New Learning Algorithms for the Neuro–Fuzzy Environment NEFCON–I

(Neue Lernalgorithmen für die Neuro–Fuzzy–Entwicklungsumgebung NEFCON–I)*

Detlef Nauck, Rudolf Kruse and Roland Stellmach

Department of Computer Science

Technical University of Braunschweig

Bültenweg 74 / 75, D-38106 Braunschweig, Germany

Tel.: +49.531.391.3155, Fax: +49.531.391.5936, Email: nauck@ibr.cs.tu-bs.de

*This paper is going to appear in the Proceedings of the Third German GI–Workshop "Fuzzy–Neuro–Systeme'95" to be held in Darmstadt, Germany, November 15 – 17, 1995

Kurzfassung

NEFCON–I ist eine X–Window–basierte graphische Simulationsumgebung für Neuro–Fuzzy–Regler. Die Software ist auf dem Internet frei verfügbar. Das NEFCON–Modell ist von einem generischen Fuzzy–Perceptron abgeleitet. Es ist in der Lage, sowohl Fuzzy–Mengen als auch Fuzzy–Regeln zu erlernen, indem es ein verstärkendes Lernverfahren einsetzt, das ein Fuzzy–Fehlermaß nutzt. Die Vorgängerversion von NEFCON hatte einige Einschränkungen hinsichtlich der Form der Zugehörigkeitsfunktionen in den Regelkonklusionen und verwendete einen aufwendigen Regellernalgorithmus. Die neue Version des NEFCON–Modells setzt neue Lernverfahren sowohl für die Erzeugung von Fuzzy–Mengen als auch von Fuzzy–Regeln ein und hebt die Einschränkungen für die Zugehörigkeitsfunktionen der Konklusionen auf.

Abstract

NEFCON–I is an X–Window based graphical simulation environment for neuro–fuzzy controllers, and it is freely available on the Internet. The NEFCON model is based on a generic fuzzy perceptron, and it is able to learn fuzzy sets and fuzzy rules by a reinforcement learning algorithm that uses a fuzzy error measure. The former version of NEFCON had some restrictions on the form of the membership functions of the conclusions, and an expensive rule learning procedure. The new version of the NEFCON model incorporates new learning algorithms for both the fuzzy sets, and the fuzzy rules, and it removes the restrictions on the conclusion fuzzy sets.

1 Introduction

The NEFCON model is a neuro–fuzzy system for control applications [3, 5, 6, 7]. The system is able to learn fuzzy sets and fuzzy rules on–line by reinforcement learning, i.e. it learns while trying to control a dynamic system without knowing the correct output value. The learning process is guided by a linguistic description of the desired system behavior – the fuzzy error measure. NEFCON has been implemented in the graphical simulation environment NEFCON–I which is freely available, and has already been downloaded over 500 times. There is also an application of the NEFCON approach in the area of waste water neutralization [1].

The NEFCON model is based on a generic fuzzy perceptron [4, 8, 9] that defines a 3–layer neural–network–like structure with fuzzy sets as weights. A NEFCON model is derived from it by applying certain semantical constraints, e.g. there are shared weights to ensure that identical linguistic terms are always represented by identical membership functions. A NEFCON system has an input layer representing input variables, a hidden layer of *rule units* representing fuzzy rules, and a single output unit η. For a full description of the NEFCON model please refer to one of the former publications [3, 5, 6, 7].

In the first version of NEFCON some limitations had to be introduced, to make the learning algorithms work. It was not possible to use regular, e.g. triangular, membership functions in the conclusions of the fuzzy rules encoded in NEFCON. The membership functions used had to be of the Tsukamoto type, i.e. monotonic over their support. The learning process did also not move the membership functions, it did only optimize the system by making the fuzzy sets wider or smaller.

In this enhanced NEFCON model we have realized additional learning capabilities, and removed the above mentioned restrictions. In the following sections we will describe the new neuro–fuzzy learning algorithms, and conclude with brief review of the software tool.

2 Learning Fuzzy Sets

For a NEFCON system the fuzzy error $e \in [0, 1]$ is determined from a set of linguistic *error rules* describing the size of the error for certain states. The extended fuzzy error $E = e \cdot \mathrm{sgn}(\eta_{\mathrm{opt}})$ carries also information about the direction of the necessary modifications. This information is given in form of the sign of the otherwise unknown optimal output value η_{opt}, i.e. the size of the output may be unknown, but it must be known, whether it is greater or less than zero [5, 7]. In the first NEFCON version only the extended fuzzy error $E \in [-1, 1]$ was considered, and used to adapt the membership functions. Now we also consider the change of E. Let E be the extended fuzzy error at time t and E' the extended fuzzy error at time $t + 1$. The error tendency τ is defined as

$$\tau = \begin{cases} 1 & \text{if } (|E'| \geq |E|) \wedge (E' \cdot E \geq 0) \\ 0 & \text{if } (|E'| < |E|) \wedge (E' \cdot E \geq 0) \\ -1 & \text{if } (E' \cdot E < 0) \end{cases}$$

The algorithm for modifying the membership functions of the antecedents, or the conclusion, respectively, is defined as follows:

Definition 1 *Let S be a dynamical system with n state variables $\xi_i \in X_i$, and one control variable $\eta \in Y$ (for the ease of computation we consider $Y = \{y_{\min}, \ldots, y_{\max}\}$ to be finite). For each input variable ξ_i there are triangular fuzzy sets $\mu_j^{(i)}$, $j \in \{1, \ldots, p_i\}$, and for the output variable η there are triangular fuzzy sets ν_j, $j \in \{1, \ldots, q\}$. For the fuzzy sets there are constraints Ψ given that determine whether a modification can be carried out. The centers of the triangular membership function are denoted by c. To adapt the membership functions of a NEFCON system with k rule units, the following steps are iterated until an end criterion is met (e.g. when the fuzzy error stays small for an acceptable time).*

(i) Determine E and $\operatorname{sgn}(\eta_{\text{opt}})$ from the current state.

(ii) Determine the absolute contribution ac_r and the relative contribution rc_r of each rule node r to the output value o_η (o_r is the output value (activation, degree of fulfillment) of rule node R_r):

$$ac_r := \nu_{j_r}^{-}(o_r),$$
$$rc_r := \frac{o_\eta - ac_r}{y_{\max} - y_{\min}},$$

where $\nu_{j_r}^{-}(o_r)$ is a pseudo inverse of ν_{j_r} at o_r, and it is calculated by a local defuzzification procedure:

$$\nu_{j_r}^{-}(o_r) = \frac{\sum\limits_{\substack{y \in Y \\ \nu_{j_r}(y) > 0}} y \cdot \min(o_r, \nu_{j_r}(y))}{\sum\limits_{\substack{y \in Y \\ \nu_{j_r}(y) > 0}} \min(o_r, \nu_{j_r}(y))}.$$

(iii) Determine the output value o_η, and apply it to the dynamical system S to obtain a new state.

(iv) Determine the new extended fuzzy error E' and the error tendency τ according to the new state of S.

(v) Determine the shift for the conclusion fuzzy sets ν_{j_r}:

$$\Delta c_{j_r} := \tau \cdot \operatorname{sgn}(\eta_{\text{opt}}) \cdot |E'| \cdot o_r \cdot |o_\eta - ac_r| \cdot \sigma,$$

where $\sigma > 0$ is a learning rate. Modify ν_{j_r}, if all constraints $\Psi(\nu_{j_r})$ are met.

(vi) *Determine the shift for the antecedent fuzzy sets* $\mu_{j_r}^{(i)}$ *(x_i is the current input value for ξ_i):*

$$\Delta c_{j_r}^{(i)} := \tau \cdot \text{sgn}(\eta_{\text{opt}}) \cdot (x_i - c_{j_r}^{(i)}) \cdot |E'| \cdot rc_r \cdot \sigma.$$

Modify $\mu_{j_r}^{(i)}$*, if all constraints* $\Psi(\mu_{j_r}^{(i)})$ *are met.*

This definition provides a learning procedure that optimizes the fuzzy sets by shifting them. The algorithm is a simple heuristic, and tries to modify as few parameters as possible to keep the learning procedure stable. In the previous NEFCON model only the spreads of triangular fuzzy sets were modified. In this extension, we keep the spreads, but allow to change the positions. By this the learning algorithm can cope with worse initial fuzzy partitions than the previous algorithm.

Althoug the algorithm can easily be extended to modifying the spread of the fuzzy sets also (and the implementation offers this option), we discovered that this does not seem to be necessary. Shifting the fuzzy sets instead of modifying their width does also result in the advantage that superfluous fuzzy sets will be discovered; they will become identical to other fuzzy sets.

With this learning algorithm NEFCON is now able to emulate a standard Mamdani controller using COA defuzzification.

3 Learning Fuzzy Rules

The original NEFCON model was able to learn fuzzy rules, by creating all possible rules, and later deleting erroneous and superfluous rules.

The new rule learning algorithm goes the opposite way, and creates a rule base from scratch by adding rule by rule. It does this by first classifying an input vector, i.e. finding that membership function for each variable that yields the highest membership value for the respective input value. By this a rule antecedent is formed. Then the algorithm tries to guess the output value by deriving it from the current fuzzy error. In a second phase the rule base is optimized by changing the conclusion to an adjacent membership function if necessary. To start the algorithm, an initial fuzzy partitioning must be assigned to each variable.

Definition 2 *We use the same assumptions as in Def. 1. To determine the rule units of a NEFCON system the following algorithm is carried out.*
Phase 1: Repeat the following steps m_1 *times:*

(i) *For the current input vector* $(x_1, \ldots, x_n)$ *find those fuzzy sets* $\hat{\mu}^{(1)}, \ldots, \hat{\mu}^{(n)}$ *such that* $(\forall i, j)(\hat{\mu}^{(i)}(x_i) \geq \mu_j^{(i)}(x_i))$ *holds.*

(ii) If there is no rule with the antecedent "**If** ξ_1 is $\hat{\mu}^{(1)}$ and ... and ξ_n is $\hat{\mu}^{(n)}$", *then find the fuzzy set* $\hat{\nu}$ *such that* $(\forall j)(\hat{\nu}(o) \geq \nu_j(o))$ *holds, where the heuristic output value* o *is determined by*

$$o = \begin{cases} m + |E| \cdot (y_{\max} - m) & \text{if } E \geq 0 \\ m - |E| \cdot (m - y_{\min}) & \text{if } E < 0, \end{cases}$$

with $m = \frac{y_{\max} + y_{\min}}{2}$.

(iii) Enter the rule "**If** ξ_1 is $\hat{\mu}^{(1)}$ and ... and ξ_n is $\hat{\mu}^{(n)}$, **then** η is $\hat{\nu}$" *into the NEFCON system.*

Phase 2: Repeat the following steps m_2 *times:*

(i) Propagate the current input vector through the NEFCON system and determine for each rule unit R_r *the desired contribution to the system output value:* $t_r = \nu_{j_r}^{-}(o_r) + \Delta c_{j_r}$.

(ii) For each rule unit R_r *determine the new output membership function* $\hat{\nu}_r$ *such that* $(\forall j)(\hat{\nu}_r(t_r) \geq \nu_j(t_r))$*, and change the conclusion of the rule unit* R_r *accordingly.*

Delete all rules that have not been used in more than in $p\%$ *of all propagations.*

This definition provides a rule learning algorithm that is less expensive than the one previously used for NEFCON. It is not necessary to handle all possible rules at once, what even becomes impossible, when there are a lot of variables or a lot of membership functions, respectively. After the rule base has been learned, the learning procedure for the fuzzy sets can be invoked to fine tune the membership functions. The tool does also allow to modify the fuzzy sets during rule learning, but it is not advisable to do so, because the learning algorithm usually produces worse results in this case.

4 Conclusions

The aforementioned learning algorithms have all been implemented in a graphical simulation environment for Unix–based workstations. The tool is called NEFCON–I (I is short for InterViews, the graphical X–Window development libraries that have been used), and like the former version it is freely available on the Internet at ftp.ibr.cs.tu-bs.de in the directory /pub/local/nefcon, or via the World Wide Web at http://www.cs.tu-bs.de/ibr/projects/nefcon (availability due in Oct. '95). The figure on next page shows the graphical surface of the controller part of the software. The user can define initial fuzzy sets and rules by graphical editors, and view the modifications of the learning process on–line. In the same way the user defines the fuzzy error rules that guide the learning process. These error rules can be either evaluated like normal

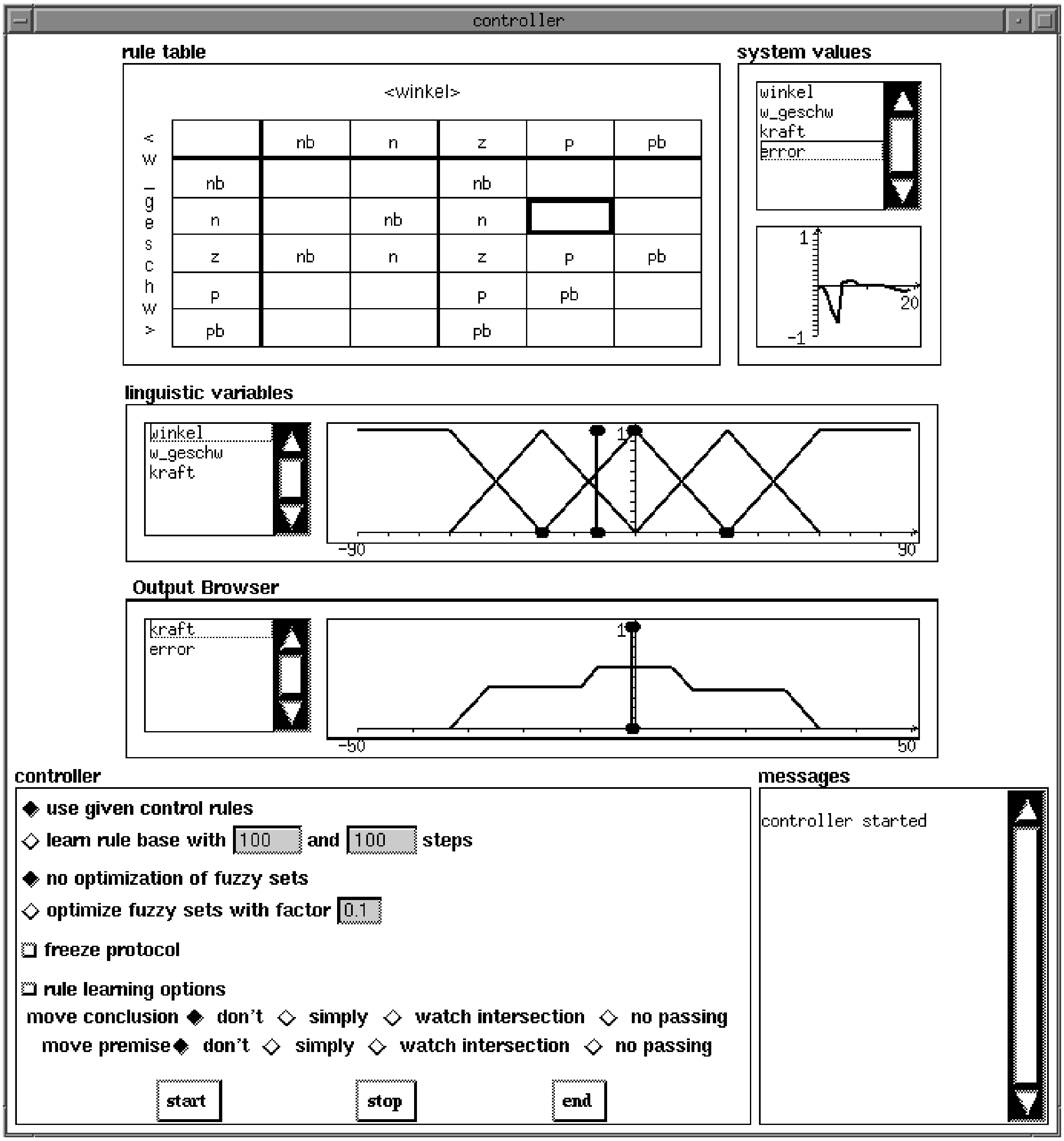

Figure 1: Graphical surface of the new version of NEFCON–I

fuzzy control rules, or by using the Goedel relation. The Goedel relation can be used when a logic–based evaluation method for the error is preferred to the usual interpolation method [2].

NEFCON–I communicates with an external process that implements a dynamical system. Currently there are only simulations of the inverted pendulum, and of a fan balancing a ball. Due to lack of space we can only provide a small glance on the simulation results here. The new version has the same capabilities than the previous one, and is able to learn rule bases faster, and can deal with unsuitable fuzzy partitions (see examples and data accompanying the software and its description [10]).

The figures 2 and 3 show the learning result of a test run. The rule bases for the controller and for the definition of the fuzzy error were given. NEFCON had to control a simulation of an inverted pendulum. Learning was stopped after 5000 cycles, and the controller is able to cope with angles in $[-40°, 40°]$. At the beginning of the learning process the controller was not able to balance the pendulum. When the pendulum was near the optimal angle, it was deliberately pushed every 20 cycles. It is interesting to see that the fuzzy partitioning of the variable *force* in fact consists of mere 3 fuzzy sets after learning, because some of the initial fuzzy sets have been shifted to the same position.

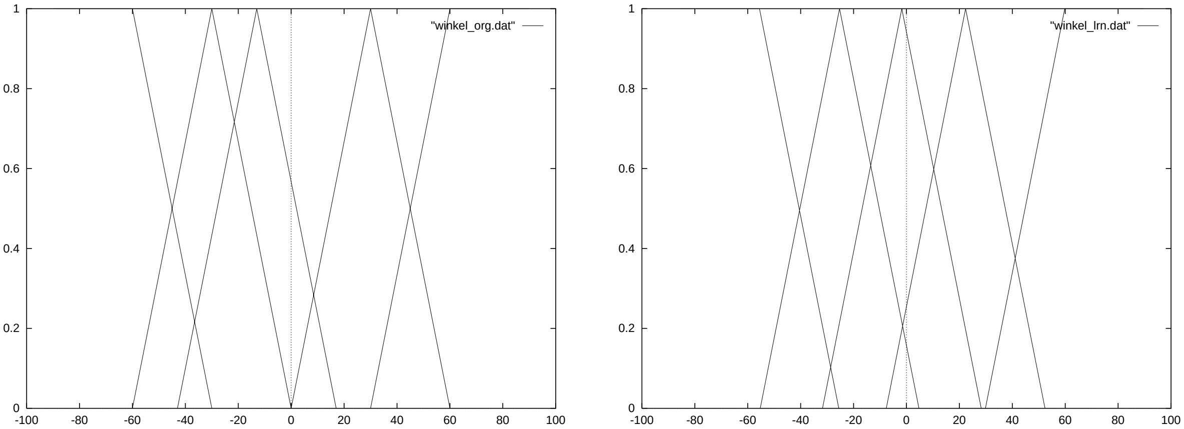

Figure 2: Fuzzy sets of the variable *angle* before (left) and after (right) learning

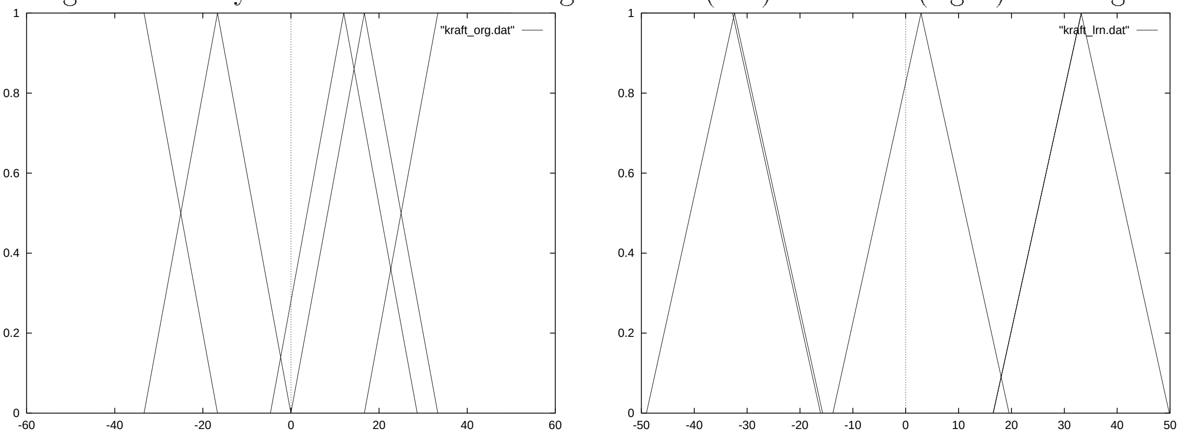

Figure 3: Fuzzy sets of the variable *force* before (left) and after (right) learning

The idea behind NEFCON is to provide a neuro–fuzzy model that is capable of learning fuzzy sets and fuzzy rules by applying simple heuristics, and modifying only few parameters. In a situation where the exact output value is unknown, it usually makes no sense to adapt gradient descent methods, because it is necessary to make assumptions about the unknown gradient that are often not justified. Choosing only a few parameters for modification by the learning algorithm ensures its stability and does not restrict the flexibility of the model. This has already be shown in the previous NEFCON model or in data analysis approaches like NEFCLASS [8].

References

[1] B. Eikens, V. Saucedo, M.N. Karim, and A.J. Morris. Waste water neutralization using a fuzzy neural network controller. In *Proc. American Control Conference 1995*, 1995. To be published.

[2] Rudolf Kruse, Jörg Gebhardt, and Frank Klawonn. *Foundations of Fuzzy Systems*. Wiley, Chichester, 1994.

[3] Detlef Nauck. Building neural fuzzy controllers with NEFCON–I. In Rudolf Kruse, Jörg Gebhardt, and Rainer Palm, editors, *Fuzzy Systems in Computer Science*, Artificial Intelligence, pages 141–151. Vieweg, Wiesbaden, 1994.

[4] Detlef Nauck. A fuzzy perceptron as a generic model for neuro–fuzzy approaches. In *Proc. Fuzzy–Systeme'94*, Munich, October 1994.

[5] Detlef Nauck, Frank Klawonn, and Rudolf Kruse. *Neuronale Netze und Fuzzy–Systeme*. Vieweg, Wiesbaden, 1994.

[6] Detlef Nauck and Rudolf Kruse. A fuzzy neural network learning fuzzy control rules and membership functions by fuzzy error backpropagation. In *Proc. IEEE Int. Conf. on Neural Networks 1993*, pages 1022–1027, San Francisco, March 1993.

[7] Detlef Nauck and Rudolf Kruse. NEFCON–I: An X–Window based simulator for neural fuzzy controllers. In *Proc. IEEE Int. Conf. Neural Networks 1994 at IEEE WCCI'94*, pages 1638–1643, Orlando, June 1994.

[8] Detlef Nauck and Rudolf Kruse. NEFCLASS – a neuro–fuzzy approach for the classification of data. In K.M. George, Janice H. Carrol, Ed Deaton, Dave Oppenheim, and Jim Hightower, editors, *Applied Computing 1995. Proc. of the 1995 ACM Symposium on Applied Computing, Nashville, Feb. 26–28*, pages 461–465. ACM Press, New York, February 1995.

[9] Detlef Nauck and Rudolf Kruse. Neuro–fuzzy systems research and applications outside of japan (in japanese). In M. Umano, I. Hayashi, and T. Furuhashi, editors, *Fuzzy–Neural Networks (in Japanese)*, Soft Computing Series. Asakura Publ., Tokyo, 1995. To be published.

[10] Roland Stellmach. NEFCON–I: Entwurf und Implementierung einer Erweiterung des Softwaretools NEFCON–I für Neuronale Fuzzy–Regler. Diplomarbeit, Technische Universität Braunschweig, 1995.