IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans, Vol.35, No.1, pp.55-65, January 2005.

An Intelligent Fuzzy Agent Approach for Realising Ambient Intelligence in Intelligent Inhabited Environments

Faiyaz Doctor, Hani Hagras, Victor Callaghan

Department of Computer Science, University of Essex, Wivenhoe Park, Colchester, CO4 3SQ, UK

Email: fdocto@essex.ac.uk

Abstract---In this paper we describe a novel life long learning approach for intelligent agents that are embedded in intelligent environments. The agents aim to realise the vision of Ambient Intelligence in Intelligent Inhabited Environments (IIE) by providing 'ubiquitous computing intelligence in the environment supporting the activities of the user. An unsupervised, data-driven, fuzzy, technique is proposed for extracting fuzzy membership functions and rules that represent the user's particularised behaviours in the environment. The user's learnt behaviours can then be adapted online in a life long mode to satisfy the different user and system objectives. We have performed unique experiments in which the intelligent agent has learnt and adapted to the user's behaviour, during a stay of five consecutive days in the intelligent Dormitory (iDorm) which is a real ubiquitous computing environment test bed. Both offline and online experimental results are presented comparing the performance of our technique with other approaches. The results show that our proposed system has outperformed the other systems while operating online in a life long mode to realise the ambient intelligence vision.

Index terms--- Fuzzy Systems, Learning, Ambient Intelligence, Intelligent Inhabited Environments, Agents, Ubiquitous computing environments.

1. Introduction

The envisaged widespread proliferation of networked embedded computer products into people's everyday lives is driving research into how such technology could be deployed

without incurring prohibitive cognitive loading on the users to create ubiquitous computing environments [13]. Ubiquitous computing, alternatively referred to as pervasive computing, is a paradigm in which computing technology becomes virtually invisible by being embedded in our environments.

As embedded computers get smaller and become integrated into non-computing artefacts, they effectively can physically disappear. With the addition of communication capability to these artefacts and the use of pervasive networking, such artefacts can be associated together and remotely accessed in both familiar and novel arrangements to make highly personalised systems. The challenge however is how to manage, program or direct such systems, a task that could quickly become prohibitive and an obstacle to the achievement of the pervasive computing. The vision of *ambient intelligence* can help to address this challenge [9, 19].

Ambient Intelligence is a new information paradigm where people are empowered through a digital environment that is "aware" of their presence and context, and is *sensitive, adaptive and responsive* to their needs [9]. Ambient intelligence improves the quality of life through creating desired environmental conditions and functionality via intelligent, personalised interconnected systems and services [9]. An environment with ambient intelligence can be characterised by its ubiquity, transparency and intelligence [9]. It is ubiquitous because the user is surrounded by a multitude of inter-connected embedded systems which form a pervasive infrastructure. Its transparency is due to the invisible nature of the computing based artefacts being seamlessly integrated into the surrounding environment. Its intelligence spawns from the fact that the technology is able to recognise the users and program itself to their needs by learning from their behaviour. In addition, environments constructed in such a way would be adaptive to changing conditions and user preferences.

As was suggested above for the vision of ambient intelligence to be realised people have to be able to use and configure the computer-based artefacts and systems present in their ubiquitous environments in a seamless, unobtrusive and non-intrusive way; without being

cognitively overloaded by having to program each device and workout how to connect them together to achieve the desired functionality [9]. One approach to achieve this vision of ambient intelligence is to embed intelligent agents in the user environments so that they can control them according to the needs and preferences of the user. Embedded intelligence is the inclusion of some capacity for reasoning, planning and learning in an artefact. Embedded-computers that contain this kind of intelligent capacity are normally referred to as "embedded-agents" [6]. Each embedded agent is an autonomous entity, and as was mentioned earlier, in a pervasive computing environment it is common for such embedded-agents (as intrinsic parts of intelligent artefacts") to have network connections allowing them to communicate and cooperate with other embedded agents, as part of a multi embedded agent system [6].

In this paper, we will present a novel fuzzy learning and adaptation technique for agents that can be embedded in ubiquitous computing environments. This we hope will be a step towards the realisation of the vision of ambient intelligence [9]. Each agent is connected to sensors and effectors comprising a ubiquitous computing environment. The intelligent learning mechanism used would learn and predict the needs of the user and automatically adjust the agent controller based on a wide set of parameters [5] in a non-intrusive and invisible way, satisfying one of the main requirements for ambient intelligent systems [9]. Due to the fact that these agents need to be located on small embedded computers with limited processor and memory capacities, our learning and adaptation system is a one pass method which does not require heavy computation.

Most automation systems that have minimal intelligence utilise mechanisms that generalise actions across a population e.g. set temperature or loudness that is an averaged compromise of the needs of a group of individuals. However in achieving the vision of ambient intelligence, we argue that any type of intelligence applied to personal artefacts and spaces needs to particularise to the individual [6]. Furthermore it is essential that any agent serving a person, should always and immediately carry out any requested actions, i.e. people are always in control, subject to overriding safety considerations, to achieve the responsive property implied

in the ambient intelligence vision [9]. So our intelligent agent will seek to provide an online, life-long, personalised learning of anticipatory adaptive control supporting the vision of ambient intelligence in Intelligent Inhabited Environments (IIE).

We have performed unique experiments in which our intelligent agent has learnt and adapted to the behaviour of a user who spend five consecutive days in the Essex intelligent Dormitory (iDorm) which is a real ubiquitous computing environment test bed.

The rest of this paper is organised as follows: In Section (2), we will introduce Intelligent Inhabited Environments. In section (3) we describe our test bed for ubiquitous environments; the iDorm. In section (4) a detailed description of the proposed life-long learning technique is presented. Section (5) presents results on offline comparison of the proposed technique with three other soft-computing approaches, and results obtained from the online performance of the agent in the iDorm. Section (6) provides some concluding remarks and future research directions.

2. Intelligent Inhabited Environments

Intelligent Inhabited Environments (IIE) are enclosed living spaces (such as a car, shopping mall, home or even our own body) equipped with embedded intelligent technology that would allow the environment to respond "thoughtfully" to the users' needs. These environments could consist of a multitude of possibly disconnected active spaces supporting ambient intelligence and providing ubiquitous access to system resources according to the current situation of the user. These intelligent environments will personalise themselves in responses to our presence and behaviour. Precursors to such environments can be found now in Intelligent Buildings (IBs) [6]. The heterogeneity, dynamism and context-awareness in a building make it useful for exploring the design challenges concerned with ubiquitous systems. One view of an IB is a computer-based system, gathering information from a variety of sensors (and other computers), and using intelligent mechanisms to determine the appropriate control actions of various

devices [6, 11, 10]. In controlling such systems one is faced with several challenges as there are a large number of information sources within such environments which these systems must monitor and use to learn and adapt to the needs of the user. In addition, the sensors themselves may be imprecise due to noise or the quality of the sensor, and the system must be able to compensate for these inaccuracies. Moreover, there is a lack of adequate models for many of the processes present in such systems. Also, these systems have to contend with the dynamic non-deterministic aspects of human behaviour.

Currently there have been several research projects concerned with applying intelligent agents to IIE. In Sweden Davidsson [4] utilised multi-agent principles to control building services. In Colorado Mozer [18] used a soft computing approach based on neural networks which was focused on the intelligent control of lighting within a building. Work at MIT in the HAL project [8] concentrated on making a room responsive to the occupant by adding intelligent sensors to the user interfaces. Context Aware systems are the focus of the Aware Home work at Georgia Tech [1]. These projects represent a large body of current research effort; however they are mostly concerned with time independent context rather than temporal history, learning or adaptation which is central to our requirements for agents supporting the vision of ambient intelligence [9]. There are other high profile intelligent environments projects such as the Microsoft Smart House, BT's Tele-care and Cisco's Internet Home [20]. However most of these industrial projects including home automation technologies, like Lonworks and X10 are geared toward using networks and remote access with some autonomous control. In general, this is mostly simple automaton with sparse use of Artificial Intelligence (AI) and little emphasis on learning and adaptation to the user's behaviour. In our previous work we developed an Incremental Synchronous Learning technique for an embedded agent [12]; however, it learnt only the user rules and not all the parameters of the embedded agent controller.

As mentioned earlier, we will use the Essex intelligent Dormitory (iDorm) as a test-bed for IIE and ubiquitous computing environments and thus we describe this environment in the following section.

3. The Essex intelligent Dormitory (iDorm)

The iDorm shown in Fig 1 is test bed for ubiquitous and pervasive computing environments. It looks like any other room and it comprises of a large number of embedded sensors, actuators, processors and a heterogeneous network [13].

Fig. 1. The iDorm.

The iDorm is multi-user space containing areas for different activities such as sleeping, working and entertaining. It contains the normal mix of furniture, found in a typical student study/bedroom environment, including a bed, work desk and a wardrobe. There is a standard multi-media PC that combines a flat screen monitor and a multi-media video projector which can be used for both working and entertainment as shown in Fig 2. Any networked computer that can run a standard Java process can access and control the iDorm directly, thus this PC can also act as an interface to control the devices in the room as shown in Fig 3a. Equally the interface to the devices could be operated from physically portable computational artefacts that can monitor and control the iDorm wirelessly. The handheld PDA (Compaq iPAQ) shown in Fig 3b contains a standard Java process that can access and control the iDorm directly. Because the iPAQ supports Bluetooth wireless networking, it was possible to adjust the environment

from anywhere inside and nearby outside the room, this forms a type of "remote control" interface that would be particularly suitable to elderly and disabled users.

Fig. 2. Video Projector and Multi-Media PC in the iDorm.

It is also possible to interact with the iDorm through mobile phones as shown in Fig 3c using a WAP interface which is a simple extension of a web interface. There is also an internet Fridge in the iDorm provided by LG consumer electronics that incorporates an intelligent user friendly server with touch screen capability, which can also be used to control the devices in the room, see Fig 3d.

Our agent learning mechanism and interface currently operates from the standard multi-media PC in the iDorm. It is possible however for our agent to be embedded into any part of the environment. In terms of software the cross platform versatility of the Java programming language which the agent was written with, could allow it to be embedded onto internet devices. By embedding agents into such devices and integrating wireless communications (including wireless based interfaces, such as PDAs), this will lead to the kind of pervasive transparent infrastructure characteristic of an ambient intelligent system.

The iDorm is fitted with a liberal placement of sensors which include the following: internal light level sensor, external light level sensor, internal temperature sensor, external temperature sensor, humidity sensor, chair Pressure sensor, bed pressure sensor, occupancy sensor, telephone status sensor.

a) b) c) d)

Fig. 3. a) PC based interface. b) Portable iPAQ interface. c) Mobile phone interface. d) iFridge interface.

The effectors one can control in the room consist of the following: heater, cooler fan, four dimmable spot lights, window blinds, desk lamp, bedside lamp, PC based Word processing application, PC based Media playing application.

The sensors and actuators in the room are concealed (e.g. buried in walls) with the intention that the user should be completely unaware of the intelligent infrastructure of the room which is required by the ambient intelligence vision [9].

The iDorm is based around three networks, Lonworks, 1-wire (TINI) and IP which provide a diverse infrastructure allowing the development of network independent solutions. Lontalk is Echelon's proprietary network protocol and forms a standard for building automation networks. There are many commercially available sensors and actuators for this system. The physical network installed in the iDorm is the Lonworks TP/FP10 network running Lontalk protocol. The gateway to the IP network is provided by Echelon's iLON 1000. This allows the states and values of sensors and actuators to be read or altered via Echelon's proprietary LNS server. The majority of the sensors and effectors inside the iDorm are connected via a Lonworks network. The 1-Wire network, developed by Dallas semiconductor was designed for simple devices to be connected over short distances. It offers a wide range of commercial devices including small temperature sensors, weather stations, ID buttons and switches. The 1-Wire network is

connected to a host of Tiny Internet Interface boards (TINI board), which run an embedded web server serving the status of the networked devices using a Java servlet. The servlet collects data from the devices on the network and responds to HTTP requests. The IP network forms a backbone to interconnect all networks and other devices like the multi-media PC.

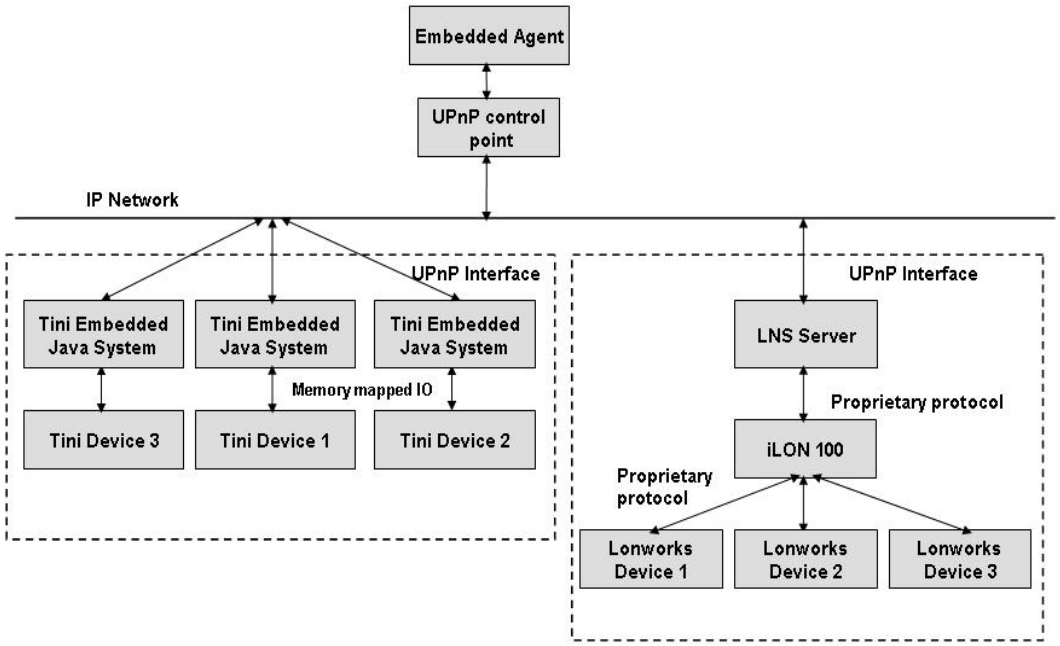

Fig. 4. The iDorm logical network infrastructure.

A common interface to the iDorm and its devices is implemented through Universal Plug & Play (UP&P) which is an event-based communication middleware for allowing devices to be plug & play enabling automatic discovery and configuration. A gateway server is used to run the UP&P software devices that interface the hardware devices on their respective networks. The agent implementing our learning and adaptation mechanism was built on top of the low level UP&P control architecture enabling it to communicate with the UP&P devices in the iDorm and thus allowing it to monitor and control these devices. Fig 4 shows the logical network infrastructure of the iDorm.

4. Proposed Architecture

The task of designing an intelligent agent to effectively fulfil the needs of the user in IIE is akin to finding a solution to a highly challenging control problem. The environment within which the agent must operate can be viewed as a very complex control system, in which the user controlling it forms an essential part. The environment facing the human controller is so complicated that any mathematical model, if it exists, is strongly non-linear. In addition, the human controller, in their own right, is largely non-deterministic and a highly individual part of this system. The task here is to design an intelligent control system to realise the ambient intelligence vision [9] and control the environment on behalf of the human user [21].

Fuzzy logic offers a framework for representing the imprecise and uncertain knowledge of the real world. It has similarities with the way people make decisions as it uses a model of approximate reasoning that allows it to deal with vague and incomplete information. Fuzzy controllers also exhibit robustness with regard to noise and variations of the system parameters. Fuzzy Logic Controllers (FLCs) are widely used in engineering and control application. A FLC is a model free approach which converts linguistic control information into mathematical control information and can represent a non-linear mapping of inputs to outputs. FLCs also provide transparent and flexible representations which can be easily adapted due to the ability of fuzzy rules to approximate independent local models for mapping a set of inputs to a set of outputs.

Our proposed technique is an unsupervised data-driven one-pass approach for extracting fuzzy rules and membership functions from data to learn a fuzzy controller that will model the user's behaviours. The data is collected by monitoring the user in the environment over a period of time. The learnt FLC provides an inference mechanism that will produce output control responses based on the current state of the inputs. Our adaptive FLC will therefore control the environment on behalf of the user and will also allow the rules to be adapted online

as the user's behaviour drifts over time. Our proposed approach aims to realise the vision of Ambient Intelligence in the following ways:

- The agent is responsive to the particular needs and preferences of the user.
- The user is always in control and can override the agent at any time.
- The agent learns and controls its environment in a non-intrusive way (although the user may be aware of the high-tech interface, he is unaware of the agent's presence).
- The agent uses a simple one pass learning mechanism for learning the user's behaviours, and thus it is not computationally expensive.
- The agent's learnt behaviours can be adapted online as a result of changes in the occupant's behaviour.
- Learning is life-long in that agent behaviours can be adapted and extended over a long period of time as a result of changes in the environment.

These features fulfil many of the requirements for the ambient intelligence vision defined by the Information Society Technologies Advisory Group (ISTAG) to the European Commission [9]. For the reminder of this paper the proposed approach will be referred to as an Adaptive Online Fuzzy Inference System (AOFIS).

AOFIS comprises of five phases: Monitoring the user's interactions and capturing input/output data associated with their actions, extraction of the fuzzy membership functions from the data, extraction of the fuzzy rules from the recorded data, the agent control and the life long learning and adaptation mechanism. The last two phases are control loops that once initiated receive inputs as either: monitored sensor changes that produce appropriate output control responses based on the set of learnt rules; or user action requests that cause the learnt rules to be adapted before an appropriate output control response is produced. Fig 5 illustrates a flow chart of these five phases.

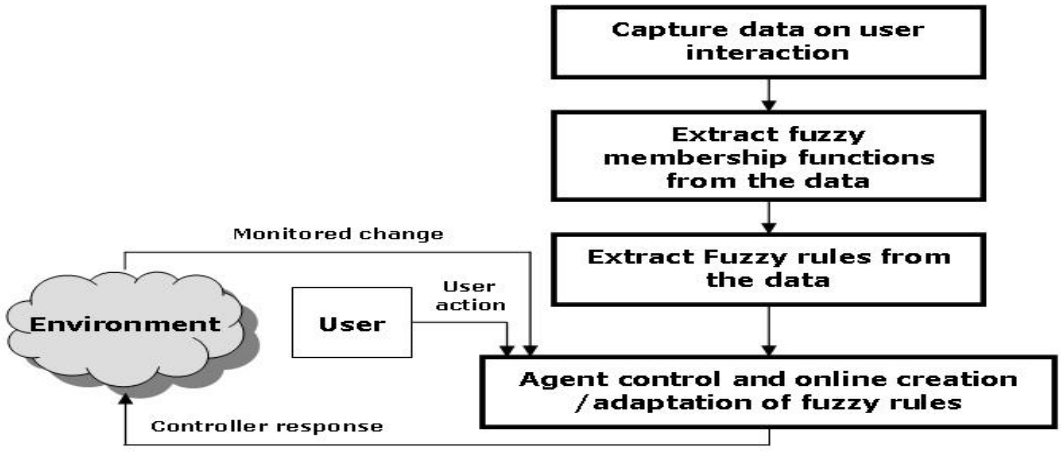

Fig. 5. Flow diagram showing five phases of AOFIS.

4.1 Capturing input/output data

The agent initially monitors the user's actions in the environment. Whenever the user changes actuator settings, the agent records a 'snapshot' of the current inputs (sensor states) and the outputs (actuator states with the new altered values of whichever actuators were adjusted by the user). These 'snapshots' are accumulated over a period of time (three days the in case of our experiments) so that the agent observes as much of the user's interactions within the environment as possible. AOFIS learns a descriptive model of the user's behaviours from the data accumulated by the agent. Therefore given a set of multi-input multi-output data pairs:

$$(x^{(t)}; y^{(t)}), \quad t = 1,2,..., N \tag{1}$$

where N is the number of data instances, $x^{(t)} \in R^n$ and $y^{(t)} \in R^k$. AOFIS extracts rules which describe how the k output variables $y = (y_1,..., y_k)$ are influenced by the n input variables $x = (x_1,..., x_n)^T \in R^n$ based on the sampled data. In our experiments in the iDorm we used 7 sensors for our inputs and 10 actuators for our outputs. The fuzzy rules which are extracted represent local models that map a set of inputs to the set of outputs without the need for formulating any mathematical model. Individual rules can therefore be adapted online influencing only specific parts of the descriptive model learnt by the agent.

4.2 Fuzzy membership function extraction

It is necessary to be able to categorise the accumulated user input/output data into a set of fuzzy membership functions which quantify the raw crisp values of the sensors and actuators into linguistic labels such as normal, cold or hot. AOFIS is based on learning the particularised behaviours of the user and therefore requires these membership functions be defined from the user's input/output data recorded by the agent. A Double Clustering approach [7] combining Fuzzy-C-Means (FCM) and hierarchical clustering, is used for extracting fuzzy membership functions from the user data. This is simple and effective approach to fuzzy information granulation [22, 16] where the objective is to build models at a certain level of information granulation that can be quantified in terms of fuzzy sets.

4.2.1 The FCM algorithm

The FCM [3] is a fuzzy partitioning prototype-based clustering algorithm. It takes a dataset of instances represented by r attributes that have been arbitrarily classified on a pre-determined number of clusters reflecting a p partitioning of the dataset. The algorithm then calculates the centres for each cluster from the arbitrarily classified instances. Using these centres, the distances of each instance from each cluster centre is calculated and used to assign each instance with a degree of membership to each cluster. In this way the partitioning of the dataset becomes fuzzified. The algorithm aims to iteratively adapt the partitioning of the dataset so as to minimise a dissimilarity function of a weighted sum of squared errors between data points and cluster centres in the feature space. FCM uses features in the r-dimensional Euclidean space to determine the geometric closeness of data points by grouping them into clusters and then determining the distance between those clusters [3]. FCM allows operations on fuzzy data in which a data point may have a degree of fuzzy membership to more than one cluster at any given time. In our case the number of dimensions r was 17 corresponding to number of input and output parameters (7 input sensors and 10 output actuators) that were used in the iDorm.

4.2.2 Double clustering

The Double clustering technique uses a combination of FCM and Hierarchical clustering for extracting a predefined number of membership functions for the input and output parameters from the sampled user data. An initial clustering of the dataset is performed using the FCM algorithm that defines a set of p clustered regions over the sampled data. Hence there are p centres $\bar{c}_1, \bar{c}_2, ..., \bar{c}_p \subseteq \mathbf{R}^r$ defined for these clustered regions. The number of clusters p is predefined and in our case was set to 90. Each centre is an r-dimensional vector $\bar{c}_i = (c_{i1}, c_{i2}, ..., c_{ir})$ and there are therefore p one-dimensional centroid values for each input and output parameter of the user data. The centroid values for each separate input and output dimension are then iteratively clustered again to form a new set of centres which represent the rough centres of the membership functions that will be extracted for each input and output parameter. Specifically let c_{ij} be the j-th component of the i-th cluster centre. For each dimension $j = 1,2,..., r,$ we perform clustering on the set of one-dimensional centroid values $C_j := \{c_{ij} | i = 1,2,..., p\}$. The approach used for this secondary clustering is an agglomerative hierarchical clustering approach [14]. Here the elements in C_j are sequentially clustered together reducing the number of elements at each step by merging together the two most similar consecutive elements. This is repeated until the number of elements corresponds to the number of membership functions we want to extract for each input and output parameter. The similarity between two elements is measured based on the closeness between their values. The number of membership functions to be defined for each input and output parameter is predefined in advance.

4.2.3 Agglomerative hierarchical clustering approach

The agglomerative hierarchical clustering algorithm used in AOFIS can be formally described as follows: Let K_j $(j = 1,2,..., r)$ represents the required number of centres and the

corresponding number of membership functions to be derived from each set C_j $(j = 1,2,...,r)$, where K_j is fixed for each input and output dimension r. The elements of C_j are initially sorted such that $i_1 < i_2 \rightarrow c_{i_1 j} \leq c_{i_2 j}$. Hence the initial set of elements in C_j is defined as:

$$pr^{(0)} := \left\{pr_1^{(0)}, pr_2^{(0)},..., pr_p^{(0)}\right\} := \left\{c_{1j}, c_{2j},..., c_{pj}\right\} \tag{2}$$

For $v = 1,2,..., p - K_j$ find the two nearest consecutive elements in $pr^{(v-1)}$, denoted by $pr_{i^*}^{(v-1)}$ and $pr_{i^*+1}^{(v-1)}$.

Define the new set of elements as $pr^{(v)} := \left\{pr_1^{(v)}, pr_2^{(v)},..., pr_{p-v}^{(v)}\right\}$,

$$pr_i^{(v)} := \begin{cases} pr_i^{(v-1)}, i < i^* \\ \left(pr_{i^*}^{(v-1)} + p_{i^*+1}^{(v-1)}\right)/2, i = i^* \\ pr_{i+1}^{(v-1)}, i > i^* \end{cases} \tag{3}$$

Until: $pr^{(p-K_j)} := \left\{pr_1, pr_2,..., pr_{K_j}\right\}$

Therefore at each step of the algorithm the two nearest consecutive elements are merged into a single cluster where the new centre of the cluster is the average of the two merged elements. After the hierarchical clustering is completed on each set C_j $(j = 1,...,r)$, we have derived K_j new centres for each input and output dimension of the dataset. We represent the set of these centres by $pr := pr^{(p-K_j)} =: \left\{pr_1, pr_2,..pr_{K_j}\right\}$ which correspond to the rough centres for the fuzzy membership functions that AOFIS will extract for each input and output parameter.

4.2.4 Quantification of fuzzy membership functions

The K_j cluster centres defined on each dimension $j = 1,2,...,r$ are then converted to fuzzy membership functions, which involves the quantification of the centres in terms of interpretable fuzzy sets [7]. As mentioned the value of K_j defines the number of fuzzy membership functions which are to be extracted for each input and output parameter. Gaussian

membership functions are used to describe the fuzzy sets A_z^j, (where $z = 1,2,...,K_j$) the mathematical definition of which is

$$\mu_{A_z^j}(x) = \exp\left\{-\frac{1}{2}\left(\frac{x - w_z^j}{\sigma_z^j}\right)^2\right\} \tag{4}$$

where the value of the centre w_z^j and the spread σ_z^j for each gaussian membership function *z*, for the *j*-th input/output parameter is derived as follows

The sampled data is defined as a hyper-interval $X := \underset{j=1}{\overset{r}{\times}}[m_j, M_j]$ where m_j and M_j are the minimum and maximum values respectively, of the *j*-th input/output dimension of the sampled dataset. The set of cuts T_j is defined as $T_j := \{t_0^j, t_1^j, ..., t_{K_j}^j\}$, where:

$$t_d^j := \begin{cases} 2m_j - t_1^j, & d = 0 \\ (pr_d + pr_{d+1})/2, & 0 < d < K_j \\ 2M_j - t_{K_j-1}^j & d = K_j \end{cases} \tag{5}$$

The centre w_z^j and spread σ_z^j of each membership function A_z^j for all $z = 1,2,...,K_j$ is derived from the set T_j as follows:

$$w_z^j := \left(t_{z-1}^j + t_z^j\right)/2 \tag{6}$$

$$\sigma_z^j := (t_z^j - t_{z-1}^j)/2\sqrt{-2\ln\varepsilon} \tag{7}$$

where ε is the maximum overlap between two adjacent fuzzy sets.

We therefore obtain the centres and spreads for a set of K_j fuzzy membership functions defined for each input and output parameter of the user data that was sampled. These membership functions are distributed over the range of values of each parameter. The membership functions at the boundaries are modified such that they are extended indefinitely beyond their respective centres with a membership value of 1. A semantic meaning can be

associated with each of the resulting fuzzy sets. Specifically depending on the value of index *z*, a meaningful symbolic label can be given to A_z^j.

4.3 Fuzzy Rule Extraction

The defined set of membership functions are combined with the existing user input/output data to extract the rules defining the user's behaviours. The fuzzy rule extraction approach used by AOFIS is based on an Enhanced version of the Mendel Wang (MW) method [21] developed by L.X. Wang. This is a one pass technique for extracting fuzzy rules from the sampled data. The fuzzy sets for the antecedents and consequents of the rules divides the input and output space into fuzzy regions.

AOFIS extracts multi-input multi-output rules which describe the relationship between $y = (y_1,..., y_k)$ and $x = (x_1,...,x_n)^T$, and take the following form:

$$IF\ x_1 \text{ is } A_1^{(l)} \ and \ldots and\ x_n \text{ is } A_n^{(l)}, THEN\ y_1 \text{ is } B_1^{(l)} \ \ and \ldots and \ \ y_k \text{ is } B_k^{(l)} \qquad (8)$$

$l = 1,2,...,M$, where *M* is the number of rules and *l* is the index of the rules. There are V fuzzy sets $A_s^q, q = 1,...,V,$ defined for each input x_s. There are *W* fuzzy sets $B_c^h, h = 1,...,W,$ defined for each output y_c. AOFIS now extracts rules in the form of Equation (8) from the data.

4.3.1 Process of extracting fuzzy rules from data

To simplify the following notation, the method for rules with a single output is shown, as the approach is quite easily expanded to rules with multiple outputs. In the following steps we will show the different steps involved in rule extraction:

Step 1: For a fixed input-output pair $(x^{(t)}; y^{(t)})$ in the dataset (1) ($t = 1,2,...,N$), compute the membership values $\mu_{A_s^q}(x_s^{(t)})$ for each membership function $q = 1,...,V$, and for each input variable s $(s = 1,...,n)$, find $q^* \in \{1,...,V\}$, such that

$$\mu_{A_s^{q^*}}(x_s^{(t)}) \geq \mu_{A_s^q}(x_s^{(t)}) \tag{9}$$

for all $q = 1,...,V$.

Let the following rule be called the rule generated by $(x^{(t)}; y^{(t)})$:

$$IF\ x_1^t \text{ is } A_1^{q^*}\ and \ldots and\ x_n^t \text{ is } A_n^{q^*},\ THEN\ y\ is\ centred\ at\ y^{(t)} \tag{10}$$

For each input variable x_s there are V fuzzy sets $A_s^q, q = 1,...,V$, to characterise it; so that the maximum number of possible rules that can be generated is V^n. However given the dataset only those rules among the V^n possibilities whose dominant region contains at least one data point will be generated. In step 1 one rule is generated for each input –output data pair, where for each input the fuzzy set that achieves the maximum membership value at the data point is selected as the one in the IF part of the rule, as explained in Equations (9),(10).

This however is not the final rule which will be calculated in the next step. The weight of the rule is computed as

$$w^{(t)} = \prod_{s=1}^{n} \mu_{A_s^q}(x_s(t)) \tag{11}$$

The weight of a rule $w^{(t)}$ is a measure of the strength of the points $x^{(t)}$ belonging to the fuzzy region covered by the rule.

Step 2: Step 1 is repeated for all the *t* data points from 1 to *N* to obtain *N* data generated rules in the form of Equation (10). Due to the fact that the number of data points is quite large, many rules are generated in step 1, that all share the same IF part and are conflicting, i.e. rules with the same antecedent membership functions and different consequent values. In this step rules with the same *IF* part are combined into a single rule.

The *N* rules are therefore divided into groups, with rules in each group sharing the same IF part. If we assume that there is *M* such groups. Let group *l* have N_l rules in the following form:

$$IF\ x_1 \text{is}\ A_1^{(q^l)}\ and \ldots and\ x_n\ \text{is}\ A_n^{(q^l)}, THEN\ y\ is\ centred\ at\ y^{(t_u^l)} \tag{12}$$

Where $u = 1,...N_l$ and t_u^l is the index for the data points in group *l*. The weighted average of all the rules in the conflict group is then computed as

$$av^{(l)} = \frac{\sum_{u=1}^{N_l} y^{(t_u^l)} w^{(t_u^l)}}{\sum_{u=1}^{N_l} w^{(t_u^l)}} \tag{13}$$

We now combine these N_l rules into a single rule of the following form:

$$IF\ x_1 \text{is}\ A_1^{(l)}\ and \ldots and\ x_n\ \text{is}\ A_n^{(l)}, THEN\ y\ is\ B^{(l)} \tag{14}$$

Where the output fuzzy set B^l is chosen based on the following. Among the *W* output fuzzy sets $B^1,...,B^W$ find the B^{h*} such that

$$\mu_{B^{h*}}(av^{(l)}) \geq \mu_{B^h}(av^{(l)}) \tag{15}$$

for $h = 1,2,...,W$, B is chosen as B^{h*}.

As mentioned above AOFIS deals with input-output data pairs with multiple outputs. Step 1 is independent of the number of outputs for each rule. Step 2 is simply expanded to allow rules to have multiple outputs where the calculations in Equations (13) and (15) are repeated for each output value.

4.4 Agent controller

Once the agent has extracted the membership functions and the set of rules from the user input/output data, it has then learnt the FLC that captures the human behaviour. The agent FLC can start controlling the environment on behalf of the human according to his desires. The agent starts to monitor the state of the environment and affect actuators based on its learnt FLC that approximate the particularised preferences of the user. Fig 6 shows a block diagram of the FLC which consists of a fuzzifier, rule base, fuzzy inference engine and defuzzifier.

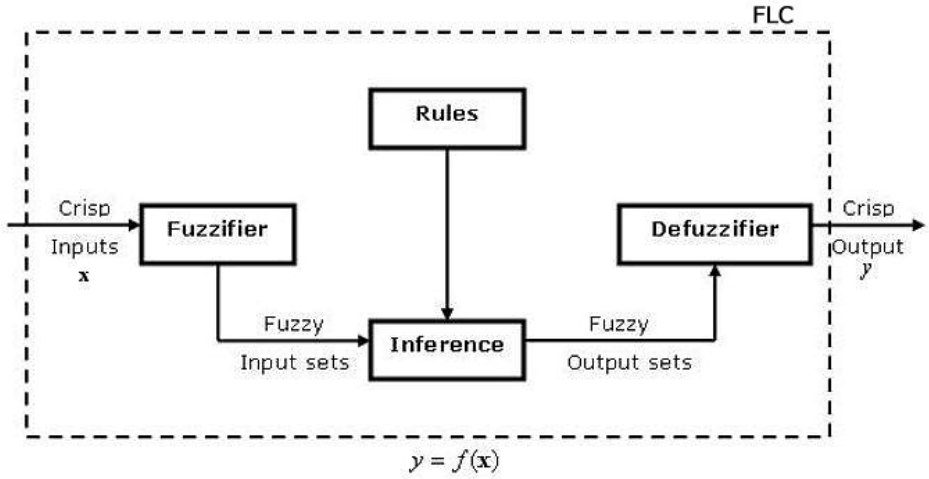

Fig. 6. Block Diagram of the FLC.

In our agent we will use singleton fuzzification, max-product composition, product implication, and height defuzzification [17]. This formula that maps a crisp input vector **x** into a crisp output $y = f(\mathbf{x})$ can be written as follows [17]:

$$y(\mathbf{x}) = f_s(\mathbf{x}) = \frac{\sum_{l=1}^{M} \bar{y}^l \prod_{i=1}^{n} \mu_{F_i^l}(x_i)}{\sum_{l=1}^{M} \prod_{i=1}^{n} \mu_{F_i^l}(x_i)} \tag{16}$$

Where M is the total number of rules in the rule base, $\bar{y}^l$ is the point having the maximum membership in the l^{th} rule output fuzzy set B^l, $\prod_{i=1}^{n} \mu_{F_i^l}(x_i)$ is the product of the membership functions of each rule's inputs and n is the number of inputs.

Equation (16) encapsulates the process of following the signal **x** through the fuzzifier, where it is converted to the input fuzzy sets and then into the inference block where it is converted to the output fuzzy sets and finally into the defuzzifier where it is converted to a crisp output $y = f(\mathbf{x})$. For multiple outputs the Equation in (16) is repeated for each output parameter.

The fuzzy sets used for the antecedent and consequent parameters are gaussian membership functions which can be calculated as in Equation (4) and have been learnt in subsection (4.2) and can be written as follows :

$$\mu_{F_i^l}(x_i) = \exp\left\{-\frac{1}{2}\left(\frac{x_i - m_{F_i^l}}{\sigma_{F_i^l}}\right)^2\right\} \quad (17)$$

where $i = 1,...,n$ and $l = 1,...,M$, $m_{F_i^l}$ is the centre of the fuzzy set and $\sigma_{F_i^l}$ is the spread of this fuzzy set.

4.5. Online adaptation and life long learning

In the previous subsections we have shown how our agent can learn an FLC that approximates the user's behaviour. However, the user may need to make adjustments to tune the system or their behaviour might change as the user requirements change over time. So our agent needs to adapt to the user's behavioural changes in a non intrusive manner and in a short time interval.

In realising the non-intrusive aspect of ambient intelligence [9] whenever the user is not happy with the agent's actions, he can always override the agent's control responses by simply altering the manual control of the system. When this occurs the agent will adapt its rules online or add new rules based on the new user preferences. This process could also incorporate what we term as 'learning inertia' where the agent would delay adapting its learnt rules until the user preference for changing a particular set of actuator values has reoccurred several times. This would prevent the agent adapting its rules in response to 'one off' user actions that don't reflect a marked change in the user's habitual behaviour. As the rules are adapted it would be necessary to preserve the old rules so they can be recalled by the agent in the future.

Whenever the user overrides the agent's control responses and actuates any of the controlled output devices, a snapshot of the state of the environment is recorded and passed to the rule adaptation routine. Each input parameter in the input vector **x** is compared to each of the antecedent sets $A_s^{(l)}$ of a given rule in the rule base to determine its membership value. The weight of the rule is then calculated to determine if the product of the input membership function (degree of firing of the rule) in Equation (11) $w^{(l)} > 0$, meaning that the rule fired, and would therefore have contributed to the overall control response generated by the agent's FLC.

The consequent membership functions that give the highest membership values to the user defined actuator values are selected to replace the consequent sets of all fired rules in the rule base.

$$\mu_{B_c^{h^*}}(y_c) \geq \mu_{B_c^h}(y_c) \tag{18}$$

for $h = 1,2,...,W$. The B_c is chosen as $B_c^{h^*}$. Where *c=1,2..,k.*

The fired rules are therefore adapted to better reflect the user's updated actuator preferences given the current state of the environment.

If none of the existing rules fired, new rules are added based on forming rules from the input fuzzy sets. For each input parameter x_s the fuzzy sets that give a membership value where $\mu_{A_s^q}(x_s^{(t')}) > 0$ are identified. This leads to a grid of identified fuzzy set(s) for each input parameter. From this grid new rules are constructed based on each unique combination of consecutive input fuzzy sets. The consequent fuzzy sets for each of the new rules are determined using Equation (18). This allows new rules to be gradually added to the rule base. The agent will also add new rules when the currently monitored environmental state is undefined by the existing rules in the rule base; i.e. none of the existing rules fired. In this case the agent will create new rules where the antecedent sets reflect the current input states of the environment and the consequent fuzzy sets are based on the current state of the actuators.

The agent adopts life long learning where it adapts its rules as the state of the environment and the preferences of the user change over a significantly long period of time. Due to the flexibility of AOFIS the initially learnt FLC can be easily extended to change both existing rules as well as add new rules. The fuzzy nature of the rules permits them to capture a wide range values for each input and output parameter. This allows the rules to continue to operate even if there is a gradual change in the environment. If however there is a significant change in the environment or the user's activity which is no longer captured by the existing rules, then as

mentioned previously the agent will automatically create new rules that satisfy the current conditions. The agent will therefore unobtrusively and incrementally extend its behaviours which can then be adapted to satisfy the user.

5. Experimental Results

We have performed unique experiments in which a user (shown in Fig 7) lived in the iDorm for a period of five consecutive days. During the monitoring phase which lasted for three consecutive days the agent recorded the user interactions with the environment. On each day the user's activity in the room was recorded. Seven input sensors were monitored which are: internal light level, external light level, internal temperature, external temperature, chair pressure, bed pressure and time measured as a continuous input on an hourly scale. Ten output actuators were controlled consisting of the four variable intensity spot lights, the desk and bed side lamps, window blinds, the heater and the two PC based applications comprising of a word processing program and a media playing program. The outputs thus covered the spectrum of physical devices and computer based applications found in a typical study bedroom environment.

Fig. 7. User in the iDorm.

The data from the iDorm that was captured during the monitoring phase was used to compare the offline performance of our approach with three other soft-computing based techniques which are Genetic Programming (GP), the Adaptive-Neuro Fuzzy Inference System

(ANFIS) [15] and the Multi-Layer Perceptron Neural Network.

The dataset obtained from the iDorm during the monitoring phase comprised of 408 instances and was randomised into six samples. Each sample was then split into a training and test set consisting of 272 and 136 instances respectively. The performance error for each technique was obtained on the test instances as the Root Mean Squared Error which was also scaled to account for the different ranges of the output parameters.

The GP used a population of 200 individuals evolving them over 200 generations. The GP evolved both the rules and the fuzzy sets. Each individual was represented as a tree composed of 'and' and 'or' operators as the internal nodes and triangular and trapezoidal membership functions as terminal nodes. The parameters of the membership functions were also evolved in parallel with the structure. The search started with a randomly generated set of rules and parameters, which were then optimised by means of genetic operators. The GP based approach for optimising an FLC was tested with different numbers of fuzzy sets.

In ANFIS subtractive clustering is used to generate an initial TSK-type fuzzy inference system. Back propagation is used to learn the premise parameters while least square estimation is used to determine the consequent parameters. An iteration of the learning procedure consists of two parts where the first part propagates the input patterns and estimates optimal consequent parameters through an iterative least squares procedure. The second part uses back propagation to modify the antecedent membership functions [2]. We tested ANFIS with a range of different cluster radii values.

The Multi-Layer Perceptron (MLP) back-propagation Neural Network was tested with different numbers of hidden nodes in a single hidden layer.

We tested our AOFIS with different number of fuzzy sets and the membership function overlap threshold was set to 0.5 as this gave both a sufficient degree of overlap while allowing the system to distinguish between the ranges covered by each fuzzy set. This value was also used for evaluating the double clustering approach presented in [7]. Tables 1 and 2 illustrate

the scaled Root Mean Squared Error (RMSE) and scaled Standard Deviation (STD) for each technique averaged over the six randomised samples, and corresponding to the values of the variable parameter tested for each approach.

Table I

Average Scaled RMSE For AOFIS, GA, ANFIS & MLP

Average Scaled Root Mean Squared Error (SRMSE) for six randomised sample of the dataset							
AOFIS		GA		ANFIS		MLP	
Num of fuzzy sets	SRMSE	Num of fuzzy sets	SRMSE	Cluster Radii	SRMSE	Num of hidden nodes	SRMSE
2	0.2148	2	0.1235	0.3	1.3269	2	0.2129
3	0.1476	3	0.1156	0.4	0.9229	4	0.1718
4	0.1461	4	0.1189	0.5	0.2582	6	0.1732
5	0.1364	5	0.1106	0.6	0.1661	8	0.1571
6	0.1352	6	0.1210	0.7	0.1669	10	0.1555
7	0.1261	7	0.1193	0.8	0.1418	20	0.1621
8	0.1326	8	0.1173	0.9	0.1213	30	0.1705
9	0.1472	9	0.1202	1.0	0.1157	40	0.1667
10	0.1537	10	0.1235	1.1	0.1201	50	0.1768
11	0.1696	11	0.1110	1.2	0.1168	60	0.1711
12	0.1999	12	0.1201	1.3	0.1131	70	0.1712
13	0.2246	13	0.1169	1.4	0.1131	80	0.1770
14	0.2337	14	0.1120	1.5	0.1118	90	0.1767
15	0.2460	15	0.1089	1.6	0.1130	100	0.1924
16	0.2459	16	0.1225	1.7	0.1115	200	0.2027
17	0.2732	17	0.1146	1.8	0.1137	300	0.2258
18	0.2747	18	0.1188	1.9	0.1182	400	0.2365
19	0.2771	19	0.1159	2.0	0.1189	500	0.2424
20	0.2839	20	0.1143				

Table II

Average Scaled STD For AOFIS, GA, ANFIS & MLP

Average scaled standard deviation (SSTD) for six randomised sample of the dataset							
AOFIS		GA		ANFIS		MLP	
Num of fuzzy sets	SSTD	Num of fuzzy sets	SSTD	Cluster Radii	SSTD	Num of hidden nodes	SSTD
2	0.1896	2	0.1128	0.3	1.2839	2	0.1499
3	0.1350	3	0.1063	0.4	0.9001	4	0.1299
4	0.1354	4	0.1094	0.5	0.2440	6	0.1277
5	0.1277	5	0.1026	0.6	0.1522	8	0.1193
6	0.1280	6	0.1121	0.7	0.1518	10	0.1160
7	0.1200	7	0.1107	0.8	0.1257	20	0.1198
8	0.1266	8	0.1085	0.9	0.1038	30	0.1229
9	0.1409	9	0.1117	1.0	0.0972	40	0.1234
10	0.1472	10	0.1145	1.1	0.1007	50	0.1245
11	0.1626	11	0.1026	1.2	0.0961	60	0.1234
12	0.1912	12	0.1115	1.3	0.0920	70	0.1222
13	0.2133	13	0.1084	1.4	0.0924	80	0.1283
14	0.2218	14	0.1031	1.5	0.0906	90	0.1272
15	0.2323	15	0.1007	1.6	0.0911	100	0.1333
16	0.2318	16	0.1128	1.7	0.0891	200	0.1366
17	0.2557	17	0.1063	1.8	0.0909	300	0.1503
18	0.2568	18	0.1090	1.9	0.0951	400	0.1674
19	0.2588	19	0.1075	2.0	0.0937	500	0.1676
20	0.2646	20	0.1051				

The results above show that the optimum number of fuzzy sets for AOFIS is 7 and on average AOFIS generated 186 rules. The GP in comparison gives a marginally lower error for

7 fuzzy sets. Both ANFIS and the MLP on average give a higher error than AOFIS. The ANFIS only learns a Multi-Input Single Output (MISO) FLC and had to be run repeatedly for each output parameter. The FLC produced is therefore only representative of a MISO system. Another restriction with ANFIS is that it generates TSK FLCs [2] where the consequent parameters are represented as either linear or constant values, rather than linguistic variables as is the case with Mamdani FLCs [17], these linguistic variables are very important to understanding the human behaviour. Our AOFIS generates Multi-Input, Multi-Output (MIMO) Mamdani FLCs that represent the rules in a more descriptive human readable form required for an ambient intelligent system.

The iterative nature of the GP makes it highly computationally intensive and this also applies to both ANFIS and the MLP which are also iterative based approaches. AOFIS is far less computationally intensive due to the one-pass procedure it employs, and is therefore more favourable for an embedded agent. Both ANFIS and the GP based approach cannot easily be adapted online as this would require their internal structures to be re-learnt if either new rules were to be added or existing rules were adapted. So our method is unique in that it can learn a good model of the user's behaviour which can then be adapted online in a life long mode in a non intrusive manner, unlike other methods which need to repeat a time consuming learning cycle to adapt the user's behaviour.

The online performance of the agent was evaluated on how well AOFIS could capture the monitored state of the environment and the user's behaviours in the iDorm, over a period of time. The data from the user's interactions in the iDorm acquired over the initial period of three days was used by AOFIS to learn an initial FLC. A fuzzy membership function overlap threshold of 0.5 was used and the number of fuzzy sets representing the input and output parameters was set to 7 as this was found to be the optimum number from the offline

experiments as shown in Table 1,2. Fig 8 show the membership functions which AOFIS produced for one of the input parameters, namely the external light level.

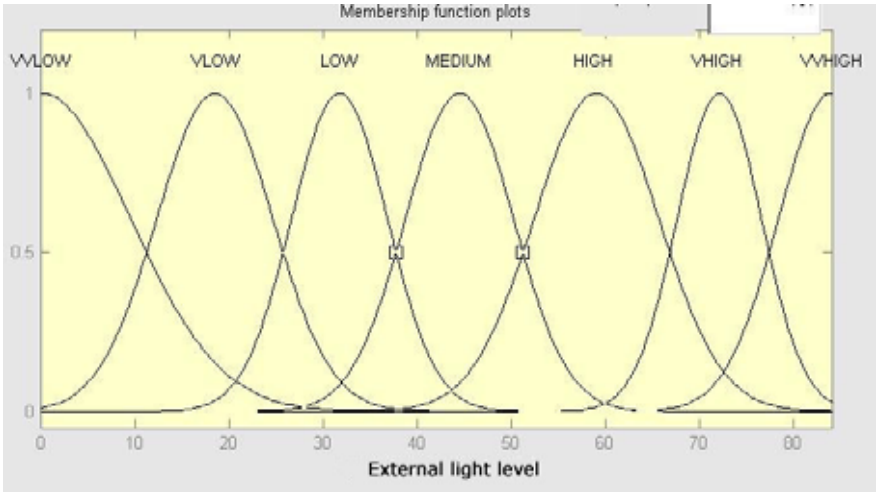

Fig. 8. Membership functions for external light level.

The agent was then run online for a further two days during which it monitored the environment and user's activities, and it produced the appropriate control responses based on its learnt rules. During this time the user was allowed to override and adapt the agent's learnt control responses, if it was necessary to modify and tune them further. As mentioned previously, one of the characteristics of the agent is that the user is always in control and he can override the agent at any time and his instructions are executed immediately, to achieve the responsive property implied in the ambient intelligence vision unless safety is compromised. Thus whenever changes to controls were made by the user, the agent received the request, generated new rules or adjusted previously learnt rules and allowed the action through. The agent would autonomously continue to monitor the environment and generate new rules when the state of the environment was not captured by its existing rule base.

The performance of the agent could be gauged on the number of occasions when the user had to override the agent's control responses and adapt the rules over time, as this can reflect

how satisfied the user is with the agent control actions. The success of the agent could be measured by monitoring how well it adjusted the environment to the user's preferences such that the user intervention was reduced over time. Fig 9 shows a graph plotting the number of online rule adaptations against time measured in minutes.

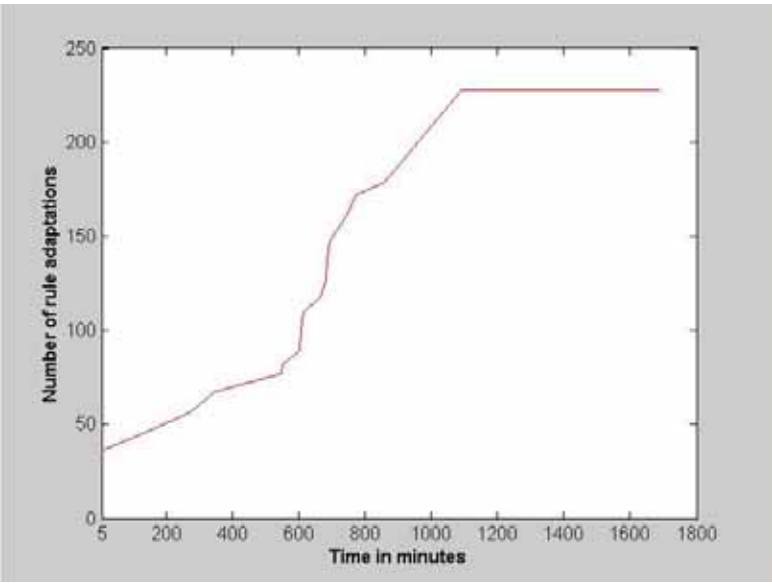

Fig. 9. Number of online rule adaptations.

The agent initially learnt 186 rules and that over the course of the two days 120 new rules were added. Fig 9 shows the number of user induced rule adaptations that occurred over the course of the two days. This can be seen to stabilise by early afternoon on the second day. One of the concerns of using our one pass approach was its potential to generate too many rules. However given the theoretical maximum number of rules that the agent could generate, the total of 306 rules produced after the online execution of the agent was well below this number. The agent was able to learn in a non intrusive way most of the user's preferences for various weather and environmental conditions over the duration of the two days, including specific behaviours associated with user activity such as lying on the bed and listening to music or sitting at the desk to word process a document. An example of the types of rules that the agent produced are shown in Fig 10.

From the experiments we can deduce that the agent has tried to realise the vision of ambient intelligence as it was intelligent and it learnt the user particularised behaviour and adapted it online to any changes in a life long learning mode in a non intrusive way. The agent was also responsive to the user commands. In addition, the intelligent environment in the iDorm was transparent and ubiquitous in that the pervasive interconnected embedded systems were seamlessly integrated into it. The user was therefore unaware of the invisible intelligently responsive infrastructure of the environment.

InternalLightLevel *is* VVLOW *AND* ExternalLightLevel *is* VVLOW *AND*
InternalTemperature *is* VVHIGH *AND* ExternalTemperature *is* MEDIUM *AND*
ChairPressure *is* OFF *AND* BedPressure *is* ON *AND* Hour is Evening *THEN*
ACTION_Light1_value *is* VHIGH *AND* ACTION_Light2_value *is* HIGH *AND*
ACTION_Light3_value *is* LOW *AND* ACTION_Light4_value *is* VVLOW *AND*
ACTION_Blind_state *is* CLOSED *AND* ACTION_BedLight_state *is* ON *AND*
ACTION_DeskLight_state *is* OFF *AND* ACTION_Heater_state *is* OFF *AND*
ACTION_MSWord_state *is* STOPPED *AND* ACTION_MSMediaPlayer_state *is*
RUNNING

Fig. 10. Typical rule produced by AOFIS.

6. Conclusion

In this paper we presented a novel fuzzy learning and adaptation technique for agents that can be embedded in ubiquitous computing environments. This we hope will be a step towards the realisation of the vision of ambient intelligence.

Our agent learnt a FLC that modelled the user's particularised behaviour and it was adaptive as it allowed the learnt behaviours to be modified and extended online and in a life-long learning mode as the user's activity and environmental conditions changed over time. The intelligent learning and adaptation occurred in a non intrusive manner while the user carried out his normal activities in the environment. The agent was always responsive to the user's commands. The iDorm environment was also transparent and ubiquitous in that the pervasive

infrastructure of the interconnected embedded systems was seamlessly integrated into it. The user was therefore surrounded by an invisible though intelligently responsive ambience.

Our technique was a simple one-pass method and thus it is not computationally expensive and could be incorporated in many embedded devices within intelligent environments.

We carried out unique experiments in which a user stayed in the iDorm for five consecutive days. The proposed AOFIS technique was compared with other soft-computing based approaches; namely a GP, ANFIS and an MLP; using data acquired from the iDorm. The results showed that the optimum performance of AOFIS produced on average a lower error than both ANFIS and the MLP, and was computationally less intensive and better suited to online learning than the other approaches compared. The online operation of the agent showed that AOFIS was effective at both learning the behaviours of a user and adapting and tuning its rules online to meet the user's preferences, without incurring any kind of cognitive loading on the user.

In our future work we intend to perform more experiments with multiple users and multiple occupancy and we also intend to extend the current system which is based on a type-1 FLC into a type-2 FLC [17].

References

[1] G. Abowd, M. Eibling, G. Hunt, H. Lei. "Context Aware Computing," *IEEE Pervasive Computing,* Volume 1, No 3, pp. 22-23, 2002.

[2] A. Abraham, "Analysis of Hybrid Soft and Hard Computing Techniques for Forex Monitoring Systems," *2002 IEEE International Conference on Fuzzy Systems (IEEE FUZZ'02), 2002 IEEE World Congress on Computational Intelligence,* pp. 1616-1622, 2002.

[3] J. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms,* Plenum New York, 1981.

[4] M. Boman, P. Davidsson, N. Skarmeas, K. Clark, and R. Gustavsson, "Energy Saving and Added Customer Value in Intelligent Buildings," *In Third International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology (PAAM'98)*, pages 505-517, 1998.

[5] A. Bonarini, "Comparing Reinforcement Learning Algorithms Applied to Crisp and Fuzzy Learning Classifier Systems," *Proceeding of the Genetic and Evolutionary Computation Conference,* pp. 52-60, Jul, 1999.

[6] V. Callaghan, M. Colley, G. Clarke, H. Hagras, "Embedded Intelligence: Research Issues for Ubiquitous Computing," *Proceedings of the Ubiquitous Computing in Domestic Environments conference, Nottingham,* Sep, 2001.
[7] G. Castellano, A. M. Fanelli, C. Mencar, "Generation of Interpretable Fuzzy Granules by a Double-Clustering Technique," *Archives of Control Science, Special Issue on Granular Computing,* 2002 to appear.
[8] M. Coen. "Design Principles for Intelligent Enviornments," *In Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI'98),* 1998.
[9] K. Ducatel, M. Bogdanowicz, F. Scapolo, J-C. Burgelman, "Senarios For Ambient Intelligence in 2010," *IPTS, Seville,* Feb, 2001.
[10] H. Hagras, V. Callaghan, M. Colley, G. Clarke, "A Hierarchical Fuzzy Genetic Multi-Agent Architecture for Intelligent Buildings Learning, Adaptation and Control," *The International Journal of Information Sciences,* Volume 150, pp. 33-54, Mar, 2003.
[11] H. Hagras, V. Callaghan, M. Colley, G. Clark, H. Duman. "Online Learning and Adaptation for Intelligent Embedded Agents Operating in Domestic Environments," *In the book entitled Fusion of Soft Computing and Hard Computing for Autonomous Robotic Systems. (Eds.: Zhou, Dario Maravall and Da Ruan), Studies in Fuzziness and Soft Computing Series, Physica-Verlag,* Volume 116, pp. 293-323, Nov, 2002.
[12] H. Hagras, M. Colley, V. Callaghan, G. Clark, H. Duman, A. Holmes. "A Fuzzy Incremental Synchronous Learning Technique for Embedded-Agents Learning and Control in Intelligent Inhabited Environments," Proceedings *of the 2002 IEEE International Conference on Fuzzy systems, pp. 139-145, Hawaii-USA, 2002.*
[13] A. Holmes, H. Duman, A. Pounds-Cornish, "The iDorm: Gateway to Heterogeneous Networking Environments," *International ITEA Workshop on Virtual Home Environments,* 2002.
[14] A. K. Jain, R. C. Dubes. *Algorithms for clustering data*, Prentice Hall, 1998.
[15] J-S. R. Jang, "ANFIS: Adaptive-Network-Based Fuzzy Inference System," *IEEE Trans. On Systems, Man and Cybernetics,* vol. 23, pp. 665-684, May/Jun, 1993.
[16] S. Kobashi, N. Kamiura, Y. Hata, F. Miyawaki, "Fuzzy Information Granulation on Blood Vessel Extraction from 3D TOF MRA Image," *International Journal of Pattern Recognition and Artificial Intelligence,* vol. 14, No. 4, pp. 409-425, 2002.
[17] J. M. Mendel, *Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions,* Prentice Hall PTR, Prentice Hall Inc, 2001.
[18] M. Mozer, "The Neural Network House: An Environment That Adapts to its Inhabitants," *In Proc of American Association for Artificial Intelligence Spring Symposium on Intelligence Environments,* pp. 110-114, AAAI Press, 1998.
[19] G. Riva, P. Loreti, M. Lunghi, F. Vatalaro, F. Davide, "Presence 2010: The Emergence of Ambient Intelligence," *Ios Press Amsterdam, , The Netherlands,* 2003.
[20] A. Sherwin, "Internet Home Offers a Life of Virtual Luxury," *The Times,* pp. 10, 3rd Nov, 1999.
[21] L. X. Wang, J. M. Mendel, "Generating Fuzzy Rules By Learning From Examples," *IEEE Trans. On Systems Man and Cybernetics*, vol. 22, pp. 1414-1427, Nov/Dec, 1992c.
[22] L. A. Zadeh, "Fuzzy Sets and Information Granularity," *In Gupta M. M., Ragade R. K., and Yager R. R, eds., Advances in Fuzzy Sets Theory and Applications,* pp. 3-18, Amsterdam, 1979.