

Learning and adaptive fuzzy control system for smart home

Vainio A.-M., Valtonen M., Vanhala J.

Tampere University of Technology, Institute of Electronics
P.O. Box 692, 33101 Tampere, Finland
{antti-matti.vainio, miika.valtonen, jukka.vanhala}@tut.fi

Abstract

Automated smart homes have widely established their position as a research field during the last decade. More and more context sensitive concepts are being studied and at the same time proactivity has broken through in ambient intelligence research. Technology has advanced towards an adaptive and autonomous home, which can take care of the inhabitants' well-being in numerous ways.

We propose to use a context sensitive and proactive fuzzy control system for controlling the home environment. Since humans communicate using fuzzy variables, we see that applying context recognition to a fuzzy control system is straightforward. The designed control system is adaptive, and it can accommodate to changing conditions of inhabitants. Our system is designed to operate fully in the background and needs very little effort from its users. The system utilizes a principle of continuous learning so that it does not require any training prior to use.

This paper describes a lighting control system implemented using the fuzzy control system designed. We concentrate on the basic operation of such systems and present findings from the design process and initial tests.

1 Introduction

Imagine someone living in a smart home equipped with a context sensitive and adaptive control system. Most of the time the home is able to react correctly to the actions of the resident or take proper proactive actions, both based on a world model and measured variables. Inevitably at some stage the system makes an error and the resident has to take a corrective action. This enables the system to update its world model in order. Two research questions rise from this simple scenario: How to construct and update a world model? And how can the system differentiate between a corrective interaction and normal behavior of the resident. Our system answers these questions by utilizing context recognition and fuzzy control.

Context recognition is an effective method for providing application-specific information and enabling context

triggered actions [1] [2]. It can be used for creating intelligent environments, and by utilizing machine learning algorithms with context recognition proactive systems can be established. [3] [4] Adaptive homes have been built by using context recognition [5] [2] and neural networks as a learning mechanism. A context-aware home can serve its inhabitants more flexibly and adaptively [4]. Context recognition deals with fuzzy quantities of environment, like warmth, brightness and humidity. Humans also perceive their environment with fuzzy variables [1]. Therefore, we see that a fuzzy control system would be suitable for controlling the home. Although the context-aware neural networks, which have already been utilized, can be converted to fuzzy systems, the implementation of a fuzzy system is much easier [6]. Moreover, by utilizing fuzzy systems for controlling the home and as a pre-processing and context learning mechanism, we believe that a more genuine and natural environment can be achieved.

We have developed a fuzzy control system that can learn its rule table without any predefined information and doesn't need any training prior to use. It can add and remove rules and modify existing ones based on learned information. With this versatile learning ability, we see that the control system can become unnoticeable after its initialization. Our implementation consists of a lighting control system that is implemented into a smart home.

1.1 Background

Smart environments have long been researched at the Institute of Electronics. Different technologies for smart spaces and their devices, networks, user interfaces and software solutions have been studied extensively in different projects. [7] The first smart space constructed was Living room, a former laboratory converted into a living room with small kitchen and a hall. The eHome-project took testing smart spaces to another level with ordinary people living in a smart apartment equipped with smart appliances and control for a long period of time. These studies highlighted the need for a control system that could learn from the behaviour of its users and use that information to adapt to people's needs without explicitly configuring the system. In other words, it would make the home actually feel smart.

In the study we focused on fuzzy systems, because they are well suited for dealing with imprecise quantities used by

humans. Fuzzy systems are fairly easy to understand and construct, because they are based on natural language. The construction of the rule base in fuzzy systems is also quite easy to automate, because the method using natural language is simple to turn into a computer program. Control systems using fuzzy logic are generally fast, user friendly, cheap and they don't need much memory. [8]

We decided to use a lighting system as a prototype, because its operation could be easily observed visually and we had an existing infrastructure for such a system. The infrastructure allowed us to fully concentrate on developing the fuzzy control system instead of hardware components. The prototype was built to be an integral part of the smart home laboratory that is used as a testing space for new technologies.

1.2 Objectives

The starting point of the research was to study modelling of the dynamic behaviour of a home environment using fuzzy logic. A key element was to find algorithms to adapt the designed model to changing situations. After the design process we intended to create a control system for a smart home test environment. The study focuses greatly on context-based control and on learning process with fuzzy control.

The objective for this control system was to build a fuzzy architecture that would support learning from user actions and proactively anticipate users' needs based on the learned data. From the early stages of the control system design, one major target was to keep the system as unobtrusive to a home's inhabitants as possible. This of course determined that we would have to use user-friendly sensing methods and user interfaces. At the end of the study we aimed to validate the system with functional and user tests.

The study began from the premise that no predefined rule base, however intelligently and thoroughly devised, can possibly suit for everyone. Our habits and needs are just too different. Therefore the only one who can make a good rule base for a given user is that user himself. To accomplish this, inhabitants' actions would have to be monitored, and system would have to learn through these observations. The learning process would need to be continuous, because our habits and routines change over time.

Another objective for the system was that it would have to need minimum effort from the inhabitant. The less the inhabitant has to know about this system, the better. This objective benefits from systems continuous learning process, as it doesn't need a specific teaching mode or period. Also the system would have to need a minimum number of specific controls in addition to normal controls for lights and Venetian blinds. Small number of controls does not imply lack of control though. The inhabitant has to have the ultimate control over his home. Therefore the inhabitant has to have a chance to override all decisions that the fuzzy system makes.

The designed system should have two different control modes: autonomous control and event-based control.

Autonomous control would have total control of the environment and it would anticipate users' needs based on the learned information. Event-based mode would react to inhabitant's overrides and enable system to trigger chains of controls based on immediate user actions. These two control modes would enable the system to flexibly adapt itself to the users' needs.

In addition the system was designed to be fairly easy to implement and existing infrastructure was to be used where applicable. In practise this meant that the system would be built on a software platform already in place in the smart home. It would also need to be easily modifiable to other areas than lighting. If a new system would be needed, it should be enough to define input and output devices with their membership functions.

2 Methods

2.1 Research facilities

The smart home laboratory was built during the eHome-project in 2002-2004, and it is located in the Institute of Electronics facilities. This smart home is a 60 m² apartment with some special modifications to help make technology invisible. It has removable floor and ceiling tiles, lots of space for equipment and customized electrics, which allow us to reconfigure lights, wall sockets and switches as needed. A picture of the smart home is shown in figure 1.



Figure 1 The smart home

The smart home is equipped with various devices and technologies, including several wireless communications networks, host of sensors that monitor conditions of the apartment, motorised curtains and front door, fingerprint scanner, infrared tag ID and positioning system and a speech recognition system. All this is hidden from the user as well as possible, so that technology remains invisible until it is actually used. To make the apartment look and feel comfortable it is furnished just like a normal apartment. [9]

The user can control the smart home through various user interfaces (UI). There are many different kinds of user

interfaces, as no interface is convenient in every situation. Provided control methods are a tablet PC, a mobile phone, TV and a speech recognition system. All interfaces connect to the home controller server that acts as a central gateway for the user interfaces and the devices. The server is connected to all devices of the apartment through different network technologies. Home controller offers a set of services to UIs, such as constant stream of up-to-date information of conditions in the home, grouping services for devices, timers and a logging service. Home controller is designed to work as a platform, on which intelligent services can be built by specializing certain classes. [10] The fuzzy system prototype was built this way. This way we were able to use the entire existing infrastructure in the smart home with minimum effort.

2.2 Simulations

Prior to implementation the fuzzy system was tested in a special simulator, which has a simple model of all of the system components. The simulator is a computer program with a user interface that enables us to monitor the system's operation at run time. This way we could first verify that the prototype was viable. The simulator was also very handy because it had a feature that allowed us to simulate passing of time with an accelerated rate. This saved much time in early stages of testing. The simulator is quite simple to configure for individual test arrangements and it could easily be used with other systems besides the lighting prototype.

2.3 Fuzzy systems

Since human speech and communication use fuzzy variables that have inexact bounds, fuzzy systems present an easy way to communicate between humans and computers. We propose that by using fuzzy systems a more natural and realistic environment can be established.

2.3.1 Fuzzy logic

Fuzzy systems theory is based on uncertainty and on imprecision. Uncertainty is very natural to humans and people usually make decisions based on indiscrete observations. [8] A man can reason that he will not need an umbrella when going outdoors, if it does not rain much. Here the meaning of the fuzzy word 'much' is dependent on context and from the point of comparison.

The presentation of uncertainty is complicated to a computer using traditional methods. Among scientist uncertainty has been considered to be an inevitable part of speech and a precise expression has been an objective of research for a long period of time. [8] For example, for a computer it is very hard to understand what dark and bright means. Let us define a border between dark and bright to 100 luxes. If the sun shines outdoors, the illumination can be measured to be 50 000 luxes. Here it is quite easy to say that it is bright outside. However, if we measure a workspace to have 110 luxes, the room is still brightly illuminated. That doesn't sound very meaningful to a man. Hence, the borderline cases are difficult to handle.

The problem exists in the binary logic of the computer. Outdoor light level can be either dark or bright. Fuzzy logic presents a solution called multi-value logic. Using multi-value logic illumination level can be dark and bright at the same time, that is more than two truth-values exist. Here, it possible for illumination level to be at the same time much bright but little dark. There is no need to define exact bounds for continuous values and the different regions of truth can overlap each other. [8]

Linguistic variables are used in fuzzy systems as a method for performing calculations. The values of linguistic variables can be presented using membership functions that define the degree of membership in real input or output spaces. [8] Figure 2 demonstrates how the linguistic variable 'illumination' is defined with values 'dark', 'normal' and 'bright'. For example, with a real input value of 150 from a light sensor illumination is characterized to be dark in degree of 0, normal in degree of 0.8 and bright in degree of 0.2.

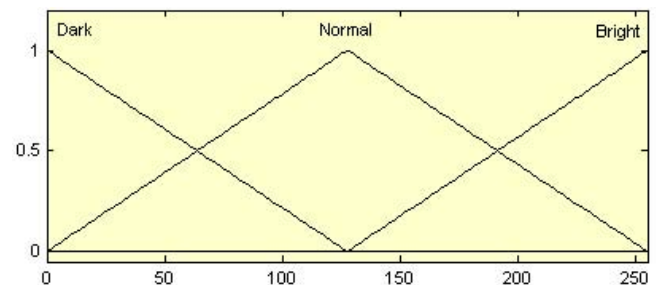


Figure 2 Linguistic variable 'illumination' defined with three membership functions

2.3.2 Fuzzy control

Fuzzy control systems have many abilities that conventional control systems don't possess because of their multi-value logic. Fuzzy control includes three major steps: fuzzification, fuzzy inference and defuzzification. [8]

Fuzzification

In order to be able to do fuzzy inference the real measurement values must be fuzzified. Fuzzification is performed using membership functions that define the input spaces and the values' degrees of membership for each variable separately. During fuzzification, every value of a linguistic variable receives a degree of membership based on the definition of the membership function and on the real measurement. An example of this was already presented with the linguistic variable 'illumination' in the last chapter.

Fuzzy inference

Fuzzy systems utilize a relatively simple rule table that represents the behavior of the whole system. The rules are of 'if - then' type. In the 'if' part the inputs' and in the 'then' part the outputs' linguistic variables' values are defined. 'If' part determines whether the rule is or is not valid in the current situation. 'Then' part is used to define the states of the outputs.

All the fuzzified input values must be aggregated to achieve a complete degree of truth to a rule. The aggregation is usually done using a minimum operator that takes the smallest degree of membership of the values of the input variables.

Composition follows next from the input aggregation. In the composition step the aggregated degree of truth of the 'if' part is multiplied by a weighing factor to get a degree of support (DoS) for the whole rule. Weighing factor or weight is associated with each rule and describes the significance of the rule. The weighing factor is usually defined to be in the interval [0, 1].

Since many of the rules can be true at the same time to a different degree, the result must be aggregated for each output variable's value separately. This is usually done by selecting the biggest result after the composition step controlling the output's value. This value determines the linguistic variables' output's degree of membership.

Defuzzification

After result aggregation the linguistic variables are defuzzified to real output signals to actuators. Defuzzification can be done in several ways. One common method is the center of gravity defuzzification. All the values of the output linguistic variables are filled from zero to the degree of the membership of the value. Then the center of the filled area is calculated and the real result is read from that point.

3 System design

3.1 Fuzzy variables

The fuzzy control system uses seven linguistic input variables. The main input variables are 'outdoor lighting level', 'person activity' and 'time'. In addition to these, the states of the two outputs and their corresponding override flags are considered as inputs. The two linguistic output variables used are 'ceiling lighting power' and 'Venetian blinds position'.

3.1.1 Inputs

The operation of a fuzzy control system is based on context recognition. Only a few sensors are used to recognize contexts in the home environment. Many different types of sensors could be used, but we prefer to measure variables that directly relate to the use of lighting in a home. Measuring only a few variables makes also the implementation of the system simpler. The sensors used are person activity and outdoor lighting level.

During the design of the lighting control system we saw a need for the system to behave differently at different times of the day. Using time as an input the system is able to react differently in the morning and in the evening while the other

conditions can be closely the same defined by the other two sensors.

Outdoor lighting level

The system measures outdoor lighting level constantly to be able, for example, to close the Venetian blinds when it gets dark outside. We have presupposed that the outdoor lighting level affects the use of Venetian blinds and ceiling lights that are used as outputs. The outdoor lighting level sensor gives a real measurement value between 0 and 255. This value is fuzzified to a linguistic variable using similar membership functions as shown in figure 2. The membership functions overlap each other so that a softer control is achieved.

Person activity

It is essential to know whether the inhabitant is in the controlled space. This is because we see that the control system must act differently when a person is present and when he is not. A person activity sensor is used to give out binary information. The sensor gives a real output value of 0 or 255. This value is fuzzified to two different values of the linguistic variable 'absent' and 'in place'. Using membership functions shown in figure 3 it can be seen, that only one membership function can receive a membership degree of one at a time.



Figure 3 Two membership functions of the linguistic variable 'person activity'

Time

The system needs to be able to react differently to same conditions at different times of day. For example, an inhabitant may want to keep Venetian blinds closed in the morning and have them open in the evening. In this case, the measured conditions by the outdoor lighting and person activity sensors might be close to the same as in the morning. If time is not kept as an input to the system, the system cannot differentiate between the morning and the evening. Consequently time is an essential input of the fuzzy control system.

The system does not use an accurate clock with even a minute precision. Instead, a day has been divided to multiple time zones, which all cover several minutes of time. By using time zones instead of minutes, the amount of rules can be kept remarkably lower. If a minute scale should be used, there would be many rules for each minute. The usage of time zones also benefits the learning process.

Long-time transient conditions that last for several minutes can easily be ignored, if a time zone is many times wider than the maximum ignorance time wanted in learning process.

There is no exact limit for the minimum number or width of time zones. However, the system's control time response is affected greatly, if a time zone is many hours in width. That happens, because within a single time zone only a limited set of rules can be effective. All of these rules must have the currently active time zone defined on the input side of the rule. These rules also remain effective throughout the time zone, if no new rules are learned.

It can be seen, that the optimal amount of overlap of adjacent time zones is one half. If the adjacent membership functions overlap each other by more than a half, the control accuracy remains still quite the same, if the width of a time zone still remains the same. However, more rules that have more closely the same outputs will generate to the rule base during learning as more than two membership functions overlap each other. In contrast, if the adjacent membership functions overlap each other by less than a half, there will be shorter transition time from one time zone to the next, resulting in sharper changes in the environment. These changes will happen at predefined times, which do not have correspondence in the real environment.

In our implementation we have divided a day into 50 different time zones. Each time zone covers approximately 58 minutes of the day. Each of these time zones overlaps adjacent time zones or membership functions by half as shown in figure 4. The spacing of time zones is hence approximately 29 minutes. The used spacing was selected to get quite precise control as time passes by, but also to keep the maximum ignorance time used in the learning long enough. Using the selected spacing we have seen that the number of rules will stay as low as in a few hundred rules.

The system receives the time as an ordinary input from the server's system clock. The time is represented in minutes (0 to 1440) from the beginning of the day. A problem arises when time passes midnight and the real time input value changes from 1440 to 0. Rules having time defined by the first and the last membership functions may have different outputs. This can cause the outputs to suddenly change at midnight, when the effective rules change. A smooth transition is required at midnight as it is on other times of the day.

Since time's continuity can be visualized with a circle, but the real input range of the linguistic variable 'time' is represented with a line, time must be artificially made continuous. To accomplish this, one extra membership function is presented. This extra membership function is inserted in the beginning of the real time scale and the other membership functions are moved 28 minutes further. Hence, both the first and the last membership functions reside half out of the used time scale. If the rules including the first and the last membership functions have same outputs, a smooth transition is achieved when time passes midnight. Using this method, only 50 membership functions

actually can be thought to reside in the used time scale and the time is handled as continuous.

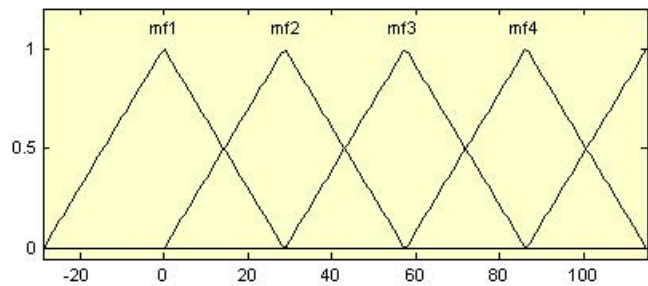


Figure 4 The first four membership functions of the linguistic variable 'time'. The real time is represented in minutes from the beginning of the day.

Override flags

The fuzzy control system must know if the output devices are overridden, in order to be able to swap to event based control. The home controller keeps record on the user overrides and gives out the real values to fuzzy control system. The state of the overrides is represented with linguistic variables 'ceiling lighting override flag' and 'Venetian blinds override flag'. Both variables' values are 'off' and 'on' and are determined by similar shaped membership functions as shown in figure 3. The first membership function defines the 'off' state and the second membership function the 'on' state.

Actuators as inputs

The states of the actuators are used as inputs to enhance the accuracy of the context recognition when event based control is used. The linguistic variables of these inputs are named with the same names as used with the outputs. The input linguistic variables utilize exactly the same membership functions as the output linguistic variables use.

3.1.2 Outputs

Two different kinds of actuators are used to change the lighting environment in a home. The ceiling lights provide artificial light from above and Venetian blinds adjust the amount of light coming in through windows.

Ceiling lights

Ceiling lights utilize a linguistic variable 'ceiling lighting power'. The variable has five different linguistic values as shown in figure 5. The membership functions of these values are laid evenly and overlap each other. The overlap makes a soft control more achievable. During defuzzification, the linguistic values are converted to a real value using center of gravity method. This real control value range of ceiling lights is defined to be 0 to 255. However, larger scale is used to set the 'small' and the 'much' membership functions enough out from the used scale, to achieve real outputs values of 0 or 255 using center of gravity method for result aggregation. If a real value outside

the used scale is received from defuzzification, it is limited to the scale used.

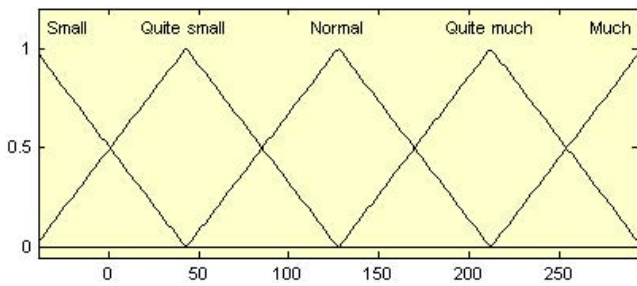


Figure 5 The membership functions of the linguistic variable ‘ceiling lighting power’

Venetian blinds

The position of the Venetian blinds is defined in the fuzzy control with a linguistic variable ‘Venetian blinds position’. The variable has five different values: ‘closed down’, ‘down’, ‘center’, ‘up’ and ‘closed up’. All of these values have membership functions, which are similar to membership functions shown in figure 5. The real scale used for controlling the blinds is also from 0 to 255. A zero marks the ‘closed down’ position and 255 the ‘closed up’ position. For same reasons as described above, a larger real scale is used with Venetian blinds as well as with ceiling lighting.

3.2 Fuzzy control

The fuzzy control system monitors the context of the home with input devices and changes the environment using its actuators according to learned rules. Inhabitants don’t need to interfere with the control system at all, but can override the actuators if needed. Two control modes, autonomous control and event-based control, are provided to make the system function better in different situations.

Conventional switches on the walls or other user interfaces let the user define an exact state for the actuator. If an actuator is manually adjusted, the control of the actuator is taken away from the fuzzy control system and the user is given a direct control over that device. At the same time, autonomous control is halted and event-based control takes charge of all other devices that are not overridden.

Autonomous control can be resumed by inactivating overrides. A special button for this use is provided to inhabitants. No automatic method to return to autonomous control is provided, because the user may want to teach the house even while he is no longer present.

3.2.1 Autonomous control

In autonomous control mode the system is totally self-ruling and independent. The user does not need to concentrate on controlling the home, for the home adapts to the changes observed. The context of the home is analyzed at all times and the actuators are proactively adjusted when a change in the context is recognized.

Inference

The problem solving or inference between input and output variables in the system is based on a rule table. Each rule has its own row in the table. The columns represent the values of the linguistic variables of inputs and outputs. Each rule defines a certain context with its inputs and the corresponding outputs define the states of the actuators.

In fuzzy inference, rules that depict the current situation, that is have linguistically same fuzzy input values as currently measured, receive a DoS value over zero. Here minimum operator is used for input aggregation and maximum operator for the composition step. All rows that get a DoS value above zero define the linguistic output values targeted. If no rule gets a DoS value over zero, the actuators positions are not altered.

An example of a rule base with three rules is shown in table 1. The first value or membership function of a linguistic variable is marked with one, the second with number two and so on. The first two rows show an example of rows used for autonomous control. The override rule type in the last row is used in event-based control.

Table 1 An example of a rule table

Linguistic variable / Rule type	Inputs								Outputs	
	Ceiling lighting override flag	Ceiling lighting power	Venetian blinds override flag	Venetian blinds position	Room lighting level	Outdoor lighting level	Person activity	Time	Ceiling lighting power	Venetian blinds position
Autonomous	1	0	1	0	3	3	1	12	2	3
Autonomous	1	0	1	0	2	1	2	40	5	1
Override	1	0	2	5	3	2	2	0	4	0

Table 2 shows all the possible types of rules used and the possible values in the rule table with the used rules. In autonomous control, the override flags of outputs on the input side are defined to be off, marked with number one. The output states on the input side are marked with zeros, so that the state of an output is ignored during the input aggregation. All the other values of the variables can be anything in the range of the used membership functions count. The override type rules used in event-based control are covered in the next chapter.

Table 2 Rule base definition with all possible types of rules

Linguistic variable / Rule type	Inputs							Outputs		
	Ceiling lighting override flag	Ceiling lighting power	Venetian blinds override flag	Venetian blinds position	Room lighting level	Outdoor lighting level	Person activity	Time	Ceiling lighting power	Venetian blinds position
Autonomous	1	0	1	0	Any				Any	
Override	2	1-5	1	0					0	1-5
Override	1	0	2	1-5					1-5	0
Not possible	2	0	2	0					Any	

3.2.2 Fuzzy control process

The fuzzy control process runs continuously. The home controller enquires sensors every second and passes the information to the fuzzy control system if any of the input variables change. The measurements are fuzzified, fuzzy outputs are calculated in fuzzy inference and then defuzzified using the center of gravity method into real control signals. The real output values are then forwarded to the home controller, which sets the actuators to the positions wanted. If a user generates an event by overriding an actuator, all inputs are immediately measured and passed to the fuzzy control process.

3.2.3 Event based control

To achieve a more proactive and context sensitive control system, we propose that in addition to autonomous control the actuators should also be controlled as they are manually being adjusted. For example, as a user manually overrides autonomous control and closes the Venetian blinds, the system can adjust the ceiling lights to another power setting learned.

The context is sensed at the time of an event also with the state where the actuator was displaced and with the other input parameters as well. However, time is excluded, because we see that the learning process of these event based rules would take too much time, if separate rules for each time zone ought to be learned. The control system couldn't do anything in case of an event, if the time zone was wrong.

To enable these proactive actions based on user adjustments, a class of override rules is introduced as shown in tables 1 and 2. Contrary to autonomous rules, these event-based rules have one or more override flags lifted up. However, not all override flags can be up in a rule at the same time. The rules with all override flags lifted up would be useless, because the fuzzy control system couldn't control the actuators in any case.

These override rules tell the fuzzy control system to control all the other outputs that are not overridden based on the

current situation or context. During fuzzy inference, the rules that depict the current environment become effective and the outputs are changed in a normal fashion. The value of the corresponding linguistic variable of the manually adjusted actuator is marked in the rule base with a zero, to show that the actuator will not be adjusted.

3.3 Learning process

The fuzzy control system can learn its rule table without prior knowledge and does not need any training prior to use. It can add and remove rules and modify existing ones based on the learned information. With this versatile learning ability, a fuzzy control system can become totally unnoticeable after its initialization.

Learning process is based on monitoring the context in the home. Fuzzy control system monitors input and output devices, so it is always aware of the state of the home. Teaching is done in four steps, which are gathering of data, fuzzification of data, filtering and updating the rule base. These steps and the connection between the home controller and the learning process are shown in figure 6. Teaching is controlled by timers that define how often these steps occur.

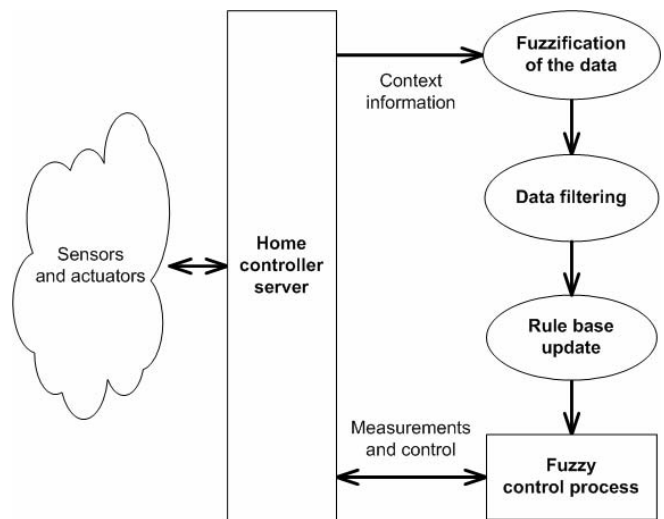


Figure 6 The learning process as a part of the system structure

3.3.1 Saving context information

Gathering the data is a continuous process. The purpose of this step is to get enough data of users actions so that routines become clear. As mentioned, the fuzzy control system monitors both the input and the output devices. It writes their current values periodically to storage. A timer controls this process. The interval of the timer therefore defines the amount of data gathered. Because this data is filtered at later stages, this interval does not really define what the system will learn. Instead, it defines the sampling period and the duration of which the system does not have any information.

If data is written once a minute, the system does not know what happened during that minute. User may have changed some output to some other value and back, and the system

has no idea what happened. Therefore this writing interval should not be very long. On the other hand, there is no use to write information for example once per second. The conditions of the home do not change that often, even when user controls it, and most of the information would be filtered out anyway. We came to a conclusion that one minute was a reasonable timer interval for writing data. With that sampling rate the system gets enough data to process, and the system is not needlessly encumbered by collecting too much data that is not needed in the first place.

3.3.2 Fuzzifying the data

Processing the data starts with fuzzification. Both the inputs and the outputs are read from their storage and fuzzified to a value representing their membership function with the greatest degree of membership. The value of this linguistic variable is then stored in order to be used in the learning process. The time input variable is treated slightly differently. Instead of choosing the membership function with the greatest degree of membership, system picks all membership functions whose degree of membership exceeds a given threshold. All these time values are then used in the learning process. This way the system learns also a little for times that are adjacent to the current time. This smoothens the transitions from one time membership function to another. In addition, it is useful in early stages of teaching when there are only a few or no rules in rule base. When the next time zone is used as well as the current one, the system always has at least one rule to use when time advances to the next zone.

3.3.3 Filtering the data

When the data is fuzzified, it is filtered. System searches the most common combination of inputs and outputs from each data batch, but disregards the time input values. All other combinations than the most common one are ignored. Time input is ignored at this moment so that every situation under the measurement time can be compared to each other regardless of the time zone. The purpose of this filtering is to ignore all transient states and discover the dominant conditions for that time period. It also removes the element of chance that comes from sampling.

Here the size of the batch is important. It must not be longer than one membership function of the time input variable, because then it would be possible that system would not generate rules for some membership functions. This sets the upper limit, which is 58 minutes. The teaching interval also defines the maximum ignorance time. That is the shortest time of constant conditions that is guaranteed to be taught. This time is same as the teaching interval if the interval is an odd number of minutes. With an even number of minutes this time is one minute less than the teaching interval.

We chose to use the teaching interval of 15 minutes. This sets the maximum ignorance time to 15 minutes, so the system guarantees to teach all conditions that last at least that long. Conditions that do not last that long are most likely transients and not relevant to the system. When the most common combination of the inputs and the outputs is found, the system finds all times from the data batch that this combination has occurred. This way the input and the

output pairs are completed. These combinations are then edited into the rule base.

3.3.4 Updating the rule base

The rule base is edited in three ways. The rules are added, possibly removed, and their weighing factors are edited. The rule base is searched for input combinations that were found in the previous step. If the input combination is not found, then the corresponding rule is added to the rule base with a small initial weighing factor. If one or more input combinations are found however, their output combinations are then compared to the new rule. If a rule has the same output combination as the new rule, its weighing factor is increased and so that rule becomes more dominant. On the other hand, if the output combinations differ, the weighing factor is decreased respectively. If the weighing factor becomes zero, the rule is removed from the rule base. The amount of increase is constant, which is designed to take the weighing factor from zero to one in approximately two days, assuming constant conditions. Reasoning behind the two-day period is that the constant conditions observed during one day might be an exception, but two consecutive days of the same conditions should have an impact on the system behaviour.

3.3.5 Teaching override rules

In teaching, the overrides are managed in two ways. The overrides are the mechanism used to tell the system what it should learn. Therefore the normal rules used in the autonomous mode must always be taught. This means that even when one override for some actuator is on, the normal autonomous control is taught as if it was not. These rules then determine how the system behaves when it is in the autonomous mode.

The override rules, which define the event-based actions, are learned in a normal fashion, by adding rules and increasing and decreasing the weighing factors. However, there is one big difference in handling the override rules: they are time-independent. In practise these rules have a zero in their time input field and therefore they can be active no matter what the time is.

4 Discussion

The system was tested in a smart home described above. Users could use all methods for controlling the apartment that were available already before this study, so no new user interfaces were necessary. Testing gave us valuable feedback that was not easy to obtain through the simulations. Some of the enhancements were implemented as the work progressed, but most are to be addressed to in the next version of the fuzzy control system.

4.1 Acceptance

Both the simulations and the testing clearly showed that a fuzzy control system is very well suited for proactive control. The system is capable of learning through observing the users' actions, and it is quick to anticipate the

users' routines. After the initialization, it takes only a few days to learn enough rules that the need for user intervention decreases dramatically. If routines are changed, it takes only a couple of days to consolidate new rules for that routine and get rid of the old ones.

Since inhabitants' needs are not constant from day to day, it is inevitable that from time to time the needs will be different from what the fuzzy system determines. However, we found out that the decisions of the fuzzy system do not need to correspond exactly with what the user is planning to do. We see that most of the time we do not have very strict demands for how our lighting should be adjusted. Since we only notice when the lighting is clearly wrong, the anticipation of the process is much easier. Therefore it suffices that the proactive actions take the environment close enough to the wanted conditions.

4.2 Managing the time

The time handling proved to be a quite important part of the system operation. Three significant time related parameters were identified to have a great influence on the behaviour of the system. These parameters are the input variable 'time' and its membership functions, the data gathering interval and the teaching interval. They all are dependent from each other and a careful cross evaluation is needed while selecting them. Especially the behaviour of the learning process changes with the parameter modifications.

The dominating parameter that sets the limit to other parameters is the input variable 'time' and its membership functions. The width of the membership functions defines the upper limit for the number of membership functions, because it is not practical to have membership functions that almost completely overlap each other. The number of membership functions has a significant effect on the control accuracy. A large number of membership functions mean a more accurate control, but it forces the teaching interval to be shorter, as the teaching interval must not be any longer than the duration defined by the membership function. A shorter teaching interval also means that shorter lasting conditions are taught to the system. This is not desirable, since transients should not be learned. The teaching interval must therefore be selected so that the transient conditions are filtered out. Finally, the teaching interval sets the upper limit to the data gathering interval. The data gathering interval must clearly be much shorter than the teaching interval. Otherwise the filtering becomes pointless.

As the system is designed for a long-term control it does not have very strict requirements for the time parameters. It is practically irrelevant whether there are 50 time zones rather than 49. Due to a continuous nature of these parameters it is also difficult to pinpoint any specific optimal value for them. Therefore the initial values of these parameters were selected with an educated guess. Naturally the parameters were trimmed during the simulations until we reached the current values and concluded that they were working well. They are by no means the only suitable values, but we found that they are feasible for this prototype. In the future a method for initializing these parameters should be developed.

4.3 Future work

During the research we found out some difficulties that the system had. One such problem was the behaviour of the event-based control. It was meant to work as a means to create event-based chains of events, but we found out it did not have enough information about the context. When every user override action created an override rule, the system could not distinguish if the adjustment had anything to do with the other actuators. Based on the small number of sensors used it is very hard to recognize the context and reason why a user made that change. This added to the fact that as the event-based control is not time-dependent the user overriding actions often caused automatic control on the other actuators. That appeared to the user as an unexpected and random behaviour. We found out that observing chains of events does not fit well to fuzzy systems, and therefore it is not practical to create an event-based system using fuzzy control. Instead, a method of sequential pattern mining first introduced in [11] and moreover utilized in [12] could be more efficient in discovering these sequences.

We also came up with another problem with the override rules. The issue was that the override rules paralysed the autonomous control process. Since every user override action was treated as an event that activated some override rules, it meant that all actions caused the normal control process to stop, even when no suitable override rules were present. It would have been better if overriding one actuator would only stop autonomous control for that single actuator and leave the rest as they were. This problem should be solved in the future versions of the system.

At the beginning of the study we considered to use an indoor lighting level sensor in the apartment as an input device. The goal was to enable us to keep lighting level inside the apartment at a constant level even when the lighting level outside would change over time. However, the simulations illustrated that it was a bad idea to create a feedback like this into the fuzzy system. If a rule has an input that is affected by the outputs of the rule and the rule becomes effective, the system ends up in a situation where the outputs of the rule cause it to be instantly made ineffective. With two or more rules of this kind this can lead to oscillation between the rules. Since the outputs may swing drastically or the rules can make themselves ineffective, the effect cannot be accepted. Therefore the indoor lighting level sensor was dropped out at early stages of this study. However, a new method for including the indoor lighting level sensor should be researched.

Some difficulties arose with the learning when the user was absent. Since overrides are on indefinitely and the system learns continuously, a user can easily ignore the fact that when he leaves the room, the learning does not stop. Instead the system learns from whatever conditions the user left the room with. This might not be a problem, but in practise it means that every once in a while the system learns something that is not desired. Because of the nature of the learning, these rules are then strengthened if the user does not actively intervene with the overrides. Our objective was that the user should need to be aware of the operation of the

system as little as possible, so this is not desirable. It became clear that handling situations, where the user is absent, must be reconsidered in the future versions of this system.

The large number of time zones is good for handling the time fairly accurately, but it also creates a challenge for the learning process. Lots of time zones produce a large number of input combinations, which means that it takes a long time to learn the rules for each input combination. Especially at the early stages of learning it is quite common that no applicable rule is found. If an effective rule cannot be found, then no automatic adjustments can be made and the system is not very useful. When the system starts learning without prior knowledge, it is important to generate quickly enough rules to function properly. Therefore we seek to find a solution, which minimizes these no-rule situations.

The currently used method for editing the weighing factors must be enhanced. Since humans' routines are quite imprecise, the rules get both strengthened and weakened by a constant value at a regular basis. The current method is quite functional, but we have seen that it leads the system to situation where the rule base has a lot of medium strength rules that try to control the lighting to different directions. In practice, this can be seen as lots of loose adjustments near the center of the real control range of the actuators. From the users point of view this is not very useful. Hence, we must design a better, nonlinear algorithm for editing the weighing factors.

5 Conclusion

Already at the beginning of the research we saw a fuzzy control system to be an interesting tool in order to reach the objectives. The simulations were promising and an actual working system was implemented for the smart home laboratory. Some user tests were carried out, which highlighted many of the development areas in the system. The event-based control mode was found inadequate and the times when the user is absent problematic. However, the basic system functioned as planned and many good results were obtained. Particularly the learning method used proved to be successful.

In conclusion, we found that the basic fuzzy control system structure is a suitable method. In the lighting control purposes the developed system has been seen to perform quite well and it is plausible that the basic structure could be brought to use on other home control systems as well. Nevertheless, it is clear that much additional work will be required before such a control system will be ready to be taken into extensive use.

References

- [1] Mäntyjärvi, J., Seppänen, T. Adapting applications in handheld devices using fuzzy context information, *Interacting with Computers*, vol. 15, issue 4, p. 521-538, August 2003
- [2] Pirttikangas, S., *Routine Learning: from Reactive to Proactive Environments*, University of Oulu, Faculty of Technology, 2004
- [3] Tennenhouse D., *Proactive Computing*, *Communications of the ACM*, May 2000, vol. 43, no. 5, pp. 43-50
- [4] Byun, H. E., Cheverst, K. Supporting proactive "intelligent" behaviour: the problem of uncertainty, *Proc. Workshop on User Modelling for Ubiquitous Computing, User Modeling 2003*
- [5] Mozer, M.C. *Intelligent Systems and Their Applications*, IEEE, vol. 2, no. 2, March/April, p. 11-13, 1999
- [6] Buckley J. J., Hayashi Y. & Czogala E. On the equivalence of neural networks and fuzzy expert systems, *Proc. IJCNN-92, Baltimore*, vol. 2, p. 691-695, 1992
- [7] Kaila, L., Vainio, A.-M., Vanhala, J., *Connecting the smart home*, IASTED, *Networks and Communication Systems*, April 18-20, 2005, pp. 445-450
- [8] Wang, L. X., *A course in fuzzy systems and control*, Prentice Hall PTR, 1997
- [9] Kaila, L., *The Intelligent home of the future*. Master of Science Thesis, Tampere University of Technology, Institute of Electronics, 2001, 93 p. (in Finnish)
- [10] Vainio, A.-M., *Control system for Intelligent Home Environment*, Master of Science Thesis, Tampere University of Technology, Institute of Software Systems, 2006, 71 p. (in Finnish)
- [11] Agrawal, R., Srikant, R., *Mining Sequential Patterns*, ICDE '95: Proceedings of the Eleventh International Conference on Data Engineering, IEEE Computer Society, Washington, DC, USA, pp. 3-14, 1995
- [12] Guralnik, V., Haigh, K. Z., *Learning Models of Human Behaviour with Sequential Patterns*, Proceedings of the AAAI-02 workshop "Automation as Caregiver", pp. 24-30, 2002