U'}(i,k).$$

The significance weighted calculation of the similarity of partial order relations, $s_{por}(U,V)$, is based on the set of off-diagonal upper triangular indices of $E_{U'}$, i.e., all *(i,j)* where $j > i$. In particular, the measure uses only the subset of indices, I', in which actions i and j are not most likely discarded. Based on $P_{U',i}$ and $E_{U'}$(i, j), the similarity of partial order relations is defined as

$$s_{por}(U,V) = 1 - \left(\sum_{(i,j)\in I'} \emptyset\left(E_U(i,j), E_V(i,j)\right)\right)/|I'|,$$

$$\emptyset\big(E(i,j), E_V(i,j)\big) =$$

$$\begin{cases} 0 & if\ E_U(i,j) = E_V(i,j) \\ min(P_{U,i}, P_{V,i}) & else \end{cases},$$

and the final measure for comparing human behaviors is

$$s_{final}(U,V) = s_{cp}(U,V) \times s_{sub}(U,V) \times s_{por}(U,V).$$

Thus, $s_{final}(U,V)$ measures how well actions group into clusters, $s_{cp}(U,V)$, share a set of non-discarded actions in common, $s_{sub}(U,V)$, and have similar probabilistic partial order relations, $s_{por}(U,V)$.

III.D Numeric Example

In order to understand the measure, we present an example. Table 1 contains three behaviors designed by hand by the authors to show the inner workings of $s_{final}(U,V)$. The component measure values for $U_{(a)}$ and $U_{(b)}$ are $s_{cp}(U_{(a)},U_{(b)})$ = 0.89, $s_{sub}(U_{(a)},U_{(b)})$ = 0.8, $s_{por}(U_{(a)},U_{(b)})$ = 1, and $s_{final}(U_{(a)},U_{(b)})$ = 0.71, while $U_{(a)}$ and $U_{(c)}$ result in $s_{cp}(U_{(a)},U_{(c)})$ = 0.9, $s_{sub}(U_{(a)},U_{(c)})$ = 1, $s_{por}(U_{(a)},U_{(c)})$ = 1, and $s_{final}(U_{(a)},U_{(c)})$ = 0.9. The result is what we expect, $s_{final}(U_{(a)},U_{(c)}) > s_{final}(U_{(a)},U_{(b)})$. Specifically, $s_{cp}(U_{(a)},U_{(c)})$ is only slightly greater than $s_{cp}(U_{(a)},U_{(b)})$, but $s_{sub}(U_{(a)},U_{(c)})$ is larger than $s_{sub}(U_{(a)},U_{(b)})$ because an action is removed. Lastly, $s_{por}(U_{(a)},U_{(c)}) = s_{por}(U_{(a)},U_{(b)})$ = 1. Even though an action was removed, the remaining set of partial order relations is the same, producing the same high value.

IV. Experiments

We previously developed an algorithm that takes action and temporal constraints and generates synthetic action data sets to test a wide range of human behavior scenarios [12]. These data sets are used in both learning and recognition.

Both normal and abnormal behaviors are included in the synthetic data sets. According to Definition 2 [11][12], normal behavior is defined as a sequence of relevant tasks (actions that

have to be performed in order to recognize a behavior as correct) of non-specified length and a set of temporal constraints between the aforementioned tasks. On the other hand, abnormal behavior is an action sequence in which constraints are violated. Therefore, our synthetic data includes both correct and incorrect sequences of tasks for the studied behavior. In addition, every sequence of actions might contain irrelevant actions, which are known as *rubbish actions*. These actions are not important for the correct performance of the behavior, but the user might do it among the relevant actions and the synthetic data generation should take it into consideration.

The following experiments study five different common elderly behaviors: Wake Up (B_1), Morning Routine (B_2), Have Breakfast (B_3), Kitchen-Based Coffee Break (B_4) and Kitchen and Living Room-Based Coffee Break (B_5). This set of behaviors is selected because a few are similar, i.e., the coffee breaks and possibly breakfast, while the others are very dissimilar, i.e., wake up and morning routine. The constraints used are shown in Table 3. The set of actions are provided in Table 4. We generated the following collection to explore how the proposed measure responds to different scenarios.

- DS_1: The elder always performs the behavior correctly, i.e., normal behaviors. Herein, DS_1 is used to learn the initial baseline model. It has 365 days worth of this behavior sequence.

- DS_2: A catastrophic event, e.g., fall, occurred. This event is severe and it results in restricted mobility. After the event, an elder's routine has changed significantly as a result. This data set contains both normal and abnormal behaviors. There are 149 days before the event, 1 day in which the event occurred (day 150), and 215 days of different behavior afterwards.

Table 3: Behavior and Temporal Constraints

Behavior	Order relationship constraints	Temporal constraints
B_1 (Wake up)	AsleepInBed ≺ AwakeInBed AwakeInBed ≺ GetOutOfBed GetOutOfBed ≺ GetSlippers GetSlippers ≺ GotoBathroom	6:50 to 7:10
B_2 (Morning Routine)	GotoBathroom ≺ { UseToilet, TurnShowerOn} {UseToilet, TakePijamaOff , TurnShowerOn } ≺ TakeShower TakeShower ≺ { TurnShowerOff, UseTowel, BrushTeeth, WashFace, CombHair} UseTowel ≺ GetDressed {TurnShowerOff, UseToilet, BrushTeeth, CombHair, WashFace } ≺ GotoBedroom	7:00 to 7:40
B_3 (Have Breakfast)	Gotokitchen ≺ { OpenCabinet, TurnStoveOn, OpenRefrigerator} OpenCabinet ≺ {GetCereal, GetCoffee} TurnStoveOn ≺ HeatWater {GetCoffee, HeatWater} ≺ PrepareCoffee PrepareCoffee ≺ TurnStoveOff OpenRefrigerator ≺ GetMilk {GetCereal, GetMilk} ≺ PrepareCereals {PrepareCereals, PrepareCoffee} ≺ EatBreakfast EatBreakfast ≺ CleanDishes { CleanDishes , TurnStoveOff} ≺ LeaveKitchen	7:30 to 8:30
B_4 (Kitchen-based Coffee Break)	Gotokitchen ≺ { OpenCabinet, TurnStoveOn} OpenCabinet ≺ GetCoffee TurnStoveOn ≺ HeatWater {GetCoffee, HeatWater} ≺ PrepareCoffee PrepareCoffee ≺ TurnStoveOff TurnStoveOff ≺ LeaveKitchen	11:10 to 11:30
B_5 (Kitchen and Living Room-Based Coffee Break)	Gotokitchen ≺ { OpenCabinet, TurnStoveOn} OpenCabinet ≺ GetCoffee TurnStoveOn ≺ HeatWater {GetCoffee, HeatWater} ≺ PrepareCoffee PrepareCoffee ≺ TurnStoveOff TurnStoveOff ≺ LeaveKitchen LeaveKitchen ≺ GotoLivingRoom GotoLivingRoom ≺ TurnTvOn	11:10 to 11:30

Table 4: Possible actions in the environment

Item	Action	Item	Action
A_1	AnswerPhone	A_{18}	GotoLivingroom
A_2	AsleepInBed	A_{19}	HeatWater
A_3	AwakeInBed	A_{20}	LeaveKitchen
A_4	BrushTeeth	A_{21}	OpenCabinet
A_5	CleanDishes	A_{22}	OpenRefrigerator
A_6	CombHair	A_{23}	PrepareCereals
A_7	EatBreakfast	A_{24}	PrepareCoffee
A_8	GetCereal	A_{25}	TakePijamaOff
A_9	GetCoffee	A_{26}	TakeShower
A_{10}	GetDressed	A_{27}	TurnShowerOn
A_{11}	GetMilk	A_{28}	TurnShowerOff
A_{12}	GetOutOfBed	A_{29}	TurnStoveOff
A_{13}	GetSlippers	A_{30}	TurnStoveOn
A_{14}	GotoBathroom	A_{31}	TurnTvOn
A_{15}	GotoBedroom	A_{32}	UseToilet
A_{16}	GotoDoor	A_{33}	UseTowel
A_{17}	GotoKitchen	A_{34}	WashFace

Table 5: Behavior constraints after catastrophic event

Behavior	Order relationship constraints
B_1^* (Wake up)	AsleepInBed ≺ AwakeInBed AwakeInBed ≺ GetOutOfBed GetOutOfBed ≺ GotoBathroom
B_2^* (Morning Routine)	GotoBathroom ≺ { GetDressed, UseToilet, BrushTeeth, CombHair, WashFace} { UseToilet, BrushTeeth, CombHair, WashFace} ≺ GotoBedroom
B_3^* (Have Breakfast)	Gotokitchen ≺ { OpenCabinet, TurnStoveOn, OpenRefrigerator} OpenCabinet ≺ {GetCereal, GetCoffee} TurnStoveOn ≺ HeatWater {GetCoffee, HeatWater} ≺ PrepareCoffee OpenRefrigerator ≺ GetMilk {GetCereal, GetMilk} ≺ PrepareCereals {PrepareCereals, PrepareCoffee} ≺ EatBreakfast EatBreakfast ≺ CleanDishes CleanDishes ≺ LeaveKitchen

The changes in behavior due to a catastrophic event are shown in Table 5. Only three behaviors are changed: Wake Up (B_1^*), Morning Routine (B_2^*) and Have Breakfast (B_3^*).

Experiments are performed in two steps. First, we use DS_1 to learn five behavior models, one for each behavior in Table 3. Next, we use DS_2 as input to the recognition and adaptation process for the specific behaviors Wake Up (B_1^*), Morning Routine (B_2^*) and Have Breakfast (B_3^*). Our algorithm continuously adapts a behavior model regarding the inputs and recognized behaviors, and that adaptation is studied as one output of our experiment.

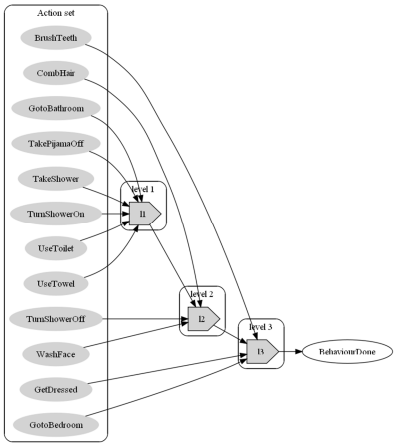


Fig. 3: "Morning Routine" behavior (B_2)

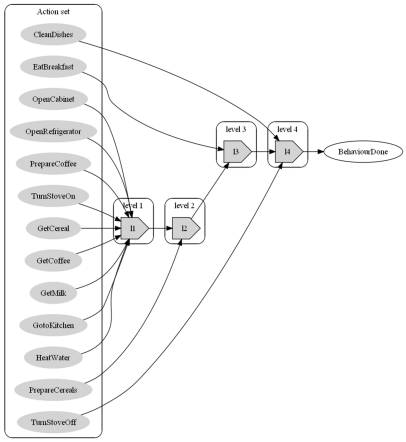


Fig. 4: "Have Breakfast" behavior (B_3)

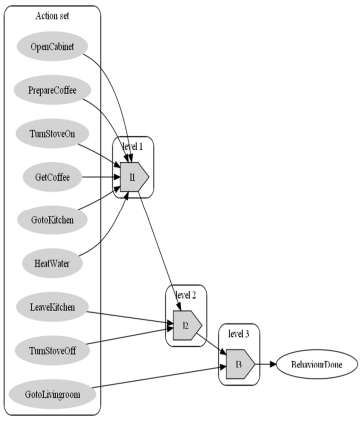


Fig. 5: "Kitchen and Living Room-Based Coffee Break" behavior (B_5)

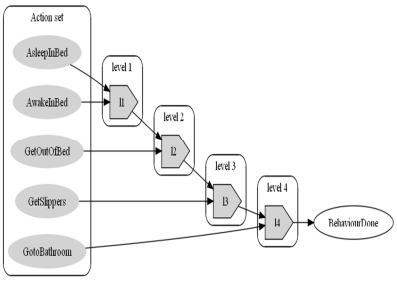


Fig. 6: "Wake Up" behavior (B_1)

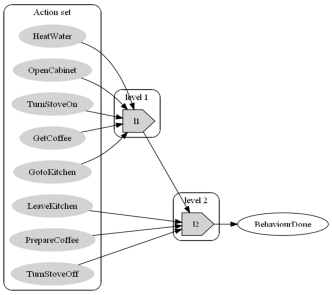


Fig. 7: "Kitchen-based Coffee Break" behavior (B_4)

In general, the proposed measure has multiple uses in the areas of comparing *different* and *same* behaviors. The next section shows how *different* behaviors can be compared. The second experiment discusses the measures utility for studying how a specific behavior evolves over time, after an adaptation process regarding elder daily activity.

V. Experiment 1: Comparing Different Behaviors

In experiment 1, five morning behaviors, $\{B_1,\ldots,B_5\}$, are learned using DS_1 (Figs. 3-7). The purpose of this experiment is to show that similar behaviors are indeed identified by $s_{final}(U,V)$.

For comparing *different* behaviors, the measure can be used in an exploratory way to help understand the similarity of independent machine learned behaviors for a single resident or

across residents in an elderly community. This information could be used by a higher level algorithm trying to assess more abstract/complex types of human behavior, e.g., is a person's overall morning behavior in July similar to that of June? Recall that one does not have to perform the exact set of behaviors every day to have similar routines. This approach can also be used to help determine if multiple reference behaviors have converged into a similar newly refined behavior, i.e., did a resident previously have separate coffee break and snack behaviors and now he/she only has a single similar breakfast behavior?

A similarity matrix between behaviors obtained using DS_1, Figs. 3-7, is shown in Table 6.

Table 6: Behavior similarity matrix for Figs. 3 through 7.

$s_{final}(B_i,B_j)$	B_1	B_2	B_3	B_4	B_5
B_1	1	0	0	0	0
B_2	0	1	0	0	0
B_3	0	0	1	0.33	0.35
B_4	0	0	0.33	1	0.73
B_5	0	0	0.35	0.73	1

Table 6 reinforces what we expect. The measure indicates that the two coffee break-based behaviors are the most similar (0.73) and they share some similarity with breakfast (0.33 and 0.35). In particular, "Having Breakfast" is slightly more similar to the kitchen-based coffee break behavior. In contrast, "Wake Up" and "Morning Routine" do not appear to be similar to any other behaviors (scores of 0).

VI. Experiment 2: Catastrophic Event

Experiment one is focused on the task of comparing *different* behaviors. In experiment two, the focus is comparing the *same* behavior which has been adapted by the learning system over time. A short time period adverse event, like a fall, can be recognized by our LA-based [11, 12] or

linguistic summarization [1,2, 3] systems. The procedure introduced in this article is developed to detect longer time period changes in behavior (days, weeks, etc.). The anomaly detection approach here is independent of the success of any short time period recognition algorithm. That is, if a fall is not detected, we still expect significant change in longer time period behavior. Applications of this approach include:

(1) Has there been a significant change in behavior over a long time period as a result of some adverse event? If the adverse event was detected, then this approach can assist in monitoring the health of an individual while recovering. If the adverse event was not detected, then this procedure could alert a caregiver and prompt investigation into what caused the change.

(2) In the absence of any adverse event, are there signs of significant change in behavior that might allow an algorithm to predict an upcoming catastrophic event?

(3) Has an adaptation algorithm acquired a normal pattern of adapted behavior? Specifically, has the structure of learned behavior converged into some stable state?

To investigate these questions, we simulate a fall, a common elderly adverse event. The fall happened 150 days after a behavior baseline is established where the initial models were learned using DS_1. Data set DS_2 has 365 days of behavior, in which 149 days are before the event, day 150 is the adverse event, and 215 days afterwards are the new changed behavior. Both normal and abnormal behaviors are part of DS_2.

In section 3.A, the normalization value

$$norm_1(U,V) = maximum(\, s(U,U)\,, s(V,V)\,),$$

$$s_{cp}(U,V) = s(U,V) \,/\, norm_1(U,V),$$

was introduced to produce a more human interpretable value to compare *different* behaviors. We extend this normalization technique for comparing the *same* behavior which has been adapted by a system over time,

$$norm_2(U, \{V_1, \dots, V_n\}) = maximum(\{ norm_1(U, V_1), \dots , norm_1(U, V_n) \}),$$

$$s_{cp2}(U, V_i) = s(U, V_i) / norm_2(U, \{V_1, \dots, V_n\}),$$

where *U* is the reference behavior and V_i $(1 \leq i \leq N)$ is an adapted behavior, i.e., N = 365 here. Note that the behavior model is updated each day. The combined measure used is

$$s_{final2}(U, V_i) = s_{cp2}(U, V_i) \times s_{sub}(U, V_i) \times s_{por}(U, V_i),$$

Fig. 8 tells the following story. In approximately the first 100 days, the system is establishing an adapted pattern of normal behavior. This can be seen from the fact that the measure is continuously under change. Remember the system is continuously adapting the model, trying to obtain the minimal adapted behavior model that better represents the studied behavior. From approximately day 100 to 150, the system has established a fairly stable refined description. This is inferred from the fact that the trend of the curve is mostly flat (non-decreasing and non-increasing) when compared to the baseline normal pattern. What is observed in the set of behaviors is a large decrease in the similarity value after the adverse event (day 150). The morning behavior exhibits the most severe change. Tables 3 and 5 show that this behavior is the most affected by the fall. In any case, the behaviors all to a rather stable pattern soon afterwards, though one that is considerably different from the original normal prototype.

In addition, we observe greater fluctuation in the first half of Fig. 8 in comparison to the second half of Fig. 8. This is because during the first part of the process the algorithm expends some time obtaining a minimal adapted behavior model. Comparing the three calculated measures, we

observe that the fluctuations are greater for Wake Up (B_1^*) than for Morning Routine (B_2^*) and Have Breakfast (B_3^*). The reason is due to the number of elements and order relationships are fewer (see Fig. 6), and therefore, the required number of levels might be reduced. The reduction makes that bigger changes need to be performed.

Figs. 9 and 10 are in support of Fig. 8. They show the behavior model evolution. Specifically, Fig. 9 shows the representation obtained just before the catastrophic event, whereas Fig. 10 presents the evolution of the model once the catastrophic event has occurred and the person has changed his/her behavior as result of that event. The system must adapt the model to the new routine (forgetting to take a shower). The behavior in Fig. 10 remains structurally the same for the remainder of the year. This indicates the elder has not recovered from the catastrophic event.

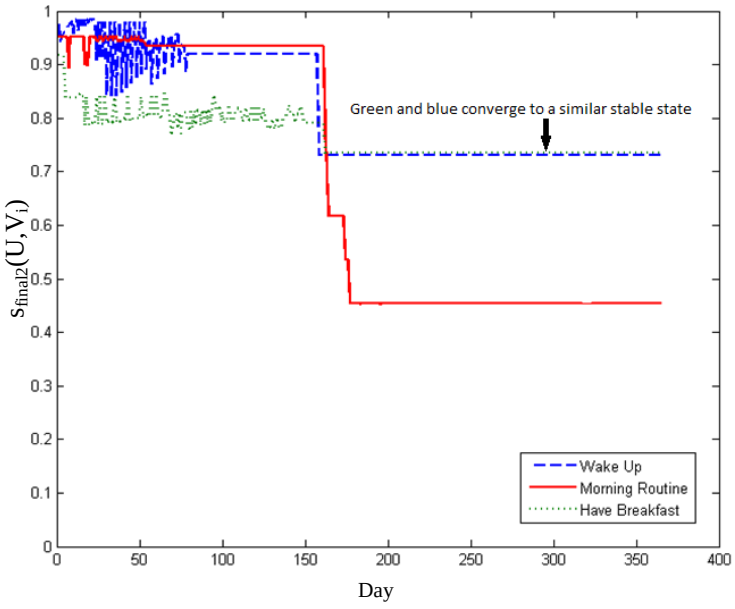


Fig. 8: Similarity of adapted behaviors to learned reference behaviors. A catastrophic event (fall) occurred at day 150. The behavior in the time periods before and after the adverse event are different.

As already stated, the LA-based system has a user defined adaptation parameter that controls how much the system is changed with respect to daily behavior [3,4]. For these experiments, we use a moderate adaptation rate, specifically 0.5, where the parameter ranges from [0,1] with 0

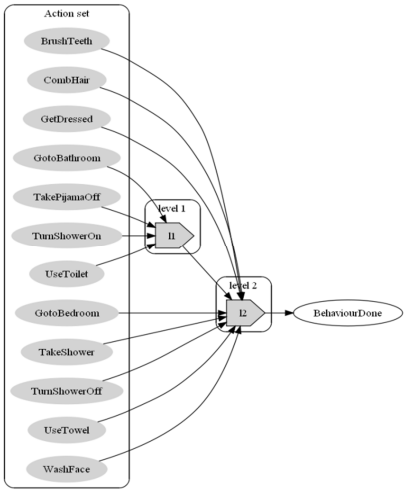


Fig. 9: "Morning Routine" before catastrophic event.

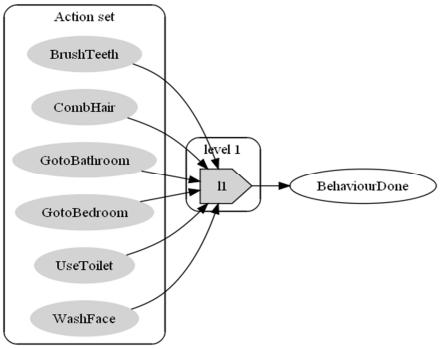


Fig. 10: "Morning Routine" 30 days after the catastrophic event. The behavior stabilizes into this structure for the remainder of the year.

being no change and 1 being full change. The reason we selected a mid-range adaptation rate is so that we can detect important changes in the behavior model quickly but not change the model too quickly in response to common daily fluctuations in behavior. What this all means is that the rate at which anomalies occur in the time series is controlled in part by this parameter. If one selects a high adaptation rate, then the behavior models will change quickly and the anomaly will be detected quicker. However, the day-to-day similarity of the adapted behavior to a reference behavior will also change more. A complete study of the adaptation rate parameter influence is presented in [11, 12]. Here, we selected a single learning rate to demonstrate the utility of this approach. However, one could run multiple behavior sets in parallel with different adaptation rates looking for significant change at different resolutions of time.

VII. Conclusion

In summary, we have presented a high level behavior analysis procedure for (1) comparing *different* behaviors and (2) comparing the *same* behavior that has been adapted by a system over time. Applications of the work include discovery of underlying similarity between behaviors for

a single resident or different residents in an elderly community, and anomaly detection in the context of behavior over long time periods. Our prior work in clustering and comparing soft partitions is combined with a measure of similarity for partial order relations to create a new similarity measure for structurally analyzing change in human behavior learned using our prior research in LAs. We demonstrated, using a wide range of different behavior data sets based on behavior and temporal constraints, that the measure does indeed calculate results that can be understood by a caregiver or by a higher order computational algorithm.

Future work includes the following. First, a synthetic data set was used to test different behavior scenarios. We will investigate using the LA approach from the University of Granada Spain on real sensor data collected from the assisted living facility, TigerPlace [23,24], in Columbia MO, USA. The sensor data collected from TigerPlace over the last several years is the subject of investigation already by our diverse research group of Engineers, Nurses, Physical Therapists, and others collaborating at the University of Missouri [1-7, 41-46].

The next future work is to take the similarity measure introduced here and use it to investigate similarity at a more abstract/complex time scale, i.e., typical overall morning behavior. We will test the hypothesis that a set of behavior models can individually be investigated using the measure presented in this work and a fusion procedure, such as fuzzy logic, fuzzy integrals, etc., can be used to aggregate evidence over a longer period of time.

References

[1] D. T. Anderson, R.H. Luke, J.M. Keller, M. Skubic, M. Rantz and M. Aud, “Linguistic Summarization of Video for Fall Detection Using Voxel Person and Fuzzy Logic,” Computer Vision and Image Understanding, vol. 113 (1), pp. 80-89, 2009

[2] D. T. Anderson, R. H. Luke, J. M. Keller, M. Skubic, M. Rantz, M Aud, “Modeling Human Activity From Voxel Person Using Fuzzy Logic,” IEEE Transactions on Fuzzy Systems, vol. 17 (1), pp. 39-49, 2009

[3] D. T. Anderson, R. H. Luke, J. M. Keller, M. Skubic, “Extension of a Soft-Computing Framework for Activity Analysis from Linguistic Summarizations of Video,” FUZZ-IEEE at WCCI, pp. 1404-1410, China, 2008 (best student paper award)

[4] M. Rantz, M. Aud, G. Alexander, B. Wakefield, M. Skubic, R. Luke, D. T. Anderson, J. M. Keller, “Falls, Technology, and Stunt Actors,” Journal of Nursing Care Quality, vol. 23 (3), pp. 195-201, 2008

[5] E. Stone, D. T. Anderson, M. Skubic, James M. Keller, "Extracting Footfalls from Voxel Data," 32nd Annual Int. Conf. of the IEEE Engineering in Medicine and Biology Society, pp. 1119-1122, 2010.

[6] F. Wang, M. Skubic, C. Abbott and J. M. Keller, "Body Sway Measurement for Fall Risk Assessment Using Inexpensive Webcams," 32nd Annual Intl. Conf. of the IEEE Engineering in Medicine and Biology Society, pp. 2225-2229, 2010.

[7] F. Wang, E. Stone, W. Dai, M. Skubic and J. Keller, "Gait Analysis and Validation Using Voxel Data," 31st Annual Intl. Conf. of the IEEE Engineering in Medicine and Biology Society, pp. 6127-6130, 2009.

[8] S. Morris and J. Paradiso, "A Compact Wearable Sensor Package for Clinical Gait," in Offspring, vol 1 (1), pp. 7-15, 2003.

[9] J. Xiong, B. Seet and J. Symonds, "Human Activity Inference for Ubiquitous RFID-Based Applications,” Sixth Intl. Conf. on Ubiquitous Intelligence and Computing, pp.304-309, 2009.

[10] W.L. Buntine, "Operations for Learning with Graphical Models," Journal of Artificial Intelligence Research, vol 2 (1), pp. 159-225, 1994.

[11] M. Ros, M. Cuellar, M. Delgado, A. Vila, H. Hagras “Detection of abnormal human activities by means of learning automata,” under review by IEEE Trans. on Systems, Man and Cybernetics, Part A, 2010.

[12] M. Ros, M. Cuellar, M. Delgado, A. Vila, “Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows,” Information Sciences, 2011, DOI: 10.1016/j.ins.2011.10.005. To be published.

[13] M. Skubic and M. Rantz, Active Elders: Center for Eldercare and Rehabilitation Technology, http://eldertech.missouri.edu/, 2011.

[14] M. Rantz, M. Aud, G. Alexander,D. Oliver, D. Minner, M. Skubic, J. Keller, Z. He, M. Popescu, G. Demiris, S. Miller, "Tiger place: An innovative educational and research environment," in AAAI in Eldercare: New Solutions to Old Problems, 2008.

[15] D. Cook, M. Youngblood, E. Heierman, K. Gopalratnam, S. Rao, A. Litvin, F. Khawaja, "Mavhome: An agent-based smart home," in proc. of the first IEEE Intl. Conf. on Pervasive Computing and Communications, pp. 521-524, 2003.

[16] F. Doctor, H. Hagras and V. Callaghan, "A fuzzy embedded agent-based approach for realizing ambient intelligence in intelligent inhabited environments," IEEE Transactions on Systems, Man, and Cybernetics, Part A, vol. 35 (1), pp. 55–65, 2005.

[17] P. Rashidi and D. Cook, "Keeping the resident in the loop: Adapting the smart home to the user," IEEE Transactions on Systems, Man, and Cybernetics, Part A, vol. 39 (5), pp. 949-959, 2005.

[18] J. Kientz, S. Patel, B. Jones, E. Price, E. Mynatt and G. Abowd, "The Georgia tech aware home," in CHI '08 extended abstracts on Human factors in computing systems, pp. 3675–3680, 2008.

[19] R. Droes, M. Mulvenna, C. Nugent, D. Finlay, M. Donnelly, M. Mikalsen, S. Walderhaug, T. v. Kasteren, B. Krose, S. Puglia, F. Scanu, M. O. Migliori, E. Ucar, C. Atlig, Y. Kilicaslan, O. Ucar, and J. Hou, "Healthcare systems and other applications," in IEEE Pervasive Computing, vol. 6 (1), pp. 59–63, 2007.

[20] M. Philipose, K. Fishkin, M. Perkowitz, D. Patterson, D. Fox, H. Kautz, D. Hahnel, "Inferring activities from interactions with objects," in IEEE Pervasive Computing, vol. 3, pp. 50–57, 2004.

[21] L. Liao, D. Patterson, D. Fox and H. Kautz, "Learning and inferring transportation routines," in journal of Artificial Intelligence, vol. 171, pp. 311-331, 2007.

[22] J. Boger, J. Hoey, P. Poupart, C. Boutilier, G. Fernie, A. Mihailidis, "A planning system based on Markov decision processes to guide people with dementia through activities of daily living," in IEEE Transactions on Information Tech. in Biomedicine, vol. 10, pp. 323 –333, 2006.

[23] D. Zhang, D. Gatica-Perez, S. Bengio, I. McCowan, G. Lathoud, "Modeling individual and group actions in meetings with layered HMMs," in IEEE Trans. on Multimedia, vol. 8, pp. 509–520, 2004.

[24] U. Naeem and J. Bigham, "A comparison of two hidden Markov approaches to task identification in the home environment," in IEEE 2nd Intl. Conf. on Pervasive Computing and App., pp. 383–388, 2007.

[25] E. Kim, S. Helal, D. Cook, “Human activity recognition and pattern discovery,” in IEEE Pervasive Computing, vol. 9, pp. 48–53, 2010.

[26] G. Acampora and V. Loia, “Fuzzy control interoperability and scalability for adaptive domotic framework,” in IEEE Transactions on Industrial Informatics, vol. 1, pp. 97 – 111, 2005.

[27] V. Jakkula and A. Crandall, “Enhancing anomaly detection using temporal pattern discovery,” Springer. chapter 8. Advanced Intelligent Systems, pp. 175–194, 2008.

[28] M. Delgado, M. Ros, M. Amparo, and M. Vila, “Correct behavior identification system in a tagged world,” Expert System Applications, vol. 36, pp. 9899–9906, 2009.

[29] M. Ros, M. Delgado and A. Vila, “Fuzzy method to disclose behavior patterns in a tagged world,” in Expert Systems With Applications, vol. 38, pp. 3600-3612, 2011.

[30] L. Rabiner, "A tutorial on Hidden Markov Models and selected applications in speech recognition," in Proceedings of the IEEE, vol. 77 (2), pp. 257-286, 1989.

[31] L. Sebastian, B. Hung, V. Svetha and G. West, "Recognition of Human Activity through Hierarchical Stochastic Learning," in Proceedings of the First IEEE International Conference on Pervasive Computing and Communications, pp. 416, 2003.

[32] A. Poznyak and K. Najim, “Learning Automata and Stochastic Optimization,” Springer-Verlag New York, Inc., 1997.

[33] M. Thathachar and P. Sastry, “Varieties of learning automata: an overview,” IEEE Transactions on Systems, Man, and Cybernetics, Part B, vol. 32 (6), pp. 711–722, 2002.

[34] M. Thathachar and K. Ramakrishnan, “A cooperative game of a pair of learning automata,” Automatica, vol. 20 (6), pp. 797 – 801, 1984.

[35] D. T. Anderson, J. C. Bezdek, M. Popescu, and J. M. Keller, “Comparing Fuzzy, Probabilistic, and Possibilistic Partitions,” IEEE Transactions on Fuzzy Systems, vol. 18 (5), pp. 906-918, 2010.

[36] L. Hubert and P. Arabie, Comparing partitions, Journal of Classification, vol. 2, pp. 193-218, 1985.

[37] P. Sneath and R. Sokal, Numerical Taxonomy - The Principles and Practice of Numerical Classification, W. H. Freeman, San Francisco, CA, 1973.

[38] E. Hullermeier and M. Rifqi, “A fuzzy variant of the Rand index for comparing clustering structures,” Proc. IFSA, Portugal, 1-6, 2009.

[39] D. T. Anderson, J. C. Bezdek, J. M. Keller and M. Popescu, “A comparison of five fuzzy Rand indices,” IPMU, Germany, 2010.

[40] M. Rantz, R. Porter, D. Cheshier, D. Otto, C. Servey, R. Johnson, M. Skubic, H. Tyrer, Z. He, G. Demiris, J. Lee, G. Alexander and G. Taylor, "TigerPlace, a State-Academic-Private Project to Revolutionize Traditional Long Term Care," Journal of Housing for the Elderly, 2008.

[41] T. Banerjee, J. M. Keller, M. Skubic and C. Abbott, "Sit-To-Stand Detection Using Fuzzy Clustering Techniques," IEEE Int. Conf. on Fuzzy Systems, pp. 1-8, 2010.

[42] N. Harvey, Z. Zhou, J. M. Keller, M. Rantz and Z. He, "Automated Estimation of Elder Activity Levels from Anonymized Video Data," in 31st Annual Intl. Conf. of the IEEE Engineering in Medicine and Biology Society, pp. 7236-7239, 2009.

[43] S. Wang, M. Skubic and Y. Zhu, "Activity Density Map Dis-similarity Comparison for Eldercare Monitoring ," in 31st Annual Intl. Conf. of the IEEE Engineering in Medicine and Biology Society, pp. 7232-7235, 2009.

[44] M. Popescu, Y. Li, M. Skubic and M. Rantz, "An Acoustic Fall Detector System that Uses Sound Height Information to Reduce the False Alarm Rate," in 30th Annual Intl. Conf. of the IEEE Engineering in Medicine and Biology Society, pp. 4628 – 4631, 2008.

[45] I. Sledge, J. M. Keller and G. Alexander, "Emergent Trend Detection in Diurnal Activity," in 30th Annual Intl. Conf. of the IEEE Engineering in Medicine and Biology Society, pp. 3815-3818, 2008.

[46] I. Sledge, J. M. Keller, T. Havens and G. Alexander, "Temporal Activity Analysis," in AI in Eldercare: New Solutions to Old Problems, AAAI Technical Report FS-08-02, pp. 100-108, 2008.

4.2. Linguistic summarization of long-term trends for understanding change in human behavior

The conference paper associated to this part is:

- Ros, M.; Pegalajar, M.; Delgado, M.; Vila, A.; Anderson, D.T.; Keller, J.M.; Popescu, M.; , "Linguistic summarization of long-term trends for understanding change in human behavior," *Fuzzy Systems (FUZZ), 2011 IEEE International Conference on* , vol., no., pp.2080-2087, 27-30 June 2011
  - Status: **Published**.
  - Conference Category (CORE 2008): A

2011 IEEE International Conference on Fuzzy Systems
June 27-30, 2011, Taipei, Taiwan

Linguistic Summarization Of Long-Term Trends For Understanding Change In Human Behavior

María Ros, Manuel Pegalajar, Miguel Delgado and Amparo Vila
Computer Science and Artificial Intelligent
University of Granada
Granada, Spain
{marosiz, manupc, mdelgado, vila}@decsai.ugr.es

Derek T. Anderson, James M. Keller and Mihail Popescu
Electrical and Computer Engineering
University of Missouri
Columbia, MO 65211 USA
{andersondt, kellerj, popescum}@missouri.edu

***Abstract*—In this paper, we propose a linguistic summarization procedure for describing long-term trends of change in human behavior. Our objective consists of defining methods that provide information to elders, caregivers, social workers or even family in an understandable language. We adapt a measure that we defined in previous work on soft cluster partition similarity for comparing behaviors that are adapted over time. From that measure, we are able to produce a time series that numerically describes change in behavior over time. In this article, the resulting time series is partitioned and linguistically summarized depending on a user's (caregiver, social worker, etc.) desired time resolution. Simulated resident behavior is used in order to explore a range of different scenarios and the response of the proposed linguistic summarization process is investigated.**

Keywords- comparing soft partitions, human behavior, anomaly detection, fuzzy linguistic summarization

I. INTRODUCTION

In recent years, successful aging has become one of the most important problems in our society. Life expectancy has improved, and along with it the resources required to maintain quality and style of life. Emerging applications combining *technology* and *medicine* provide one possible solution. Multiple large scale projects combine these subjects. One such project is the center for Eldercare and Rehabilitation Technology (CERT) [1] and the TigerPlace facility [2]. This effort is a large interdisciplinary collaboration between electrical and computer engineers, gerontological nurses, social workers, physical therapists and others at the University of Missouri. This group has installed multiple sensing technologies in the homes of elders. One goal of these technologies includes assisting residents with "aging in place" as independently as possible. Other projects include the MavHome project [3], iDorm [4], CASAS [5] and the Georgia Tech Aware Home [6]. In addition, in [7] eight other relevant projects in this field are summarized.

Important advantages of behavior analysis in the eldercare domain include: reducing dependency, enhancing personal comfort and safety and delaying the process of moving to a nursing home. Many projects are centered around Activities of Daily Living (ADL) [8]. From the set that focus on the ADL, we emphasize [9], an approach based on multiple behavioral hidden Markov models (MBHMM). The authors propose the creation of multiple HMMs for each variation of an action. The most important advantage of this proposal is that the system is able to determine which tasks are currently active, even if the user has not yet completed the activity.

A related fuzzy approach is [10]. In that work, the authors propose an AmI fuzzy computing system, which is a multilayer architecture to learn and model human behavior. They attempt to join the advantages of a multi-agent based framework with fuzzy control techniques to improve the recognition process and continuously enrich the knowledge about the user and environment.

The majority of existing efforts are focused on the detection of normal and abnormal behaviors at the moment the resident is performing them. In previous work [11], we introduced a method to identify changes in patterns of behavior over longer periods of time (days, weeks or months). Identification of changes in behavior over such time frames is significant in the regard that it can be used to help predict cognitive and/or functional decline that affects overall quality of life [12, 13, 14]. If significant change in human behavior can be recognized early, then caregivers can take action to help avoid or postpone catastrophic events. However, the information obtained is a numerical data set. In this article our objective consists of developing a method to produce linguistic information that is of greater use to users. Our approach is based on the concept of linguistic summarization. A linguistic data summary is a concise, human-consistent description of a (numerical) data set [15, 16, 17]. This process allows us to summarize a data time series by segmenting and describing long-terms trends of change in behavior.

In the existing literature, there are different approaches to deal with time series summarization. The application range is very assorted, but the business field stands out over the rest. In [18, 20] Kacprzyk and Wilbik deals with time series used for an analysis of the past performance of investment (mutual) funds and extracts summaries to help to make future investment decisions. Focusing on human activities, in [19] the authors propose a novel method for recognizing human

978-1-4244-7317-5/11/$26.00 ©2011 IEEE

activity from linguistic summarizations of temporal fuzzy inference curves representing the states of a three-dimensional object called voxel person. However, the approach taken in this article is most similar to that of Kacprzyk et al. [20]. Primarily, they focus on stock market analysis and finance applications. They propose a way to summarize trends identified with straight line segments of a piecewise defined linear approximation of time series of change in long-term human behavior. They employ a set of features (attributes) to characterize the trends such as the slope of the line segment. They then use a linguistic quantifier driven aggregation of trends to perform the linguistic summary. In addition, they also study the description of duration and variability.

Herein, we introduce a method to obtain a natural language description about change in long-term adapted behavior. The user selects a time period amount (resolution) to summarize, e.g., day, week, month, etc. A resident history is partitioned according to this time window and we extract features to characterize the change. In particular, we look for decreasing, constant, or increasing change (similarity) in behavior. This makes it possible to monitor cases such as a degenerative disease or a rehabilitation process without the need to continuously bother an elder.

This paper is organized as follows. Section 2 is previous work, which is required in order to understand this article. Next, in section 3 we introduce the linguistic summarization process, specifying the parameters used and the structure of the sentences. Section 4 is experiments. We present simulated user behavior in order to demonstrate a wide range of different possible scenarios and the response of our proposed approach.

II. Related Work

A. Modeling Human Behavior

Modeling human behavior in eldercare is a field that studies computational techniques to represent and analyze human daily activities in order to provide people with services if required, but without loss of independence. In this article, our objective is to extract a linguistic description of trends in long-term change of human behavior. This approach depends on the choice of representation. Here, we focus on a computational framework investigated by our research team in unpublished [21, 22]. The choice of representation is a sequence of cascaded AND gates with an underlying probability distribution that characterizes temporal dependency requirements over a set of actions for the different ways in which a user can perform a behavior. The work in [21, 22] is summarized in this subsection.

An abnormal situation occurs when a person does not perform the behavior as expected, i.e., forgetting things, performing actions in a dangerous order, etc. Example 1 demonstrates the role of detecting abnormal situations in the context of the goals of this work.

Example 1: Suppose that every morning a resident wakes up, goes to the bathroom, takes his pajamas off, takes a shower, washes his face, gets dressed and returns to the bedroom. All of these actions could be embraced as a single behavior concept, "morning routine". However, suppose that for a period of time the person does not take a shower. This is considered as a change in the "morning routine" behavior. This abnormal situation needs to be identified and the behavior model needs to be updated accordingly.

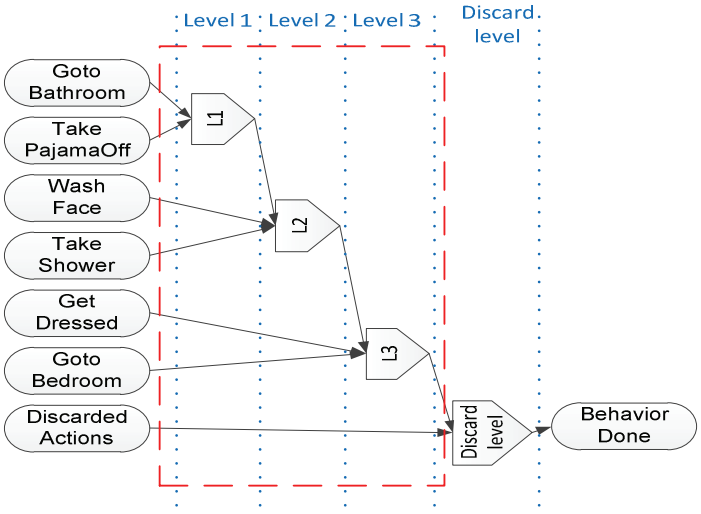


Fig.1. Morning routine behavior model (example 1).

In example 1, two important concepts are introduced, *action* and *behavior*. Informally, an action is the output of a sensor or a low-level algorithm interpreting sensor firings, e.g., the refrigerator door is closed, the front door is opened, a person is sitting on the couch, etc. Behavior is the set of actions and the order in which the actions need be performed. Formally, these concepts are characterized as follows.

Definition 1 (Human Action) [21, 22]: A human action h_i is a triple $h_i =< l_i, s_i, t_i >$, where l_i is a label used to identify the action, s_i is a sensor event and t_i is a time stamp in which the action occurs.

Definition 2 (Behavior) [21, 22]: A human behavior B_i is a quadruple $B_i =< H, R_i, C_i, \prec >$, where $H = \{h_1, h_2, \dots, h_n\}$ is a set of available human tasks, $R_i = \{r_1^i, r_2^i, \dots, r_{|R_i|}^i\} \subseteq H$ is the subset of available human tasks that are required to complete the behavior, and C_i is a set of temporal constraints between the tasks in R_i defined over a partial order relationship operator $\prec$. The partial order relationship $\prec$ defines the temporal dependency requirements over the actions of a behavior, and its formal definition is

$$r_a^i \prec r_b^i \leftrightarrow t_a \prec t_b. \quad (1)$$

Thus, a behavior is represented as a set of relevant actions, which maintain a specific order of performance between them. In [21, 22], we demonstrated that a behavior can be represented as a set of cascaded AND gates. An action in the system corresponds to an input of a specific AND gate. Each input is true if the action has been performed, or false if not. On the other hand, although all the tasks in the H set may be taken into consideration for a specific behavior, only those

actions that are included in the R set are really required for it. To handle this situation, we introduced a discard-level in which those actions that are irrelevant are assigned.

Besides relevant actions, this representation contains more knowledge about a specific behavior, such as the order relationship between diverse actions. As previously pointed out, a behavior is designed as a set of actions that maintains an order relationship between them. The level-representation for a behavior simulates the order relationship between the actions in such a way that those actions that are the input to an AND gate in Level_n have to be done previously to those that are the inputs to the following AND gate, in Level_{n+1}. Fig. 1 shows the sequence of AND gates obtained from example 1.

Nevertheless, a behavior does not have a single way to be performed. Hence, there may be several combinations of the same set of actions for a behavior, i.e., that some tasks may be performed in different levels of the structure without loss of meaning. For instance, in example 1, the action "wash face" could be performed either in level 2 or in the level 3 and be a legitimate Morning Routine. To deal with this disparity, in [21, 22] a probability matrix, $P(i,j)$, was computed, whose rows, $1 \leq i \leq |H|$, are actions, H set, and columns, $1 \leq j \leq |L|$, are available levels in the sequence of ANDs. Each element represents the probability of an action being performed in a particular level. Therefore, the behavior model represents, for each action, the most expected level for it being carried out. Next, keeping on with the prior example, an achievable matrix is presented below.

Continuation of example 1: Let us suppose that we want to study a "morning routine" behavior. In addition, let us suppose that we have the actions presented in example 1, plus the action "go to kitchen". Then, if the maximum number of levels was fixed to four, a possible P matrix to represent the model presented in fig. 1 could be

	L_1	L_2	L_3	L_4	$L_5(discard\ level)$
go to bathroom	0.7	0.2	0	0	0
take pajama off	0.6	0.4	0	0	0
take shower	0.2	0.7	0.1	0	0
wash face	0	0.5	0.3	0	0.2
get dressed	0	0.2	0.6	0.2	0
go to bedroom	0	0.1	0.5	0.3	0.1
go to kitchen	0	0	0	0	1

Among the advantages of our approach, the independence between the behavior representation and the method used to learn, recognize or adapt is the most important. This separation allows us to achieve better system modularization and to use a higher number of learning methods for the same activity model [21]. In [21, 22] we constructed a learning and recognition/adaptation process that will be used in this article to obtain the input to the process presented here. However, as this is not the focus of our article, we refer the reader to our previous work [11, 21, 22].

B. Comparing Soft Partitions

The next tool used in this investigation comes from the field of clustering. This method is used herein to compute the similarity between two behaviors according to how closely actions group into levels. Let $O = \{o_1, \ldots, o_n\}$ denote n objects. When each object in O is represented by a (column) vector x, the set $X = \{x_1, \ldots, x_n\} \subset \Re^p$ is an *object data representation* of O. When each object in $o_i \in O$ has a *physical label*, O is a set of *labeled data*; otherwise, O is unlabeled. Let integer c denote the number of groups, $1 < c < n$. Clustering in unlabeled data is the assignment of one of four types of labels to each object in O. After clustering, the label vectors of the objects are the columns of c-partitions of O, which are sets of (cn) values $\{ u_{ik} \}$ that can be conveniently arrayed as $(c \times n)$ matrices, $U = [u_{ik}]$. These labels can be crisp, probabilistic, fuzzy, or possibilistic.

When clustering produces more than one candidate for partitioning a finite set of objects O, there are two approaches to validation, i.e., selection of a "best" partition, and implicitly, a best value for c, the number of clusters in O. First, one could use an internal index, which evaluates each partition separately. Second, one could compare pairs of candidate partitions to each other, or to a reference partition that purports to represent the "true" cluster structure in the objects. In [23], we generalized many of the classical indices that have been used with outputs of crisp clustering algorithms so that they are applicable for candidate partitions of any type, i.e. crisp or soft, soft comprising the fuzzy, probabilistic and possibilistic cases. In particular, we concentrated on the Rand index and its modifications. We showed that our extension of the Rand index is $\boldsymbol{O}(n)$.

Let U and V be crisp partitions of O. Here, U and V need not possess the same number of clusters, i.e., r need not equal to c ($r \neq c$). The four classical combinations for pairs of objects from $O \times O$ in clusters of U and V are: (a) paired in U and V; (b) not paired in U nor in V; (c) paired in V but not in U; and (d) paired in U but not in V [24]. The comparison of U to V with a similarity measure s begins with the $r \times c$ contingency matrix $N = UV^T$ that contains counts of the number of occurrences of each of the four types over the $n(n-1)/2$ distinct, unordered pairs in $O \times O$. Entry n_{ij} is the number of objects common to subgroups U_i and V_j. The building blocks of many similarity measures for $s(U,V)$ are the four expressions (a)-(d) below. These four calculations count the number of occurrences amongst the $n(n-1)/2$ pairs of each of the four types of unordered pairs:

$$a = \frac{1}{2} \sum_{i=1}^{r} \sum_{j=1}^{c} n_{ij}(n_{ij} - 1); \tag{2}$$

$$d = \frac{1}{2} \left[n^2 + \sum_{i=1}^{r} \sum_{j=1}^{c} n_{ij}^2 - \left(\sum_{i=1}^{r} n_{i\bullet}^2 + \sum_{j=1}^{c} n_{\bullet j}^2 \right) \right]; \tag{3}$$

$$b = \frac{1}{2} \left[\sum_{j=1}^{c} n_{\bullet j}^2 - \sum_{i=1}^{r} \sum_{j=1}^{c} n_{ij}^2 \right]; \tag{4}$$

$$c = \frac{1}{2} \left[\sum_{i=1}^{r} n_{i\bullet}^2 - \sum_{i=1}^{r} \sum_{j=1}^{c} n_{ij}^2 \right]. \tag{5}$$

where $n_{i\bullet}$ and $n_{\bullet j}$ referring respectively to the number of objects in classes U_i and V_j, using standard "dot" notation for row and column sums.

The sums $(a+d)$ and $(b+c)$ are usually interpreted, respectively, as (the total number of) *agreements* and *disagreements* between U and V. Anderson et al. [23] tabulate a (non-exhaustive) list of 14 coefficients that have been proposed for $s(U,V)$ based on functions of a, b, c and d; Sokal and Sneath list many others [25]. In addition, Anderson et al. also present a scaling value in the case of possibilistic labels. Here, we focus on the Rand, the classical form of which is

$$s_R(U,V) = (a+d)\,/\,(a+b+c+d). \quad (6)$$

The measure $s(U,V)$ can be used to address a variety of clustering problems [23]. For example, it can be used to compare a *clustering* to a reference partition that purports to represent the "true cluster structure", measure the effect of noise or missing data with respect to a specific clustering algorithm, study the convergence of a clustering algorithm at successive iterations, perform prediction (regression), as well as compare two different clustering algorithms.

C. Similarity measure for comparing human behavior

In [11], a similarity measure is introduced for comparing pairs of learned behavior and subsequently anomaly detection. Our new measure for analyzing long term human activity tends includes three different components: (1) comparing partitions, (2) subsethood of non-discarded actions, and the (3) similarity of partial order relations.

- *Comparing soft partitions*
  We determine the similarity between pairs of behaviors according to the strength of actions grouping into similar levels. Specifically, $s(U,V)$ is used, where U and V are behavior probability matrices acquired using the procedures described in subsection IIA, i.e. $U(i,j)$ is the probability that action i belongs to level j. In [11], we presented a scaling of $s(U,V)$ to produce a [0,1] value that is more appealing to human interpretation. The formula used to compare two different behaviors is

$$s_{cp}(U,V) = s(U,V)\,/\,maximum(\,s(U,U)\,,s(V,V)\,) \quad (7)$$

- *Subsethood of Non-Discarded Actions*
  This term is included to reduce the influence of the discarded level on $s_{cp}(U,V)$. This is because U and V generally share many actions in that level, independently of the represented behavior. First, the most likely level, $L_{i,max} = argmax_j\, P(i,j)$, for each action is found. Next, the crisp sets, $R_U \subseteq H_U$ and $R_V \subseteq H_V$, of actions in which $L_{i,max}$ does not equal the discard level is found. The matrix P contains evidence about the importance of actions to a behavior and the criteria of picking all actions that are not most likely in the discard level is feature selection. The subsethood measure of non-discarded actions over R_U and R_V is

$$s_{sub}(U,V) = |R_U \wedge R_V| \,/\, |R_U \vee R_V| \quad (8)$$

where $\wedge$ is intersection, $\vee$ is union and $|\bullet|$ is cardinality. The resulting similarity is

$$s_{cp\_sub}(U,V) = s_{cp}(U,V) \times s_{sub}(U,V). \quad (9)$$

- *Similarity of Partial Order Relations*
  The definition of $s_{cp\_sub}(U,V)$ does not take into account the similarity between the partial order relations. That is, two behaviors may share a set of non-discarded actions and have a strong similarity of actions grouping into levels (clusters), but the order of levels is not consistent. Hence, to truly compare two behaviors, the partial order relations need to be factored into the measure. We resolve this by counting the number of disagreements between U and V and weight the disagreements by their individual importance.

  First, a $|H|x|H|$ matrix, E_U, is created for behavior matrix U. For each action, the level with the greatest probability (most likely explanation) is found, i.e. $L_{i,max}$. Next, $E_U(i,j)$ is assigned a label from the set $\{B,A,S\}$, where 'B' (before) means $L_{i,max} < L_{j,max}$, 'A' (after) means $L_{i,max} > L_{j,max}$, and 'S' (same) means $L_{i,max} = L_{j,max}$. Cell $E_U(i,i) =$ *'S'* and if $E_U(i,j)$ is 'B' ('A') then $E_U(j,i)$ is 'A' ('B'). Remember, $L_{i,max}$ is the most likely level that action i belongs to. For that reason, we cannot just count the number of disagreements in E_U. Instead, we calculate an importance value per action. For action i, $\mathrm{P_{U,i}}$ measures the ability to decide that action i belongs to some particular level,

$$\mathrm{P_{U,i}} = \max_j(\mathrm{P_U(i,j)}) - \max_{\substack{k \\ j\neq k}} \big(\mathrm{P_U}(i,k)\big) \quad (10)$$

The significance weighted calculation of the similarity of partial order relations, $s_{por}(U,V)$, is based on the set of off-diagonal upper triangular indices of E_U, i.e. all (i,j) where $j > i$. In particular, the measure uses only the subset of indices, I', in which actions i and j are not most likely discarded. Based on $P_{U,i}$ and $E_U(\mathrm{i,j})$, the similarity of partial order relations is now defined as

$$s_{por}(U,V) = 1 - \left(\textstyle\sum_{(i,j)\in I'} \emptyset(\,\mathrm{E_U(i,j)}\,,\mathrm{E_V(i,j)}\,)\right)/|I'|, \quad (11)$$

where

$$\emptyset\big(\mathrm{E(i,j)},\mathrm{E_V(i,j)}\big) = \begin{cases} 0 & if\ \mathrm{E_U(i,j)} = \mathrm{E_V(i,j)} \\ \min\,(\mathrm{P_{U,i}},\mathrm{P_{V,i}}) & else \end{cases}, \quad (12)$$

and the final measure for comparing human behaviors is

$$s_{final}(U,V) = s_{cp}(U,V) \times s_{sub}(U,V) \times s_{por}(U,V) \quad (13)$$

With this measure, we are able to detect longer time period changes in the *same* behavior (days, weeks, months, etc). The anomaly detection approach in this article is independent of the success of any short time period recognition algorithm. That is, if a serious short-term event is undetected, we still expect significant change in longer time period behavior. In [11], we also presented a time series normalization of $s(U,V)$,

$$norm_1(U,V) = maximum(\, s(U,U)\,, s(V,V)\,), \quad (14)$$

$$norm_2(U,\{V_1,\dots,V_n\}) = maximum(\,\{\, norm_1(U,V_1)\,,\dots\,, norm_1(U,V_n)\,\}\,), \quad (15)$$

$$s_{cp2}(U,V_i) = s(U,V_i) \,/\, norm_2(U,\{V_1,\dots,V_n\}), \quad (16)$$

where U is the reference behavior and V_i $(1 \le i \le N)$ is an adapted behavior. We update the behavior model each day. The final measure presented in [11] is

$$s_{final2}(U,V_i) = s_{cp2}(U,V_i) \times s_{sub}(U,V_i) \times s_{por}(U,V_i) \quad (17)$$

Fig. 2 is an example time series from [11] for the behavior 'Wake up". The annotations describe what the person was doing during the one year time period. We see that the proposed similarity measure captures these important changes.

III. LINGUISTIC SUMMARIZATION

Using the measure discussed in section 2c, we are able to study longer time period changes in behavior. However, this representation is not the most understandable by nurses, social workers, families or residents. Our objective is to produce linguistic summarizations of the time series. Example summarizations include *"the quality of the 'wake up' behavior is decreasing in the last month"*, *"the quality of the 'have breakfast' behavior is slowly increasing in the last week"*, or *"the quality of 'morning routine' is constant but highly unstable in the last month"*. These examples illustrate cases in which less information is more useful. That is, summarization identifies the important trends and it is understandable by a domain expert. In the following sections, we present a method to measure the type and amount of change as well as the stability in comparing behavior time series. We then present a fuzzy set theoretic approach for linguistically interpreting the numeric values for the eldercare problem domain.

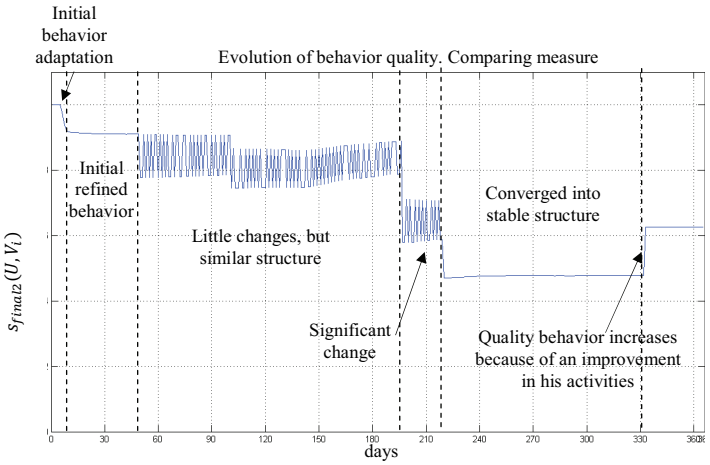


Fig 2. Output of our comparing behavior approach [10] for 360 days of adapted behavior for the activity "Wake up" (which is reported in table 1). Annotations show what happened in this year. Each day is compared to an initial base refined behavior (the base-line behavior).

A. Temporal resolution of summarization

As already discussed, in Kacprzyk et al. [20] a time series is algorithmically segmented. The proposed technique can address identifying trends that take varying amounts of time to occur, but only one set of summarizations are produced. That is, different summarizations could be produced if the problem is viewed in different ways, like being sampled differently. For example, a summary of increasing might be produced if we analyzed day by day, while a summarization of constant might be produced if the time scale is month. Our objective is not to study automatically segmented global trends, rather trends for a user specified time scale. We are interested in detecting gradual changes in the elder's behavior as soon as possible. Kacprzyk et al. differentiate between global trends, which concern the entire time series (or some, probably large, part of it), and partial trends concerning a smaller time span (e.g. a window). Our focus is on the second (partial trends). We provide the person with the option of selecting the time period that he/she would like summarized. That defined time window has a considerable influence on the linguistic summarizations produced. Thus, the user will select it based on a specific hypothesis to be tested. For example, has there been change this week, this month, etc.

B. Partial trend summarization using linear regression

The global trend segmentation and resulting variability calculations of Kacprzyk et al. did not work particulary well for our problem domain. Their focus is stock market analysis and financial domains. The particular trends that we are looking to identify are not reliably segmented using their technique. Instead, we rely on the well-known method of linear regression in order to numerically summarize a partial trend time window. Linear regression is used here to fit a predicted model (linear equation) to observed data, a window of our time series. We calculate the slope of the estimated linear equation, which resides in $[-90°, 90°]$. Additionally, we compute the mean squared error (mse), which varies in [0, 1]. This simple but effective method provides us with the two desirable quantities, the angle of the line segment that characterizes the rate of change of the time series and the variability of the samples to the estimated linear equation.

Herein, linguistic summarization of change in human

behavior over long time periods according to a reference baseline model for a given user time resolution consists of the generation of understandable sentences of the form:

Behavior [X] is [angle] with [mse] in a time period equal to [time window].

C. Fuzzy sets

In this article, we leverage the concept of *dynamics of change* (or speed of change) identified by Kacprzyk et al [20]. The authors propose linguistic terms corresponding to various inclinations of a trend line: *quickly decreasing, decreasing, slowly decreasing, constant, slowly increasing, increasing and quickly increasing*. Each term represents a fuzzy granule of direction. Following the authors' indication, we define the following trapezoidal fuzzy sets on the angle domain in degrees, and illustrated in fig. 3:

- *Quickly decreasing* [-90 -90 -80 -60]
- *Decreasing* [-75 -60 -40 -25]
- *Slowly decreasing* [-40 -30 -20 -10]
- *Constant* [-20 -10 10 20]
- *Slowly increasing* [10 20 30 40]
- *Increasing* [25 40 60 75]
- *Quickly increasing* [60 80 90 90].

Next, we define a linguistic variable for the concept stability of a time series in a user defined time window. While this is similar to Kacprzyk's notion of variability, they are calculated differently. Our terms for stability are defined using linear regression, specifically mse. The fuzzy sets represent the possible levels of instability existing in a time window. We define five terms: *no instability, low instability, medium instability, high instability* and *extreme instability*. The trapezoidal membership functions for these fuzzy sets are (see fig. 4 for a graphical representation):

- *no instability* [0 0 0.0001 0.0012]
- *low instability* [0.0001 0.0005 0.001 0.0014]
- *medium instability* [0.0001 2 0.002 0.004 0.0048]
- *high instability* [0.004 0.009 0.04 0.07]
- *extreme instability* [0.05 0.071 1 1].

These fuzzy sets have been defined as a generalization of a wide range of cases studied, using the trial and error technique.

IV. Experiments

Our experiments study different common elderly behaviors, such as Wake Up, Morning Routine or Have Breakfast. Here, we show results for *Wake Up* and *Morning Routine*, but more results can be found in [11].

As previously indicated, the behavior model is an abstract architecture completely independent of method used to learn, recognize or adapt it. Therefore, our outcomes will be also independent of the type of sensors used to detect the user activities required to those processes. In order to test a wide range of human behavior scenarios, we have developed an

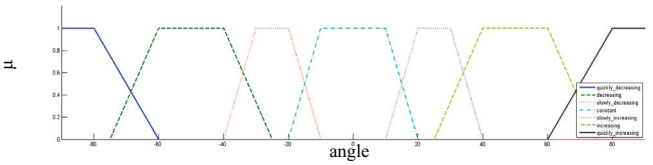


Fig. 3: Fuzzy sets for the angle of the estimated linear equation.

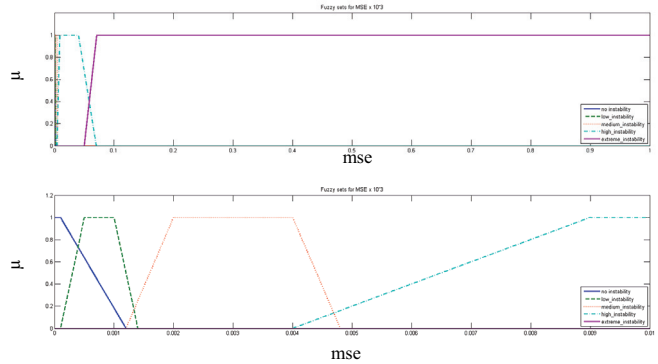


Fig. 4: Fuzzy sets for the mse in the user specified time window.

algorithm that takes action and temporal constraints and generates *synthetic data sets* [21, 22]. The input to this algorithm consists of a set of constraints (table 1), where the partial order relations between the actions are defined. These data sets are used in both learning and recognition [21]. We generated the following data sets to explore how the proposed measure responds to different scenarios.

- DS_1: The resident always performs the behavior correctly, i.e. normal behaviors. Herein, DS_1 is used to learn the initial baseline model. It has 365 days worth of behavior.
- DS_2: The resident has a degenerative disease, and his behavior deteriorates continuously and slowly.
- DS_3: The resident has a *catastrophic event*, e.g. a fall. This event is severe and it results in restricted mobility. After the event, a user's routine has changed significantly as a result. However, after a period of time the user is able to start *rehabilitation*, and his behavior improves. There

TABLE 1: BEHAVIOR AND TEMPORAL CONSTRAINTS

Behavior	Order relationship constraints	Temporal constraints
B_1 (Wake up)	AsleepInBed ≺ AwakeInBed AwakeInBed ≺ GetOutOfBed GetOutOfBed ≺ GetSlippers GetSlippers ≺ GotoBathroom	6:50 to 7:10
B_2 (Morning Routine)	GotoBathroom ≺ { UseToilet, TurnShowerOn} {UseToilet, TakePijamaOff , TurnShowerOn } ≺ TakeShower TakeShower ≺ { TurnShowerOff, UseTowel, BrushTeeth, WashFace, CombHair} UseTowel ≺ GetDressed {TurnShowerOff, UseToilet, BrushTeeth, CombHair, WashFace } ≺ GotoBedroom	7:00 to 7:40

TABLE 2: BEHAVIOR CONSTRAINTS AFTER CATASTROPHIC EVENT

Behavior	Order relationship constraints
B_1^* (Wake up)	AsleepInBed ≺ AwakeInBed AwakeInBed ≺ GetOutOfBed GetOutOfBed ≺ GotoBathroom
B_2^* (Morning Routine)	GotoBathroom ≺ { GetDressed, UseToilet, BrushTeeth, CombHair, WashFace} { UseToilet, BrushTeeth, CombHair, WashFace} ≺ GotoBedroom

are 110 days before the event, 1 day in which the event occurred (day 111), 139 days of convalescence and others 115 days of different behavior afterwards when the user is undergoing rehabilitation. During the rehabilitation, his behavior is improving slowly.

The changes in behavior due to the catastrophic event are shown in table 2.

A. *Experiment 1. Degenerative disease*

In the first experiment, we show how our system is able to detect a degenerative disease. First, we learn a correct behavior using DS_1. After that, we use DS_2 as input to the recognition and adaptation process. Once the process has finished, the evolution graph is generated using the method in [11]. This time series is used as input to our linguistic summarization method. First of all, we define the *time window resolution*. Although the system is able to deal with different resolutions, here we study the results monthly in order to simplify the problem.

The results for *Wake up* behavior are shown in fig. 5. The graph represents the measured change in behavior. The continuous curve is the similarity measure, while the discontinuous lines are the linear regression results at each non-overlapping window. The related linguistic summarization is presented in table 3.

Fig. 5 shows a behavior whose quality is decreasing very slowly. This corresponds to a resident with a degenerative disease reducing the resident mobility and, therefore, changing his normal activity. However, there are time periods when his/her behavior is considered constant although not stable. Our summary is able to detect those moments of instability, describing it as medium in *February* or low in *November,* when the behavior has a little decrease and later an increase.

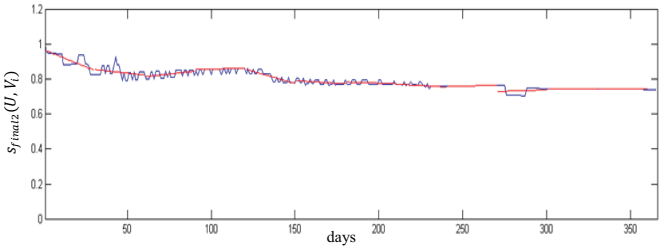


Fig. 5: Similarity of adapted behavior to a learned reference behavior for "Wake up" behavior. In this situation, we simulate a user with a degenerative disease. The rate of change for this behavior is slow.

TABLE 3: LINGUISTIC SUMMARIZATION OF TIME SERIES IN FIGURE 5

Days	Linguistic Summarization
1-30	B_1 is *slowly decreasing* in a time period equal to *30 days*
31-60	B_1 *is constant* with *medium instability* in a time period equal to *30 days*
61-90	B_1 *is constant* with *low Instability* in a time period equal to *30 days*
91-120	B_1 *is constant* with *no instability* in a time period equal to *30 days*
121-150	B_1 *is slowly decreasing* in a time period equal to *30 days*
151-180	B_1 *is constant*, with *no instability* in a time period equal to *30 days*
181-210	B_1 *is constant*, with *no instability* in a time period equal to *30 days*
211-240	B_1 *is constant*, with *no instability* in a time period equal to *30 days*
241-270	B_1 *is constant*, with *no instability* in a time period equal to *30 days*
271-300	B_1 *is constant*, with *low Instability* in a time period equal to *30 days*
301-330	B_1 *is constant*, with *no instability* in a time period equal to *30 days*
331-365	B_1 *is constant*, with *no instability* in a time period equal to *30 days*

B. *Experiment 2. Catastrophic event and rehabilitation*

For the second experiment, once again DS_1 is used to learn a reference behavior. However, DS_3 is used during the recognition and adaptation process. The database represents user activities for a year. The user suffers a *catastrophic event*, as a fall. This event provokes the elder to change his/her activity significantly, and then undergo *rehabilitation*. The linguistic summarization for *Morning Routine* behavior obtained from these results is presented in table 4 regarding fig. 6.

We would like to emphasize how the linguistic summarization is able to delineate the important changes occurring during the year. During the first three months, the behavior is *constant* because the elder maintains his/her normal activities. When the catastrophic event happens, at the middle of the fourth month, the behavior model adapts to the new normal. The model requires some time period to adapt completely. However, it does reflect those little changes, as does the linguistic summary. The forth month is characterized as *medium instability*, since the behavior is summarized as *constant*. During the fifth and sixth months, the behavior changes significantly (declines), resulting in *behavior decreasing* and *behavior slowly decreasing* respectively. These months are included in the convalescence period: the user stabilizes her behavior and she recovers from the catastrophic event. The ninth month is when the elder starts rehabilitation. As expected, rehabilitation is not an instant process, it requires effort and time. During the ninth and tenth months, she is changing her behavior slowly without great advance (*behavior constant* and *low instability*). Nevertheless, after two months undergoing rehabilitation, the elder improves her behavior more rapid than before (*behavior increasing*), until a point where the behavior is stable again (*behavior constant* and *low instability*).

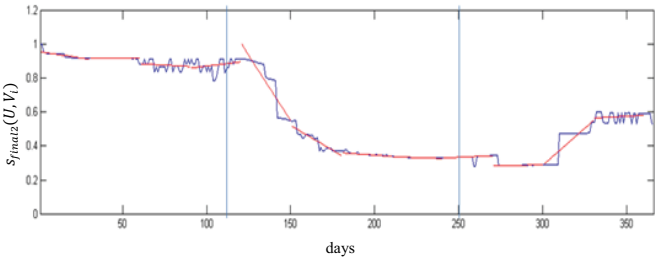


Fig. 6: Similarity of adapted behavior to a learned reference behavior for "Morning Routine" behavior. This event, e.g. a fall, results in a large change in subsequent behavior (i.e. table 2). We also simulate a scenario in which the user undergoes rehabilitation and his/her behavior improves but never goes back to what it was before.

TABLE 4: LINGUISTIC SUMMARIZATION OF TIME SERIES IN FIG. 6

Days	Linguistic Summarization
1-30	B_2 is *constant*, with *no instability* in a time period equal to *30 days*
31-60	B_2 is *constant*, with *no instability* in a time period equal to *30 days*
61-90	B_2 is *constant* with *low Instability* in a time period equal to *30 days*
91-120	B_2 is *constant* with *medium instability* in a time period equal to *30 days*
121-150	B_2 is *decreasing* in a time period equal to *30 days*
151-180	B_2 is *slowly decreasing* in a time period equal to *30 days*
181-210	B_2 is *constant*, with *no instability* in a time period equal to *30 days*
211-240	B_2 is *constant*, with *no instability* in a time period equal to *30 days*
241-270	B_2 is *constant* with *low Instability* in a time period equal to *30 days*
271-300	B_2 is *constant* with *low Instability* in a time period equal to *30 days*
301-330	B_2 is *increasing* in a time period equal to *30 days*
331-365	B_2 is *constant* with *low Instability* in a time period equal to *30 days*

V. CONCLUSIONS

In this paper, we presented a method to produce a linguistic summarization about the quality of an elder's behavior as it adapts over time. In previous work, we presented a method to obtain behavior representations [21, 22] and a measure to analyze change in behavior models [11]. This paper is focused on defining a fuzzy set-based method to summarize the measured information, with the goal of reporting information in a natural language to nurses, doctors or social workers, which makes it simpler for them to understand what is going on. We introduced a user-defined time resolution approach that allows the expert to investigate different time specific health queries. In addition, we defined a set of features that allows us to detect if the behavior is changing, the speed of the change and the stability of the behavior. Finally, we tested our method in a simulated resident scenario to show how the method helps a person (e.g. elder, caregiver, social workers or even family member) to monitor the activities of an elderly resident.

Future work includes the following. First, a synthetic data set was used to test different behavior scenarios. We will investigate using the LA approach from the University of Granada, Spain on real sensor data collected from the assisted living facility, TigerPlace [2], in Columbia MO, USA. Secondly, we propose to obtain the expert opinion to evaluate the method and improve it by adding more attributes. Both the membership functions and the methods of producing the linguistic summarization can be learned.

VI. ACKNOWLEDGMENT

This work has been partially supported by TIN2009-14538-C02-01 project, I+D+I national program from Spain Government and by National Science Foundation NFS IIS-0428420.

REFERENCES

[1] M. Skubic and M. Rantz. (2011) Active elders: Center for eldercare and rehabilitation technology. [Online]. Available: http://eldertech.missouri.edu/

[2] M. Rantz, M. Aud, G. Alexander, D. Oliver, D. Minner, M. Skubic, J. Keller, Z. He, M. Popescu, G. Demiris, , and S. Miller, "Tiger place: An innovative educational and research environment," AAAI in Eldercare: New Solutions to Old Problems, Tech. Rep, November 2008.

[3] D. Cook, M. Youngblood, I. Heierman, E.O., K. Gopalratnam, S. Rao, A. Litvin and F. Khawaja. "Mavhome: An agent-based smart home", in *Pervasive Computing and Communications, 2003. (PerCom 2003). Proceedings of the First IEEE International Conference on*, march 2003, pp. 521 – 524.

[4] F. Doctor, H. Hagras, and V. Callaghan, "A fuzzy embedded agentbased approach for realizing ambient intelligence in intelligent inhabited environments," *Systems, Man and Cybernetics, Part A: Systems and Humans. IEEE Transactions on*, vol. 35, no. 1, pp. 55–65, jan. 2005.

[5] P. Rashidi and D. Cook. "Keeping the resident in the loop: Adapting the smart home to the user". *Systems,Man and Cybernetics,Part A:Systems and Humans. IEEE Transactions on*, vol.39, no.5, pp.949 –959. sept.2009

[6] J.A. Kientz, S.N.Patel, B. Jones, E. Price,E.D. Mynatt, G.D. Abowd, "The georgia tech aware home", in *CHI '08: CHI '08 extended abstracts on Human factors in computing systems*, ser CHI EA '08. New York, NY, USA: ACM, 2008, pp. 3675–3680.

[7] R.-M. Droes, M. Mulvenna, C. Nugent, D. Finlay, M. Donnelly, M. Mikalsen, S. Walderhaug, T. v. Kasteren, B. Krose, S. Puglia, F. Scanu, M. O. Migliori, E. Ucar, C. Atlig, Y. Kilicaslan, O. Ucar, and J. Hou, "Healthcare systems and other applications," *Pervasive Computing, IEEE*, vol. 6, no. 1, pp. 59 –63, jan.-march 2007.

[8] M. Philipose, K. P. Fishkin, M. Perkowitz, D. J. Patterson, D. Fox, H. Kautz, and D. Hahnel, "Inferring activities from interactions with objects," *Pervasive Computing, IEEE*, vol. 3, no. 4, pp. 50 – 57, oct.-dec. 2004.

[9] U. Naeem and J. Bigham, "A comparison of two hidden markov approaches to task identification in the home environment," in *Proceedings of the 2nd International Conference on Pervasive Computing and Applications*, 2007, pp. 383–388.

[10] Acampora, G., Loia, V. "Fuzzy control interoperability and scalability for adaptive domotic framework". *Industrial Informatics, IEEE Transactions on,* vol. 1, no. 2, pp. 97 – 111, may 2005.

[11] Derek T. Anderson, María Ros, James M. Keller, Manuel P.Cuéllar, Mihail Popescu, Miguel Delgado and Amparo Vila. Similarity measure for anomaly detection and comparing human behaviors. Submitted to IEEE Transactions on Knowledge and Data Engineering, 2011.

[12] Stone E, Anderson D, Skubic M & Keller J, "Extracting Footfalls from Voxel Data," *Engineering in Medicine and Biology Society(EMBC), 2010 Annual International Conference of the IEEE,*vol.1, no. 1, sept. 2010, pp. 1119 –1122.

[13] Wang F, Skubic M, Abbott C & Keller J, "Body Sway Measurement for Fall Risk Assessment Using Inexpensive Webcams," *Engineering in Medicine and Biology Society(EMBC), 2010 Annual International Conference of the IEEE,*vol.1, no.1, sept 2010, pp. 2225 –2229.

[14] Wang F, Stone E, Dai W, Skubic M & Keller J, "Gait Analysis and Validation Using Voxel Data," in *Engineering in Medicine and Biology Society, 2009 EMBC. Annual International Conference of the IEEE*, sept, 2009, pp. 6127–6130.

[15] R.R. Yager. "A new approach to the summarization of data". *Information Sciences*, vol. 28, no. 1, pp. 69 – 86, 1982.

[16] J. Kacprzyk, RR. Yager, "Linguistic summaries of data using fuzzy logic". *International Journal of General Systems* vol. 30, no. 2, pp. 133 – 154, 2001.

[17] J. Kacprzyk, RR. Yager, S. Zadrozny, "A fuzzy logic based approach to lingusitic summaries of database". *International Journal of Applied Mathematics and Computer Science*, vol. 10, pp. 813–834, 2000.

[18] Kacprzyk, Janusk and Wilbik, Anna. "Temporal Linguistic Summaries of Time Series Using Fuzzy Logic", in *Information Processing and Management of Uncertainty in Knowledge-Based Systems. Theory and Methods,* ser. Communications in Computer and Information Science, 2010, vol. 80, pp. 436–445.

[19] Derek Anderson, Robert H. Luke, James M. Keller, Marjorie Skubic, Marilyn Rantz, and Myra Aud. "Linguistic summarization of video for fall detection using voxel person and fuzzy logic". *Computer Vision and Image Understanding,* vol. 113, no. 1, pp. 80 – 89, 2009

[20] J. Kacprzyk, A. Wilbik, and S. Zadrony. "Linguistic summarization of time series using a fuzzy quantifier driven aggregation". *Fuzzy Sets Syst.*, vol. 159, pp. 1485–1499, June 2008

[21] Cuellar, MP., Ros M., Delgado M, Vila A, Detection of abnormal human activities by means of learning automata. Submitted to IEEE Trans. on Systems, Man and Cybernetics, Part A .

[22] Ros M., Cuellar, MP., Delgado M, Vila A, Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows. Submitted to Spec. Iss.: On-Line Fuzzy ML and DM. Information Sciences.

[23] D. T. Anderson, J. C. Bezdek, M. Popescu, and J. M. Keller, "Comparing Fuzzy, Probabilistic, and Possibilistic Partitions," *Fuzzy Systems, IEEE Transactions on*, vol. 18, no. 5, pp. 906 –918, oct. 2010.

[24] Hubert, L. and Arabie, P., "Comparing partitions". *Journal of Classification*, vol. 2, pp. 193–218, 1985.

[25] Sneath, P. H. A. and Sokal, R. R.. *Numerical Taxonomy - The Principles and Practice of Numerical Classification*, Freeman, 1973.

Bibliography

[1] Healthy places terminology. Last access 2012-01-22.

[2] Proyección de la población de españa a largo plazo, 2009-2049, January 2009. Last access 31/03/2012.

[3] G.D. Abowd, M. Ebling, G. Hung, Hui Lei, and H.-W. Gellersen. Context-aware computing. *Pervasive Computing, IEEE*, 99(3):22 –23, july-sept. 2002.

[4] G. Acampora and V. Loia. Fuzzy control interoperability and scalability for adaptive domotic framework. *Industrial Informatics, IEEE Transactions on*, 1(2):97 – 111, may 2005.

[5] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, VLDB '94, pages 487–499, San Francisco, CA, USA, 1994. Morgan Kaufmann Publishers Inc.

[6] Derek T. Anderson, María Ros, James M. Keller, Manuel P.Cuéllar, Mihial Popescu, Miguel Delgado, and Amparo Vila. Similarity measure for anomaly detection and comparing human behaviors. *International Journal of Intelligent Systems*, -:–, 2011. Submitted to.

[7] D.T. Anderson, J.C. Bezdek, M. Popescu, and J.M. Keller. Comparing fuzzy, probabilistic, and possibilistic partitions. *Fuzzy Systems, IEEE Transactions on*, 18(5):906 –918, oct. 2010.

[8] J Augusto and C. Nugent. The use of temporal reasoning and management of complex events in smart homes. In L.Saitta R.L. deMantaras, editor, *Proceedings of European Conference on Artificial Intelligence*, pages 778–782. IOS Press, August 22-27 2004.

[9] J. Boger, J. Hoey, P. Poupart, C. Boutilier, G. Fernie, and A. Mihailidis. A planning system based on markov decision processes to guide people with dementia through activities of daily living. *IEEE Trans Inf Technol Biomed*, 10(2):323–333, 2006.

[10] Koen Bonenkamp, Dario Chiappetta, and Benjamin Bittner. Variable mapping for transfer learning. Technical report, University of Amsterdam, 2012.

[11] Juan A. Botia, Ana Villa, and Jose Palma. Ambient assisted living system for in-home monitoring of healthy independent elders. *Expert Systems with Applications*, 39(9):8136 – 8148, 2012.

[12] Juan A. Botia, Ana Villa, Jose T. Palma, David Pérez, and Emilio Iborra. Detecting domestic problems of elderly people: Simple and unobstrusive sensors to generate the context of the attended. In *Proceedings of the 10th International Work-Conference on Artificial Neural Networks: Part II: Distributed Computing, Artificial Intelligence, Bioinformatics, Soft Computing, and Ambient Assisted Living*, IWANN '09, pages 819–826, Berlin, Heidelberg, 2009. Springer-Verlag.

[13] Harry Chen, Tim Finin, and Anupam Joshi. An ontology for context-aware pervasive computing environments. *Knowl. Eng. Rev.*, 18(3):197–207, sep 2003.

[14] Liming Chen and Chris D. Nugent. Ontology-based activity recognition in intelligent pervasive environments. *IJWIS*, 5(4):410–430, 2009.

[15] M.M. Ciampi and C. da Rocha Brito. Human side of engineering: Dealing with complex and ethical challenges. In *Frontiers in Education Conference (FIE), 2011*, pages T1G 1 – T1G 4, oct. 2011.

[16] Diane J. Cook, Juan C. Augusto, and Vikramaditya R. Jakkula. Ambient intelligence: Technologies, applications, and opportunities. *Pervasive and Mobile Computing*, 5(4):277 – 298, 2009.

[17] Diane J. Cook, Michael Youngblood, and Sajal K. Das. Designing smart homes. In Juan Carlos Augusto and Chris D. Nugent, editors, *Designing smart, The role of Artificial Intelligence*, chapter A multi-agent approach to controlling a smart environment, pages 165–182. Springer-Verlag, Berlin, Heidelberg, 2006.

[18] D.J. Cook, M. Youngblood, III Heierman, E.O., K. Gopalratnam, S. Rao, A. Litvin, and F. Khawaja. Mavhome: an agent-based smart home. In *Pervasive Computing and Communications, 2003. (PerCom 2003). Proceedings of the First IEEE International Conference on*, pages 521 – 524, march 2003.

[19] M. Delgado, M. Ros, and A. Vila. Correct behavior identification system in a tagged world. *Expert Systems with Applications*, 36(6):9899–9906, 2009.

[20] F. Doctor, H. Hagras, and V. Callaghan. A fuzzy embedded agent-based approach for realizing ambient intelligence in intelligent inhabited environments. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 35(1):55 – 65, jan. 2005.

[21] Rose-Marie Droes, Maurice Mulvenna, Chris Nugent, Dewar Finlay, Mark Donnelly, Marius Mikalsen, Stale Walderhaug, Tim van Kasteren, Ben Krose, Stefano Puglia, Fabio Scanu, Marco Oreste Migliori, Erdem Ucar, Cenk Atlig, Yilmaz Kilicaslan, Ozlem Ucar, and Jennifer Hou. Healthcare systems and other applications. *IEEE Pervasive Computing*, 6(1):59–63, January 2007.

[22] K. Ducatel, M. Bogdanowicz, F. Scapol, J. Leijten, and J-C. Burgelman. Scenarios for ambient intelligence in 2010. Technical report, IST Advisory Group (ISTAG), 2002.

[23] Carmen Garrido, Nicolas Marin, and Olga Pons. Fuzzy intervals to represent fuzzy valid time in a temporal relational database. *International Journal of Uncertainty, fuzzyness and Knowledge-Based Systems*, 17(1):173–192, August 2009.

[24] Thomas R. Gruber. Toward principles for the design of ontologies used for knowledge sharing. *Int. J. Hum.-Comput. Stud.*, 43(5-6):907–928, December 1995.

[25] H. Hagras, V. Callaghan, M. Colley, G. Clarke, A. Pounds-Cornish, and H. Duman. Creating an ambient-intelligence environment using embedded agents. *Intelligent Systems, IEEE*, 19(6):12 – 20, nov.-dec. 2004.

[26] Hani Hagras, Faiyaz Doctor, Victor Callaghan, and Antonio Lopez. An incremental adaptive life long learning approach for type-2 fuzzy embedded agents in ambient intelligent environments. *Fuzzy Systems, IEEE Transactions on*, 15(1):41 –55, feb. 2007.

[27] S. Helal, W. Mann, H. El-Zabadani, J. King, Y. Kaddoura, and E. Jansen. The gator tech smart house: a programmable pervasive space. *Computer*, 38(3):50 – 60, march 2005.

[28] T. Hester, D.M. Sherrill, M. Hamel, K. Perreault, P. Boissy, and P. Bonato. Identification of tasks performed by stroke patients using a mobility assistive device. In *Engineering in Medicine and Biology Society, 2006. EMBS '06. 28th Annual International Conference of the IEEE*, pages 1501 –1504, 30 2006-sept. 3 2006.

[29] V. Jakkula and D.J. Cook. Enhancing smart home algorithms using temporal relations. In *Technology and Aging, IOS Press*, 2008.

[30] Vikramaditya R. Jakkula, Aaron S. Crandall, and Diane J. Cook. Enhancing anomaly detection using temporal pattern discovery. In Achilles D. Kameas, Victor Callagan, Hani Hagras, Michael Weber, and Wolfgang Minker, editors, *Advanced Intelligent Environments*, pages 175–194. Springer US, 2009.

[31] V.R. Jakkula, A.S. Crandall, and D.J. Cook. Knowledge discovery in entity based smart environment resident data using temporal relation based data mining. In *Data Mining Workshops, 2007. ICDM Workshops 2007. Seventh IEEE International Conference on*, pages 625 –630, oct. 2007.

[32] Julie A. Kientz, Shwetak N. Patel, Brian Jones, Ed Price, Elizabeth D. Mynatt, and Gregory D. Abowd. The georgia tech aware home. In *CHI '08 extended abstracts on Human factors in computing systems*, CHI EA '08, pages 3675–3680, New York, NY, USA, 2008. ACM.

[33] Dae-Jin Kim, R. Lovelett, and A. Behal. An empirical study with simulated adl tasks using a vision-guided assistive robot arm. In *Rehabilitation Robotics, 2009. ICORR 2009. IEEE International Conference on*, pages 504 –509, june 2009.

[34] Eunju Kim, Sumi Helal, and Diane Cook. Human activity recognition and pattern discovery. *IEEE Pervasive Computing*, 9(1):48–53, 2010.

[35] Sang Wan Lee, Yong Soo Kim, Kwang-Hyun Park, and Zeungnam Bien. Iterative bayesian fuzzy clustering toward flexible icon-based assistive software for the disabled. *Inf. Sci.*, 180(3):325–340, February 2010.

[36] Victor Lesser, Michael Atighetchi, Brett Benyo, Bryan Horling, Victor Lesser Michael Atighetchi, Anita Raja, Regis Vincent, Ping Xuan, Shelley XQ. Zhang, Thomas Wagner, Ping Xuan, and Shelley Xq. Zhang. The intelligent home testbed. In *Proceedings of the Autonomy Control Software Workshop*, 1999.

[37] Lin Liao, Donald J. Patterson, Dieter Fox, and Henry Kautz. Learning and inferring transportation routines. *Artif. Intell.*, 171(5-6):311–331, 2007.

[38] J.M. Mendel. *Uncertain rule-based fuzzy logic systems: introduction and new directions*. Prentice Hall PTR, 2001.

[39] Xiaoning Meng, Ka Keung Lee, and Yangsheng Xu. Human driving behavior recognition based on hidden markov models. *IEEE International Conference on Robotics and Biomimetics*, 0:274–279, 2006.

[40] Michael C. Mozer. *Lessons from an Adaptive Home*, chapter 12, pages 271–294. John Wiley & Sons, Inc., 2005.

[41] U. Naeem and J. Bigham. A comparison of two hidden markov approaches to task identification in the home environment. In *Proceedings of the 2nd International Conference on Pervasive Computing and Applications*, pages 624–634, 2007.

[42] C.D. Nugent, M. D. Mulvenna, F. Moelaert, B. Bergvall-Kereborn, F. Meiland, D. Craig, R. Davies, A. Reinersmann, M. Hettinga, A-L. Andersson, R-M. Dries, and J. Bengtsson. Home-based assistive technologies for people with mild dementia. In *Pervasive Computing for Quality of Life Enhancement*, volume 4541, pages 63–69, 2007.

[43] Matthai Philipose, Kenneth P. Fishkin, Mike Perkowitz, Donald J. Patterson, Dieter Fox, Henry Kautz, and Dirk Hahnel. Inferring activities from interactions with objects. *IEEE Pervasive Computing*, 3:50–57, October 2004.

[44] Martha E. Pollack, Laura Brown, Dirk Colbry, Colleen E. McCarthy, Cheryl Orosz, Bart Peintner, Sailesh Ramakrishnan, and Ioannis Tsamardinos. Autominder: an intelligent cognitive orthotic system for people with memory impairment. *Robotics and Autonomous Systems*, 44(3):273 – 282, 2003.

[45] Alexander S. Poznyak and Kaddour Najim. *Learning Automata and Stochastic Optimization*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1997.

[46] Marilyn J. Rantz, Myra A. Aud, Gregory Lynn Alexander, Debra Oliver, De Manner, Marge Skubic, James M. Keller, Zhihai He, Mihail Popescu, George Demiris, and Steve Miller. Tigerplace: An innovative educational and research environment. In *AAAI in Eldercare: New Solutions to Old Problems*, 2008.

[47] Parisa Rashidi and Diane J. Cook. Keeping the resident in the loop: adapting the smart home to the user. *Trans. Sys. Man Cyber. Part A*, 39:949–959, September 2009.

[48] M. Ros, M.P. Cuéllar, M. Delgado, and A. Vila. Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows. *Information Sciences*, -:–, 2012.

[49] M. Ros, M. Delgado, and A. Vila. An ambient intelligent approach to control behaviours on a tagged world. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5518 LNCS(PART 2):764–771, 2009.

[50] M. Ros, M. Delgado, and A. Vila. A system for recognizing activities of daily living using everyday objects. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6279 LNAI(PART 4):337–346, 2010.

[51] M. Ros, M. Delgado, and A. Vila. A system to supervise behaviours using temporal and sensor information. In *2010 IEEE World Congress on Computational Intelligence, WCCI 2010*, 2010.

[52] M. Ros, M. Delgado, and A. Vila. Fuzzy method to disclose behaviour patterns in a tagged world. *Expert Systems with Applications*, 38(4):3600–3612, 2011.

[53] M. Ros, M. Delgado, A. Vila, Hani Hagras, and Aysenur Bilgin. A fuzzy logic approach for learning daily human activities in an ambient intelligent environment. In *IEEE International Conference on Fuzzy Systems*, page In press, 2012.

[54] M. Ros, M. Pegalajar, M. Delgado, A. Vila, D.T. Anderson, J.M. Keller, and M. Popescu. Linguistic summarization of long-term trends for understanding change in human behavior. In *IEEE International Conference on Fuzzy Systems*, pages 2080–2087, 2011.

[55] M. Ros, M. Pegalajar, M. Delgado, A. Vila, and Hani Hagras. Detection of abnormal human activities by means of learning automata. *Transactions on Systems, Man and Cybernetics – Part A: Systems and Humans*, -:–, 2011. Submitted to.

[56] María Ros, Miguel Delgado, and Amparo Vila. A fuzzy approach for the model of sliding window. an application to behaviour patterns mining. In *IFSA/EUSFLAT Conf.*, pages 1211–1216, 2009.

[57] Fariba Sadri. Ambient intelligence: A survey. *ACM Comput. Surv.*, 43(4):36:1–36:66, October 2011.

[58] N. Sahab and H. Hagras. Adaptive non-singleton type-2 fuzzy logic systems: A way forward for handling numerical uncertainties in real world applications. *International Journal of Computers Communications & Control*, 6(3):503–529, 2011. ISSN 1841-9836.

[59] G Singla, DJ Cook, and M. Schmitter-Edgecombe. Tracking activities in complex settings using smart environment technologies. *Int J Biosci Psychiatr Technol.*, 1(1):25–35, Jan 2009.

[60] E.E. Stone, D. Anderson, M. Skubic, and J.M. Keller. Extracting footfalls from voxel data. In *Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE*, pages 1119 –1122, 31 2010-sept. 4 2010.

[61] Emmanuel M. Tapia, Stephen S. Intille, and Kent Larson. Activity recognition in the home using simple and ubiquitous sensors. In Alois Ferscha and Friedemann Mattern, editors, *Pervasive Computing*, volume 3001 of *Lecture Notes in Computer Science*, pages 158–175–175. Springer Berlin / Heidelberg, 2004.

[62] M.A.L. Thathachar and P.S. Sastry. Varieties of learning automata: an overview. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 32(6):711 – 722, dec 2002.

[63] T. van Kasteren and B. Krose. Bayesian activity recognition in residence for elders. *IET Conference Publications*, 2007(CP531):209–212, 2007.

[64] J.A. Ward, P. Lukowicz, G. Troster, and T.E. Starner. Activity recognition of assembly tasks using body-worn microphones and accelerometers. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(10):1553 –1567, oct. 2006.

[65] M. Weiser. The computer for the 21st century, 1991. Last access 2012-01-22.

[66] Lotfi A. Zadeh. Outline of a new approach to the analysis of complex systems and decision processes. *Systems, Man and Cybernetics, IEEE Transactions on*, SMC-3(1):28 –44, jan. 1973.

[67] Dong Zhang, Daniel Gatica-Perez, Samy Bengio, Iain McCowan, and Guillaume Lathoud. Modeling individual and group actions in meetings with layered hmms. *IEEE Trans. on Multimedia*, 8:509–520, 2004.