Bulletin of the *Transilvania* University of Braşov
Series I: Engineering Sciences • Vol. 4 (53) No. 1 - 2011

A PRACTICAL REVIEW OF A DESIGN METHOD FOR FUZZY CONTROLLERS BASED ON SELF-LEARNING ALGORITHM

C. BOLDIŞOR[1] V. COMNAC[1] S. COMAN[1]

Abstract: *A method for building the rule-base of a fuzzy controller, using the iterative learning and adaptive neural fuzzy training is tested in practical conditions. This method aims to engage intelligent features to controller design procedure, by implying concepts and techniques from artificial intelligence as learning or adapting. An iterative self-learning algorithm is used to gather useful and trustful control data for the process. These are subsequently used as training data for the ANFIS structure. The method is verified by constructing the rule-base of a fuzzy controller for a DC drive. System's performances and method's viability are analyzed.*

Key words: *fuzzy control, iterative learning, ANFIS training.*

1. Introduction

Design of fuzzy logic controllers (FLC) has been intensively treated from both theoretical and practical viewpoints, leading to numerous successful strategies, from the simplest empirical one to complex algorithms [1], [5], [8-10]. New trends engage hybrid fuzzy neural strategies that lead to intelligent control systems, aiming to somehow imitate the knowledge of an expert operating the process for which a control system is developed. These modern approaches simulate human specific reasoning and computations methods, or, in other words, they embed intelligent concepts as learning or adapting.

Intelligent feature is often extended to the design stage, meaning that simulated human actions are used to build the fuzzy rule-base. Implying intelligent techniques has an important advantage: little or even no knowledge about the process is needed. No expert, plant's operator, process model or identification stage would be necessary (often mentioned as the data sources used in fuzzy controller design). Knowledge is automatically gathered and numerically represented. The main guidelines to follow in intelligent fuzzy controller design would be: (*i*) the control system should satisfy the desired performance and (*ii*) the required knowledge about the process should be kept as little as possible [5], [7].

When it comes to the design of a fuzzy control system, the extended self-learning algorithm refers to the ability of extracting control rules from an automatic process of data recording and processing [3], [7]. Two main stages in this procedure can be defined. The first stage aims to gather correct and relevant data connecting the command actions and the corresponding

[1] Centre "Process Control Systems", *Transilvania* University of Braşov.

process' outputs. One method of obtaining those data is using the iterative algorithm proposed in [2], and presented in [6]. The second stage consists in forming the control rules themselves and it is strongly correlated with the fuzzy inference mechanism involved.

A clear example of using the self-learning concept in fuzzy rule-base construction is the one thoroughly presented in [7]. That method is used to design a fuzzy rule-base for a 2-input-2-output control system, for a table-based fuzzy inference (the table-based inference is detailed in [2]). Fuzzy rules are obtained by a *statistical* processing of the data recorded in the first stage, applied mainly on the values of command action. That solution is intuitive and proven efficient in the simulations presented in [7].

However, the method in [7] finds only the range for the input variables on the fuzzy rule-base and the constant values for the output variable in the rules' table. The number of fuzzy sets for each input variable, their membership functions, parameters and position relatively to that range are fixed and defined by the designer. Hence, the method lacks on generalisation and strongly relies on designer's experience.

Therefore, it is worth testing another solution for the second stage on the design method, in order to reduce the needs of an expert, and so to have a more independent design method. The solution proposed is based on the adaptive neural fuzzy inference system (ANFIS), as introduced by [4]. ANFIS training algorithm can be efficiently used to build fuzzy rules from correct input-output numerical data pairs. The main motivations for such an investigation are: *i*) the ANFIS is a well-known and successful solution [1], and it is widely used in adaptive fuzzy control; *ii*) it can be used directly on the data recorded in the learning stage, and so it can be further considered for a real-time implementation; *iii*) it stands as a classical algorithm, with trustful implementation as the one included in Matlab.

2. Iterative Self-Learning Algorithm

The block diagram of the self-learning system is presented in Figure 1 [6-7]. Desired control performances are defined through a reference model that can be a first-order or second-order element, possibly with an additional delay.

The reference output y_{ref} is compared with the plant's output, $y(t)$, at every iteration of the algorithm. The purpose is to find the command action that determines the plant's output to be very close to the reference output.

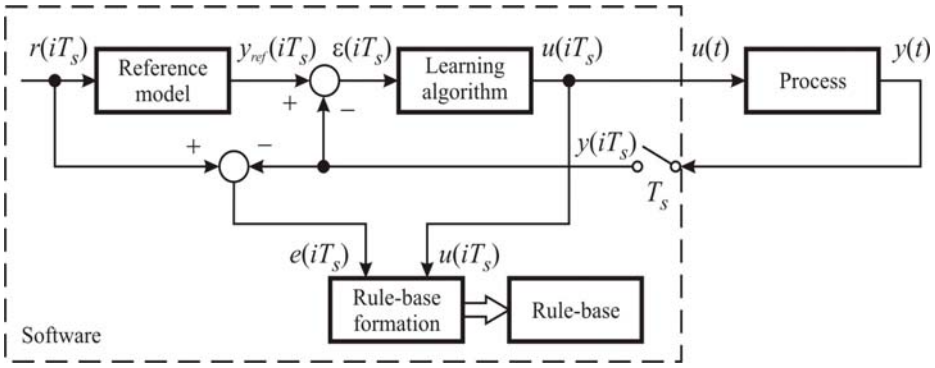

Fig. 1. *The block diagram of the iterative self-learning system*

Learning progress is evaluated for every iteration (trial) of this algorithm using the learning error, which for the *k*-th iteration is defined as:

$$\varepsilon_k(iT_s) = y_{ref}(iT_s) - y_k(iT_s), \quad (1)$$

where $i = \overline{0,I}$ is the sample index of all signals recorded with the sampling time T_s. Notice that the variables that changes from one iteration to the next are marked with the iteration number *k*.

The learning algorithm block in the diagram, also called *PID-type update law*, adjusts the command action, based on the data recorded (*learned*) at the previous trial

$$u_{k+1}(iT_s) = u_k(iT_s) + g_k\varepsilon_k(iT_s), \quad (2)$$

with g_k as a learning gain for current iteration. In the most simple case, learning gain is constant for every iteration, $g_k = g$. However, a varying gain is possible for faster algorithm convergence. Notice that the command update refers to all $I + 1$ values of command action which are going to be applied to the process in the next $k + 1$ trial, regardless the process output. This stage does not focus on control, but on data recording.

The control action is adjusted with every iteration, so that the learning error asymptotically tends to zero, or a pre-specified small value ε_{max}. The initial command output is conveniently chosen to avoid critical situations: $u_0(iTs) = 0$.

To evaluate the learning, a criterion is defined as:

$$\left\| \varepsilon_k(iT_s) \right\| = \sum_{i=0}^{I} \varepsilon_k(iT_s), \quad (3)$$

and the iterations will stop when:

$$\left\| \varepsilon_k(iT_s) \right\| \le \varepsilon_{max}. \quad (4)$$

However, this criterion is calculated with all the error values recorded ($I + 1$ values), or it is defined over the entire recording time period. Hence, it is necessary to choose a reasonable value for ε_{max}, strongly correlated with the recording interval and the sampling period.

We consider that the system learned enough when the learning error at every sampling moment iT_s is less that 0.5% from the reference output. Hence, for all recording period, condition in (4) becomes:

$$\left\| \varepsilon_k(iT_s) \right\| \le 0.005 \cdot I. \quad (5)$$

The convergence of the iterative self-learning algorithm was intensively treated and proven in [7]. Thus, it is reasonable to consider that using the algorithm for a process with no previously known model, allows gathering relevant and useful data, with no risk of instability. Further, this data can be safely used to design the control strategy, either this is a classical PID or a fuzzy controller.

3. The Combined Self-Learning ANFIS Design Method

The approach presented here is to use the adaptive neural fuzzy inference system (ANFIS) on the data recorded through self-learning. The basic idea is that, if learned data are correct for the control viewpoint, then an ANFIS could be trained with those data to build a reliable fuzzy controller.

The ANFIS training requires a sufficient and correct set of input-output pairs:

$$\{e_i; ce_i\} \sim \{u_i\}, \; i = \overline{0,I}. \quad (7)$$

Designer's only concern is only to find these correct training data. For that, the previously presented iterative algorithm is a good solution.

Suppose that, at the *K*-th learning iteration

of the iterative learning algorithm, the correct control action:

$$u_K(iT_s),\ i=\overline{0,I}, \tag{8}$$

is learned, so that the desired output specified by the reference model is achieved, $y_K(iT_s)\cong y_{ref}(iT_s)$. At the same time, the control error defined as:

$$e_K(iT_s)=r(iT_s)-y_K(iT_s), \tag{9}$$

is recorded. The change-in-error values (error derivative) are calculated from the recorded error values:

$$ce_K(iT_s)=\frac{e_K(iT_s)-e_K(iT_s-T_s)}{T_s}, \tag{10}$$

The three data vectors, from (8), (9) and (10), are organized into data pairs:

$$\{e_i;ce_i\}\sim\{u_i\}, \tag{11a}$$

where ~ means "corresponds to". The iteration number and the sampling time are no longer needed.

The current $I+1$ groups are obtained from a positive step reference, or positive command action. If the process' output is symmetrical around zero when command action sign is reversed:

$$u(iT_s)\to y(iT_s)\Rightarrow -u(iT_s)\to -y(iT_s),$$

which is most likely to be true, then another I data pairs will be obtained:

$$\{-e_i;-ce_i\}\sim\{-u_i\}. \tag{11b}$$

Hence, at the end of learning algorithm, there are $2I+1$ pairs available for further processing.

The correspondence described in (11a) and (11b) is further exploited using ANFIS training algorithm, with the $2I+1$ pairs as input-output training samples, resulting in a Sugeno-Takagi fuzzy rule-base.

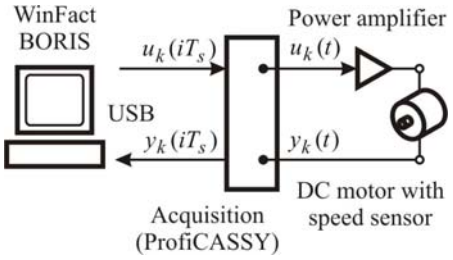

Fig. 2. *The practical experiment*

4. Constructing a Rule-Base for a DC Drive Fuzzy Controller

The practical application subjected to the presented methodology is the fuzzy controller design for a DC drive. The main reason is that DC drive control applications are wide spread, and so it can be considered a benchmark application.

The DC drive model is not relevant, since not knowing the model is one of the reasons for using intelligent design strategies. Hence, the experiment does not include identification or parameters estimation procedures.

The learning scheme in Figure 1 was implemented using a software application (WinFact/BORIS) that enables both data acquisition and real-time processing (see the diagram in Figure 2). Data acquisition is done by using the Profi-CASSY acquisition module, connected to a computer on USB port. The drive produces 3000 rpm for 10 V voltages applied on the power amplifier. The speed sensor generates 1 V signal for 1000 rpm speed. An additional scaling factor of 10/3 is used to adjust sensor's voltage to the [–10; +10] V reference range, so the maximum speed will correspond to the maximum command signal.

For the learning algorithm, the following settings were made:

i) reference is: $r(t)=5\cdot 1_+(t)$;

ii) first order reference model, with no dead-time, having $K_{ref} = 1$ and $T_{ref} = 1$;

iii) initial command is null: $u_0(iT_s) = 0$;

iv) learning gain is constant $g_k = g = 1$ (this value is intuitively chosen and was experimentally verified);

v) the sampling period is $T_s = 0.1\,\text{s}$;

vi) the recording interval is $T = 10\,\text{s}$;

vii) the learning error value to stop iterative learning is: $\varepsilon_{max} = 0.5$.

With these settings, the iterative learning was performed, by repeatedly powering the DC motor. As expected, the learning evaluation criterion exponentially tends to zero (see Figure 3). It reaches the stop condition (5) at the 10th iteration:

$$\left\| \varepsilon_k(iT_s) \right\|_{k=K=10} = 0.483 .$$

The recorded values for error, change-in-error and command action, when learning is complete are used as training data for ANFIS. A Matlab program was written and run, to process data and to implement the ANFIS. The initial settings for the ANFIS structure and training are:

i) 2 input variables - error and change-in-error;

ii) 1 output variable - the command action;

iii) triangular MFs, 7 for error and 3 for change-in-error;

iv) complete rule-base;

v) constant MFs for output variable;

vi) "and" method is prod;

vii) implication method is prod;

viii) 200 epochs for ANFIS training.

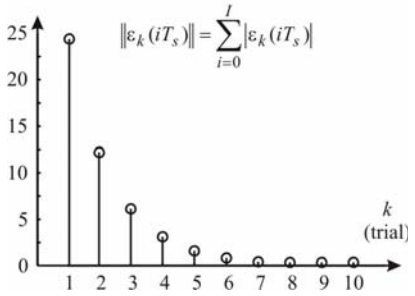

Fig. 3. *The learning error criterion value*

After running the Matlab code, the obtained rule-base (Table 1) was verified by using again WinFact BORIS software, which also provides a powerful tool to run real-time tests and analyze fuzzy control systems. The results are depicted in Figure 4. The DC speed closely follows the output of the chosen reference model. A small steady-state error is recorded and small oscillations around the desired output, which

Fuzzy controllers' rule-bases for simulation and practical test Table 1

Command		**Change-in-error**		
		ce_mf1 (N) (–100; –49.96; 1.452)	ce_mf2 (Z) (–50.01; 0.0; 50.01)	ce_mf3 (P) (–1.121; 49.96; 100)
error	e_mf1 (NB) (–6.644; –5.037; –3.467)	0.2004	–4.0564	–46.8566
	e_mf2 (NM) (–4.758; –3.183; –1.331)	91	–5.164	–21.3495
	e_mf3 (NS) (–3.331; –0.644; 0.284)	–52.5125	–4.0868	–6.9086
	e_mf4 (ZE) (–1.620; –0.620; 0.574)	12.6835	–4.7505	–58.4937
	e_mf5 (PS) (–0.682; 0.298; 3.347)	13.994	5.0468	45.0779
	e_mf6 (PM) (1.084; 3.028; 4.771)	–39.7715	1.419	–7.0527
	e_mf7 (PB) (3.428; 5.016; 6.644)	49.9223	3.6714	–0.2002
All fuzzy sets for input variables have triangular membership functions				

are caused by noise in the recording circuit or inaccurate data acquisition.

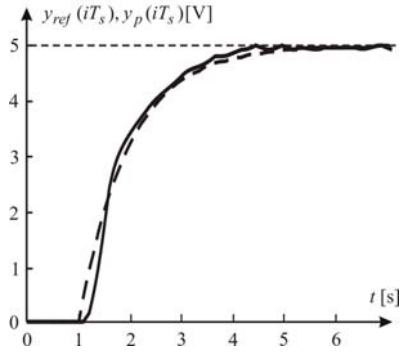

Fig. 4. *The step response of the designed fuzzy control system*

5. Conclusions

A method for building the rule-base of a fuzzy logic controller (FLC), using a self-learning algorithm and ANFIS training was tested in practical conditions. The method is aiming to engage intelligent features to the fuzzy controller design procedure, by implying techniques that somehow simulates human learning. The self-learning algorithm is used to gather useful data, which are subsequently used as training data for the ANFIS structure.

The presented method is verified by constructing the rule-base of a fuzzy controller for a DC drive application, a widely and intensively treated type of control application with well known results.

A custom Matlab code was used to process recorded data and the obtained fuzzy controller was tested in real-time conditions by using WinFact BORIS environment. The general fuzzy design guidelines were followed and fuzzy control advantages were exploited. The system has good control performances, and the controller is built without including the model of the DC drive in the design procedure, which is the main objective of an intelligent design procedure.

References

1. Abraham, A.: *Neuro-Fuzzy Systems: State-of-the-Art Modelling Techniques in Connectionist Models of Neurons, Learning Processes, and Artificial Intelligence*. In: Proc. of the 6th Int. Conf. on Artificial and Natural Neural Networks, Springer-Verlag, part 1, Granada, Spain, 2001, p. 269-276.
2. Arimoto, S., Kawamura, S., et al.: *Bettering Operation of Robots by Learning*. In: J. of Robotic Systems **1** (1984), p. 123-140.
3. Duka, A.-V., Abrudean, M., et al.: *Positioning System Based on Electromagnetic Levitation using Fuzzy Learning Control.* In: CEAI Journal **10** (2008), p. 60-69.
4. Jang, J.S.R.: *ANFIS: Adaptive Network Based Fuzzy Inference System.* In: IEEE Trans. on Systems, Man. and Cybernetics **23** (1993) No. 3, p. 665-684.
5. Jantzen, J.: *Foundations of Fuzzy Control*. Chichester, West Sussex, UK. John Wiley & Sons, 2007.
6. Moore, K.L.: *Iterative Learning Control: An Expository Overview.* In: Datta, B.N. (ed.). Applied and Computational Control, Signals, and Circuits, Vol. 1, Springer-Birkhauser, 1999, p. 151-214.
7. Nie, J., Linkens, D.A.: *Fuzzy-Neural Control: Principles, Algorithms and Applications*. New Jersey, USA. Prentice Hall, 1995.
8. Passino, K. Yurkovich, S.: *Fuzzy Control*. Boston, USA. Addison-Wesley, 1998.
9. Preitl, Ş., Precup, R.E.: *Introduction to Fuzzy Control* (*Introducere în conducerea fuzzy a proceselor*). Bucureşti. Editura Tehnică, 1997.
10. Sala, A., Guerrab, T.M., et al.: *Perspectives of Fuzzy Systems and Control*. In: IEEE Fuzzy Sets and Systems **156** (2005) No. 3, p. 432-444.