In Proceedings of the 2004 World Automation Conference, pp. 101-106, Seville, Spain, July 2004

An Adaptive Fuzzy Learning Mechanism for Intelligent Agents in Ubiquitous Computing Environments

FAIYAZ DOCTOR, HANI HAGRAS and VICTOR CALLAGHAN
Department of Computer Science, University of Essex, Wivenhoe Park, Colchester, CO4 3SQ, UK
fdocto@essex.ac.uk, hani@esex.ac.uk and vic@essex.ac.uk

ABSTRACT

In this paper we describe a novel system for learning and adapting fuzzy controllers for intelligent agents that are embedded in ubiquitous computing environments to support the activities of the user. We have performed unique experiments in which the intelligent agent has learnt and adapted online 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 learning mode.

KEYWORDS: Fuzzy Systems, Learning, Intelligent Buildings, Adaptive Systems, Agents, Ubiquitous computing environments

INTRODUCTION

Ubiquitous computing also referred to as pervasive computing, is a paradigm in which computing technology becomes virtually invisible by being embedded in our environments. These environments will contain networked embedded computer artefacts that can interact with the users living or working within them. The challenge however is how to manage and configure the computer-based artefacts and systems present in these ubiquitous environments in a seamless, unobtrusive and non-intrusive way; without the user being cognitively overloaded by having to manually configure these devices to achieve a desired functionality. Embedded intelligent mechanisms can go some way to achieve this goal.

Embedded intelligence is the inclusion of some capacity for reasoning, planning and learning in an artefact. Embedded-computers containing this kind of intelligent capacity are referred to as "embedded-agents". Each embedded agent is an autonomous entity in a pervasive computing environment.

In this paper, we will present a novel system for learning and adapting fuzzy controllers for agents that can be embedded in ubiquitous computing environments. Each agent is connected to sensors and effectors integrated within the environment. The intelligent learning mechanism learns the particularised needs of the user and adjusts the agent controller based on a wide range of parameters in a non-intrusive and invisible way. It is also able to adapt online to changing conditions and user preferences in a life-long learning mode. Our technique is a one pass method which does not require heavy computation so it is suitable for embedded computers which have limited computational abilities.

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) shown in Figure 1. This is test bed for ubiquitous and pervasive computing environments comprising of a large number of embedded sensors, actuators, processors and a heterogeneous network [3].

Figure 1: the iDorm.

ADAPTIVE ONLINE FUZZY INFERENCE SYSTEM (AOFIS)

Our proposed Adaptive Online Fuzzy Inference System (AOFIS) technique is an unsupervised data-driven one-pass approach for extracting fuzzy rules and membership functions from data, to learn a Fuzzy Logic Controller (FLC) 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 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. AOFIS comprises of five phases which are illustrated in Figure 2.

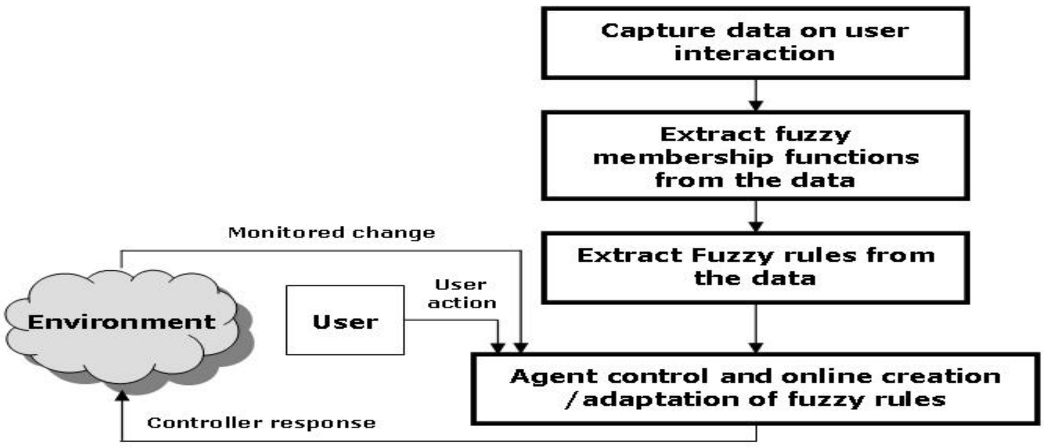

Figure 2: Flow diagram showing five phases of AOFIS.

Phase 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 in the 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.

Phase 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 [2] combining Fuzzy-C-Means [1] and hierarchical clustering [4], is used for extracting fuzzy membership functions from the user data.

The Double clustering technique uses a combination of Fuzzy-C-Means [1] 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 $\overline{c}_1, \overline{c}_2,..., \overline{c}_p \subseteq \mathbf{R}^r$ 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 $\overline{c}_i = (c_{i1}, c_{i2},..., c_{ir})$, in our case the number of dimensions r was 17 corresponding to number of input and output parameters (7 sensors inputs and 10 actuators outputs) that were used in the iDorm. Therefore there are 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 [4]. 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.

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.

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 [2]. 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 $X := \times_{j=1}^{r}\left[m_j, M_j\right]$ 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 $T_j := \left\{t_0^j, t_1^j, ..., t_{K_j}^j\right\}$, 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, so depending on the value of index *z*, a meaningful symbolic label can be given to A_z^j.

Phase 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 [7] 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:

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

$l = 1,2,..., M$, 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.

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, where n is the total number of input variables. 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.

Phase 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. In our agent we use singleton fuzzification, max-product composition, product implication, and height defuzzification [6]. The fuzzy sets used for the antecedent and consequent parameters are gaussian membership functions of the type shown in Equation (4).

Phase 5: Online Adaptation

In the previous phases we have shown how our agent can learn a 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.

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 functions (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.

$$\mu_{B_c^{h*}}(y_c) \geq \mu_{B_c^h}(y_c) \tag{16}$$

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 (16). 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 extends its rules as the state of the environment and user activity change over a significantly long period of time.

EXPERIMENTAL RESULTS

We have performed unique experiments in which a user 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. 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.

The data from the iDorm that was captured during the monitoring phase was used to compare the offline performance of AOFIS with three other soft-computing based techniques which are Genetic Programming (GP), the Adaptive-Neuro Fuzzy Inference System (ANFIS) [5] and the Multi-Layer Perceptron Neural Network. The dataset 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 offline 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. We tested our AOFIS with different number of fuzzy sets and the overlap between the membership functions 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 [2]. Figure 3a illustrates the Scaled Root Mean Squared Error (SRMSE) for each technique averaged over the six randomised samples, and corresponding to the values of the variable parameter tested for each approach.

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				

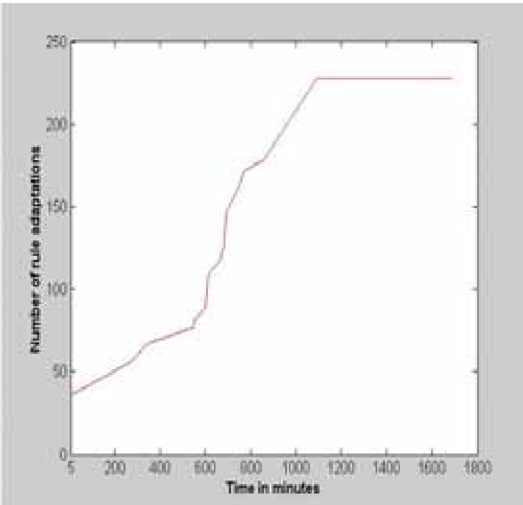

Figure 3 a): Average SRMSE For AOFIS, GA, ANFIS & MLP. **b): Number of online rule adaptations against time in minutes.**

The offline results above show that the optimum number of fuzzy sets for AOFIS is 7 and on average AOFIS generated 155 rules from the 272 training instances. 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 iterative nature of the compared approaches makes them more computationally intensive than the one pass AOFIS technique which makes it suitable for embedded agents. The other approaches cannot easily be adapted online as this would require their internal structures to be re-learnt every time either new rules were 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.

The online performance of the agent was evaluated on how well AOFIS could model the user's behaviour from their observed activity that had been recorded over the initial three days of monitoring. The performance of the learnt FLC could then be gauged online in its ability to control the environment and satisfy the preferences of the user. The agent was therefore 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 FLC. During this time the user could override and adapt the agent's learnt control responses, if it was necessary to modify and tune them further. The agent could also autonomously add new rules to its rule base.

The online performance 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. This can be shown in Figure 3b which plots the number of online rule adaptations against time measured in minutes that occurred over the course of the two days. This initially shows the user intervention to be high but seems to stabilise by early afternoon on the second day. The agent initially started with the 186 rules it learnt from the 408 training instances accumulated during the three days of monitoring the user. Over the course of the subsequent two days 120 new rules were added. The number of fuzzy sets representing the input and output parameters were set to 7 which was the optimum number derived from the offline experiments. The agent was therefore able to learn and adapt in a non intrusive way to most of the user's preferences for various 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.

CONCLUSION

In this paper we presented a novel system for learning and adapting fuzzy controllers for agents that can be embedded in ubiquitous computing environments. 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.

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. AOFIS allowed online learning and was computationally less intensive and better suited for embedded intelligence 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.

REFERENCES

1. J. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms,* Plenum New York, (1981).
2. 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.
3. Holmes, H. Duman, A. Pounds-Cornish, "The iDorm: Gateway to Heterogeneous Networking Environments," *International ITEA Workshop on Virtual Home Environments,* (2002).
4. K. Jain, R. C. Dubes. *Algorithms for clustering data*, Prentice Hall, (1998).
5. 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).
6. J. M. Mendel, *Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions,* Prentice Hall PTR, Prentice Hall Inc, (2001).
7. 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, (1992).