Heuro-Fuzzy Extraction of Interpretable Fuzzy Rules from Data*

Andri Riid
Dept. of Computer Control
Tallinn University of Technology
Ehitajate tee 5, Tallinn, 19086, Estonia
andri@dcc.ttu.ee

Ennu Rüstern
Dept. of Computer Control
Tallinn University of Technology
Ehitajate tee 5, Tallinn, 19086, Estonia
ennu.rystern@dcc.ttu.ee

Abstract – *The paper addresses extraction of linguistic fuzzy rules from data, paying specific attention to such properties of the resulting fuzzy model as interpretability and generalization ability. A modeling technique, combining some previously known heuristic modeling approaches, is developed. Experiments of controller identification based on the truck backer-upper application demonstrate that the proposed technique is able to capture the relevant information even if the data sets used for model extraction are insufficient and/or contain noise.*

Keywords: Fuzzy modeling, modeling for control, interpretability, generalization.

1 Introduction

Many applications of fuzzy logic use it as the underlying logic system for expert systems consisting of a collection of IF-THEN rules, to reason about data. Fuzzy logic systems are able to capture the inexact and approximate nature of human knowledge and provide a man-machine interface to make accumulated human experience available for the applications in different fields of automated decision-making, e.g. in control. Note that in this case the interface translates empirical human knowledge into exact numerical relationship between observed input-output variables. This poses a question – could fuzzy systems be used for the opposite task, i.e. for extracting interpretable linguistic information from raw identification data? Extracted fuzzy model can reveal causal relationships between system variables and this information that can be very useful in system analysis and controller design (e.g. by linguistic inversion techniques [1]). Data-driven fuzzy modeling today, however, is primarily a black-box technique; further emphasized by the fact that fuzzy system structure commonly used belongs to the class of 1st order Takagi-Sugeno systems [2], with admittedly poor interpretation values. Even if the initial model is consistent with existing human knowledge about the modeled process, it generally loses any linguistic integrity it had during the subsequent optimization.

One of the obvious reasons this situation can be contributed to is that the modeling techniques applied generally ignore the semantic aspect of fuzzy systems. The resulting fuzzy models are non-transparent to interpretation, i.e. interpretation of extracted fuzzy rules can lead to erroneous conclusions. This problem can be solved by applying transparent fuzzy modeling techniques [1] that maintain transparency of the model throughout the training process thus ensuring that combined meaning of fuzzy rules and fuzzy membership functions (MFs) of the final model is consistent with the observed numerical behavior of the model.

Transparency, however, is the measure of model internal consistency and does not give any guarantee that the model is consistent with training data (in this context, external consistency). Obviously, the primary measure of the latter is the approximation error but, in particular, difficulties arise when fuzzy model is identified from scarce (and possibly noisy) data where the algorithm has to generalize on the basis of existing samples. Note that this situation is quite common because it is usually difficult to get good coverage of the input space by training data as the number of inputs increases and for the same reason it may be difficult to obtain adequate validation data sets.

The generalization ability of the model depends on the distribution of training data and on how the modeling algorithm uses the parameters of the model. Fuzzy modeling methods (e.g. neural network inspired learning algorithms [3]) generally rely on global learning techniques driven by numerical approximation error and tend to obtain the missing rules by drawing conclusions through the extrapolation of existing data samples often resulting in fuzzy rules that are unrealistic or simply untrue for the given application, interpretation of which would lead to invalid conclusions. For example, least squares estimation [4], commonly used for consequent parameter identification, has such properties.

* 0-7803-8566-7/04/$20.00 © 2004 IEEE.

This paper investigates the properties of alternative algorithms ([5, 6]) and ends up with the unified method for antecedent and consequent MF identification from data with subsequent fuzzy model refinement technique to merge compatible membership functions and remove redundant rules. The approach is tested on the simplified truck backer-upper application [7], where our goal is to replicate the existing controller from control data using the developed method.

2 System definition

Throughout the paper we consider linguistic fuzzy systems with the following rule format

$$\text{IF } x_1 \text{ is } A_{1r} \text{ AND} \dots \text{AND } x_N \text{ is } A_{Nr} \text{ THEN } y \text{ is } B_r, \quad (1)$$

where A_{ir} and B_r denote the linguistic labels (such as "small", "hot", etc.) of input variables x_i and output variable y, respectively ($i = 1\dots N$, $r = 1\dots R$).

It has been shown [8] that with inference operators such as product implication and sum aggregation, it is sufficient to consider triangular symmetric output MFs (which ensure output transparency condition by default [1]) that results in a computationally inexpensive inference algorithm that gives numerical relationship between the system (1) variables

$$y = \sum_{r=1}^{R} \tau_r b_r s_r \Big/ \sum_{r=1}^{R} \tau_r s_r , \qquad (2)$$

where b_r and s_r are the center and support of the output MF B_r associated with the r^{th} rule and τ_r is the activation degree of the same rule, given by

$$\tau_r = \bigcap_{i=1}^{N} \mu_{ir}(x_i), \qquad (3)$$

where μ_{ir} denotes the MF of the i^{th} input variable (representing A_{ir}) associated with the r^{th} rule.

In order to satisfy input transparency conditions [1] we require triangular input MFs (given by parameters $a_i^s, b_i^s, c_i^s, s = 1,\dots,S_i$, see Fig. 1) to form a partition (4) – i.e. $a_i^s = b_i^{s-1}, c_i^s = b_i^{s+1}$, except for the MFs at the extremes of the domain where $a_i^1 = b_i^1, c_i^{S_i} = b_i^{S_i}$.

$$\forall x_i \in X_i : \sum_{s=1}^{S_i} \mu_i^s (x_i) = 1 . \qquad (4)$$

Note that with (4), $\forall r \, (\sum_{r=1}^{R} \tau_r = 1)$.

Moreover, if $\forall r$, $s_r = \xi$, where ξ is arbitrary positive constant and taking into consideration (4) the expression (2) further simplifies

$$y = \sum_{r=1}^{R} \tau_r b_r \qquad (5)$$

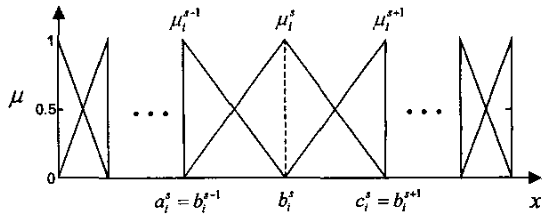

Fig. 1. Input partition style, employed in current paper.

3 Heuro-fuzzy function approximation

In function approximation the goal is usually to minimize modeling error, typicaly a root mean squared error (RMSE)

$$\varepsilon = \sqrt{\frac{1}{K} \sum_{k=1}^{K} \left(y(k) - \tilde{y}(k)\right)^2} , \qquad (6)$$

where $y(k)$ denotes the k^{th} output reading of the model and $\tilde{y}(k)$ is the respective reference output.

Heuristic approximation schemes, however, generally based on some simple principle, usually do not address the minimization of (6) directly. Consequently, the obtained RMSE value is usually higher in comparison with methods (e.g. gradient descent [3]) with built-in minimization of (6). Nevertheless, heuristic algorithms may have other attractive properties such as low computational cost, improved model generalization properties, etc.

It is observed in [9] that good learning schemes should be able to place optimal lone rules so that they cover the extremes or bumps of the approximand (and then fill in between with extra rule patches if the rule budget allows). Identification of these crucial points is not a trivial task but the method proposed by Nakoula *et al.* [5] does a rough job by placing the rules iteratively at the locations responsible for local error maxima. Model is initialized with 2^N rules, i.e. for each input variable there are two MFs placed at the extremes of its domain ($x_i^{\min}$, $x_i^{\max}$). Rule output parameters b_r for these rules are equal to the output readings $y(k)$ that correspond to the samples that provide $\max(\tau_r(k))$ for the given rules.

At each l^{th} iteration the absolute value of modeling error is computed over the training data set and the sample $[x_1(k), \dots, x_i(k), \dots, x_N(k), y(k)]$ responsible for $\max(\varepsilon(k))$ is selected (we denote it by $[x_1(l), \dots, x_i(l), \dots, x_N(l), y(l)]$). Then one MF is added to each input variable x_i, centered at

$x_i(l)$ and the MFs in its immediate neighborhood are updated to preserve (4) (see Fig. 2). Note that the number of introduced rules at each iteration is $\prod_{i=1}^{N}(S_i+1)-\prod_{i=1}^{N}S_i$, where S_i is the number of MFs per i^{th} input variable at the previous iteration and b_r must be specified for all inserted rules (unless $\tau_r = 0$, in which case we cannot add the rule).

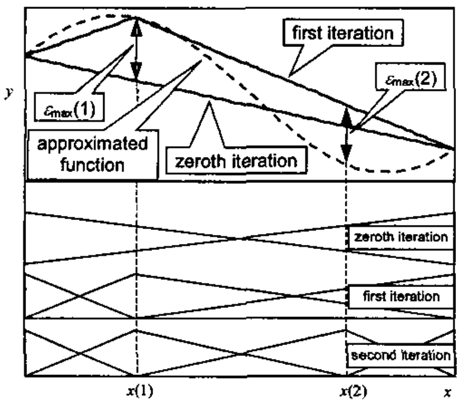

Fig. 2. First few iterations of Nakoula's algorithm.

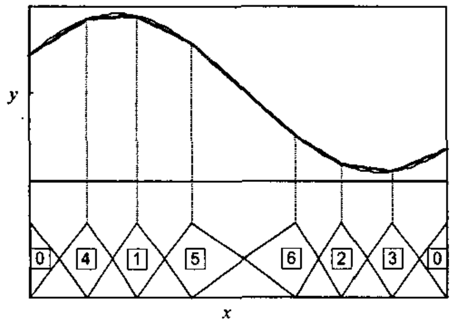

Fig. 3. Final approximation result with 8 fuzzy rules. numbers on MFs indicate the order in what they were generated.

As the final result in Fig. 3. demonstrates, what we have is a very simple yet clever method that can produce a reasonable approximation just in a few iterations. Two disadvantages of the method, however, can be pointed out. First, it may fail if training data set contains noise because, by definition, it gets hooked on outliers if they are present in identification data. This can result in a seriously biased model. The solve this problem we replace the original consequent parameter identification routine with the method of Nozaki *et al.* [6]

$$b_r = \sum_{k=1}^{K}(\tau_r(k))^{\alpha}\,y(k)\Big/\sum_{k=1}^{K}(\tau_r(k))^{\alpha}\,, \qquad (7)$$

where α is the parameter that influences model accuracy in terms of RMSE (it is reported in [6] that $\alpha = 10$ provides best results in ideal environment and that it should be smaller if data is bad). Note also that if $\alpha = 1$, (7) is the local least squares method [10] and if α is very large, (7) performs the original approach of Nakoula. The basic important characteristic of Nozaki's method is that consequent parameters for a given rule are computed as the weighted average of relevant (relevancy is expressed by rule activation degree τ_r in (7)) output samples that gives the algorithm interpolating rather than extrapolating character.

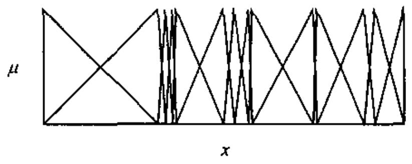

Fig. 4. Overdetermined fuzzy partition.

The second problem is that resulting input MFs of the extracted model can be very barely distinguishable from or too close to each other (see certain regions in Fig 4). Note that this primarily happens in multidimensional function approximation. On one hand, distinguishability is commonly listed as one of the requirements for interpretability, on the other hand, merging of highly similar MFs would reduce complexity of the model.

Fuzion algorithm proposed in [11] that can be used for that purpose measures the distance ($d_{is} = b_i^{s+1} - b_i^s$, $s = 1,\dots,S_i - 1$) between consecutive MF centers and compares those values with the pre-specified resolution limit $d_{\min}$. All MFs corresponding to detected sequences $s = a, \dots, b$ satisfying $d_{is} < d_{\min}$, will be replaced by a MF centered at b_{new}

$$b_{new} = \sum_{s=a}^{b+1} b_i^s \Big/ (b-a+2)\,. \qquad (8)$$

The disadvantage of FuZion is that it merges detected sequences even if the distance between the MFs at the extremes of the sequence is much greater than $d_{\min}$ which may result in considerable loss of information. The technique that we introduce here (that may be termed as FuZion II) tries to avoid it. First, we compute the distance matrix D_i (10) with elements $d_{st}^{(i)}$ for each input variable.

$$d_{st}^{(i)} = \left|b_i^s - b_i^t\right|, s = 1,\dots,S_i, t = 1,\dots,S_i \qquad (9)$$

$$D_i = \begin{bmatrix} d_{11}^{(i)} & d_{12}^{(i)} & \dots & \dots & d_{1S_i}^{(i)} \\ \dots & d_{22}^{(i)} & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ d_{S_i1}^{(i)} & \dots & \dots & \dots & d_{S_iS_i}^{(i)} \end{bmatrix} \qquad (10)$$

Note that the elements in the main diagonal of D_i are equal to zero ($|b_s^i - b_t^i| = 0$, if $s = t$) and elements below the main diagonal are the same that the respective elements above the main diagonal ($|b_s^i - b_t^i| = |b_t^i - b_s^i|$).

Each element of the matrix gives the distance between the centers of the extremal MFs of the sequence $s, \ldots, t$. The sequence can be merged only if $d_{st}^{(i)} < d_{\min}$. The merging procedure starts from the right upper element (distance between b_i^1 and $b_i^{S_i}$). Generally, if the merge for selected sequence represented by $d_{st}^{(i)}$ is accepted then the whole sequence is replaced by a MF centered at

$$b_{\text{new}} = P{\cdot}S/1{\cdot}S \tag{11}$$

where P is the row vector containing the centers b_i^s of the MFs to be merged, S is the corresponding column vector containing respective supports $(b_i^{s+1} - b_i^{s-1})$ and **1** is the unitary row vector consisting of the same number of elements as P and S. For the elements in the first row, however, $b_{\text{new}} = b_i^1$ and for the elements in the last column $b_{\text{new}} = b_i^{S_i}$ because two MFs must be kept at the extremes of the domain of the given input for numerical reasons. After the merge, D_i must be re-evaluated (it shrinks as the number of MFs is reduced) and the process restarts from the upper right element again. If there is nothing to merge in the given diagonal we simply move to the next one. In case of several merge candidates in the same diagonal the sequence with the corresponding minimum d_{st} will be chosen. The procedure concludes when we reach the main diagonal.

Fig. 5 shows how FuZion I and Fuzion II are able to handle the overdetermined partition from Fig. 4 ($d_{\min}$ is 5% of the domain). It can be seen that FuZion II is able to avoid a loss of information in comparison with FuZion I.

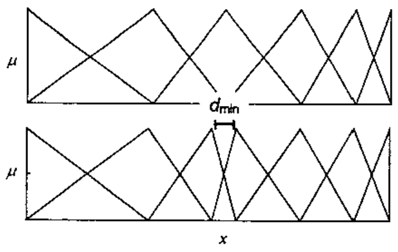

Fig. 5. Refined input partitions with FuZion I (above) and Fuzion II (below).

4 Application Example

As an application example for the combined method of [5], [6] and FuZion II we consider truck backer-upper control. The car position is determined by three state variables $x, y > 0$ and $\Phi = [-90°, 270°]$, where the latter is the angle between truck's onward direction and the x-axis (Fig. 6). The width and length of the car are 2 and 4 meters, respectively. The ultimate goal of the controller is to deliver the car to the loading dock positioned at $(x_f, y_f, \Phi_f) = (0, 0, 90°)$ from any given point in the backing field.

The control system developed for this problem in [7], consists of two units. The main component of the system is the trajectory mapping unit (TMU) that specifies the optimal car angle (Φ_r) for the given point in input space determined by car current coordinates x and y. Computation of the actual steering angle θ that would lead to the desired car orientation is carried out by a PD control loop (Fig. 7). Optimized TMU consists of 15 fuzzy rules (Fig. 8). Although the input range of the TMU is [-10, 10]×[0, 25], the limiting functions on input variables (e.g. if $x > 10$ then $x = 10$) can be used to ensure flawless performance even if the original value of the input variable is beyond the scope of the TMU. Simulations in [7] show that the controller performs on the testing set of ten car's initial positions (also used here) with 90% reliability.

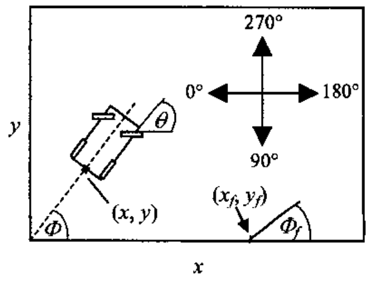

Fig. 6. Car and main variables

In this paper we attempt to replicate the TMU from data measurements by applying the method developed in this paper (HF) alongside with ANFIS [3] and variation of the heuro-fuzzy method that uses least squares estimator [2] for consequent parameter identification instead of Nozaki's method (HF/LSE). Note that the general model configuration (number of rules, type of system, MFs) is the same for all algorithms for adequate comparison (the number is self determined by Fuzion II component of HF method that is run for 5 iterations). Data distribution and noise level, however, vary, to simulate the conditions that we are likely to encounter in real life.

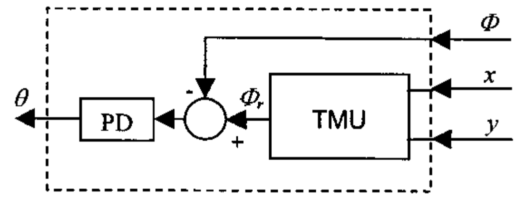

Fig. 7. Original control system from [7].

First (complete) data set is obtained from TMU evaluation with input samples that are uniformly distributed over the input space (interval between the samples for x and y is 1 m). The resulting data set contains 546 samples. Second (scarce) data set that consists of 406 samples is measured from 5 backings (Fig. 9, left). Third data set is essentially a corrupted first data set (noise level is up to 20% of the original signal, with normal distribution). Finally, fourth data set is based on the second one, but only each eight element of it is picked out bringing the number of samples down to 51, data in this set is similarly corrupted (30% noise).

Table 1 contains obtained modeling errors (RMSE) for all datasets. Note that ANFIS models are created with 200 training steps and that $\alpha = 10$ for noise-free and 1 for noisy data for HF algorithm. For noisy data, the validation error for corresponding noise-free data sets is provided in parentheses. Unsurprisingly, it can be seen that both ANFIS and HF/LSE consistently outperform our heuro-fuzzy method in terms of modeling error (except for the evaluation error for 4th dataset).

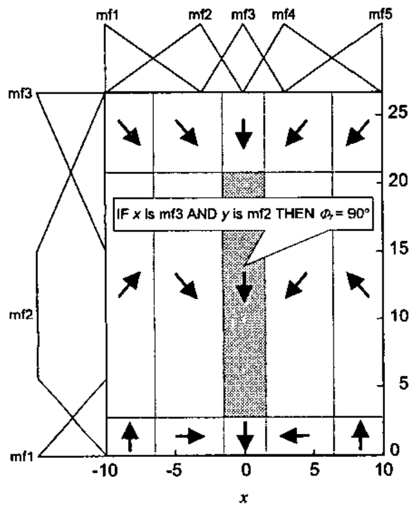

Fig. 8. TMU rulebase

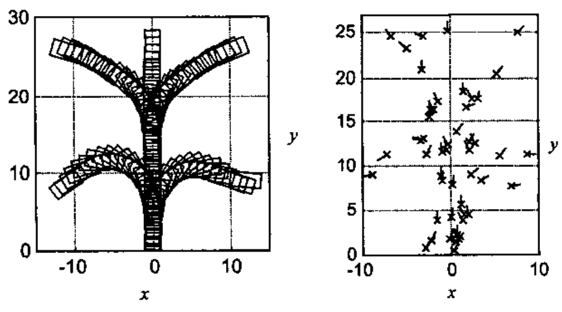

Fig. 9. 5 backings that provide data for the scarce data set (left). Corrupted scarce data set (right)

Control results are given in Table 2, where control error is evaluated by

$$\varepsilon_c = \left|x_f - x(T_f)\right| + 0.0267\left|\Phi_f - \Phi(T_f)\right|, \qquad (12)$$

where T_f is the duration of the backing. Backing is considered successful if $\varepsilon_c < 0.4$ [7].

Table 1. Modeling errors. Number of fuzzy rules is given in parentheses in the first row.

Algorithm	Data set 1 (25)	Data set 2 (34)	Data set 3 (30)	Data set 4 (23)
ANFIS	0.0445	0.8579	24.7779 (9.2115)	17.7282 (73.2060)
HF	4.5174	6.2777	26.9767 (14.3393)	25.1311 (19.0036)
HF/LSE	3.4907	1.2851	25.1864 (9.2010)	21.2912 (45.2276)

Table 2. Control results

Controller	avg(ε_c)	max(ε_c)	avg(η)	failure
Data set 1				
ANFIS	0.1096	0.4768	98.0 %	10%
HF	0.1264	0.6300	104.3 %	20%
HF/LSE	0.1280	0.5920	97.7 %	10%
Data set 2				
ANFIS	2.9606	13.3862	-	60%
HF	0.3849	1.1054	95.1 %	20%
HF/LSE	0.1948	1.0157	164.9 %	20%
Data set 3				
ANFIS	0.6595	0.9988	96.5 %	100%
HF	0.2821	0.8099	92.5 %	20%
HF/LSE	0.8332	1.0083	96.1 %	80%

With dataset 1 there are no apparent problems; the identified controllers perform as the original TMU (HF is somewhat weaker and produces one slight extra failure with $\varepsilon_c = 0.4116$). We consider the obtained trajectories in these experiments to be close to the ideal and averaged trajectory lengths $\eta_k^* = (\eta_k^{\mathrm{HF}} + \eta_k^{\mathrm{HF/LSE}} + \eta_k^{\mathrm{ANFIS}})/3$ (k = 1, ..., 10) are the basis for the secondary quality measure (13) for all experiments.

$$\mathrm{avg}(\eta) = \frac{1}{10}\sum_{k=1}^{10} \eta_k / \eta_k^* \qquad (13)$$

Problems start with the second group of controllers where the ANFIS controller clearly fails to do the job. Note that on three occassions (out of ten) the car simply gets lost. The other controllers produce much stabler results except that HF/LSE temporarily loses track on two occassions (although the final error is very small) that is reflected by a higher value of avg(η).

With the third data set both ANFIS and HF/LSE controllers consistently fail in the final stage. Closer inspection (linguistic validation) of fuzzy rules that contribute to the final position of the car sheds some light to this problem. As Fig. 11 exposes, rules of HF/LSE controller have been disturbed by the noise in training data more heavily than those of the HF controller.

Controllers identified on the basis of fourth dataset are technically nothing but failures (average errors for ANFIS, HF and HF/LSE are 7.0100, 2.6677 and 6.3043, respectively). Still, some interesting moments can be pointed out. Not only is HF the best controller in this comparison, but it is able to produce one successful result (the only one in this series) and keeps the car under reasonable control (Both ANFIS and HF/LSE mislead its car in one experiment). The controller cannot be remarkably better than the corresponding data set (see Figs. 9, left and 10 for comparison). Therefore, it can be said, the proposed modeling scheme can be very useful when we deal with noise and underdetermined data sets, which come up frequently in many real life applications.

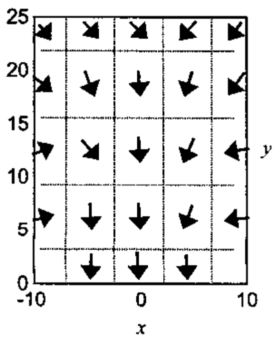

Fig. 10. Rulebase of the HF controller based on 4th dataset

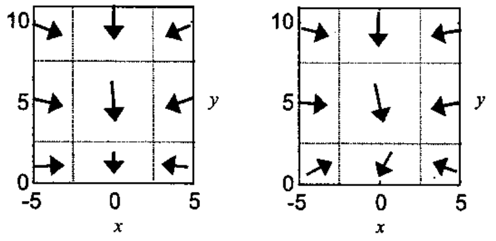

Fig. 11. Comparison of rules responsible for car final orientation. HF (left) and HF/LSE (right).

5 Conclusions

In this paper we have developed a method for fuzzy approximation, based on combining different heuro-fuzzy approaches. In the light of the results of current paper, it can be stated that heuro-fuzzy modeling can be viable alternative to immensely popular neuro-fuzzy methods, in particular when extraction of interpretable linguistic fuzzy models from data is required in real conditions because of its robustness, working speed, conservative generalization properties and despite somewhat less outstanding results in terms of direct approximation error.

References

[1] A. Riid, *Transparent Fuzzy Systems: Modeling and Control*, Ph.D. dissertation, Dept. of Comp. Control, Tallinn Technical Univ., Tallinn, 2002.

[2] T. Takagi, and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," *IEEE Trans. Syst., Man, Cybern.*, Vol. SMC-15, No. 1, pp. 116-132, 1985.

[3] J.-S. R. Jang, C.-T. Sun, and E. Mizutani, *Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence*, Prentice Hall, Upper Saddle River, 1996.

[4] R. Babuska, *Fuzzy Modeling for Control.* Kluwer Academic Publishers, Boston, 1998.

[5] Y. Nakoula, S. Galichet, and L. Foulloy, "Simultaneous Learning of Rules and Linguistic Terms", Proc. 5th IEEE Int. Conf. on Fuzzy Systems, New Orleans, pp. 1743-1749, 1996.

[6] K. Nozaki, H. Ishibuchi, and H. Tanaka, "A Simple but Powerful Heuristic Method for Generating Fuzzy Rules From Numerical Data", *Fuzzy Sets and Syst.*, Vol. 86, No. 3, pp. 251-270, 1997.

[7] A. Riid, and E. Rüstern, "Fuzzy logic in control: Truck Backer-Upper problem revisited," Proc. IEEE 10th Int. Conf. Fuzzy Systems, Melbourne, Vol. 1, pp. 513-516, 2001.

[8] A. Riid, and E. Rüstern, "On the Interpretability and Representation of Linguistic Fuzzy Systems," Proc. 3rd IASTED Int. Conf. Artificial Intelligence and Applications, Benalmadena, pp. 88-93, 2003.

[9] B. Kosko, "Optimal Fuzzy Rules Cover Extrema", *Int. J. of Intelligent Systems*, Vol. 10, No. 2, pp. 249-255, 1995.

[10] J. Abonyi, *Fuzzy Model Identification for Control*, Birkhauser, Boston, 2003

[11] J. Espinosa, and J. Vandewalle, "Constructing Fuzzy Models with Linguistic Integrity - AFRELI Algorithm", *IEEE Transactions on Fuzzy Systems*, Vol. 8, No. 5, pp. 591-599, 2000.