ANZIIS ´93, Perth Western Australia
December 1-3, 1993

SPIN-NFDS
Learning and Preset Knowledge for Surface Fusion - A Neural Fuzzy Decision System -

Jörg Bruske, Ewald von Puttkamer & Uwe R. Zimmer

University of Kaiserslautern - Computer Science Department - Research Group Prof. E. v. Puttkamer
P.O. Box 3049 - W6750 Kaiserslautern - Germany
Phone: …49 631 205 2624 - Fax: …49 631 205 2803 - Telex: 4 5627 unikl d
e-mail: uzimmer@informatik.uni-kl.de

The problem to be discussed in this paper may be characterized in short by the question: "Are these two surface fragments belonging together (i.e. belonging to the same surface)?". The presented techniques try to benefit from some predefined knowledge as well as from the possibility to refine and adapt this knowledge according to a (changing) real environment, resulting in a combination of fuzzy-decision systems and neural networks. The results are encouraging (fast convergence speed, high accuracy), and the model might be used for a wide range of applications. The general frame surrounding the work in this paper is the SPIN-project, where emphasis is on sub-symbolic abstractions, based on a 3-d scanned environment.

1. Motivation

At the actual state of research, the project SPIN (from **S**patial **P**erception to **I**dentification with **N**eural networks) is based on the data of a 3-d scanning device and designed to reach a stage of abstraction where convex clusters of surfaces are generalized, completed and classified [3].

1-1. The application

As a part of the SPIN-project introduced above, the concrete problem that is going to be discussed here may be stated as follows: "Given the representation of two surface fragments, decide whether they originally belong together (i.e. form one surface) or not?".

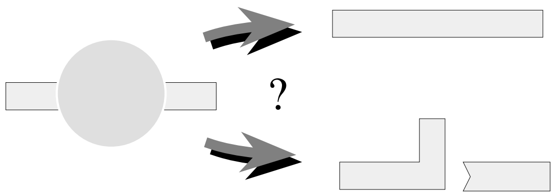

figure 1 : The surface fragmentation problem

The representation of the surfaces, the decision process is based on, includes all the curvature values at the borders (see also [3] for details) and the relative position as well as the relative orientation of both surface fragments. From this information, a group of potentially relevant features is extracted. We have chosen more features for this group than might be necessary, but we would like to discriminate important from unimportant features by learning. The tested group of features includes several curvature differences, distances between extrapolations and angle differences (see [1] for details).

1-2. Learning ***as well as*** *preset knowledge*

The initial motivation for the method presented here, is the lack of complete a-priori-knowledge for the stated problem. The correctness of the decisions depends largely on the occurring surfaces in the real environment and the correctness might alter with time. On the other hand it is quite easy to formulate a simple rule base, which is able to find "acceptable" decisions from scratch on. The combination of a preset rule-base together with a possibility to alter this rules (and something more) according to the real environment, may speed up the learning process and may result in a good performance at a very early state.

The supervisor for the learning phase is the following processing stage in the pipeline: surface-extraction (where the method discussed here is a component of this stage) ➠ surface-completion ➠ surface-classification, i.e. there is a "degree-of-recognition" signal interpreted as the error-feedback.

2. *Definitions*

Fuzzy sets and set theoretic operations

• *(Def 1):* A **fuzzy set F** in a universe of discourse U is characterized by a **membership function** $\mathbf{M_F}$: $U \to [0,1]$. A fuzzy set F in U may be represented as a set of ordered pairs of a generic element u and its grade of membership:

$$F = \{[u, M_F(u)] \mid u \in U\}. \quad (1)$$

• *(Def 2):* A **fuzzy number** in a continuous universe U is a fuzzy set F in U which is **normal** and **convex**:

$$\max \{M_F(u), u \in U\} = 1 \text{ (normal)} \quad (2)$$

$$M_F(\lambda u_1 + (1 - \lambda)u_2) \geq \min \{M_F(u_1), M_F(u_2)\}$$
$$\forall\, u_1,u_2 \in U;\ 0 \leq \lambda \leq 1 \text{ (convex)} \quad (3)$$

• *(Def 3):* The **support of a fuzzy set** F is the set of all $u \in U$ such that $M_F(u) > 0$.

• *(Def 4):* A **fuzzy singleton** is a fuzzy set whose support is a single $u \in U$ with $M_F(u) = 1$.

A, B are two fuzzy sets in U, and $A_1, \ldots, A_n$ are fuzzy sets in $U_1, \ldots, U_n$ in the following.

• *(Def 5):* The **union** $\mathbf{A \cup B}$ is characterized by the membership function $M_{A \cup B}$ with:

$$M_{A \cup B}(u) = \max \{M_A(u), M_B(u)\}, \forall\, u \in U \quad (4)$$

• *(Def 6):* The **intersection** $\mathbf{A \cap B}$ is characterized by the membership function $M_{A \cap B}$ with:

$$M_{A \cap B}(u) = \min \{M_A(u), M_B(u)\} \text{ or} \quad (5)$$

$$M_{A \cap B}(u) = M_A(u) \cdot M_B(u), \forall\, u \in U \quad (6)$$

• *(Def 7):* The **cartesian product** $A_1 \times \ldots \times A_n$ in the product space $U_1 \times \ldots \times U_n$ is characterized by the membership function $M_{A_1 \times \ldots \times A_{1n}}$ with:

$$M_{A_1 \times \ldots \times A_{1n}}(u_1, \ldots, u_n)$$
$$= \min \{M_{A_1}(u_1), \ldots, M_{A_n}(u_n)\} \text{ or} \quad (7)$$
$$= M_{A_1}(u_1) \cdot \ldots \cdot M_{A_n}(u_n),$$
$$\forall\, (u_1, \ldots, u_n) \in U_1 \times \ldots \times U_n \quad (8)$$

• *(Def 8):* An **n-ary fuzzy relation** R is a fuzzy set in $U_1 \times \ldots \times U_n$ and is expressed as:

$$R_{U_1 \times \ldots \times U_n} = \{[(u_1, \ldots, u_n), M_R(u_1, \ldots u_n)] \mid$$
$$(u_1, \ldots, u_n) \in U_1 \times \ldots \times U_n\} \quad (9)$$

Let R be a fuzzy relation in $U \times V$ and S a fuzzy relation in $V \times W$.

• *(Def 9):* A **sup-star composition** of R and S is a fuzzy relation denoted by R°S with:

$$R°S =$$
$$\{[(u, w), \sup_{v \in V} \{M_R(u, v) * M_S(v, w)\}]$$
$$\mid u \in U, w \in W\} \quad (10)$$

where * can be any operator in the class of triangular norms.

Approximate reasoning

• *(Def 10):* A **linguistic variable** is characterized by a quadruple (X, T(X), U, M(X)) in which X is the **name of the variable**; $T(X) = \{T_X^1, \ldots, T_X^k\}$ is the **term set** of X, that is the set of **linguistic values** of X with each linguistic value corresponding to a fuzzy number defined on U; $M = \{M_X^1, \ldots, M_X^k\}$ is the set of membership functions characterising these fuzzy numbers.

• *(Def 11):* A **proposition** (syntactical form: "<linguistic variable> is < linguistic value>") has the semantic: "Instantiate the linguistic variable to the fuzzy number associated with the linguistic value".

The fuzzy implication inference rule used here is a form of the **generalized modus ponens** (**gmp**):

premise 1:	X is A´
premise 2:	if X is A then Y is B
consequence:	Y is B´

where A, A´, B, B´ are fuzzy sets and X, Y are linguistic variables. Writing the premises of the gmp without the linguistic variables leads to (R is a fuzzy relation in $U \times V$)

premise 1:	$\{[u, M_{A'}(u)] \mid u \in U\}$
premise 2:	$\{[(u, v), M_R(u, v)] \mid u \in U, v \in V\}$

• *(Def 12):* The **sup-star compositional rule of inference** asserts that the fuzzy set B´ in V is given by

$$B' = A' \circ R \quad (11)$$

so the consequence may be written as

conseq.: $\{[v, \sup_{u \in U} \{M_{A'}(u) * M_R(u, v)\}] \mid v \in V\}$

If the star represents the min-operator, then this definition reduces to Zadeh´s compositional rule of inference.

3. *Some existing Neural Fuzzy Decision Systems*

Due to lack of space, this chapter lists only the negative aspects of the following models, which make them inadequate for our application. For a complete discussion of the several models see the following references or [1].

The **NARA**-system introduced by Hideyuki Takagi in his paper [5] depends largely on the selection of the neural sub-net architectures, which has to be done on the base of intuition. The **ANFIS**-model proposed by Jyh-Shing in [2] is restricted to linear consequence functions because of the employed Kalman filters. The most useful aspect of **BPFS** (by Li-Xin Wang and J. M. Mendel [6]) is, in the opinion of the authors, the proof for universal approximation, but the price paid is the huge number of terms in each variable. Finally, the **NNFDCS**-model by Chin-Teng Lin and C.S. George Lee [4] is the most similar one to our system (i.e. a lot of ideas presented here are inspired by this model). The only open problems we found in **NNFDCS** are the suspicious gradients of the min-functions.

4. *SPIN-NFDS*

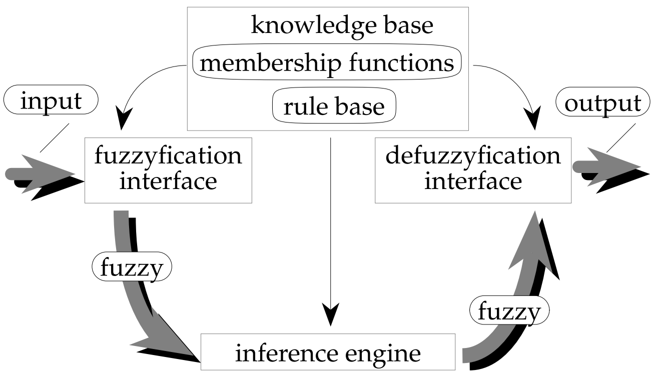

figure 2 : Fuzzy decision system (structure)

According to the general model of a **fuzzy-decision-system** (**fds**), we will first discuss all the needed components (figure 2) in order to define a mapping on a neural net structure in the following, and finally we will make some short remarks to the learning techniques.

4-1. *Rule-base*

The rule-base in SPIN-NFDS consists of linguistic rules which obey the following syntax:

<lrule> ::= if <antec> then <conseq> [else <lrule>]
<antec> ::= <propos> [and <antec>] | anything
<conseq> ::= <propos> [and <antec>]
<propos> ::= <variable name> is <term name>

where **<term name>** denotes a linguistic value of the linguistic variable **<variable name>**. The “anything” antecedent is used to define consequences as unconditioned “facts” in the rule-base.

4-2. *Membership functions*

Membership functions are used according to the definition of linguistic variables, to define the linguistic values of the associated term set. We restrict membership functions to the following basic types.

- Sigmoid function:

$$S(\mu, \sigma, x) = 1 \Big/ \left(1 + e^{-\frac{x - \mu}{\sigma}}\right) \qquad (12)$$

In the following, a sigmoid function with $\sigma > 0$ is called a **rsigmoid function** and with $\sigma < 0$ a **lsigmoid function**.

- Gaussian or radial basis function (rbf):

$$N(\mu, \sigma, x) = e^{-\frac{(x-\mu)^2}{2\sigma^2}} \qquad (13)$$

Both functions may be used for membership functions of terms appearing in an antecedent of a lrule, while only rbf-s may be used in the definition of terms appearing in a consequence.

4-3. *Fuzzyfication interface*

Non-fuzzy input data are mapped to fuzzy sets by treating them as “scaled” fuzzy singletons: If $u_0 \in U$ is a non-fuzzy input data and X is the corresponding linguistic variable with $\mathrm{dom}(X) = U$ and $T(X) = \{T_x^1, \ldots, T_x^k\}$, the input is fuzzyfied by setting

$$M^i_{X_0}(u) = \begin{cases} M^i_X(u) & ;\ \text{if } u = u_0 \\ 0 & ;\ \text{otherwise} \end{cases} \quad (1 \le i \le k) \tag{14}$$

4-4. *Inference engine*

The fuzzy implication ("then") along with the operation "and" and the combination of several fuzzy rules ("else") has to be defined.

In the first step the evaluation of the "and" in the antecedents is done by a cartesian product.

$$A \wedge B = \{[(u,v), M_A(u) \cdot M_B(v)] \mid u \in U, v \in V\} \tag{15}$$

Since all input terms are scaled fuzzy singletons, equation (15) simplifies to:

$$A \wedge B = \{[(u_0, v_0), \alpha] \mid u_0 \in U, v_0 \in V\} \tag{16}$$

$$\alpha = M_A(u_0) \cdot M_B(v_0) \tag{17}$$

with α is said to be the "degree of fulfilment" for this antecedent. The fuzzy implication is also implemented by a cartesian product, with the membership-function of the antecedents A being represented by a scalar $\alpha(u)$ (because of the scaled fuzzy singletons as the input data).

$$A \rightarrow B = \{[(u,v), \alpha(u) \cdot M_B(v)] \mid u \in U, v \in V\} \tag{18}$$

The combination of multiple fuzzy implications $A_i \rightarrow B_i$ $(1 \le i \le r)$ is interpreted as a union of the resulting fuzzy sets.

$$\{A_i \rightarrow B_i\} = \cup (A_i \rightarrow B_i);\ (1 \le i \le r) \tag{19}$$

$$= \{[(u, v), \max_{1 \le i \le r} \{M_{A_i \rightarrow B_i}(u,v)\}] \mid u \in U, v \in V\} \tag{20}$$

Since we have to keep in mind that we need differentiable functions for the learning techniques in the later neural network section, we approximate the *max*-function by *sum* as defined below.

$$\text{sum}: [0, 1]^r \rightarrow [0, 1];$$

$$\text{sum}_{1 \le i \le r}(M_{C_i}(w)) = \frac{\sum_{i=1}^{r} M_{C_i}(w)}{\sum_{i=1}^{r} \max_{w \in W}\{M_{C_i}(w)\}} \tag{21}$$

Of course the definition itself contains a *max*-function again, but this one evaluates to a linear function, as can be seen in the following example of a complete rule evaluation.

Let "**if** $X_1 = A_i$ **and** $X_2 = B_i$ **then** $Y = C_i$" be a rule base of r rules $(1 \le i \le r)$. In a first step the antecedents reduce to

$$\alpha_i = M_{A_i}(x_0) \cdot M_{B_i}(y_0);\ (1 \le i \le r) \tag{22}$$

and the output terms can be calculated to:

$$C'_i = \{[w, \alpha_i \cdot M_{C_i}(w)] \mid w \in W\} \tag{23}$$

The membership function of the union of the output terms, using the sum-equation (21), may be written as

$$M_Y(w) = \frac{\sum_{i=1}^{r} \alpha_i \cdot M_{C_i}(w)}{\text{const} \cdot \sum_{i=1}^{r} \alpha_i} \tag{24}$$

4-5. *Defuzzyfication interface*

Let Y be one of the output fuzzy sets from the inference step (25). Then the centroid method (27) is employed for the defuzzyfication to the non-fuzzy output data y.

$$Y = \{[w, M_Y(w)], w \in W\} \text{ with} \tag{25}$$

$$M_Y(w) = \text{sum}_{1 \le i \le r}(\alpha_i \cdot M_{C_i}(w)) \tag{26}$$

Note that only output terms C_i from the term set of the linguistic variable associated with Y have to be taken into account.

$$y = \frac{\int w \cdot M_Y(w)\, dw}{\int M_Y(w)\, dw} \tag{27}$$

This technique is also known as "centre of area method" (coa) or "centre of gravity".

$$y = \frac{\sum_{i=1}^{r} \alpha_i \int w \cdot M_{C_i}(w)\, dw}{\sum_{i=1}^{r} \alpha_i \int M_{C_i}(w)\, dw} \tag{28}$$

Since we restrict ourselves to gaussian functions in the output terms, (28) becomes

$$y = \frac{\sum_{i=1}^{r} \alpha_i \int w \cdot e^{-\frac{(w-\mu_i)^2}{2\sigma_i^2}}\, dw}{\sum_{i=1}^{r} \alpha_i \int e^{-\frac{(w-\mu_i)^2}{2\sigma_i^2}}\, dw} \tag{29}$$

and after some simplifications

$$y = \left(\sum_{i=1}^{r} \alpha_i \mu_i \sigma_i\right) / \left(\sum_{i=1}^{r} \alpha_i \sigma_i\right) \tag{30}$$

4-6. *Neural net architecture*

The five layered feed-forward neural network used for SPIN-NFDS is outlined in figure 3. The several layers are classified in the sense of the NFD-scheme introduced above and by the calculations being done at each layer. Note that this network architecture is *not* fully connected. A

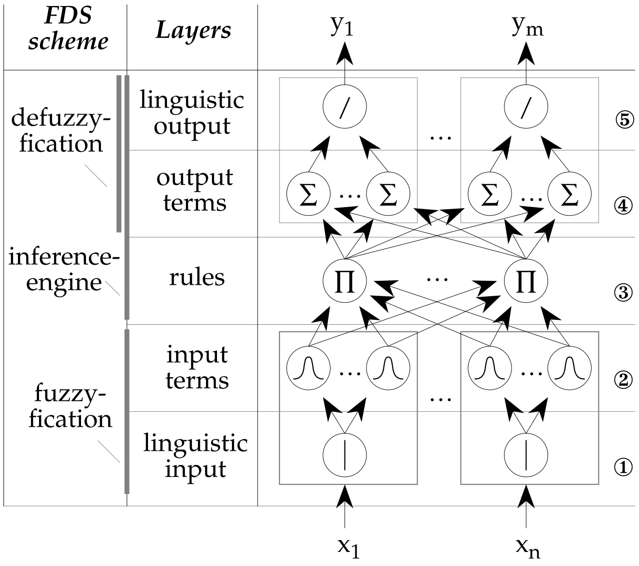

figure 3 : SPIN-NFDS network topology

description of the computations at each stage will be given in the following. The resulting set of parameters, that may be trained contains the parameters σ, μ of each membership function from the input and the output variables and a set of weights w, one for each output term in the linguistic rules.

Layer 1: Linguistic input

The operation being done here is only transmitting the input values to the terms of the corresponding (i.e. not to all) input variables at the next layer.

Layer 2: Input terms

Let n be the number of linguistic input variables with names X_i $(1 \le i \le n)$, values $(x_1, \ldots, x_n)$ and term sets $T(X_i)$. This layer computes the membership functions

$$M_{X_i}^{j_i}(x_i)\,;1 \le i \le n;\ (1 \le j_i \le |T(X_i)|) \tag{31}$$

where sigmoid (12) and gaussian functions (13) are allowed. The parameters in each node, which have to be optimized by learning (described in the next section) are

$$(\mu_i^{j_i}, \sigma_i^{j_i}) \tag{32}$$

Layer 3: Rules

From the membership functions the degree of fulfilment α_i $(1 \le i \le r)$ for each antecedent is to be calculated (with r is the number of linguistic rules in the rule base). The α_is are computed by Π-neurons[1] without any additional weight in the input terms.

Layer 4: Output terms

Let $t = |T(Y_j)|$ and n_i be the number of linguistic rules supporting the i-th term of Y_j. So α_k^i is the support of the k-th rule for the i-th term of Y_j and equation (30) may be formulated as

$$y_j = \frac{\sum_{i=1}^{t}\left(\mu_i\sigma_i\sum_{k=1}^{n_i}\alpha_k^i\right)}{\sum_{i=1}^{t}\left(\sigma_i\sum_{k=1}^{n_i}\alpha_k^i\right)} \tag{33}$$

The computational step being performed in this layer is the summation of the support for each term from the term sets of the linguistic output variables, i.e.

$$s_i = \sum_{k=1}^{n_i} w_{ki}\alpha_k^i \tag{34}$$

where weights w_{ki} are introduced as a possibility to determine the "importance" of a single rule for an output term. All these weights are initialized to 1 in the beginning and may be altered during the learning phase.

Layer 5: Linguistic output

Once the supports s_i have been calculated by the previous layer, equation (30) simplifies to

$$y_j = \frac{\sum_{i=1}^{t}\mu_i\sigma_i s_i}{\sum_{i=1}^{t}\sigma_i s_i} \tag{35}$$

Note that there is a certain redundancy concerning the parameters σ_i and w_{ki}.

4-7. Learning in SPIN-NFDS

Training is performed by standard backpropagation techniques. Two schemes are supported: The "normal" backpropagation (also called: offline, batch, epoch) and the "on-line" backpropagation (also referred to as sample, or stochastic backpropagation).

1. A Π-neuron calculates its output to $out_\Pi = \Pi x_i$ with $x_1, \ldots, x_n$ being the inputs of the neuron.

5. Experimental results

Several tests have been done, according to the following test strategies for SPIN-NFDS:

- Analyse the *training performance*, with an ad-hoc rule base for our decision task.
- Analyse the *generalisation performance*, i.e. training and testing is being done with different sets of test surface-fragments.
- Compare performances resulting from large training sets to that resulting from small but *"characteristic" training sets*.
- Compare performances resulting from changes in the predefined rule-base.
- Compare performances from SPIN-NFDS to that of a "classic" backpropagation classifier.

Some of the most interesting results are shown in the following. For the complete discussion see [1]. The training sets T_1 and T_2 consist of 300 example pairs of fragments each.

Training performance

As the reader may see from table 1, the classification rate rises really fast (remember we are regarding a backpropagation system!) even for a primitive rule base (initial classification rate is only 50%). This fast convergence is slowed down by adding noise, but nevertheless a classification rate beyond 90 % is reached with 50 training steps.

iterations:	0	1	5	10	50	175
class. rate[%] $T_1^{0\%}$:	50	50	96	96	97	99
class. rate[%] $T_1^{10\%}$:	50	50	50	63	92	98

Table 1 : Training performance with 0% noise

Generalisation performance

Interesting, but not completely surprising are the results of the tests for generalisation. We may see from table 2 that the training on unnoisy training sets shows only poor results in noisy environments, whereas training on noisy training sets shows reasonable results on noisy *and* unnoisy test sets.

Classification rates [%] after 70 iterations		trained on $T_1^{0\%}$	trained on $T_1^{5\%}$	trained on $T_1^{10\%}$
classifies	$T_2^{0\%}$	98	98	98
	$T_2^{5\%}$	83	96	97
	$T_2^{10\%}$	63	85	95

Table 2 : Generalisation performance

6. Conclusion

First, we would like to recall the benefits we have observed in the tests of SPIN-NFDS : Superior speed of convergence and generalization performance compared to conventional backpropagation-trained nets, *and* superior accuracy compared to conventional fuzzy inference system. Therefore we feel encouraged to call the concept a combination of the advantages from fuzzy inference systems and neural networks.

As one future direction, we will adapt the SPIN-NFDS on other parts of the SPIN-project, in order to get a wider testbed of applications.

References

[1] Jörg Bruske
Neural Fuzzy Decision Systems
Diploma thesis, University of Kaiserslautern ´93

[2] J. S. R. Jang
Self-Learning Fuzzy Controllers Based on Temporal Back Propagation
subm.: IEEE Trans. on Sys., Man and Cybern., ´92

[3] Herman Keuchel, Ewald von Puttkamer, Uwe R. Zimmer
SPIN - Learning and Forgetting Surface Classifications with Dynamic Neural Networks
to appear: ICANN ´93, Amsterdam

[4] C. Lin, C. S. G. Lee
Neural-Network-Based Fuzzy Logic Control and Decision System
IEEE Trans. on Comp., Vol.40, No.12, Dec.1991

[5] H. Takagi
Neural-networks designed on Approximate Reasoning Architecture and its Application
to appear: IEEE trans. Neur.Netw. Vol3, No.5, ´92

[6] Li-Xin Wang and Jerry M. Mendel
Back-Propagation Fuzzy System as Nonlinear Dynamic System Identifier
subm. to: IEEE Trans. on Neural Networks, 1991