AwarePen - Classification Probability and Fuzziness in a Context Aware Application

Martin Berchtold[1], Till Riedel[1], Michael Beigl[2], and Christian Decker[1]

[1] TecO, University of Karlsruhe
Vincenz-Priessnitz-Str. 3, 76131 Karlsruhe, Germany
[2] Distributed and Ubiquitous Systems
Muehlenpfordtstr. 23, 38106 Braunschweig, Germany

Abstract. Fuzzy inference has been proven a candidate technology for context recognition systems. In comparison to probability theory, its advantage is its more natural mapping of phenomena of the real world as context. This paper reports on our experience with building and using monolithic fuzzy-based systems (a TSK-FIS) to recognize real-world events and to classify these events into several categories. It will also report on some drawbacks of this approach that we have found. To overcome these drawbacks a novel concept is proposed in this paper. The concept incorporates fuzzy-based approaches with probabilistic methods, and separates the monolithic fuzzy-based system into several modules. The core advantage of the concept lays in the separation of detection complexity into distinct modules, each of them using fuzzy-based inference for context classification. Separation of detection functionality is supported by an automatic process using transition probabilities between context classifiers to optimize detection quality for the resulting detection system. This way our approach incorporates the advantages of fuzzy-based and probabilistic systems. This paper will show results of experiments of an existing system using a monolithic FIS approach, and reports on advantages when using a modular approach.

1 Introduction

Fuzzy systems have been known for years and have been successfully applied in many domains like control systems, medical systems and white ware electronics [1]. The characteristics of fuzzy control systems is their ability to classify noisy and vague input information in an application appropriate way. For example, in a process control system an engine recognized system status might be neither halted nor running after the machine receives the stop signal. Fuzzy systems handle this by fuzzifying membership functions, thus basically the engine state somehow belongs to both classes. This behaviour is similar to detecting situations of human activity: Between successive activities, there is a state which is between two activity classes, so it is impossible to fix which this activity would belong to. As shown in our previous work [2], the fuzzy logic approach can be successfully applied for activity recognition. This can be carried out by assigning fuzzy context classes to detected sensor events [2].

F.E. Sandnes et al. (Eds.): UIC 2008, LNCS 5061, pp. 647–661, 2008.

Another class of techniques successfully applied to the area of activity and context recognition are probabilistic methods. These systems often use a filter-based approach to improve the recognition rate of context recognition systems. This is achieved by assigning an a priori probability to the following classification and adjusting the recognized information accordingly.

This paper will show how we improve context and activity recognition systems by applying both methods in combination. Section 2 will shortly introduce characteristics of fuzzy set theory and discuss some difference between fuzzy set and probability theory. In section 3 we will illustrate our initial fuzzy inference (FI) based context recognition method and also discuss recognition results for this approach. In section 4 we will argument that we can further improve FI based methods by separating FI functionality into modules. Section 5 will demonstrate how to fuse our fuzzy-based approach with probabilistic methods to further increase recognition rate. A discussion section 6 shows which degrees of freedom are accessible through our approach. The paper is summarized in section 7.

2 Probability vs. Fuzziness

2.1 Fuzzy Set Theory

A general fuzzy set $(x, \mu_{\tilde{A}}(x))$ is a tuple of a value x and a membership $\mu_{\tilde{A}}(x)$. The membership function $\mu_{\tilde{A}} : U \rightarrow [0, 1]$ expresses the degree of which an element - e.g. a sensor value - belongs to the fuzzy set $\tilde{A}$ - e.g. a context class. In a crisp set A the membership $\mu_A : U \rightarrow \{0, 1\}$ would equal 1 for all members. Typical fuzzy membership functions are Gaussian-, triangular-, trapezoid-, etc. functions $\mu : U \rightarrow [0, 1]$ with a maximum of one and a minimum zero. In general, the fuzzy sets are an extension of the crisp set theory, and therefore fuzzy logic is an extension of the Boolean logic. The fuzzy equivalents of Boolean operators are functions $f : [0, 1]^2 \rightarrow [0, 1]$.

2.2 Differences of Fuzzy Set to Probability Theory

Since the invention of the fuzzy set theory by Lofti A. Zadeh [3] a controversy has evolved on whether to use the fuzzy logic or the probability theory. A probability based view of fuzzy sets and differences to probability was published by Laviolette, et al [4] and is briefly summarized in the following:

Basic Differences: In value a membership is the same as a probability, both are elements of the interval $[0, 1]$, but the semantic is unequal and can therefore not be compared. A membership does not follow the laws of probability.

The included Middle: A non-zero membership can simultaneously be held to several fuzzy sets. The probability theory defines states in a distinct way, so only the probability to one state can be expressed at one point in time, which was criticized by Zadeh [5].

Toleration of Vagueness: The key idea in the fuzzy set theory is the definition of vagueness [6]. In the probability theory there is no such concept.

Promotion of Emulation: The fuzzy logic is designed to emulate human behaviour and thinking about real-world concepts, whereas the probability theory does not fully harness this natural potential [7]. Others [8] claim that the emulation of human reasoning contradicts many empirical studies.

The Axiomatic 'Inadequacy' of Probability: Kandel and Byatt [6] claim that the laws of probability are not realistic since human beings do not conform to them. Kosko [9] claims that the conditional probability theory lacks an adequate axiomatic foundation. This is disproved by Fishburne [10].

Fuzzy Set Operations: The fuzzy operators are generalizations of the operators from the crisp set theory [3].

We claim that there are sufficient reasons to use both - as suggested by Zadeh himself [11]. Contextual coherence can in some cases be more consistently expressed in fuzzy sets, while in other cases it is more clearly defined through probability states. We will study the limitations and mutual complementariness based on a ubiquitous appliance that we built, the AwarePen.

2.3 Applications Using Fuzzy Logic and/or Statistical Methods

There is a wide use of fuzzy logic techniques to model contextual coherences on a semantic level. Guarino and Saffiotti [12] used fuzzy logic to monitor the state of ubiquitous systems by the everyday user of the system. They claim that fuzzy logic is especially suited to overcome heterogeneity in ubiquitous sensing systems. The modelling is done by hand on a semantic level, which is not applicable in our case. Another human computer interface approach using fuzzy context can be found in [13]. One of the arguments stated here for using fuzzy logic, is that it provides a continuous control signal, whereas Boolean logic does only provide discontinuous threshold based control signals. The system is also modelled manually. An application using adaptive network-based fuzzy inference systems (ANFIS) is introduced in [2]. Here the fuzzy system is used to represent the error another system may cause when recognizing contexts. Our goal is to maximize context recognition rate and minimize calculation effort at the same time.

A system that uses a probabilistic approach to represent regular activities in a smart house based on the concept of anxiety is introduced in [14]. The anxiety is formulated using probabilistic models. Another work [15] does localization based on Bayesian networks. The location is inferred of a wireless client from signal quality levels. It additionally discusses how probabilistic modelling can be applied to a diverse range of applications using sensor data.

There is also a hybrid approach using both fuzzy logic and probabilistic models in a context aware system. The proposed system [16] uses Bayesian networks for deriving consequences and fuzzy logic to draw attention to subjective decisions. In this paper we focus more on context recognition and classification.

3 The AwarePen Architecture

The general architecture of the AwarePen consists of a hardware and software part, whereas the software model is obtained offline through intelligent system identification. Overall architecture are shown in fig. 1.

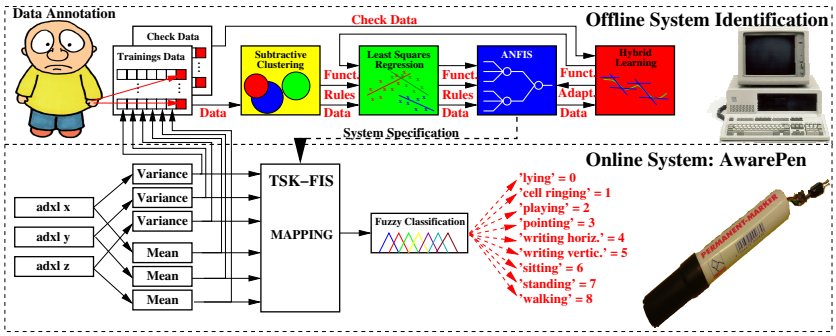

Fig. 1. General software architecture of AwarePen artefact

Our software design is based on long experience with smart artefacts in general and the AwarePen [2] especially. A central aspect of our design is a fuzzy inference system (FIS): The system is capable of automatically obtaining the necessary parameters. The mapping can be fuzzily interpreted and thus stability towards unknown input is much better in our system design than with neural networks. The fuzziness of context classes is also a central aspect in our design principles. Looking at the fuzzy side of the system design, there is initially no probabilistic aspect in our architecture of the AwarePen, but a step by step analytical and experimental approach will deliver distinct facts to apply probabilistic in system identification and design.

3.1 The Hardware

The software system is implemented using the AwarePen hardware. This hardware consists of a small electronic PCB (fig. 2 middle and right) containing a digital signal processing microcontroller unit (DSP-MCU, dsPIC by Microchip) and a collection of sensors. The DSP-MCU is dedicated to processing sensor data allowing fast, real-time sampling and FIS computation. For communication and main system operation we used the Particle pPart [17] platform, which is plugged onto the sensor PCB. In this paper we focus on 3D acceleration sensors only. Boards and battery are applied inside a marker pen (fig. 2 left).

3.2 Online Fuzzy Inference System (FIS)

One way to map queued sensor data (mean and variance) onto context classes is through a fuzzy inference system. The results of this mapping can be again

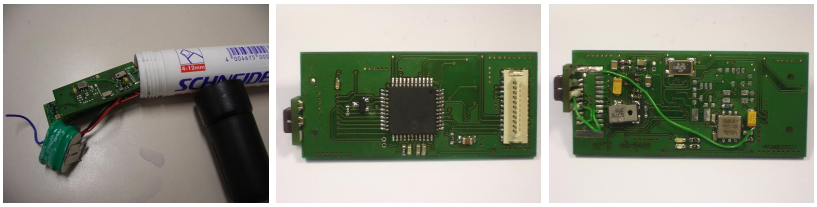

Fig. 2. Pictures of AwarePen hardware with pen (left), sensor board with dsPIC on top side (middle) and sensor board with sensors on top side (right)

interpreted as fuzzy classes. The fuzziness is the actual error caused by the mapping FIS. Here the fuzzy mapping is semantically correct for ALL context classes that have overlapping patterns and therefore are not clearly distinguishable. With the AwarePen, classes 'pointing' and 'playing' are good examples for fuzziness. Whilst 'playing' around movements can occur which are typical for 'pointing'. These circumstances are semantically correctly expressible with fuzziness, and in particular with the 'included middle' concept of fuzzy sets.

Takagi-Sugeno-Kang-FIS. Takagi, Sugeno and Kang [18][19] (TSK) fuzzy inference systems are fuzzy rule-based structures, which are especially suited for automated construction. The TSK-FIS also maps unknown data to zero, making it especially suitable for partially incomplete training sets. In TSK-FIS the consequence of the implication is not a functional membership to a fuzzy set but a constant or linear function. The consequence of the rule j depends on the input of the FIS:

$$f_j(\overrightarrow{\mathbf{v}}_t) := \sum_{i=1}^{n} a_{ij} v_i + a_{(n+1)j}$$

The linguistic equivalent of a rule is formulated accordingly:

$$\text{IF } F_{1j}(v_1) \text{ AND } F_{2j}(v_2) \text{ AND .. AND } F_{nj}(v_n) \text{ THEN } f_j(\overrightarrow{\mathbf{v}}_t)$$

The membership functions of the rule are non-linear Gaussian functions. The antecedent part of the rule j determines the weight w_j accordingly:

$$w_j(\overrightarrow{\mathbf{v}}_t) := \prod_{i=1}^{n} F_{ij}(v_i)$$

The projection from input $\overrightarrow{\mathbf{v}}_t := (v_1, v_2, .., v_n)$ onto the classifiable one-dimensional set is a weighted sum average, which is a combination of fuzzy reasoning and defuzzification. The weighted sum average is calculated according to the rules $j = 1, .., m$ as follows:

$$\mathbf{S}(\overrightarrow{\mathbf{v}}_t) := \frac{\sum_{j=1}^{m} w_j(\overrightarrow{\mathbf{v}}_t) f_j(\overrightarrow{\mathbf{v}}_t)}{\sum_{j=1}^{m} w_j(\overrightarrow{\mathbf{v}}_t)}$$

Fuzzy Classification. The outcome of the TSK-FIS mapping needs to be assigned to one of the classes the projection should result in. This assignment is done fuzzy so the result is not only a class identifier, but also a membership identifying the clearness of the classification process. Each class identifier is interpreted as a triangular shaped fuzzy number. The mean of the fuzzy number is the identifier itself with the highest membership of one. The crisp decision which identifier is the mapping outcome, is carried out based on the highest membership to one of the class identifiers. The overall output of the FIS mapping and the classification is a tuple (c, μ_c) of class identifier and membership to it.

3.3 Offline System Identification

The system identification of the FIS is not performed on the embedded device, but on a PC. The computed high-efficient FIS is then downloaded onto the AwarePen embedded device for processing in-situ FIS-based classification. It is important that the resulting FIS represents the mapping function as precise as possible, but the precision results in more rules. These conflicts with efficiency, because the more rules a FIS has the better the mapping, but the less efficient.

Subtractive Clustering. An unsupervised clustering algorithm is needed to perform system identification. Each cluster results in a fuzzy rule representing the data in the cluster and its influences on the mapping result. Since there is no knowledge about how many clusters are required, an algorithm is needed that computes the number of clusters automatically. For example, the mountain clustering [20] could be suitable, but is highly dependent on the grid structure. We instead opt for a subtractive clustering [21]. This algorithm estimates every data point as a possible cluster center, so there are no prior specifications required. Chiu [22] gives a description of parameters subtractive clustering needs for a good cluster determination. Throughout this paper we use different parameters for the subtractive clustering to achieve different numbers of clusters. The subtractive clustering is used to determine the number of rules m, the antecedent weights w_j and the shape of the initial membership functions F_{ij}. Based on initial membership functions a linear regression can provide consequent functions.

Least Squares Linear Regression. The weights a_{ij} of the consequent functions f_j are calculated through linear regression. The least squares method fits the functions f_i into the data set that needs to be adapted. A linear equation for the differentiated error between designated and actual output - which can be calculated with the rules and initial membership functions the subtractive clustering identified - is solved for the whole data set with a numeric method. Single value decomposition (SVD) is used to solve the over-determined linear equation. Using the initial membership functions F_{ij}, the rules j and the linear consequences f_j, a neural fuzzy network can be constructed. The neural fuzzy network is used to tune the parameters a_{ij}, m_{ij}, and σ_{ij}^2 in an iterative training towards a minimum error.

Adaptive-Network-based FIS. A functionally identical representation of an FIS as a neural network is an Adaptive-Network-based FIS (ANFIS) [23]. Most of the network's neurons are operators and only the membership functions F_{ij} and the linear consequences f_j are adaptable neurons. This neural fuzzy network is used to tune the adaptable parameters a_{ij} of the linear consequences, and m_{ij} and σ_{ij}^2 of the Gaussian membership functions. The tuning process is done iteratively through a hybrid learning algorithm.

Hybrid Learning. The learning algorithm is hybrid since it consists of a forward and a backward pass. In the backward pass we carry out a back-propagation of the error between designated and real output of the ANFIS to the layer of the Gaussian membership functions. The back-propagation uses a gradient descent method that searches a preferably global minimum for the error in an error hyper plane. The forward pass performs another iteration of the least squares method with the newly adapted membership functions from the backward pass. The hybrid learning stops, when a degradation of the error for a different check data set is continuously observed. The resulting ANFIS represents the qualitative non-normalized TSK-FIS **S**.

4 Divide and Conquer: Fragmentation of Complex FIS Rules

We have shown in [2] that a FIS is capable of detecting even complex context information from sensor readings. Nevertheless, handling and detection quality can be improved by dividing the processing of a FIS into several subparts. Throughout this section such a FIS is step-by-step 'divided' into subparts to maximize classification results and to keep the calculation effort to a minimum. The division is not done literally, instead of dividing the amount of classes each FIS should map onto. At first, a single FIS with a varying number of rules, represents all classes (nine in the example of figure 1). In the next step, two FIS do the same mapping, also with a variable set of rules each. In the last subsection the mapping is done via four queued FISs, whose order is determined statically according to best mapping result.

4.1 Analysis of Complex FIS Mapping Error

The usage of one FIS for mapping onto nine context classes with a reasonable amount of rules is nearly impossible. Separating the patterns from acceleration sensors is difficult when the patterns are too similar. For example, separating the patterns of 'writing horizontally' and 'pointing' is hardly possible with only mean and variance data, because they are nearly the same for both. In theory of fuzzy sets it was proofed [24] that every function can be represented to an infinite precision with a FIS with an infinite number of rules, but such theory is of less value for resource limited embedded system. The chosen clustering shows best results with a low amount of fuzzy rules, but still is not able to separate the

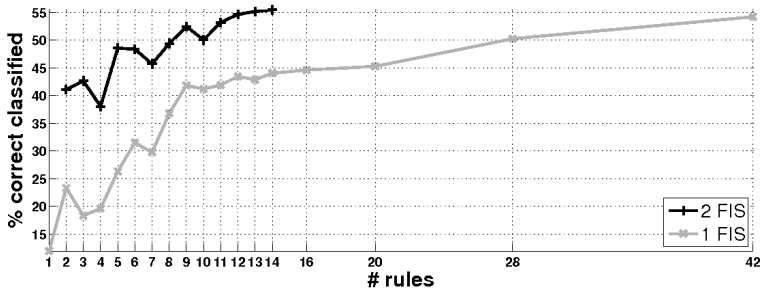

Fig. 3. Percent of correct classified data pairs for one FIS classifying all classes (grey ×) or two FIS mapping onto classes (black +)

patterns to a satisfying degree. The percentage of correct classified data pairs for a varying number of rules is shown in figure 3 as grey curve and × markers.

4.2 Dual-FIS Context Recognition

A better result according to the last section can be achieved when the big FIS representing all classes is divided into several FISs. We start here by explaining how to separate into two FISs, and later explain how to apply this method to an n-FIS system.

In our first example, each FIS maps onto four (FIS A) or five (FIS B) basic contextual classes (as 'writing', 'playing' etc., see fig. 4). To allow the transition from one FIS to another, we add an additional classifier that represents all complementary classes. To train each FIS correctly, an equal amount of data pairs for basic classes and for the complementary class is required. In our two-FIS example, the training data for the complementary class consists of data for all classes the second FIS is mapping on. Correct selection of the training data is important, as performance depends on the correct detection of basic classes, but also on correct detection of complementary class. The recognition percentage for an average rule evaluation based on a test data set is plotted in Fig. 3 (black curve with + markers) against the equal amount of rules for the FIS recognizing all context classes. For this plot the order giving the best classification result is chosen in our example (first FIS A than FIS B).

For evaluation purposes a different amount of rules for each FIS are combined and analyzed upon the average recognition rate for a test data set. The recognition rate in percentage is plotted as a surface in figure 6 on the left and as a contour plot on the right. A brief analysis of these plots can be found in discussion section.

To implement each of the two FIS above, it is important to know the number of rules that have to be processed by each FIS for correct classification. Due to the high variance of possible input data, it is difficult to give a general estimation

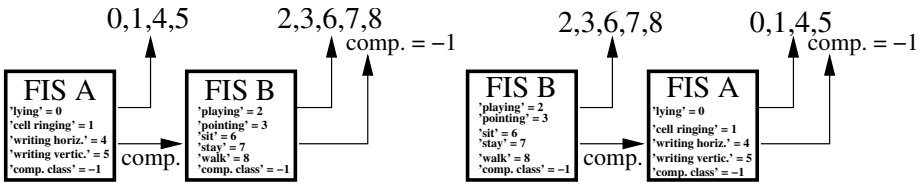

Fig. 4. Schematic of contextual classifier working with two FIS - (left) combination with 50% and (right) with 48% correct classifications

of the mean number of rules. A mean estimation for the number of rules to be processed for an input data can be given:

$$\begin{aligned} N_{AB} &= \mathbb{P}(A)(n_A + \mathbb{P}(B|A)n_B) + \mathbb{P}(B)(n_B + \mathbb{P}(A|B)n_A) \\ &= n(\mathbb{P}(A)\mathbb{P}(B|A) + \mathbb{P}(B)\mathbb{P}(A|B)), \text{ for } n_A = n_B = n \text{ and } \mathbb{P}(A) + \mathbb{P}(B) = 1 \end{aligned}$$

Probabilities are only distinguished by the amount of context classes represented by each FIS and not through the average recognition rate for the classes or the complementary one. Hereby $\mathbb{P}(X)$ with $X \in \{A, B\}$ is the probability of the FIS X to be evaluated according to the number of classes it classifies on. The probability that the second FIS $Y \in \{A, B\}$ needs to be executed - if the first X is not capable of classifying the data - is expressed through the formula $\mathbb{P}(Y|X)$. The number of rules each FIS consists of is n_X with $X \in \{A, B\}$, where in our case $n_A = n_B$. If each FIS is handling the same amount of rules, the mean rule evaluation is $\frac{1}{2}n$.

4.3 Queued Fuzzy Inference Systems

We can continue the above approach by further subdividing the classification into four FIS. For example, one FIS (FIS_{lying}) represents 'lying' and 'cell ringing', the next one (FIS_{hand}) classifies 'playing' and 'pointing', another one (FIS_{write}) maps 'writing horizontally' and 'writing vertically' and the last (FIS_{pants}) manages the classes 'sitting', 'staying' and 'walking'. FISs are queued after another. The average amount of rule evaluations of 16,15 rules is a bit high, but the recognition rate has improved up to 68,44%.

5 Stateful Interpretation and Probabilistic Optimizations

Up to this point the best order of the FISs in a queue was based on the overall classification accuracy of the different sub-classifiers from a test-data run, and therefore static at run-time. Even better results can be reached using a dynamic ordering of FIS based on current state of the stochastic process. Basic knowledge about the statistical behaviour of the classifiers and the underlying process that is classified, allows us to optimize the order of the FIS queue dynamically,

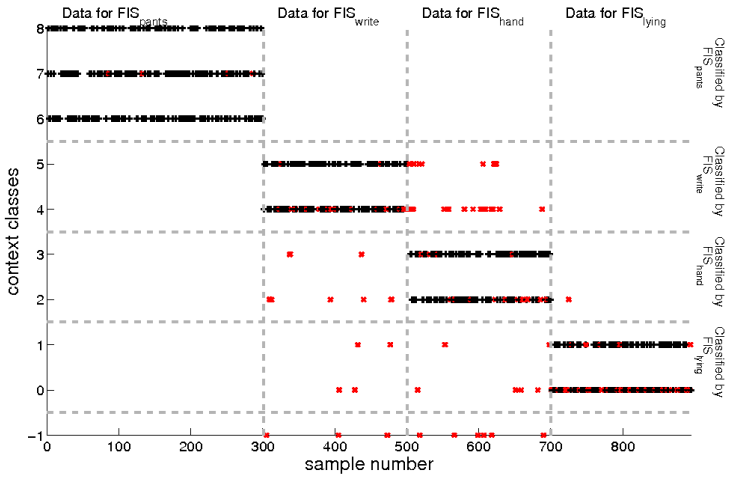

Fig. 5. 4 FISs dynamically queued - black (+) data pairs are correct classified and red (×) are false classified - 85,35% correct classified with an average of 10 rules evaluation

and avoid unnecessary execution of FIS modules. Although adding some complexity here, we can show that, we do not only save resources, but also improve recognition rates.

One discovery we made looking at typical AwarePen states in a typical usage process is that the transition of a classified state to itself during following classification is more likely than to any other state. Furthermore, we also see that some classes, like 'playing' after 'writing', have higher transition probabilities than for example 'walking' after 'writing'. In order to make use of this statistical feature in our experiments, we intuitively grouped classes in a way that similarly form a kind of 'macro state'. The transition probability from this state towards a state with classes from another FIS is much lower than towards itself.

We used this property to pre-compute an oder of a classification queue such that we always execute the classifier first that matched in the last step. This way we reduce the expected rule executions, but also improve the recognition rates, because it becomes less likely to make mistakes in previous classifiers. The results of such a scheme are shown in figure 5 and as a confusion matrix in table 1. The recognition rate is again improved and the dynamic FIS queue classifies 85% of the check data correctly.

5.1 Modelling and Optimization of Classification Probabilities

From this last dynamic experiment we can see that probabilistic modelling can help us maximize the statistical classification correctness. The underlying trained fuzzy system provides us with a quality of classification that is valuable when interpreting a single classification in an application context. When quantitatively optimizing the system the previous experiments suggest that we should leave

Table 1. 4 FISs dynamically queued - Confusion matrix

	Error	$\hat{0}$	$\hat{1}$	$\hat{2}$	$\hat{3}$	$\hat{4}$	$\hat{5}$	$\hat{6}$	$\hat{7}$	$\hat{8}$
0	0	94.0594	4.9505	0.9901	0	0	0	0	0	0
1	0	30.8511	69.1489	0	0	0	0	0	0	0
2	5.7692	3.8462	1.9231	63.4615	3.8462	13.4615	7.6923	0	0	0
3	0	1.0417	0	12.5000	80.2083	6.2500	0	0	0	0
4	2.1053	1.0526	1.0526	6.3158	0	87.3684	2.1053	0	0	0
5	0.9524	0.9524	0.9524	0.9524	2.8571	15.2381	78.0952	0	0	0
6	0	0	0	0	0	0	0	100.0000	0	0
7	0	0	0	0	0	0	0	0	100.0000	0
8	0	0	0	0	0	0	0	0	4.3478	95.6522

the fuzzy domain in favour of probabilistic modelling, as we do not look at a single classification any more but multiple samples. In this section we therefore provide a first probabilistic formalization based on our experimental findings to explain the relationship between recognition rates, class grouping and different classifier sequences.

In order to model the overall statistical performance of the proposed queued interpretation system, we model the probability for a correct classification $\mathbb{P}(\hat{X}_t = X_t)$, i.e. that the real state X equals the classified state $\hat{X}$. Both random variables are defined over the same set of $\mathcal{C}$. In the dynamic scheme we optimized the classification queues on the previous state, which need to be analyzed separately:

$$\mathbb{P}(\hat{X}_t = X_t) = \sum_{\hat{x}_{t-1}} \left(\mathbb{P}(\hat{x}_{t-1}) \sum_{x_t} \mathbb{P}(x_t|\hat{x}_{t-1}) \Big(\mathbb{P}(\hat{X}_t = x_t|x_t, \hat{x}_{t-1}) \Big) \right)$$

To reflect the different precomputed classifier sequences we introduce a total order relation $<_i$. The classifier queue $<_i$ orders $(K_a <_i \cdots <_i K_z)$ by their execution precedence. As in our experiments each classifier K_i in the queue classifies into a specific set of classes $\mathcal{C}_{K_i} \subset \mathcal{C}$, so that $\mathcal{C}_{K_i} \cap \mathcal{C}_{K_j} = \emptyset$, or into the complementary state $\perp$ (previously assigned with -1). For convenience we additionally define a helper function, that selects the index of the classifier responsible for a given class: $m(c) = (i|c \in \mathcal{C}_{K_i})$. We further use the random variable $K_{x,t}$ to denote the classifiers result at time t.

In our system the classifications inside a queue are only based on the current state X_t, while the choice of the queue depends solely on the previously classified state $\hat{X}_{t-1}$ via a static lookup. If we classified state $\hat{x}_t$ by a matching on non $\perp$ classifier result $K_{m(x_t)}$ we will in the next steps execute the classifiers in a sequence according to $<_{\hat{x}_t}$.

As discussed before a classifier in a split classifier system, can only classify correctly if the complementary state is classified for all previous classifiers. If we take this process into account we can calculate the true positive probability, the recognition rates of the classifiers and the queue system state:

$$\mathbb{P}(\hat{X}_t = x_t|x_t, \hat{x}_{t-1})) = \mathbb{P}(K_{m(x_t)} = x_t|x_t, \hat{x}_{t-1}) \prod_{i|K_i <_{\hat{x}_{t-1}} K_{m(x_t)}} \mathbb{P}(K_{i,t} = \perp|x_t, \hat{x}_{t-1})$$

Because the internal classifiers are stateless, i.e. independent of $\hat{x}_{t-1}$, we finally obtain a model for the overall recognition performance based on the recognition rates, the class grouping and the classifier sequences:

$$\mathbb{P}(\hat{X}_t = X_t) = \sum_{\hat{x}_{t-1}} (\mathbb{P}(\hat{x}_{t-1}) \sum_{x_t} (\mathbb{P}(x_t|\hat{x}_{t-1}) \mathbb{P}(K_{m(x_t)} = x_t|x_t) \prod_{i|K_i < \hat{x}_{t-1} K_{m(x_t)}} \mathbb{P}(K_{i,t} = \perp|x_t)))$$

5.2 Definition of a Cost Function

We can use this model to derive optimization strategies for designing classifiers. However, because in theory every classifier function can be represented by an infinite number of rules in our TSK-FIS [24], we have to consider execution complexity of the TSK-FIS when optimizing classification. To model this trade-off, we introduce a composite cost function that incorporates the amortized recognition rates and resource usage. The proposed optimization function

$$\text{minimize } \mathit{cost} = E[\mathit{class\_cost}] + E[\mathit{exec\_cost}]$$

is composed by the expectation of false classification costs and the expected number of rule execution costs. We use $\mathbb{P}(\hat{X}_t = x_t|x_t, \hat{x}_{t-1})$, which is associated with $\mathit{class\_cost}_1 = 0$ costs and the inverse $\mathbb{P}(\hat{X}_t \neq x_t|x_t, \hat{x}_{t-1}))$, which associate with an application specific cost $\mathit{class\_cost}_0$ to define the expectation:

$$E[\mathit{class\_cost}] = (1 - \mathbb{P}(\hat{X}_t = x_t|x_t, \hat{x}_{t-1}))\mathit{class\_cost}_0$$

Because of the strictly compositional setup of the constructed FIS the execution time is proportional to the number of rules. In order to obtain the expected execution costs, we generalize the equation of section 4.2 in a similar way as the equation for the recognition rates based on the number of rules in a FIS $|K_{\hat{S}}|$ and a cost factor *rule_cost*:

$$E[\mathit{exec\_cost}] = \mathit{rule\_cost} \sum_{\hat{x_t-1}} \sum_{x_t} \mathbb{P}(x_t|\hat{x}_{t-1}) \sum_{K_i} |K_i| \prod_{K_j r(\hat{x}_t) K_i)} \mathbb{P}(Kj, t = \perp|x_t)$$

With a given algorithm for implementing classifiers K like our fuzzy system as an internal classification algorithm, we can optimize the class mapping m and queue mapping r to achieve a better recognition rate. Both the number of classifiers and the number of rules in a classifier can additionally be adapted. Resulting from this modelling we can easily see the interconnection of classifier grouping and recognition rates via the state and the transition probabilities. Because the true positive rates of the classifiers are dependent on an initial grouping and the transition probabilities $\mathbb{P}(x_t|\hat{x}_{t-1})$ are again recursively dependent on the recognition rates, we get a highly non-linear optimization problem. We can solve this problem heuristically based on expert knowledge as we did at the beginning of this section, or we can do an automated design space exploration using empirical data.

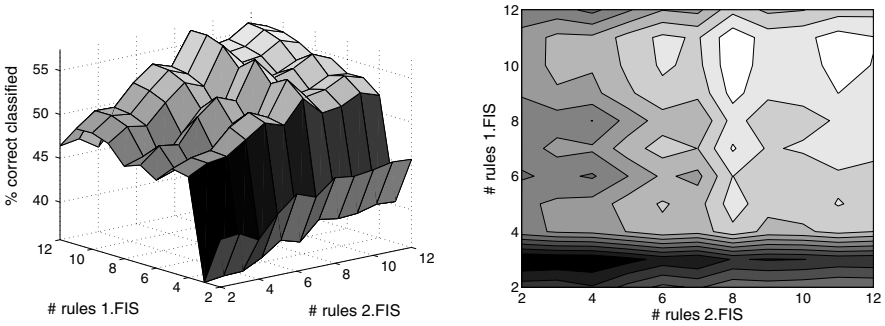

Fig. 6. Surface plot (left) and contour plot (right) for percent of right classified data pairs for two FIS - better FIS first in line

6 Discussion

As we saw in our earlier experiments, we can find interesting trade-offs between execution time and classification accuracy. Understanding those trade-offs in relation to the underlying statistical process contains a high potential for a model-guided optimization. Analyzing the cost function we can see different possibilities for minimization. In the simplest case above, that splits a FIS into two separate classifiers, we only changed execution order of a split classifier. Because we used only static queues - i.e. the classification process is independent on $\hat{x}_t$, and $\mathbb{P}(x_t)$ was equally distributed - the expectation for the cost is solely dependent on the recognition rates of the correct and complementary state, as already seen in figure 4.

The second degree of freedom is modifying the internal FISs classifiers themselves by changing the rule numbers and thus also adapting the execution cost expectation. We plotted the resulting optimization space in figure 6 comparing recognition rates depending on the number of rules in a classifier that show a certain degree of linearity in the distribution, which suggests that classifier systems can be modelled statistically. We can see that the maximal recognition rate is roughly distributed around the "natural" cluster amount returned by our initial subtractive clustering. Knowing this optimal number of rules and the according recognition rate allows us to estimate the recognition rates of FISs with a different number of rules. This would mean, that even if the optimization space already grows quickly for the static case, we could easily find good solutions. The figure also shows that we can find Pareto optimal solutions for cost (proportional to the distance to zero) and recognition rate (height/colour).

The third degree of freedom was discussed here and is represented in our model is the grouping of classes by common or separate classifiers. Naturally this affects the local recognition rates, as seen already in 3. In our dynamic approach, which can change the queue order by a pre-computed lookup of based on the previous match, the grouping additionally affects the joined probability

for having to detect previous complementary classes. This probability would be 1 if $K_{m(x_t)} = K_{m(\hat{x}_t)}$ and thus optimal, if the previously classified state has the same classifier as the current. We also see potential to heuristically optimization such dynamic cases based on the model. By grouping classes into classifiers by their transition probabilities at the beginning of the last section, we exploited the observation that $\mathbb{P}(x_t|\hat{x}_{t-1})$ is dominated by $\mathbb{P}(x_t|x_{t-1})$ for high overall recognition rates. With the results shown in 5 already hint the potential for such an approach.

7 Conclusion and Future Work

Motivated by the problems of extending our fuzzy classification system to support a larger quantity and diversity of recognizable context classes on a single, resource-constraint device, we started looking at the intrinsic problems of monolithic classifiers like our initial TSK-FIS. Because we strongly believe in the expressiveness and the intuitiveness of automatically learned fuzzy inference systems, we looked for ways to increase scalability. We showed in this paper, that a divide and conquer approach allows complex classifications while maintaining low execution overhead. Our experiments indicated, that maximum gain can be obtained by carefully designing the division of the classifiers. Different groupings, rules amounts and execution sequences for the classifiers all have effects on the recognition rates. It soon becomes clear that many of those effects are interdependent. Already a simple set-up, which compromises a classifier split into sub-classifiers acting in a chain of responsibility pattern, allows many degrees of freedom for optimizations. They are difficult to overlook by an appliance developer, who only wants to optimize the recognition rates while being bound in complexity by hardware constraints. Therefore we reviewed our experimental findings and formalized the classification process in a probabilistic model. This model captures the different effects of a divided classifier system and directly motivates different optimization strategies for maximizing the expected recognition rate while minimizing the expected costs.

In this paper we outlined how a fuzzy inference system can profit from a complex top down probabilistic analysis, which can be done off-line. In future work we plan to develop a tool chain that helps the artefact developer in designing classifiers that are automatically optimized based on the parameters of the underlying process. We motivated and showed that a combination of a bottom-up, pre-trained classification system with an probabilistic approach is advantageous from both a system performance perspective, but also from the simplicity to understand and use such a system in ubiquitous computing settings. We believe that the complexity of embedded ubiquitous applications can only be dealt with by an intuitive design abstraction like fuzzy inference systems paired with optimization technologies for an automated design space exploration based on the probabilistic modelling of the appliance and its context.

References

1. Elkan, C.: The paradoxical success of fuzzy logic. IEEE Expert: Intelligent Systems and Their Applications 9(4), 3–8 (1994)
2. Berchtold, M., Decker, C., Riedel, T., Zimmer, T., Beigl, M.: Using a context quality measure for improving smart appliances. In: IWSAWC (2007)
3. Zadeh, L.A.: Fuzzy sets. Information and Control 8, 338–353 (1965)
4. Laviolette, M., Seaman, J., Barrett, J.D., Woodall, W.: A probabilistic and statistical view of fuzzy methods. Technometrics 37, 249–261 (1995)
5. Zadeh, L.A.: Is probability theory sufficient for dealing with uncertainty in ai? a negativ view. In: Uncertainty in Artificial Intelligence. Elsevier, Amsterdam (1986)
6. Kandel, A., Byatt, W.: Fuzzy sets, fuzzy algebra and fuzzy statistics. IEEE, Los Alamitos (1978)
7. Kosko, B.: The probability monopoly. Fuzzy Systems 2 (1994)
8. Kahneman, D., Slovic, P., Tversky, A.: Judgement under uncertainty: Heuristics and biases. Cambridge University Press, Cambridge (1982)
9. Kosko, B.: Fuziness vs. probability. International Journal of General Sys. (1990)
10. Fishburn, P.: The axioms of subjective probability. Statistical Sci. (1986)
11. Zadeh, L.A.: Discussion: Probability theory and fuzzy logic are complementary rather than competitive. Technometrics 37, 271–276 (1995)
12. Guarino, D., Saffiotti, A.: Monitoring the state of a ubiquitous robotic system: A fuzzy logic approach. In: Fuzzy Systems Conference, pp. 1–6 (2007)
13. Mäntyjärvi, J., Seppänen, T.: Adapting applications in mobile terminals using fuzzy context information. In: Mobile HCI, London, UK. Springer, Heidelberg (2002)
14. West, G., Greenhill, S., Venkatesh, E.: A probabilistic approach to the anxious home for activity monitoring. In: Computer Software and Applications Conf. (2005)
15. Castro, P., Chiu, P., Kremenek, T., Muntz, R.R.: A probabilistic room location service for wireless networked environments. Ubiquitous Computing (2001)
16. Park, J., Lee, S., Yeom, K., Kim, S., Lee, S.: A context-aware system in ubiquitous environment: a research guide assistant. Cybernetics and Intelligent Sys. (2004)
17. TecO: Telecooperation Office, Univ. of Karlsruhe (2006), `http://particle.teco.edu`
18. Tagaki, T., Sugeno, M.: Fuzzy identification of systems and its application to modelling and control. IEEE Trans. Syst. Man and Cybernetics (1985)
19. Sugeno, M., Kang, G.: Structure identification of fuzzy model. Fuzzy Sets and Systems 26(1), 15–33 (1988)
20. Yager, R., Filer, D.: Generation of fuzzy rules by mountain clustering. Journal on Intelligent Fuzzy Systems 2, 209–219 (1994)
21. Chiu, S.: Method and software for extracting fuzzy classification rules by subtractive clustering. IEEE Control Systems Magazine, 461–465 (1996)
22. Chiu, S.: 9, Extracting Fuzzy Rules from Data for Function Approximation and Pattern Classification. In: Fuzzy Information Engineering: A Guided Tour of Applications, John Wiley & Sons, Chichester (1997)
23. Jang, J.S.R.: Anfis: Adaptive-network-based fuzzy inference system. IEEE Transactions on Systems, Man and Cybernetics 23, 665–685 (1993)
24. Wang, L.X.: Adaptive Fuzzy Systems and Control. Prentice-Hall, Englewood Cliffs (1998)