In the IEEE Transactions on Fuzzy Systems, Vol. 15, No.1, pp.41-55, February 2007

An Incremental Adaptive Life Long Learning Approach for Type-2 Fuzzy Embedded Agents in Ambient Intelligent Environments

Faiyaz Doctor*, Hani Hagras*, Antonio Lopez**, Victor Callaghan*

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

**The Electrical Engineering Department, University of Oviedo,
Campus Viesques, Ed. 2, 33204, Gijon, Asturias, Spain

Abstract-----**In this paper, we present a novel adaptive embedded agent architecture for Ambient Intelligent Environments (AIEs) that is based on interval type-2 fuzzy systems. Type-2 fuzzy systems are able to handle the different sources of uncertainty and imprecision encountered in AIEs to give a very good response. The presented agent architecture uses a one pass method to learn in a non intrusive manner the user's particular behaviours and preferences for controlling the AIE. The agent learns the user's behaviour by learning his particular rules and type-2 Membership Functions (MFs) required by the type-2 fuzzy agent, these rules and MFs can then be adapted online incrementally in a life long learning mode to suit the changing environmental conditions and user preferences. The presented type-2 agent architecture is suited for the embedded platforms used in AIEs which have limited computational and memory capabilities. Moreover, we will show that the type-2 agents generated by our one pass learning technique outperforms those generated by the Genetic Algorithms (GAs). We will present unique experiments carried out by different users over the course of the year in which the type-2 agent has learnt and adapted to the user behaviour and to the short and long term environmental variations while the users stayed in the intelligent Dormitory (iDorm) which is a real AIE test bed. We will show how our type-2 agents can deal with the short term and long term uncertainties in AIEs to give a very good performance that outperforms the type-1 fuzzy agents while using smaller rule bases.**

Index Terms---**Interval Type-2 Fuzzy Logic Systems, Learning, Ambient Intelligent Environments, Embedded Agents.**

I. INTRODUCTION

Over the last several years there has been an accelerated growth in embedded computational artefacts fuelled by the recent advances in microelectronics, networks and internet technologies. Recent figures show that about 8 billion microprocessors were produced with only 2% of them going into PCs [33]. The remaining 98 % ended up as part of the pervasive fabric of computing that's being woven around and through our lives via a wide range of tiny embedded computers integrated within various artefacts. Such artefacts include mobile phones, home entertainment systems, fridges, washing machines, kitchen appliances, security systems, cars, transport systems and even our clothes and furniture [34]. Most of these artefacts are network enabled and thanks to pervasive networking such artefacts can communicate and collaborate together to support our lives. As the myriad of embedded computers get smaller and are

networked and integrated into non-computing artefacts, they can effectively physically disappear within the structure of our inhabited spaces forming *ubiquitous computing environments* [34]. However the multitude of interconnected devices, networks and services can create inherent complexities in programming and configuring the ubiquitous environments to personalise themselves to suit the individual needs which is a task that can quickly cognitively overburden the user. A solution to this problem can be found by embedding intelligent agents [4], [15] into ubiquitous computing environments.

Embedded agents [4], [15] are embedded computational artefacts integrated with intelligent reasoning and learning mechanisms. The embedded agents are able to personalise themselves to the user's needs and preferences by learning from their behaviour and thus configuring and controlling the user's environments on their behalf. Thus these agents can reduce the cognitive load associated with configuring and programming ubiquitous computing environments. Due to the limited processing power and memory capacities of embedded computers, the intelligent mechanisms that are used in embedded-agents have to be computationally undemanding. Moreover, the intelligent approaches applied should have transparent internal representations such that their learnt behaviours and decisions can be represented in the form of human readable rules which are accessible to the end user. The coupling of ubiquitous computing environments with intelligent embedded agents leads us towards environments which possess Ambient Intelligence [11].

Ambient intelligence is a new information paradigm where the user is empowered through a digital environment that is "aware" of his presence and context, and is *sensitive*, *adaptive* and *responsive* to his needs in an unobtrusive way [11]. Ambient Intelligent Environments (AIEs) consist of a multitude of interconnected embedded systems which form a pervasive infrastructure that surround the user. The computer based artefacts making up this pervasive infrastructure are transparent and seamlessly integrated into the surrounding environment. Intelligent agents are embedded into AIEs to form an intelligent "presence" allowing the AIE to recognize the user and program itself to his needs by learning from his behaviours in a non intrusive way [11]. In addition these intelligent agents must be responsive and adaptive to the changing environmental conditions and user preferences [11].

There are many research efforts in the area of AIEs. In [30] a system based on machine vision has been presented for extracting and maintaining awareness of a range of events and human activities in indoor spaces such as meeting rooms or performance spaces. In [2] a system was presented for monitoring the elderly in their environment using a virtual maid. An interactive space which learns to influence the user behaviours was outlined in [12]. In a wider context there are also several research efforts worldwide on applications of smart environments. Work at Georgia Tech has focused on context aware systems with ubiquitous sensing capabilities as was shown in their Aware Home project [1]. The MIT's Oxygen project [27] creates pervasive human-centred computing through creating intelligent spaces in which embedded devices provide large amounts of computation and interfaces to cameras and microphones allowing users to communicate through speech, vision and gesture. The Intelligent Room [6] is a related project that has applied similar concepts found in Oxygen to a room environment making

it responsive to the occupant by adding intelligent sensors to the user interfaces. The Sony Interaction Laboratory in Japan [29] focuses on developing software, hardware and sensor architectures to realize natural and intuitive interactions between the user and the information environment. In [26], Mozer has performed work on an adaptive learning approach for inhabited spaces; however the approach only works over short time intervals and the internal representations and learnt decisions of the system are not accessible in a human readable form.

Most of the aforementioned approaches for AIEs represent a large body of current research work on ambient intelligent applications and smart environments. However they primarily focus on intelligent tracking and user interaction, sensing, planning and time independent context and there was no emphasis on online learning and life long adaptation of the user behaviours that can be generated in a human readable form which is a central feature of the Ambient Intelligence paradigm.

Fuzzy Logic Controllers (FLCs) have been credited with providing appropriate framework for representing the information in a human readable form. Moreover, FLCs provide an adequate methodology for designing robust controllers that are able to deliver a satisfactory performance when contending with the uncertainty, noise and imprecision attributed to real world environments as in AIEs. 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. As a result, FLCs have been used in AIEs as in [8], [15], [16], [28]. Most of the FLCs use the traditional type-1 FLCs that utilise crisp and precise type-1 fuzzy sets and thus the type-1 FLCs can be used to model the user behaviour under specific environment conditions. In our previous work [8], we have presented an adaptive type-1 agent that can learn the rules and Membership Functions (MFs) of a type-1 fuzzy system to model the user behaviour. This agent can be fine tuned online by adding, deleting or modifying rules to adapt the agent to slight changes in the user preferences and environmental conditions [8]. Such type-1 fuzzy agent can handle the slight uncertainties faced within the short term (we will term this as *short term uncertainties*) like slight noise and imprecision associated with the inputs and outputs of the FLC as well as slight behaviour changes of the user. However, the effectiveness of the type-1 based agents will degrade as the environmental conditions and the associated user activities change due to *long term uncertainties* which can arise over longer durations of time such as:

- Uncertainties arising due to change of environmental conditions (such as the external light level, temperature, time of day (morning, evening...etc)) over a considerable long period of time due to seasonal variations. For example the difference in the position of the sun would cause a difference between the late afternoon light levels in mid summer and the late afternoon light levels in mid winter.
- Uncertainties caused by the humans occupying these environments as their behaviours, moods and activities are dynamic, unpredictable and non-deterministic and change with time and season. There is also the fact that different words mean different things in different times of the year and the values associated with a term such as '*warm*' in reference to temperature can vary from winter to summer.
- Uncertainties in the agent's controller outputs due to the change of actuator characteristics as a result of the wear and tear that occurs over time.

These uncertainties translate into uncertainties about the fuzzy set MFs [25]. The effectiveness of type-1 FLCs are limited by their use of precise type-1 fuzzy sets which handle the uncertainties associated with the inputs and outputs by using *precise and crisp* MFs [24] that will only apply under specific environment conditions. Over the longer time durations the effects of long term uncertainties would cause the values of the fuzzy sets MFs associated with the linguistic labels to change. The type-1 MFs would therefore no longer be effective at modelling the user behaviours under the new conditions and would not be able to handle the associated uncertainties.

A type-2 fuzzy set is characterized by a fuzzy membership function, i.e. the membership value (or membership grade) for each element of this set is a fuzzy set in [0,1], unlike a type-1 fuzzy set where the membership grade is a crisp number in [0,1] [24]. The MFs of type-2 fuzzy sets are three dimensional and include a Footprint Of Uncertainty (FOU), it is the new third-dimension of type-2 fuzzy sets and the FOU that provide additional degrees of freedom that can make it possible to directly model and handle the uncertainties [24], [25]. Therefore FLCs that use type-2 fuzzy sets to represent the inputs and outputs of the FLC can handle the short and long term uncertainties to produce a good performance. Moreover, using type-2 fuzzy sets to represent the FLC inputs and outputs will result in the reduction of the FLC rule base when compared to using type-1 fuzzy sets. This is because type-2 fuzzy sets rely on uncertainty represented in the footprint of uncertainty to cover the same range as type-1 fuzzy sets with a much smaller number of labels [24].

In our previous work, we have described an approach for using a type-2 agent in AIEs where the type-2 fuzzy sets had fixed FOUs that were determined empirically [9], [10]. We have shown that the type-2 agent operated in the presence of long term uncertainties and gave a very good performance that outperformed the type-1 agent while using smaller rule base than the type-1 agent. This type-2 agent also allowed the online adaptation of the rules base [9], [10]. However, using fixed type-2 MFs can eventually also become ineffective in handling the dynamic nature of AIEs over extended periods of time. For example, a FLC configured for summer will be very different from a FLC configured for winter as the values for the type-2 fuzzy sets MFs associated with the linguistic labels will change as the season, environment and the associated user preferences change. For example, a term such as '*warm*' in reference to temperature could imply turning a heater to almost its highest setting in winter while in summer it can imply shutting down the heater. Moreover, the seasonal and environmental changes will cause ongoing changes in the user's activities, for example the users will spend more time in day light hours during summer which will change their activities compared to winter when the hours of day light will be reduced. Hence, the type-2 FLC rules and MFs parameters can cover very different sets of values due to the seasonal changes. Therefore, fixed type-2 MFs with fixed FOUs will be unable to effectively handle the continually accumulating uncertainties over the extended time scales.

In this paper, we present a novel system for learning and adapting type-2 FLCs for embedded agents in AIEs. The system learns both the MFs (for all the inputs and outputs of the FLC) and rules for a type-2 FLC that represents and models the particularised behaviours of the user. The agent type-2 FLC controls

the AIE based on a wide range of parameters in a non-intrusive and invisible way. As the environmental conditions and user preferences change over short time durations our controller is able to adapt its rule base online to accommodate the changes or additions to the user preferences and hence the system can be tuned to maximize the user satisfaction. Over the long term, the agent can adapt incrementally by adapting the FLC rules and MFs in a life long learning mode to handle and accommodate for all long term uncertainties that arise due to the changes in the environment and user behaviour over extended time durations. Our technique is a one pass method which is not computationally intensive and has been embedded onto a home consumer appliance, namely an internet fridge. In order to evaluate the effectiveness of our learning technique, we have compared the type-2 FLC generated by our technique with the type-2 FLC generated by Genetic Algorithms (GAs). We will show how the type-2 FLC generated by our learning technique will better model the user behaviour and outperform the type-2 FLC generated by the GAs, while our technique is well suited for embedded devices and life long learning. We will present unique experiments that were conducted in the iDorm and were carried out by different users over the course of the year in which the type-2 agent has learnt and adapted to the user behaviour. We will show how our type-2 agents can deal with the short term and long term uncertainties in AIEs to give a very good performance that outperforms the type-1 fuzzy agents while using smaller rule bases, this rule reduction will result in the type-2 agents to use less memory and to be computationally faster than the type-1 agents.

The rest of this paper is organised as follows: In Section II, we describe the iDorm which is our test bed for AIEs. In Section III, the type-2 fuzzy sets and their associated terminologies are introduced. Our learning and incremental adaptation technique for the type-2 embedded agent is described in Section IV. In Section V, we present our experiments and results. Finally conclusions are presented in Section VI.

II. THE iDORM TEST BED FOR AMBIENT INTELLIGENT ENVIRONMENTS

The intelligent Dormitory (iDorm) shown in Fig 1 is situated in the University of Essex. The iDorm is a real AIE test bed used for conducting research into intelligent embedded agent technologies. The iDorm is a multi-user inhabited space that is fitted with a plethora of embedded sensors, actuators, processors and heterogeneous networks that are cleverly concealed (buried in the walls and underneath furniture) so that the user is completely unaware of the hidden intelligent infrastructure of the room. The iDorm looks and feels like an ordinary study/bedroom environment containing a mix of furnishings such as a bed, work desk and wardrobe. This splits the room into areas of different activity such as sleeping, working and entertaining [16]. The iDorm has 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.

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 [16]. A common interface to the iDorm and its devices is implemented through Universal Plug & Play (UPnP) 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 UPnP software devices that interface the hardware

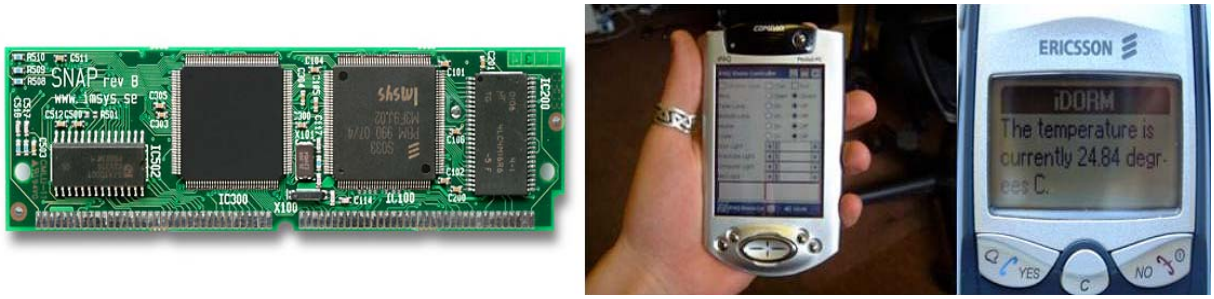
devices on their respective networks. The agent implementing our learning and adaptation mechanism was built on top of the low level UPnP control architecture enabling it to communicate with the UPnP devices in the iDorm and thus allowing it to monitor and control these devices.

Fig.1. The iDorm.

Any networked embedded computer that can run a standard Java process can access and control the devices in the iDorm directly. Our agent system is currently embedded in an internet Fridge (iFridge) located in the iDorm as shown in Fig 2. The iFridge incorporates an intelligent embedded user friendly server with touch screen capability.

Fig. 2. The user interfacing with the iFridge and embedded agent interface.

Our agent could also be embedded into any other part of the environment by operating on embedded platforms such as the Imsys technologies Simple Network Application Platform (SNAP) shown in Fig 3a. Our agent could also be embedded in physically portable computational artefacts that can monitor and control the iDorm wirelessly such as handheld PDAs (as shown in Fig 3b) and mobile phones using a WAP interface which is a simple extension of a web interface (as shown in Fig 3c).



a) b) c)

Fig. 3. a) SNAP embedded paltform. b) PDA interface. c) Mobile phone interface.

III. TYPE-2 FUZZY SETS

Type-2 fuzzy sets are able to handle the numerical and linguistic uncertainties faced by our agent controller to enable it to better model the user's behaviours while handling the changing dynamics of the environment and the user activity, which is an essential requirement for an ambient intelligent system.

Type-2 fuzzy sets are able to model the numerical and linguistic uncertainties because their MFs are themselves fuzzy [25]. One can imagine blurring the type-1 membership function depicted in Fig 4a by shifting the points on the triangle either to the left or to the right and not necessarily by equal amounts as in Fig 4b. Therefore at a specific value of *x*, say *x'*, there is no longer a single value for the membership function (*u'*); instead, the membership function takes on values wherever the vertical line intersects the blurred area shaded in grey. Those values need not all be weighted the same; hence, we can assign an amplitude distribution to all of those points. Doing this for all $x \in X$, we create a three-dimensional MF which is a type-2 MF that characterises a type-2 fuzzy set [24].

Formally a type-2 fuzzy set $\tilde{A}$ is characterised by a type-2 MF $\mu_{\tilde{A}}(x,u)$ [25] where $x \in X$ and $u \in J_x \subseteq [0,1]$, i.e.,

$$\tilde{A} = \left\{ ((x,u), \mu_{\tilde{A}}(x,u)) \mid \forall x \in X, \forall u \in J_x \subseteq [0,1] \right\} \tag{1}$$

in which $0 \le \mu_{\tilde{A}}(x,u) \le 1$. $\tilde{A}$ can also be expressed as follows [25]:

$$\tilde{A} = \int_{x \in X} \int_{u \in J_x} \mu_{\tilde{A}}(x,u) / (x,u) \quad J_x \subseteq [0,1] \tag{2}$$

At each value of *x* say *x = x'*, the 2-D plane whose axes are *u* and $\mu_{\tilde{A}}(x',u)$ is called a vertical slice of $\mu_{\tilde{A}}(x,u)$ [25]. A *secondary membership function* is a vertical slice of $\mu_{\tilde{A}}(x,u)$. It is $\mu_{\tilde{A}}(x = x',u)$ for $x' \in X$ and $\forall u \in J_{x'} \subseteq [0,1]$ [25], i.e.

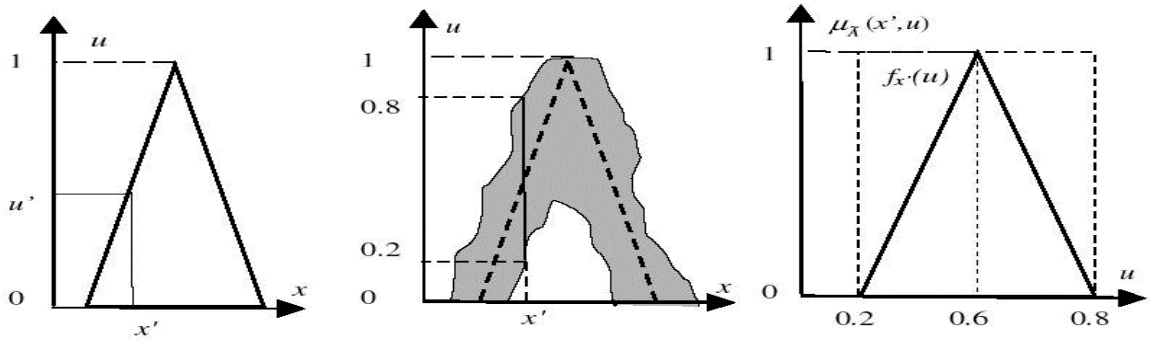
$$\mu_{\tilde{A}}(x = x', u) \equiv \mu_{\tilde{A}}(x') = \int_{u \in J_{x'}} f_{x'}(u) / (u) \quad J_{x'} \subseteq [0,1] \tag{3}$$

in which $0 \le f_{x'}(u) \le 1$. Due to $\forall x' \in X$, the prime notation on $\mu_{\tilde{A}}(x')$ is dropped and $\mu_{\tilde{A}}(x)$ is referred to as a secondary membership function [25]. According to Mendel [24] the name that we use to describe the entire type-2 membership function is associated with the name of the secondary membership functions; so, for example if the secondary membership function is triangular (as shown in Fig 4c) then we refer to $\mu_{\tilde{A}}(x,u)$ as a triangular type-2 membership function.

Based on the concept of secondary sets, type-2 fuzzy sets can be written as the union of all secondary sets as follows [25]:

$$\tilde{A} = \int_{x \in X} \mu_{\tilde{A}}(x) / x = \int_{x \in X} \left[\int_{u \in J_x} f_x(u) / u \right] / x \qquad J_x \subseteq [0,1] \tag{4}$$

The domain of secondary membership functions is called *primary membership* of *x* [25], and in Equation (4), J_x is the primary membership function of *x*, where $J_x \subseteq [0,1]$ for $\forall x \in X$ [25].



(a) (b) (c)

Fig. 4. a) Type-1 membership function. b) Type-2 membership function. c) Triangular secondary membership function plotted in thick line; interval secondary membership function plotted in dashed line.

The uncertainties in the primary membership function of $\tilde{A}$ are encapsulated within the bounded region termed as the Footprint of Uncertainty (FOU) [25], which is shown as the grey region in Fig 4b. The FOU [25] is the union of all primary memberships [25], i.e.,

$$FOU(\tilde{A}) = \bigcup_{x \in X} J_x \tag{5}$$

According to Liang *et al.* [21] a type-2 fuzzy set can be thought of as a large collection of embedded type-1 sets each having a weight associated with it.

Our agent will use the interval type-2 FLC (using interval type-2 fuzzy sets to represent the inputs and outputs). The interval type-2 fuzzy sets are characterised by interval type-2 MFs in which the secondary membership grades are equal to unity [22]. An interval type-2 membership is represented by its left and right end-points; these two end points are associated with two type-1 membership functions referred to as upper and lower membership functions which are also the upper and lower bounds for the FOU of the type-2 set [24]. Fig 4c illustrates the interval secondary membership function (plotted with the dashed line) at *x*'. Formally the upper and lower membership functions of a fuzzy set $\tilde{A}$ are denoted by $\overline{\mu}_{\tilde{A}}(x), \forall x \in X$ and $\underline{\mu}_{\tilde{A}}(x), \forall x \in X$ respectively. According to Mendel [24] we can re-express Equation (4) to represent the interval type-2 fuzzy set $\tilde{A}$ (where $f_x(u) = 1$) in terms of upper and lower membership functions as follows [24]:

$$\tilde{A} = \int_{x \in X} \left[\int_{u \in [\underline{\mu}_{\tilde{A}}(x),\ \overline{\mu}_{\tilde{A}}(x)]} 1/u \right] \Big/ x \tag{6}$$

An interval type-2 FLC is computationally far less intensive than a general type-2 FLC and thus better suited for real-time computation in embedded computational artefacts.

From the above discussion, we see that type-2 FLCs using type-2 fuzzy sets have many advantages when compared to type-1 FLCs using type-1 fuzzy sets, we summarise some of them as follows:

- Type-2 fuzzy sets are able to handle the numerical and linguistic uncertainties and hence they can accommodate the short and long term uncertainties faced by the agents in AIEs. Therefore, FLCs that are based on type-2 fuzzy sets will have the potential to produce a better performance than the type-1 FLCs.

- In a type-2 FLC each input and output will be represented by a large number of type-1 fuzzy sets which are embedded within the FOUs of the type-2 fuzzy sets. The use of such a large number of type-1 fuzzy sets to describe the input and output variables allows for greater accuracy in capturing the user behaviours.
- Using type-2 fuzzy sets to represents the FLC inputs and outputs will result in the reduction of the FLC rule base when compared to using type-1 fuzzy sets. This is because type-2 fuzzy sets rely on uncertainty represented in the FOU to cover the same range as type-1 fuzzy sets with a much smaller number of labels. As the number of inputs to the FLC increases, the potential rule reduction as a consequence of fewer labels becomes significantly greater [24], which can have an effect on reducing the memory usage and increasing processing speeds of type-2 FLCs when compared to type-1 FLCs.

IV. LEARNING AND INCREMENTAL ADAPTATION IN THE TYPE-2 AGENT

The embedded agents learn and adapt to the user behaviours in AIEs using our type-2 Incremental Adaptive Online Fuzzy Inference System (IAOFIS) technique, which builds on our previous type-1 Adaptive Online Fuzzy Inference System [8]. IAOFIS is an unsupervised data-driven one-pass approach for extracting type-2 fuzzy MFs and rules from data to learn an interval type-2 FLC that will model the user's behaviours.

The IAOFIS approach consists of eight phases of operation as described in Fig 5. In **Phase 1,** the system monitors the user interactions in the environment for a specific time (3 days in case of our experiments) to capture input/output data associated with the user actions. In **phase 2** the system learns from the data captured in phase 1 the type-1 MFs and rules needed to form a type-1 FLC that can effectively model the user's behaviours under the specific environmental conditions during phase 1. The approach used for learning this type-1 FLC can be found in [8] where the method for learning the type-1 MFs is based on a double clustering approach combining Fuzzy-C-Means (FCM) and agglomerative hierarchical clustering [8], for the reader's convenience we present this method in Appendix I. In **phase 3,** the learnt type-1 FLC operates in the environment to satisfy the user preferences under the faced environmental conditions. The type-1 FLC can handle the short term uncertainties arising from slight sensor noise or imprecision as well as slight changes in environmental conditions such as small changes in temperature and light level due to variations in the weather conditions. The type-1 agent can also adapt in the short term as shown in **phase 4** by updating the FLC rule base through adding or adapting rules to reflect the user preferences associated with the encountered environment conditions. However over a long period of time, the long term uncertainties caused by seasonal changes and the associated changes in user activity will result in a significant deviation of the type-1 MFs parameters (associated with the linguistic labels for the input and output variables) from those initially modelled by the type-1 FLC. So whatever adaptations occur to the rules, this will not improve the system performance as the MFs values attached to the linguistic labels (which are the antecedents and the consequents of the rules) no longer reflect the current environment and user preference. This will cause the performance of the type-1 FLC to degrade which can be gauged by the increase in user interaction with the system to override the FLC outputs to try to adapt the system to his desires; this reflects the user's dissatisfaction. When the type-1

FLC sufficiently degrades a *system adaptation trigger* (this will be further discussed in subsection IV.F) is activated and the system goes to **phase 5** in which the user is re-monitored again under the new environmental conditions for a specific time interval (again 3 days in case of our experiments). The system then goes to **phase 6** in which the agent learns the type-2 MFs and rules to form an interval type-2 FLC. The system then moves to **phase 7** in which the type-2 FLC controls the user environment based on the user learnt behaviours and preferences. The type-2 FLC is able to operate effectively to handle the short term and long term uncertainties in the environment. The type-2 agent can adapt in the short term as shown in **phase 8** by updating the type-2 FLC rule base through adding or adapting rules to reflect the user preferences associated with the encountered environment conditions. However after an extended period of time, new uncertainties arise due to the continual seasonal changes which occur in the environment, hence the type-2 MFs parameters associated with the linguistic labels change which will cause the performance of the type-2 FLC to degrade. The agent again enters a monitoring mode in **phase 5** to re-monitor the user behaviour under new environmental conditions, the agent will then incrementally adapt the type-2 FLC by generating a new set of type-2 MFs and rules to take into account the current and previous uncertainties which arose. Our system can therefore incrementally adapt the type-2 FLC in a life long learning mode so that its type-2 MFs and FOUs capture all the faced uncertainties in the environment during the online operation of the agent. The adapted type-2 FLC also retains all the previously learnt user behaviours captured in the FLC's rule base. The following subsections will describe each of the phases of our IAOFIS approach in more detail.

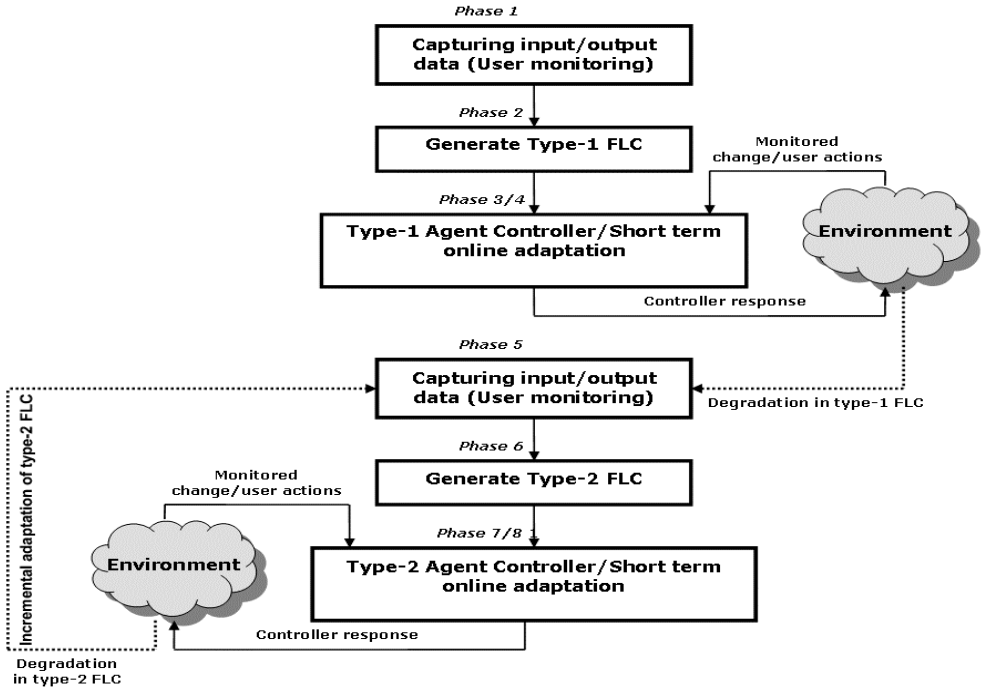


Fig. 5. Flow diagram showing the main phases of IAOFIS.

A. Phase 1 & 5: Capturing Input Output Data (User Monitoring)

In phase 1 and 5, the agent 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 current

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. IAOFIS then learns a *descriptive model* of the user's behaviours from the data accumulated by the agent. Given a set of multi-input multi-output data pairs:

$$(x^{(t)}; y^{(t)}), \quad t = 1,2,..., N \tag{7}$$

where N is the number of data instances, $x^{(t)} \in R^n$ and $y^{(t)} \in R^k$. IAOFIS extracts MFs and 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 dataset. In our experiments in the iDorm we used 8 sensors for our inputs and 10 actuators for our outputs.

B. Phases 2-4: Type-1 FLC Agent

The agent starts initially by generating a type-1 FLC that models the user behaviour under the specific environment conditions encountered during the monitoring phase (**phase 1**). The type-1 FLC can handle the faced short term uncertainties and adapt in the short term by updating the rule base to accommodate the user's preferences over the short time interval. More information about the type-1 agent and **phases 1-4** can be found in [8]. We have provided the technique for generating type-1 MFs in Appendix I. This approach generates a predefined number V of type-1 MFs that are distributed over the total range of data values for each continuous input/output variable, the input variables can have a different V number from that used by the output variables (V_i for the input variables and V_o for the output variables). For the input and output variables an overlap threshold is also set to determine the overlap point of each fuzzy set [8]. The type-1 agent uses Guassian MF, where the agent learns the mean m_z^j and the standard deviation σ_z^j for each type-1 fuzzy set A_z^j associated with the z linguistic label for all the input and output variables. **Phase 5** will be a monitoring and data capturing phase that be will be triggered based on when the type-1 FLC sufficiently degrades as signalled by *system adaptation trigger* which is described in the next subsection.

C. The System Adaptation Trigger

In realising the non-intrusive aspect of ambient intelligence [11] whenever the user is not happy with the agent's actions, he can always intervene and override the agent's control responses by simply altering the manual control of the system. Over a significantly longer duration of time the uncertainties which arise can eventually degrade the performance of the controller (type-1 or type-2) as its rules and MF parameters are no longer able to model the new conditions. Moreover, the rule base adaptation will not have any effect as the MFs values associated with the linguistic labels would have changed due to the environmental changes and the associated change in the user behaviour. This will lead to the increase of

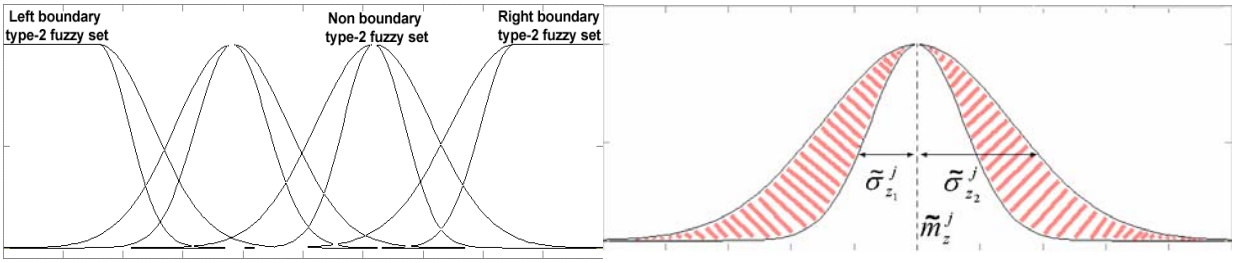
user intervention (due to his dissatisfaction with the agent's actions) and will also cause the agent to generate an increasing number of new rules to try and adapt to the new conditions. The performance degradation is determined by **two degradation criteria. The first** is an increase in the number of rule adaptations as a consequence of user intervention that exceeds a *system intervention threshold* within a specified time period. **The second criterion** is an increase in the number of new rules that exceeds a *rule creation threshold* within a specified time period. Both these time periods would be based on the average amount of time it had previously taken the controller to stabilise and totally satisfy the user (1 day in our experiments). If either of these criteria is satisfied then the *system adaptation trigger* is fired. In the next subsection, we will describe **phase 6** of our IAOFIS approach in which the agent learns the type-2 MFs and rules to form an interval type-2 FLC.

D. Phase 6: Generating Type-2 FLC

When the *system adaptation trigger* is fired, the IAOFIS will incrementally learn/adapt the agent type-2 FLC by generating the type-2 MFs and rules to handle and accommodate the long term uncertainties. In the next subsections, we will describe how to generate the type-2 MFs and rules. It is worth mentioning that we aim to generate good enough type-2 MFs that will enable the type-2 FLC to give a good enough response in face of the uncertainties encountered so far, and we do not aim to produce optimum MFs. The proposed type-2 MF generation method is a simple heuristic one pass method that is able to learn and adapt the type-2 MFs in a life long learning mode so that their FOUs are proportional to the encountered uncertainties. The method needs a very short amount of time and little memory to find the MFs which makes it well suited for the embedded platforms used in AIEs with limited processor and memory capabilities. Moreover, we will show in the experiments section that our method will outperform a global optimisation method like GAs to give better results while our method is computationally much faster and more suitable for online learning and adaptation.

1) Extracting Type-2 Membership Functions: It is necessary to be able to categorise the recorded user input/output data (captured in **phase 5**) into a set of fuzzy MFs which quantify the raw crisp values of the sensors and actuators into linguistic labels such as *Normal*, *Cold* or *Hot.* IAOFIS is based on learning the *particularised* behaviours of the user and therefore requires these MFs to be defined from the user's input/output data recorded by the agent. For each j continuous input/output variable, our IAOFIS generates the type-2 MFs for all the type-2 fuzzy sets $\widetilde{A}_z^j$, z represents the linguistic labels where $z = 1,2,...,V$ and V is the number of labels (V_i for the input variables and V_o for the output variables). The generated type-2 MFs will try to capture the uncertainties encountered in the environment during the operation of the agent. For our type-2 FLCs, it is desirable to have the non boundary type-2 fuzzy sets represented by MFs which have only one point in the middle with a maximum membership of 1.0, while it is desirable for the type-2 fuzzy sets at the boundaries of the variable range to be represented by MFs which have a certain membership of 1.0 for both the lower and upper MF beyond a fixed point as shown

in Fig 6a. Based on these requirements, we have chosen to describe the interval type-2 fuzzy set $\widetilde{A}_z^j$ by MFs based on the Guassian primary MF with uncertain standard deviations (Shown in Fig 6b).



(a) (b)

Fig. 6. a) Boundary and non-boundary type-2 fuzzy sets. b) Gaussian primary MF with uncertain standard deviation.

The chosen MF has a fixed mean $\widetilde{m}_z^j$ and uncertain standard deviation $\widetilde{\sigma}_z^j$ that takes on the values in $[\widetilde{\sigma}_{z_1}^j, \widetilde{\sigma}_{z_2}^j]$. We will denote $\widetilde{\sigma}_{z_1}^j$ as the lower standard deviation and $\widetilde{\sigma}_{z_2}^j$ as the upper standard deviation as shown in Fig 6b. The primary MF for each $\widetilde{A}_z^j$ can be written as follows [24]:

$$\exp\left\{-\frac{1}{2}\left(\frac{x-\widetilde{m}_z^j}{\widetilde{\sigma}_z^j}\right)^2\right\} \qquad \widetilde{\sigma}_z^j \in [\widetilde{\sigma}_{z_1}^j, \widetilde{\sigma}_{z_2}^j] \tag{8}$$

We will use the ~ to differentiate the $\widetilde{m}_z^j$, $\widetilde{\sigma}_z^j$, $\widetilde{\sigma}_{z_1}^j$ and $\widetilde{\sigma}_{z_2}^j$ from their type-1 counterparts. The lower MF $\underline{\mu}_{\widetilde{A}_z^j}(x)$ is associated with $\widetilde{\sigma}_{z_1}^j$ and the upper MF $\overline{\mu}_{\widetilde{A}_z^j}(x)$ is associated with $\widetilde{\sigma}_{z_2}^j$.

At a specific time instance, if we monitored the user and extracted type-1 MFs from the captured data, the extracted type-1 MFs will best characterise the environmental conditions and the user behaviour at this specific instance and will thus represent at this instance the user's particular type-1 MF parameters for the fuzzy sets associated with the given linguistic labels. We start initially in phase 2 by generating type-1 MFs at the initial monitoring instance (**phase 1**) and then whenever the system degrades and goes to **phase 5**, we generate the type-1 MFs at this learning/adaptation instance *E*. We will then end up with a set *E* type-1 MFs, where each type-1 MF will best represent the environmental conditions and user preferences captured at the monitoring phase of its specific instance. Due to the long term uncertainties, there will be a deviation of the mean m_z^j and the standard deviation σ_z^j for the type-1 MF representing the type-1 fuzzy set A_z^j associated with a given linguistic label *z*. However, in our application domain the deviation of standard deviation can be neglected when compared to the deviation of the means of the type-1 MFs. So the deviation of the means of the *E* type-1 MFs will reflect the deviation and uncertainty that has affected the linguistic label of the type-1 fuzzy set A_z^j over the *E* instances. Thus, the deviation of means of the *E* type-1 MFs for a given linguistic label will be directly proportional to the accumulated numerical and linguistic uncertainties associated with this label. Therefore, if for a given linguistic label *z* we found the smallest and largest mean values of the *E* type-1 MFs, then we can use the difference

between the smallest and largest means to reflect the accumulated numerical and linguistic uncertainties associated with this label. We will denote this mean difference as $C\nabla$, where the bigger $C\nabla$ is, the bigger will be the uncertainty associated with this linguistic label and thus the bigger the size of the FOU and vice versa. Hence, the generated type-2 fuzzy set associated with the linguistic label *z* for any input/output should have its FOU directly proportional to $C\nabla$. The generated type-2 fuzzy sets should cover the domain covered by all the generated *E* type-1 fuzzy sets (we will denote this domain by $R\nabla$). For the generated type-2 MF, the fixed mean $\tilde{m}_z^j$ will lie in the middle of the domain $R\nabla$, (this will be slightly different for boundary fuzzy sets as described later). As we aim to cover $R\nabla$, then the upper standard deviation $\tilde{\sigma}_{z_2}^j$ will be designed so that the generated type-2 MF covers $R\nabla$. In the Guassian type-2 MF we used, the FOU is proportional to the difference between $\tilde{\sigma}_{z_1}^j$ and $\tilde{\sigma}_{z_2}^j$, so the bigger $C\nabla$ is, the bigger will be the difference between $\tilde{\sigma}_{z_1}^j$ and $\tilde{\sigma}_{z_2}^j$ and vice versa, hence the lower standard deviation $\tilde{\sigma}_{z_1}^j$ will be designed according to this. In the rest of this subsection, we will describe the specific equations used to generate the type-2 MF for each $\tilde{A}_z^j$.

For each *j* continuous input/output variable and for each *z* linguistic label, the new interval type-2 MF that describes the type-2 fuzzy set $\tilde{A}_z^j$ is computed as follows: take the *E* type-1 MFs corresponding to the type-1 fuzzy set A_z^j associated with the linguistic label *z*, find the minimum and maximum mean parameters $m_z^{j(\min)}$ and $m_z^{j(\max)}$ of all the *E* type-1 MFs. $C\nabla$ can therefore be written as follows:

$$C\nabla = m_z^{j(\max)} - m_z^{j(\min)} \tag{9}$$

The total domain covered by the *E* type-1 membership functions which we denote as $R\nabla$ can be written as follows:

$$R\nabla = \left(R_z^{j(\max)} - L_z^{j(\min)} \right) \tag{10}$$

Where $L_z^{j(\min)}$ represents the left most bound of all the *E* type-1 MFs and $R_z^{j(\max)}$ represents the right most bound of all the *E* type-1 MFs. As we are using Guassian type-1 MFs that extend to infinity, $L_z^{j(\min)}$ can be found by finding $m\text{-}3\sigma$ for the associated type-1 MF while $R_z^{j(\max)}$ can be found by finding $m+3\sigma$ for the associated type-1 MF. $L_z^{j(\min)}$, $R_z^{j(\max)}$, $R\nabla$ and $C\nabla$ are shown in Fig 7.

The mean point $\tilde{m}_z^j$ for the type-2 MF will lie in the middle of $R\nabla$ domain (i.e. in the middle between $L_z^{j(\min)}$ and $R_z^{j(\max)}$) for all the non boundary fuzzy sets. For the boundary fuzzy sets $\tilde{m}_z^j$ will be the average of $m_z^{j(\max)}$ and $m_z^{j(\min)}$. So $\tilde{m}_z^j$ can be written as follows:

$$\tilde{m}_z^j = \begin{cases} \left(L_z^{j(\min)} + R_z^{j(\max)} \right) 2, & NonBoundaryMFs \\ \left(m_z^{j(\min)} + m_z^{j(\max)} \right) 2 & BoundaryMFs \end{cases} \tag{11}$$

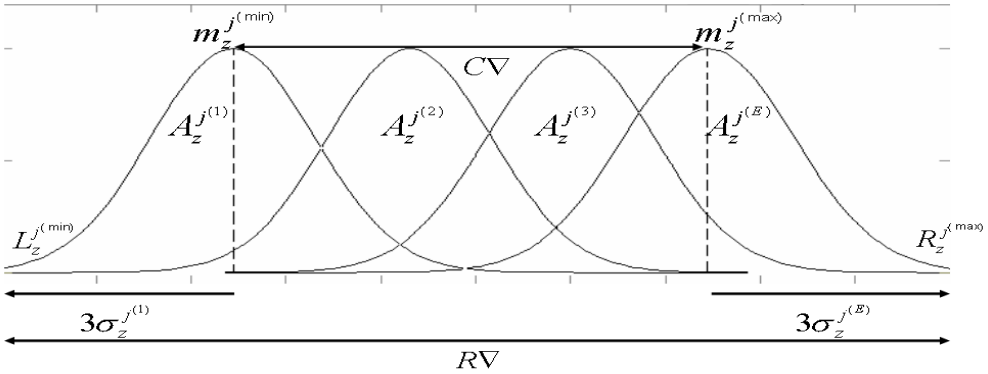


Fig. 7. Diagram showing $C\nabla$, $R\nabla$, $L_z^{j(\min)}$, $R_z^{j(\max)}$, $m_z^{j(\min)}$ and $m_z^{j(\max)}$ of *E* type-1 MFs.

The upper standard deviation $\tilde{\sigma}_{z_2}^j$ is chosen to make the type-2 MF cover the domain covered by the *E* type-1 MFs. For non boundary type-2 MFs $(\tilde{m}_z^j - L_z^{j(mim)}) = (R_z^{j(\max)} - \tilde{m}_z^j)$, however for the left and right boundary type-2 MFs we use $(R_z^{j(\max)} - \tilde{m}_z^j)$ and $(\tilde{m}_z^j - L_z^{j(mim)})$ respectively in deriving $\tilde{\sigma}_{z_2}^j$. So $\tilde{\sigma}_{z_2}^j$ can be written as follows:

$$\tilde{\sigma}_{z_2}^j = \begin{cases} (R_z^{j(\max)} - \tilde{m}_z^j)/3 \\ (\tilde{m}_z^j - L_z^{j(\min)})/3 \end{cases} \begin{matrix} Non Boundary MFs \& Left Boundary MFs \\ Right Boundary MFs \end{matrix} \quad (12)$$

We have divided by 3 in Equation (12) as we have used 3σ for determining the end points of the Gaussian MFs.

The FOU of $\tilde{A}_z^j$ lies between $\underline{\mu}_{\tilde{A}_z^j}(x)$ and $\overline{\mu}_{\tilde{A}_z^j}(x)$ which are associated with $\tilde{\sigma}_{z_1}^j$ and $\tilde{\sigma}_{z_2}^j$ respectively. As we discussed before $C\nabla$ should be proportional to the FOU and hence $C\nabla$ should also be proportional to the difference between $\tilde{\sigma}_{z_1}^j$ and $\tilde{\sigma}_{z_2}^j$. For small uncertainties which will be reflected by small values of $C\nabla$, we would aim to have the FOU small, which can be achieved by having the difference between $\tilde{\sigma}_{z_2}^j$ and $\tilde{\sigma}_{z_1}^j$ small, so for small $C\nabla$ values $\tilde{\sigma}_{z_1}^j$ can be written as follows:

$$\tilde{\sigma}_{z_1}^j = \tilde{\sigma}_{z_2}^j - C\nabla \qquad \text{For small } C\nabla \text{ values} \quad (13)$$

The criteria for judging how small is the value of $C\nabla$ depends on the application, in our domain, for all the continuous variables $R\nabla$ was always bigger than 30 and we judged any values of $C\nabla$ to be small if they are less than or equal to 1/9 $R\nabla$, this value was found empirically. Note that when the uncertainty reduces to zero which is reflected by $C\nabla$ = 0 (i.e. if all the means and standard deviations will be the same, the very rare case in our application where there is a difference in the standard deviations when $C\nabla$ = 0 is explained below), then $\tilde{m}_z^j = m_z^j$ and $\tilde{\sigma}_{z_2}^j = \tilde{\sigma}_{z_1}^j = \sigma_{z_1}^j$, thus the type-2 fuzzy set will reduce to the original certain type-1 fuzzy set. When we have large uncertainty represented by large $C\nabla$ values, then we will have $\tilde{\sigma}_{z_1}^j$ written as follows:

$$\tilde{\sigma}_{z_1}^{j} = \tilde{\sigma}_{z_2}^{j} / C\nabla \qquad \text{For large } C\nabla \text{ values} \tag{14}$$

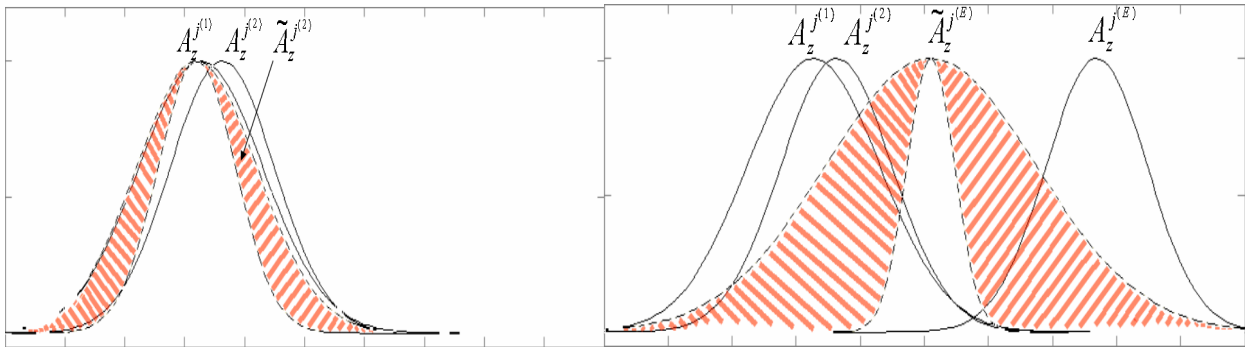
Again we judge values of $C\nabla$ to be large if they are larger than 1/9 $R\nabla$. Note that as $C\nabla$ goes to infinity $\tilde{\sigma}_{z_1}^{j}$ goes to zero so that the FOU will be maximised to represent maximum uncertainty.

The *system adaptation trigger* should be fired based on high uncertainty levels. However this high uncertainty can be associated with a given variable and the levels of uncertainly can then vary between the other variables which is why we have to address small and large $C\nabla$ values.

Although this has seldom been the case with our application, but in the case for a given linguistic label where $C\nabla$ is zero but there is a significant deviation in the E standard deviations then the standard deviations for the E MFs can be ordered and $\tilde{\sigma}_{z_2}^{j} = \sigma_z^{j(\max)}$ and $\tilde{\sigma}_{z_1}^{j} = \sigma_z^{j(\min)}$.

Using our method, we are able to continually adjust the FOU size of the type-2 MF according to the total deviation in the E type-1 MFs which describe a linguistic label over different time intervals. Fig 8a shows the first generated type-2 MF (*shaded*) for a given linguistic label when the *system adaptation trigger* is fired for the first time. Over the period of time the two type-1 MFs generated at different instances show a small deviation, and therefore the associated uncertainty represented in the type-2 fuzzy set FOU is small. In Fig 8b the *system adaptation trigger* is fired again however there is now a much larger deviation shown between the initial two type-1 MFs and a third type-1 MF generated at a new instance. The type-2 MF (*shaded*) that is created accommodates for this increased uncertainty by adjusting its parameters to have a larger FOU.

The newly generated set of type-2 fuzzy sets are combined together with the accumulated data recorded from each of the E adaptation instances to generate the rules for the type-2 FLC. The following subsection will now explain this rule generation phase of our type-2 IAOFIS approach.



(a) (b)

Fig. 8. a) Type-2 fuzzy set generated from two type-1 fuzzy sets created at different time instances where the deviation is small. b) Type-2 fuzzy set generated from three type-1 fuzzy sets created at different time instances where the deviation is large.

2) Fuzzy Rule Extraction: The set of interval type-2 MFs generated from the previous step are combined with the accumulated user input/output data to extract the rules defining the user's behaviours. The fuzzy rule extraction approach used by the type-2 IAOFIS is based on a type-2 extension to the

enhanced version of the Mendel Wang (MW) method developed by L. X. Wang [32], [24]. The rule extraction method is a one pass technique for extracting fuzzy rules from the recorded data. The fuzzy sets for the antecedents and consequents of the rules divides the input and output space into fuzzy regions.

The type-2 IAOFIS extracts multi-input multi-output rules which describe the relationship between $y=(y_1,..,y_k)$ and $x=(x_1,\ldots,x_n)^T$, and take the following form:

$$IF\ x_1 \text{ is } \widetilde{A}_1^l \ \ldots \ and\ x_n \text{ is } \widetilde{A}_n^l, THEN\ y_1 \text{ is } \widetilde{B}_1^l \ \ldots \ and\ \ y_k \text{ is } \widetilde{B}_k^l \tag{15}$$

$l = 1,2,\ldots, M$, where M is the number of rules and l is the index of the rules. There are V_i interval type-2 fuzzy sets $\widetilde{A}_s^q, q = 1,\ldots,V_i$, defined for each input x_s where $(s = 1,\ldots,n)$. There are V_o interval type-2 fuzzy sets $\widetilde{B}_c^h, h = 1,\ldots,V_o$, defined for each output y_c where $(c = 1,\ldots,k)$, the V_i input type-2 fuzzy sets and the V_o output fuzzy sets were generated in the previous subsection. IAOFIS extracts rules in the form of Equation (15) 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. We will now show the different steps involved in rule extraction:

Step 1: For a fixed input-output pair $(x^{(t)}; y^{(t)})$ in the dataset, ($t=1,\ldots N$), compute the upper and lower membership values $\overline{\mu}_{\widetilde{A}_s^q}(x_s^{(t)})$ and $\underline{\mu}_{\widetilde{A}_s^q}(x_s^{(t)})$ for each fuzzy set $q=1,\ldots V_i$, and for each input variable s $(s = 1,\ldots,n)$. Find $q^* \in \{1,\ldots V_i\}$ such that

$$\mu_{\widetilde{A}_s^{q^*}}{}^{cg}(x_s^{(t)}) \geq \mu_{\widetilde{A}_s^q}{}^{cg}(x_s^{(t)}) \tag{16}$$

for all $q=1,..V_i$, where $\mu_{\widetilde{A}_s^q}{}^{cg}(x_s^{(t)})$ is the centre of gravity of the interval membership of $\widetilde{A}_s^q$ at $x_s^{(t)}$ as follows [24]:

$$\mu_{\widetilde{A}_s^q}{}^{cg}(x_s^{(t)}) = f^{cg}_{x_s^{(t)}}(\widetilde{A}_s^q) = \frac{1}{2}\left[\overline{\mu}_{\widetilde{A}_s^q}(x_s^{(t)}) + \underline{\mu}_{\widetilde{A}_s^q}(x_s^{(t)})\right] \tag{17}$$

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

$$IF\ x_1 \text{ is } \widetilde{A}_1^{q^*(t)} \ldots and\ x_n \text{ is } \widetilde{A}_n^{q^*(t)}\ THEN\ y\ is\ centred\ at\ y^{(t)} \tag{18}$$

For each input variable x_s there are V_i type-2 fuzzy sets $\widetilde{A}_s^q$, to characterise it; so that the maximum number of possible rules that could be generated is V_i^n. However given the dataset only those rules among the V_i^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 (16), (17) and (18).

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

$$wi^{(t)} = \prod_{s=1}^{n} \mu_{\tilde{A}_s^q}{}^{cg}(x_s(t)) \tag{19}$$

The weight of a rule $wi^{(t)}$ 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 (18). Due to the fact that the number of data points is quite large consisting of many similar instances, many rules are generated in step 1, that all share the same IF part and are conflicting, i.e. rules with the same antecedent fuzzy sets 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 are *M* such groups. Let group *l* have N_l rules in the following form:

$$IF\ x_1 \text{is}\ \tilde{A}_1^l \ldots and\ x_n\ \text{is}\ \tilde{A}_n^l THEN\ y\ is\ centred\ at\ y^{(t_u^l)} \tag{20}$$

Where $u = 1,\ldots N_l$ 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

$$av^{(l)} = \frac{\sum_{u=1}^{N_l} y^{(t_u^l)} wi^{(t_u^l)}}{\sum_{u=1}^{N_l} wi^{(t_u^l)}} \tag{21}$$

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

$$IF\ x_1 \text{is}\ \tilde{A}_1^l\ and \ldots and\ x_n\ \text{is}\ \tilde{A}_n^l, THEN\ y\ is\ \tilde{B}^l \tag{22}$$

Where the output fuzzy set $\tilde{B}^l$ is chosen based on the following: among the V_o output interval type-2 fuzzy sets $\tilde{B}^1,\ldots,\tilde{B}^{V_o}$ find the B^{h*} such that

$$\mu_{\tilde{B}^{h*}}{}^{cg}(av^{(l)}) \geq \mu_{\tilde{B}^h}{}^{cg}(av^{(l)}) \tag{23}$$

for $h = 1,2,\ldots,V_o$, $\tilde{B}^l$ is chosen as $\tilde{B}^{h*}$, where $\mu_{\tilde{B}^h}{}^{cg}(av^{(l)})$ is the centre of gravity of the interval membership of $\tilde{B}^h$ at $av^{(l)}$ as in Equation (17).

After generating the type-2 MFs in the previous subsection and by generating the *M* rules from the user data, we can have an approximated model of the user's behaviour in the environment over a specific period of time. As we describe later additional new rules can be added to the rule base by the agent while operating online during the short term adaptation phase. To avoid the rule base explosion and the associated effects on the memory and processor capability, we will place a limit on the rule base size as will be discussed in the next subsection.

As mentioned above IAOFIS 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 (21), (22) and (23) are repeated for each output value.

E. Phase 7: The Agent Controller

Once the agent has extracted the type-2 MFs and the set of rules from the user input/output data, it has then learnt the type-2 FLC that captures and *models* the user behaviour. The agent FLC can start controlling the environment on behalf of the user to satisfy his preferences. The agent starts to monitor the state of the environment and affect actuators based on its learnt type-2 FLC that approximate the particularised preferences of the user. This operation is performed in a non-intrusive way to realise the ambient intelligence vision [11]. Fig 9 shows a block diagram of the interval type-2 FLC which consists of a fuzzifier, rule base, fuzzy inference engine, centre of sets type-reducer and defuzzifier, more information about this real time type-2 FLC can be found in [14].

The type-2 FLC works in the following way, the crisp inputs comprising the sensory state of the environment are first fuzzified into the input interval type-2 fuzzy sets (we will use singleton fuzzification) which then activates the inference engine and the rule base to produce output type-2 fuzzy sets. The type-2 fuzzy outputs of the inference engine are then processed by the type-reducer which combines the output sets and then performs a centroid calculation which leads to type-1 fuzzy sets called the type-reduced sets [24]. The defuzzifier can then defuzzify the type-reduced type-1 fuzzy outputs to produce crisp outputs to be fed to the actuators.

In the inference engine the firing interval for each rule based on the input and antecedent operations is calculated as follows [24]:

$$F(x') = [\underline{f}(x'),\ \overline{f}(x')] \equiv [\underline{f}^l, \overline{f}^l] \tag{24}$$

where

$$\underline{f}^l(\mathbf{x}') = \underline{\mu}_{\tilde{F}_1^l}(x_1') * \ldots\ldots\ldots * \underline{\mu}_{\tilde{F}_n^l}(x_n') \tag{25}$$

and

$$\overline{f}^l(\mathbf{x}') = \overline{\mu}_{\tilde{F}_1^l}(x_1') * \ldots\ldots\ldots * \overline{\mu}_{\tilde{F}_n^l}(x_n') \tag{26}$$

where $l = 1,\ldots, M$ refers to the l^{th} rule in the rule base and n is the number of inputs.

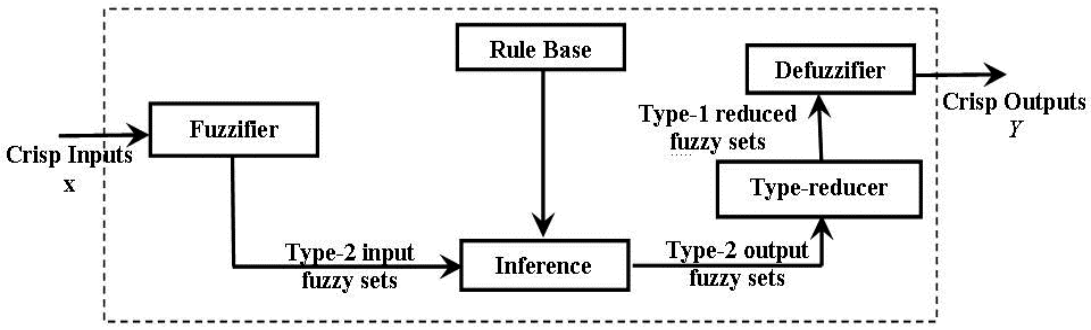


Fig. 9. Block diagram of type-2 FLC

Type-reduction was proposed by Karnik and Mendel [19], [24], it is called type-reduction because this operation takes us from the type-2 output sets of the inference engine to a type-1 set that is termed the type-reduced set [22]. These type-reduced sets are then defuzzified to obtain crisp outputs to be sent to the actuators.

As we are dealing with interval sets, the type-reduced set for the c^{th} output will also be an interval set [24]. As in [22] we use the centre of sets type reduction, as it has reasonable computational complexity that lies between the computationally expensive centroid type-reduction and the simple height and modified height type-reduction which have a problem when only one rule fires [24]. As we are using an interval type-2 FLC the type reduction calculation becomes significantly simplified by using an effective iterative procedure developed by Karnik and Mendel [22], [24]. The type reduced set using the centre of sets type-reduction can be expressed as follows:

$$Y_{\cos}(x)_c = [y_{lc}, y_{rc}] = \int_{y_c^1 \in [y_{lc}^1, y_{rc}^1]} \cdots \int_{y^{M'} \in [y_{lc}^{M'}, y_{rc}^{M'}]} \int_{f^1 \in [\underline{f}^1, \overline{f}^1]} \cdots \int_{f^{M'} \in [\underline{f}^{M'}, \overline{f}^{M'}]} 1 \Big/ \frac{\sum_{l=1}^{M'} f^l y_c^l}{\sum_{l=1}^{M'} f^l} \qquad (27)$$

Where *M'* is the number of the fired rules and $Y_{\cos}(x)_c$ for the c^{th} output is an interval set determined by its left most point y_{lc} and its right most point y_{rc}. y_c^l corresponds to the centroid of the type-2 interval consequent set $\widetilde{B}_c^l$ of the l^{th} rule for the c^{th} output; y_c^l is a type-1 interval set determined by its left most point y_{lc}^l and its right most point y_{rc}^l [22]. f^l denotes the firing strength (degree of firing) of the l^{th} rule which is an interval type-1 set determined by its left most point $\underline{f}^l$ and right most point $\overline{f}^l$ where $\underline{f}^l$ is calculated using Equation (25) and $\overline{f}^l$ is calculated using Equation (26).

The calculation of the type-reduced sets is divided into two stages. In the first stage the centroids of type-2 interval consequent sets of the l^{th} rule are calculated using the iterative procedure developed by Karnik and Mendel [19], [24]. This is conducted ahead of time and before starting the control cycle of the agent's FLC. The second stage consists of calculating the type-reduced sets using the iterative procedure reported in [22], [24]. The type-reduced sets are then defuzzified to produce the crisp output for the actuators; this occurs at each control cycle. The iterative procedure for type-reduction is proven [22], [24], [35] to converge in no more than *M'* iterations to find y_{rc} and *M'* iterations to find y_{lc}. As mentioned before *M'* is the number of fired rules at each control cycle which is a usually a small number compared to the whole rule base of *M* rules. As we will describe in the following section the rule base can be extended online in the short term allowing the addition of new rules based on the environmental conditions and user behaviour. As the number of rules increase the required computation and memory requirements will rise and this could hinder the real time operation of the agent on embedded computers. In our system we have a limit on the rules that can be stored within the embedded agent; this limit is related to the memory and processing capabilities of the embedded agent. When the number of rules

exceeds this limit, we keep the most important rules for the embedded agent operation. The *degree of importance* of each rule is related to how frequently the rule is being used. This is determined by a *frequency counter* associated with each rule which is incremented every time the rule fires during a control cycle. If two rules share the same *degree of importance*, tie-breaking is resolved by a least-recently-used strategy; derived from a "life invocation" flag that is updated each time a rule is activated. It is worth mentioning that the rules that leave the rule base of the embedded agent as a result of rule capping mechanism can be stored in an external drive for any future use.

From the type-reduction stage we produce for each output a type-reduced set $Y_{\cos}(x)_c$. Each type-reduced set is an interval type-1 set determined by its left most point y_{lc} and right most point y_{rc}. We defuzzify the interval set by using the average of y_{lc} and y_{rc} hence the defuzzified crisp output for each output *c* is [35].

$$Y_c(x) = \frac{y_{lc} + y_{rc}}{2} \tag{28}$$

F. Phase 8: Short Term Online Adaptation

In the previous steps we have shown how our agent can learn a type-2 FLC that models and 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 [11] whenever the user is not happy with the agent's actions, he can always intervene and 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.

Whenever the user intervenes to overrides the agent's control responses and actuates one or more of the controlled output devices, a snapshot of the state of the environment is recorded and passed to the rule adaptation routine. Each input variable in the input vector **x** is compared to each of the antecedent sets of a given rule in the rule base to determine its upper and lower membership values. The weight of the rule is then calculated to determine if the degree of firing of the rule in Equation (19) $wi^{(l)} > 0$, meaning that the rule fired, and would therefore have contributed to the overall control response generated by the agent's FLC. The consequent fuzzy set(s) that give the highest membership to the user altered actuator value(s) are selected to replace the consequent set(s) for the corresponding output(s) of all fired rules in the rule base. The consequent fuzzy sets are found by calculating the centre of gravity of the output interval memberships as follows:

$$\mu_{\tilde{B}_c^{h*}}{}^{cg}(y_c) \geq \mu_{\tilde{B}_c^{h}}{}^{cg}(y_c) \tag{29}$$

for $h = 1,2,\ldots,V_o$. the consequent fuzzy set $\widetilde{B}_c$ is chosen as $\widetilde{B}_c^{h*}$. 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 variable x_s the fuzzy sets that give a membership value where $\mu_{\widetilde{A}_s^q}{}^{cg}(x_s^{(t')}) > 0$ are identified. This leads to a grid of identified fuzzy set(s) for each input variable. 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 (29).

This phase allows the rules to be gradually adapted and added to the rule base which enables the agent to adapt as the state of the environment and the preferences of the user drift over the short term. The online adaptation phase can be used to fine tune the learnt type-2 FLC to the current states of the environment and user preferences. As we will show in our experiments a relatively short online tuning period (1 day in our experiments) is required before the type-2 FLC is able to stabilise and continue to control the environment with little or no user intervention, thus handling the short and long term uncertainties arising from changing environmental conditions and user behaviour. However over a significantly longer duration of time the accumulated long term uncertainties which arise can eventually degrade the performance of the controller as its rules and MFs parameters are no longer properly able model the new conditions and this can eventually lead to the firing of the *system adaptation trigger*.

V. EXPERIMENTAL RESULTS

In order to evaluate our proposed type-2 agent, we have performed unique experiments with two different users who have different preferences and activities, both users have stayed in the iDorm (as shown in Fig 10) at different periods over the course of the year. For each user, the agent will learn a type-2 FLC that models the user particular behaviours and preferences. Over the extended period of our experiments (spanning over the course of the year), we will be able to evaluate and demonstrate how the agent can adapt in a life long learning mode and handle the faced short and long term uncertainties.

During the experiments, the users were monitored in the iDorm over different times of the year where a given monitoring phase (phase 1 or 5) lasted for three consecutive days. During each monitoring phase, the user performed the normal variety of behaviours and activities one would associate with a study bedroom environment while the agent recorded the user interactions with the environment in a non-intrusive way. The agent type-2 FLC has **Eight Inputs** connected to the following 8 sensors within the iDorm: internal light level, external light level, internal temperature, external temperature, chair pressure, bed pressure, occupancy and time. The type-2 FLC outputs controlled **Ten actuators** in the iDorm which are: four variable intensity spot lights, the desk and bed side lamps, the window blinds, the heater and 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. As mentioned before, the sensors and actuators are concealed so that the user is completely unaware of the hidden intelligent and pervasive infrastructure of the room. In our experiments, we have used five fuzzy sets (type-1 and type-2) to represent the input/output variables, as we have found that this number can sufficiently describe the user behaviour by our agent [9]. As we mentioned previously the users were able to interface with the devices in the room via the iFridge on which our intelligent agent was embedded (as shown in Fig 2).

Fig. 10. Users in the iDorm

As mentioned before, the agent will start initially by generating a type-1 FLC from the data captured during phase 1 at instance 1. The type-1 FLC will then control the iDorm on behalf of the user and it will handle the short term uncertainties and adapt its rule base to accommodate for the changes of the environment and user preference over the short term. The type-1 FLC will eventually fail when the long term uncertainties will cause the *system adaptation trigger* to fire. This will lead the agent to enter a second monitoring (*phase 5*) at instance 2 to generate the type-2 FLC. For *user 2*, instance 2 occurred during mid summer (late June 2004). While for *user 1*, instance 2 happened during late spring (May 2004) and the generated type-2 FLC was further adapted due to the accumulating long term uncertainties when the *system adaptation trigger* fired again forcing the agent into a further monitoring phase at instance 3 in the middle of winter (late December 2004).

In the following subsections, we will show how the type-2 FLC generated by our agent will handle the uncertainties to better model the user behaviour and preferences when compared to other techniques. In the first part of these comparisons, each technique will produce a FLC that will model the user's behaviour from their monitored activity; each generated FLC will be then tested on unforeseen test data. We will then obtain the scaled Root Mean Squared Error (RMSE) and Standard deviation (STDEV) that will reflect the extent to which the generated user's model represented by the FLC produced the same actions as the human user over the test data. The smaller the RMSE and STDEV, the better the generated FLC will model the user behaviour, this will help us to evaluate how the generated FLC models the user behaviour in face of uncertainties.

Our type-2 agent will then be evaluated online while controlling the iDorm, in its ability to satisfy the user preferences while dealing with the short term and long term uncertainties. We will also show the

effect of the incremental adaptation of the type-2 FLC in enabling the agent to accommodate for the accumulating long term uncertainties over extended periods to better model the user behaviour while generating fewer rules.

In Subsection (V.A), we will start by showing how the type-2 FLC generated by our simple one-pass method will better model the user behaviour and outperform a global optimisation method like GAs, while our technique is better suited to online adaptation and computationally well suited for embedded devices. In Subsection (V.B), we will compare the performance of the generated type-2 agent with the type-1 agent at modelling the user behaviour, and we will show how the type-2 agent can handle the long term uncertainties to produce a better performance that will outperform the type-1 agents while the type-2 agents used fewer rules in the rule base. We will show in subsection (V.C) the online performance of our agent when controlling the iDorm on behalf of the user and how our agent can adapt in a short time interval and handle the short term and long term uncertainties. We will compare the online performance of the type-2 agent with the type-1 agent, where the performance of both agents will be measured by monitoring how well they adjusted the iDorm environment to the user's preferences such that the user intervention was reduced over time which can be used as a measure of the user's satisfaction. We will also show that the type-2 agent can be computationally faster than the type-1 agent as a result of using fewer rules in the rule base. Finally, in Subsection (V.D), we will show the effect of the incremental adaptation of the type-2 FLC in enabling the agent to accommodate for the accumulating long term uncertainties over extended periods to better model the user behaviour while generating fewer rules.

A. Comparison of IAOFIS approach with a GA based method

In this subsection, we will compare our one pass method for generating the type-2 FLC MFs and rules with the Genetic Algorithms (GAs) which is an evolutionary based stochastic search technique [13], [17] and a well know global optimisation method that has been widely used for the purpose of the learning FLC MFs and rules [7]. In [31], [36] GAs have been used to learn the MFs and rules for type-2 FLCs of simulated processes with a limited number of inputs and outputs. We have chosen to compare our method with the GAs because of their global optimisation capability [7], [13] where they need only an objective function (minimising the training error in our case) and they do not need knowledge regarding complex differentiation which is needed by other methods such as the type-2 fuzzy neural networks [31]. In our previous work [8] we have found that the GA based methods gave the closest results to our type-1 agent as opposed to the ANFIS and Neural Networks which gave worse results.

The GA used in this paper is based on an adapted version of a Pittsburgh based GA presented in [23]. We will use this GA to learn both the type-2 MF parameters (centres as well as upper and lower standard deviations for the Guassian type-2 MFs) and rules to form a type-2 FLC. This GA used multi-chromosome representations to represent the numerical parameters of the type-2 MFs for each input/output variable as well as the rules. The GA would evolve the type-2 MFs and rules together to determine an optimum set of parameters for the type-2 FLC. The GA used a modified two phase

crossover operator which would affect the parameters for MFs or the rules based on predefined probabilities [23]. Classical real valued and one point crossover operators were applied for the MF and rule parameters respectively [23], separate mutation operations were also applied on the basis of predefined probabilities to both these sets of parameters [23]. Tournament selection over five individuals was applied during the evolution.

Under the selected Pittsburgh approach, every individual in the population generated by the GA is a complete type-2 FLC that must be evaluated in order to determine its fitness as a solution to model the user behaviours. Due to the large chromosome length, this evaluation process was a highly computationally demanding task, therefore we have restricted the GA to run under a limited configuration where 200 individuals were evolved over 100 generations and at the end the type-2 FLC that resulted in the best fitness was chosen as the solution that was further tested on the test data.

For both users, the IAOFIS and the GA type-2 FLCs were generated from the data captured from instance 1 and instance 2 that occurred at different times of the year. Due to the stochastic nature of the GAs, their training was repeated 6 times while testing on unforeseen 6 randomised test sets. Both the type-2 FLCs generated by the IAOFIS and GAs were tested on the same unforeseen test sets.

For both users, we will compare which generation method (the IAOFIS or GA) will produce a type-2 FLC that will model better the user behaviours. For each user and for both the IAOFIS and the GA based type-2 FLCs, we calculated the scaled RMSE and STDEV averaged over the six test sets, the RMSE and STDEV were scaled to account for the different ranges of the output variables. The results in Table I show that for both users our IAOFIS approach produced a type-2 FLC which has a lower average RMSE and STDEV than the type-2 FLC generated by the GA and hence the type-2 FLC generated by our IAOFIS was better able to model the user behaviour. Fig 11 shows the generated type-2 MFs for the input variable *external light level*, where Fig. 11a shows the type-2 MFs generated by the GA and Fig. 11b shows the type-2 MFs generated by the IAOFIS.

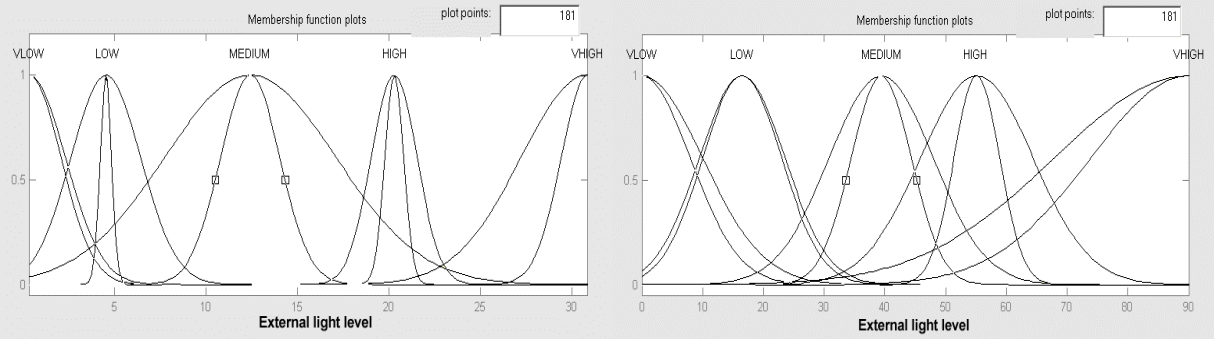
The GA generation method is based on optimising and minimising a training error to produce the FLC parameters, therefore the type-2 MFs parameters were calculated in unison to fit a globally optimal solution rather than being based on the individual uncertainties associated with each linguistic label as in IAOFIS.

TABLE I
Average SCALED RMSE AND STDEV FOR the type-2 FLCs generated by the IAOFIS AND GA.

User	GA based type-2 FLC		IAOFIS based type-2 FLC	
	Scaled RMSE	Scaled STDEV	Scaled RMSE	Scaled STDEV
User 1	0.2837	0.2329	0.1463	0.1312
User 2	0.3348	0.2729	0.2475	0.2079

Even under the limited configuration used to run the GA, it took on average several hours, to converge to its optimal parameters for the type-2 FLCs; if we increased the number iterations so that GA response approaches that of the IAOFIS, the computation time will be huge. In comparison our one pass IAOFIS approach takes a fraction of a minute to generate the agent's type-2 FLC and adapts it in seconds. The

iterative nature of the GA makes it computationally more intensive than our one pass technique which is better suited to operate on embedded computers with limited computational resources. Moreover, the GA based approach cannot easily be adapted online as this would require it to repeat its time consuming learning cycles every time the environment changes or the user behaviours are changed or adapted.



(a) (b)

Fig. 11. a) Type-2 fuzzy sets for external light level generated with GA approach. b) Type-2 fuzzy sets for external light level generated with IAOFIS approach.

B. Comparison of Type-2 IAOFIS Agent with the Type-1 based Agent at Modelling User Behaviour

In this subsection, we will show how the IAOFIS can produce a type-2 FLC that can handle the long term uncertainties to produce a very good user model that outperforms the type-1 agents while using fewer rules than that used by the type-1 agent FLCs.

For both users, we will use the datasets captured during the monitoring phases of instances 1 and 2. For *user 1*, the data recorded in instance 1 contained 408 data points while the data recorded in instance 2 contained 925 data points. For *user 2*, the data recorded in instance 1 contained 471 data points while the data recorded in instance 2 contained 967 data points. For both users, the type-1 MFs for the type-1 agents were generated from instance 1 datasets, while the type-2 MFs for the type-2 agents were generated from instance 1 and instance 2 datasets. Our aim here is to determine how the type-1 MFs generated at instance 1 would deal with the long term uncertainties arising due to the different seasonal conditions at instance 2. We want to compare this with the type-2 MFs generated using our IAOFIS which would accommodate for these uncertainties. For the sake of a fair comparison, the rule bases for both the type-1 and type-2 FLCs were generated only from the data at instance 1. In order to test the performance of the type-1 and type-2 agents at modelling the user behaviour while handling the effects of the long term uncertainties, both the type-1 and type-2 FLCs were evaluated on unseen test instances captured after instance 2. For each user, the performance for both the type-1 and type-2 FLCs was based on the scaled RMSE and scaled STDEV. Table II shows the performance on the test data for the type-1 and type-2 FLCs for both users, Table II also shows the number of rules that were generated by each FLC.

The results in Table II show that the type-2 FLC for both users produced lower values for the scaled RMSE and STDEV and hence the type-2 FLCs were better able to model the user behaviours while handling the long term uncertainties which had arisen due to the different seasonal conditions under

which the testing data was recorded. The type-2 FLC also generated fewer rules compared to the type-1 FLC. In the next subsection, we will show that this reduction in the number of rules used by the type-2 FLC will result in faster responses and less computational and memory overheads when compared to the type-1 FLCs.

TABLE II
SCALED RMSE AND STDEV FOR TYPE-1 FLC, AND IAOFIS DERIVED TYPE-2 FLC.

User	FLC	Scaled RMSE	Scaled STDEV	Number of Rules
User 1	Type-1	0.3837	0.3147	137
	Type-2	0.3244	0.2832	121
User 2	Type-1	0.3741	0.2926	194
	Type-2	0.2743	0.2208	111

C. Online Comparison of Type-2 IAOFIS Agent and Type-1 Agent at Controlling the iDorm

The analysis in the previous section had shown how well the IAOFIS could model the user's behaviour and handle the effects of the long term uncertainties which would arise from the changing conditions over different seasonal periods of the year and the associated change in user's behaviour. In this subsection, we will show how our type-2 agent performs online at controlling the real AIE of the iDorm on behalf of the user and we will show how our agent can adapt in a short time interval and handle the short term and long term uncertainties. We will compare the performance of the type-2 agent with the type-1 agent where the performance of the agents will be measured by monitoring how well they adjusted the iDorm environment to the user's preferences such that the user intervention was reduced over time, which can be used as a measure of the user's satisfaction. As the agent's performance for both users is the same and due to the space restrictions, we will present the results obtained for one of the users, namely *user 2*. The presented online experiments were performed at the end of autumn 2004, when the seasonal conditions were different from those in instance 1 and 2.

The type-2 FLC (MFs and rules) was generated from data captured in instances 1 and 2. The compared type-1 FLC had its rule base generated from data captured in instance 1 and 2 while the type-1 MFs were generated from instance 1. The reason for this was that the conditions reflected in instance 1 were closer to the unseen conditions in which the online experiments were conducted. This was to allow us to perform a fair comparison between the type-1 and the type-2 agents.

During the online experiments shown in Fig 12 both the type-1 and type-2 agents operated separately for two days where they monitored the environment and user's activities, and produced the appropriate control responses based on their learnt FLCs. The type-1 and type-2 agents controlled the environment in a non-intrusive and invisible way while the user continued to carry out his assortment of behaviours and activities. One of the characteristics of our agent is that the user is always in control and he can override the agent at any time when he is not satisfied with the agents action and his instructions are executed immediately, this is done to achieve the responsive property implied in the ambient intelligence vision [11] unless safety is compromised. Thus whenever changes to controls were made by the user (reflecting

dissatisfaction), the agent received the request, generated new rules or adjusted previously learnt rules and allowed the action through. The agent would autonomously continue to control the environment and adapt/generate new rules when the user is unsatisfied or when the state of the environment was not captured by its existing rule base and in this way the agent can adapt in the short term.

The agents' performance for these online experiments could be measured by monitoring how well the agents adjusted the iDorm environment to the user's preferences such that the user intervention was reduced over time which can be used as a measure of the user's satisfaction. Fig 12 shows over the course of the two days the number of rules that were adapted online every time the user had to override the agent's decision. From Fig 12, we can see that the type-2 agent required significantly less user interaction than the type-1 agent. The plot for the type-2 agent shows that the user intervention was initially high but then stabilised by the second day. Therefore the type-2 agent only required very short online tuning period of approximately one day. This is because the type-2 agent better modelled the user behaviour and handled the long and short term uncertainties. The plot for the type-2 agent also shows it to be more stable than the type-1 agent in controlling the environment between the points when the user had to intervene in the agents decisions and adapt the rules. These intervention points relate to the number of times or frequency of the user intervention that occurred over the duration of the two days.

In comparison the plot for the type-1 agent shows that the user intervention continued to increase and did not properly stabilise by the end of the second day. This was because type-1 agents use precise and crisp type-1 fuzzy sets that were generated under the specific environmental conditions at instance 1 and were no longer able to effectively handle the long term uncertainties associated with the new conditions encountered during the online experiments conducted in different season. The user intervention to adapt the rule base for the type-1 agent was therefore having no effect due to the change of the type-1 fuzzy set MFs values associated with the linguistic labels. As a result, the frequency of the user interaction with the type-1 agent was quite high compared to the type-2 agent as the user had to continually adjust the agent's decisions. From our experiments, we found that the type-1 agent had a 61% increase in the frequency of user interactions compared to the type-2 agent over the two days, this will eventually lead to the firing of the *system adaptation trigger* as the type-1 agent cannot handle the long term uncertainties.

As mentioned in the previous subsection, the type-2 FLC will generate fewer rules when compared to the type-1 FLC. At the start of the online experiments, the type-2 FLC started with 178 rules in the rule base compared to the type-1 FLC which started with 244 rules in the rule base. Over the subsequent two days 114 new rules were created by the type-1 agent. In comparison the type-2 agent created no new rules over the two days. The reason for this was that the type-2 agent rule base better covered the new sensor states of the environment due to the ability of the FOUs of the type-2 fuzzy sets to handle the long term uncertainties caused by seasonal variation and the associated changes in the activities of the user. The fuzzy sets for the type-1 agent were however configured for specific environment conditions and they were unable to handle the long term uncertainties. The type-1 FLC rules were therefore ineffective

at modelling the user behaviours and the agent was forced to generate a large number of new rules to try and adapt its rule base to the new conditions and this will eventually lead to the firing of the *system adaptation trigger*. The type-1 agent was therefore ineffective in handling the long term uncertainties associated with long durations of online operation. On the other hand, the type-2 FLC was able to adapt better than the type-1 FLC to the new environmental conditions with less user interaction and a fewer number of generated rules.

In order to test our rule capping mechanism to limit the rule explosion within our agents, we applied a tough limit of limiting the size of the type-2 rule base to only 100 rules, the actual limit is much bigger as it is related to the memory and processor capabilities. The type-2 agent was run online for a third consecutive day to determine how the capped rule base would affect the agent's performance. During this period the user partly altered his activities in the room and there was some additional user intervention that caused some rule adaptation as shown in Fig 12. Over the course of the third day the type-2 FLC continued to perform well without creating any additional new rules. The capped rule base maintained the most useful rules that fired most frequently and which reflected the user preferences over the previous two days. The ability for our type-2 agent FLC to handle the short and long term uncertainties allowed the reduced rule base to accommodate for the change in the user's activity under the new conditions.

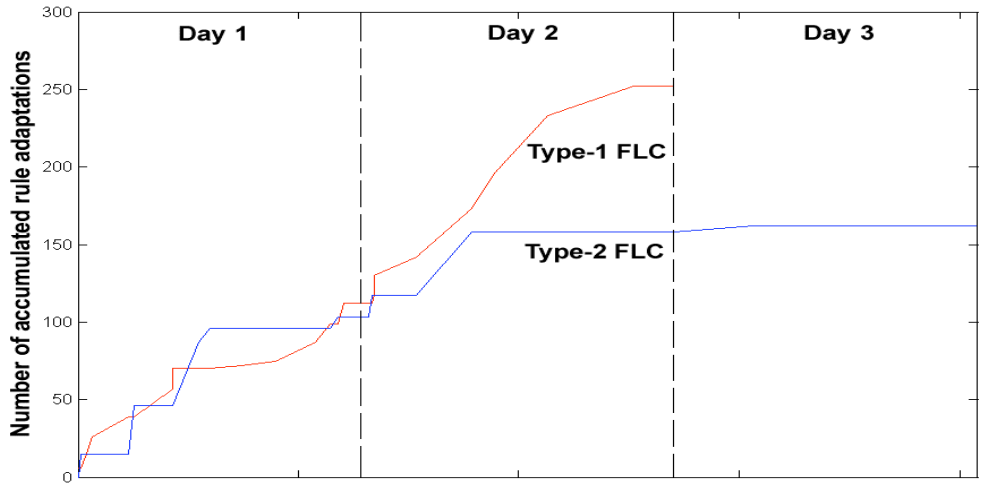


Fig. 12. Number of accumulated online user adaptations.

We have compared the computational speed of both the type-1 and the type-2 agents in which we measured the time (in milliseconds) it took for each FLC to complete a single inference and control cycle. The performance measure was based on the total generated rule base after the initial two days of online control. The type-2 agent took 1.085 ms compared with the type-1 agent which took 2.720 ms to complete a single control cycle as the type-2 agent had only 178 rules compared with the 358 rules generated for the type-1 FLC. Thus the type-2 agent was able to outperform the type-1 agent performance while achieving a 60% increase in processing speed as a result of attaining a 50% reduction in the size of the rule base, thus reducing the memory usage.

D. Incremental Adaptation of Type-2 FLC

In this subsection, we will introduce the effect of incrementally adapting the type-2 agent in a life long learning mode to accommodate for the accumulated long term uncertainties whenever *the system adaptation trigger* is fired. As mentioned above for *user 1*, the type-2 FLC needed to be adapted when the *system adaptation trigger* fired again in the middle of winter (late December 2004). We will compare type-2 FLC that operated till the *system adaptation trigger* fired in late December 2004 and the adapted type-2 FLC that was generated after the trigger was fired to accommodate for the all the long term uncertainties. Both FLCs were tested on unseen testing dataset recorded after instance 3 consisting of 374 data points. The results obtained show that the type-2 FLC agent before adaptation produced a scaled RMSE of 0.2471 and scaled STDEV of 0.2017 on the testing data, and had a rule base of 341 rules. In comparison the type-2 FLC after the adaptation produced a scaled RMSE of 0.2290 and scaled STDEV of 0.1714, and had a rule base of 328 rules. Note that the type-2 FLC before adaptation had more rules as the system tried to add more rules to accommodate for the change in the environment and user preferences which then led to the *system adaptation trigger* to fire. For the type-2 FLC before adaptation, it is obvious from that the RMSE and STDEV values that the FLC performance was degrading and it couldn't handle the new long term uncertainties that arose under the new unseen conditions of the testing data and thus it needed to be adapted. Our incremental method enabled the type-2 FLC to continuously accommodate for the accumulating long term uncertainties which resulted in a very good performance and hence the FLC continued to capture the user behaviour while using a smaller rule base.

VI. CONCLUSION

In this paper, we presented a novel type-2 fuzzy embedded agent approach for AIEs that is capable of learning and adapting to the user's particular behaviours in a non intrusive manner. The proposed type-2 agent architecture has satisfied the requirements needed for embedded agents in AIEs as it is able to handle the short and long term uncertainties and operate over long durations of time in a life long learning mode to incrementally adapt and accommodate the accumulating uncertainties. Our agents were responsive and adaptive to the needs and preferences of the user in an unobtrusive manner. The agent used the type-2 IAOFIS which a simple one pass method for generating and adapting the type-2 MFs and rules that are easily readable and interpretable by the human end user. Our agent is not computationally intensive and thus suitable for the embedded platforms available in AIEs as we have integrated our agent into an internet fridge and hence the agent was seamlessly integrated into the environment.

In order to evaluate our proposed type-2 agent, we have performed unique experiments with two different users who have different preferences and activities, both users have stayed in the iDorm at different periods over the course of the year. Over the extended period of our experiments (spanning over the course of the year), we were able to evaluate and demonstrate how the agent can adapt in a life long learning mode and handle the faced short and long term uncertainties.

We have compared the type-2 agents with the type-1 agents in their ability to model the user's behaviour while handling the long term uncertainties. The results showed that the type-2 FLC was better

able to model the user behaviour and handle the long term uncertainties while using fewer rules. We conducted online experiments in the iDorm from which we showed that our type-2 agent outperformed the type-1 agent at effectively controlling the iDorm while handling both the short term and long term uncertainties under the unseen environmental conditions. We have shown also how the type-2 agents can adapt to user behaviours and how they always generated fewer rules compared to the type-1 agents and we showed that this rule reduction will lead to faster processing and more efficient memory usage when compared to the type-1 agents. We further showed that our incremental adaptive approach could adapt the type-2 FLC such that the adapted type-2 FLC was better able to model the user behaviour and accommodate for the accumulated long term uncertainties faced while using a smaller rule base. In order to evaluate the effectiveness of our learning technique, we have compared the type-2 FLC generated by our technique with the type-2 FLC generated by Genetic Algorithms (GAs). We have shown how the type-2 FLC generated by our learning technique will better model the user behaviour and outperform the type-2 FLC generated by the GAs, while our technique is well suited for embedded devices and life long learning and adaptation as opposed to the GAs.

From this work we have shown that type-2 fuzzy logic is an appropriate framework for modelling and handling the short and long term uncertainties arising in AIEs. In our future work we aim to continue developing and testing our embedded agent based approaches using a new research facility that is currently available at Essex called the intelligent Flat (iFlat). The iFlat is a multi-room and multi-user intelligent inhabited environment, where we will be capable of running longer continuous experiments by embedding the agents in different parts of the iFlat. The iFlat will enable us to experiment with new forms and large amounts of uncertainties associated with multi users and multi rooms and this will also enable us to develop new learning techniques for the type-2 systems in AIEs.

REFERENCES

[1] G. Abowd, M. Eibling, G. Hunt, and H. Lei, "Context aware computing," *IEEE Pervasive Computing*, vol. 1, no. 3, pp. 22–23, Jul./Sep. 2002.

[2] T. Barger, D. Brown, and M. Alwan, "Health status monitoring through analysis of behaviour patterns," *IEEE Transactions on Systems Man & Cybernetics*, vol. 35, no. 1, pp. 22–27, Jan. 2005.

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

[4] V. Callaghan, M. Colley, G. Clarke and H. Hagras, "Embedding Intelligence: Research Issues for Ubiquitous Computing", in *Proceedings of the 1st Equator IRC Workshop on Ubiquitous Computing Environments*, Nottingham, pp. 110-130, September 2001.

[5] G. Castellano, A. M. Fanelli, and C. Mencar, "Generation of interpretable fuzzy granules by a double-clustering technique," *Archives of Control Science, Special Issue on Granular Computing*, vol. 12, no. 4, pp. 397-410, 2002.

[6] M. Coen, "Design principles for intelligent environments," *presented at the 15th National Conference on Artificial Intelligence*, Madison, Wisconsin, 1998.

[7] O. Cordon, F. Gomide, F. Herrera, F. Hoffmann, and L. Magdalena, "Ten years of genetic fuzzy systems: Current Framework and New Trends," *Journal of Fuzzy Sets and Systems*, vol. 141, no. 1, pp. 5–31, 2004.

[8] F. Doctor, H. Hagras, and V. Callaghan, "An intelligent fuzzy agent approach for realising ambient intelligence in intelligent inhabited environments," *IEEE Transactions on System, Man & Cybernetics*, vol. 35, no. 1, pp. 55–65, Jan. 2005.

[9] F. Doctor, H. Hagras, and V. Callaghan, "A type-2 fuzzy embedded agent to realise ambient intelligence in ubiquitous computing environments," *To appear in Journal of Information Sciences.*

[10] F. Doctor, H. Hagras and V. Callaghan, "A type-2 fuzzy embedded agent for ubiquitous computing environments," *in Proceedings of the 2004 IEEE International Conference on Fuzzy Systems*, Budapest, Hungary, July. 2004.
[11] K. Ducatel, M. Bogdanowicz, F. Scapolo, J. Leijten, and J.-C. Burgelman, "Ambient Intelligence: From Vision to Reality," IST Advisory Group Draft Report to the European Commission, 2003.
[12] K. Eng, R. J. Douglas, and P. Verschure, "An interactive space that learns to influence human behaviours," *IEEE Transactions on Systems Man & Cybernetics*, vol. 35, no. 1, pp. 66–77, Jan. 2005.
[13] D. E. Goldberg, *Genetic Algorithms in Search, Optimisation, and Machine Learning.* Reading, Massahusetts: Addison-Wesley, 1989.
[14] H. Hagras, "A hierarchical type-2 fuzzy logic control architecture for autonomous mobile robots," *IEEE Transactions on Fuzzy Systems.*, vol. 12, no. 4, pp. 524–539, Aug. 2004.
[15] 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, Nov/Dec. 2004.
[16] H. Hagras, M. Colley, V. Callaghan, G. Clark, H. Duman, and A. Holmes, "A fuzzy incremental synchronous learning technique for embedded-agents learning and control in intelligent inhabited environments," in *Proceedings of the IEEE International Conference on Fuzzy Systems*, Honolulu, Hawaii, 2002, pp. 139–145.
[17] J. H. Holland, *Adaptation in Natural and Artificial Systems.* Ann Arbor: University of Michigan Press, 1975.
[18] A. K. Jain, R. C. Dubes. *Algorithms for clustering data*, Prentice Hall, 1998.
[19] N. Karnik and J. Mendel. (1998) An introduction to type-2 fuzzy logic systems. University of Southern California, Los Angeles, California. [Online]. Available: http://sipi.usc.edu/~mendel/report/.
[20] S. Kobashi, N. Kamiura, Y. Hata, and 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.
[21] Q. Liang, N. Karnik, and J. Mendel, "Connection admission control in ATM networks using survey-based type-2 fuzzy logic systems," *IEEE Transactions on Systems, Man, & Cybernetics, part C*, vol. 30, no. 3, pp. 329–339, Aug. 2000.
[22] Q. Liang and J. Mendel, "Interval type-2 fuzzy logic systems: Theory and design," *IEEE Transactions on Fuzzy Systems*, vol. 8, no. 5, pp. 535–550, Oct. 2000.
[23] L. Magdalena, F. Monasterio, "A fuzzy logic controller with learning through the evolution of its knowledge base," *International Journal of Approximate Reasoning*, vol. 16, no. 3-4, pp. 335–358, Apr/May. 1997.
[24] J. Mendel, *Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions.* Upper Saddle River, New Jersey: Prentice-Hall, 2001.
[25] J. Mendel and R. John, "Type-2 fuzzy sets made simple," *IEEE Transactions on Fuzzy Systems*, vol. 10, no. 2, pp. 117–127, Apr. 2002.
[26] M. Mozer, "The neural network house: An environment that adapts to its inhabitants," in *Proceedings of the American Association for Artificial Intelligence Spring Symposium on Intelligence Environments*, AAAI Press, 1998, pp. 110-114.
[27] L. Rudolph, "Project oxygen: Pervasive human centric computing – An initial experience," in *Proceedings of the 13th International Conference on Advanced Systems Engineering CAiSE, Interlaken, Switzerland, June 2001*, pp. 1-13.
[28] U. Rutishauser, J. Joller, and R. Douglas, "Control and learning of ambience by an intelligent building," *IEEE Transactions on Systems, Man & Cybernetics*, vol. 35, no. 1, pp. 121–132, Jan. 2005.
[29] SONY Interaction Laboratory. Interaction Lab. [Online]. Available: http://www.csl.sony.co.jp/IL/index.html.
[30] M. M. Trivedi, K. S. Huang, and I. Mikic, "Dynamic context capture and distributed video arrays for intelligent spaces," *IEEE Transactions on Systems Man & Cybernetics* , vol. 35, no. 1, pp. 145–163, Jan. 2005.
[31] C. Wang, C. Cheng, and T. T. Lee, "Dynamical optimal training for interval type-2 fuzzy neural network (T2FNN)", *IEEE Transactions on Systems, Man and Cybernectics- Part B: Cybernetics*, vol. 34, no. 3, pp.1462-1477, June 2004.
[32] L. X. Wang, "The MW method completed: A flexible system approach to data mining," *IEEE Transactions on Fuzzy Systems*, vol. 11, no. 6, pp. 768–782, Dec. 2003.
[33] R. Want, G. Borriello, T. Pering, K. I. Farkas, "Disappearing Hardware," *IEEE Pervasive Computing.*, vol. 1, no. 1, pp. 36–47, Jan./Mar. 2002.
[34] M. Weiser, "Some computer science problems in ubiquitous computing," Communications of the ACM, July 1993. (reprinted as "Ubiquitous Computing," *Nikkei Electronics*., pp. 137-143, December 6, 1993.)
[35] H. Wu and J. Mendel, "Uncertainty bounds and their use in the design of interval type-2 fuzzy logic systems," *IEEE Transactions on Fuzzy Systems*, vol. 10, no. 5, pp. 622–639, Oct. 2002.
[36] D. Wu and W. Tan, "A type-2 fuzzy logic controller for the liquid level process," *Proceedings of the 2004 IEEE International Conference on Fuzzy Systems*, Budapest, Hungary, July. 2004.

APPENDIX I

I. EXTRACTION OF TYPE-1 FUZZY MEMBERSHIP FUNCTIONS FROM DATA

We have used an approach to categorise the accumulated user input/output data into a set of type-1 fuzzy MFs which quantify the raw crisp values of the sensors and actuators into linguistic labels. The MFs are defined from the user's input/output data recorded by the agent. A Double Clustering [5] approach combining Fuzzy-C-Means (FCM) and hierarchical clustering is used for extracting type-1 fuzzy MFs from the user data. This is a simple and effective approach to fuzzy information granulation [20] where the objective is to build models at a certain level of information granulation that can be quantified in terms of fuzzy sets.

Our dataset of user instances contains *r* attributes corresponding to the number of input and output variables. We start by generating *p* initial clusters using the FCM approach. Each cluster has a centre $\bar{c}_i$ which is an *r*-dimensional vector having *r* centroid values ci_{ij}. For each dimension $j = 1,2,...,r$, we perform agglomerative hierarchical clustering [18] on the set of one-dimensional centroid values $C_j := \{ci_{ij} | i = 1,2,..., p\}$ to obtain V_j final cluster centres to be converted to the extracted fuzzy sets (linguistic labels).

A. The FCM Algorithm

The FCM [3] is a fuzzy partitioning prototype-based clustering algorithm. It uses features in the *r*-dimensional Euclidean space to determine the geometric closeness of data points by grouping them into clusters [3]. FCM takes a dataset of instances with *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 which 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. The algorithm aims to iteratively adapt the partitioning of the dataset so as to minimise a dissimilarity function of a weighted sum of squared distances between data points and cluster centres in the feature space.

B. Double Clustering

The Double clustering technique uses a combination of FCM and Hierarchical clustering for extracting a predefined number of fuzzy sets for the input and output variables 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 $\bar{c}_i = (ci_{i1}, ci_{i2}, ..., ci_{ir})$ and there are therefore *p* one-dimensional centroid values for each input and output variable 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 fuzzy sets that will be extracted for each input and output variable. Specifically let ci_{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 $C_j := \{ci_{ij} | i = 1,2,..., p\}$. The approach used for this secondary clustering is an agglomerative hierarchical clustering approach [18]. 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 fuzzy sets we want to extract for each input and output variable. The similarity between two elements is measured based on the closeness between their values. The number of fuzzy sets to be defined for each input and output variable is predefined in advance.

C. Agglomerative Hierarchical Clustering Approach

The agglomerative hierarchical clustering algorithm which is used can be formally described as follows: Let $V_j \quad (j = 1,2,...,r)$ represents the required number of centres and the corresponding number of fuzzy sets to be derived from each set C_j, where V_j is a fixed number which can be different for the input and output dimensions of *r*. The elements of C_j are initially sorted such that $i_1 < i_2 \rightarrow ci_{i_1,j} \leq ci_{i_2,j}$. Hence the initial set of elements in C_j is defined as:

$$pr^{(0)} := \{pr_1^{(0)}, pr_2^{(0)}, ..., pr_p^{(0)}\} := \{ci_{1j}, ci_{2j}, ..., ci_{pj}\} \tag{30}$$

For $v = 1,2,..., p - V_j$ 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 $pr^{(v)} := \{pr_1^{(v)}, pr_2^{(v)}, ..., pr_{p-v}^{(v)}\}$,

$$pr_i^{(v)} := \begin{cases} pr_i^{(v-1)}, & i < i^* \\ \left(pr_{i^*}^{(v-1)} + pr_{i^*+1}^{(v-1)}\right)/2, & i = i^* \\ pr_{i+1}^{(v-1)}, & i > i^* \end{cases} \tag{31}$$

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

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 we have derived V_j new centres for each input and output dimension of the dataset. We represent the set of these centres by $pr := pr^{(p-V_j)} =: \{pr_1, pr_2, .. pr_{V_j}\}$ which correspond to the rough centres for the fuzzy sets (linguistic labels) that will be extracted for each

input and output variable.

D. Quantification of Fuzzy Membership Functions

The V_j cluster centres defined on each dimension $j = 1,2,...,r$ are then converted to the *V* type-1 fuzzy membership functions, which involves the quantification of the centres in terms of interpretable fuzzy sets [5]. As mentioned in section IV. (B) the value of *V* defines the number of fuzzy sets which are to be extracted for each input and output variable where $V_j = V_i$ for the input variables and $V_j = V_o$ for the output variables. Gaussian membership functions are used to describe the fuzzy sets A_z^j, (where $z = 1,2,...,V$) the mathematical definition of which is

$$\mu_{A_z^j}(x) = \exp\left\{-\frac{1}{2}\left(\frac{x - m_z^j}{\sigma_z^j}\right)^2\right\} \tag{32}$$

where the value of the centre m_z^j and the standard deviation σ_z^j for each gaussian membership function *z*, for the *j*-th input/output variable is derived as follows.

The sampled data is defined as a hyper-interval $X := \overset{r}{\underset{j=1}{\times}}\left[g_j, G_j\right]$ where g_j and G_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 $T_j := \left\{ti_0^j, ti_1^j, ..., ti_{V_j}^j\right\}$, where:

$$ti_d^j := \begin{cases} 2g_j - ti_1^j, & d = 0 \\ (pr_d + pr_{d+1})/2, & 0 < d < V_j \\ 2G_j - ti_{V_j-1}^j & d = V_j \end{cases} \tag{33}$$

The centre m_z^j and standard deviation σ_z^j of each fuzzy set A_z^j for all $z = 1,2,...,V$ is derived from the set T_j as follows:

$$m_z^j := \left(ti_{z-1}^j + ti_z^j\right)/2 \tag{34}$$

$$\sigma_z^j := (ti_z^j - ti_{z-1}^j)/2\sqrt{-2\ln\varepsilon} \tag{35}$$

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

We therefore obtain the centres and standard deviations for a set of *V* type-1 fuzzy sets for each input and output variable of the user data that was sampled. These fuzzy sets are distributed over the range of values of each variable. The fuzzy sets 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.

