Bulletin of the *Transilvania* University of Braşov • Vol. 10 (59) No. 2 - 2017
Series I: Engineering Sciences

SIMULATIONS OF A NUMERICAL IMPLEMENTATION FRIENDLY ADAPTIVE FUZZY CONTROL SYSTEM

C. BOLDIŞOR[1] S. COMAN[1]

Abstract: *A simplified model reference adaptive fuzzy control algorithm is tested on a few generic applications, through a set of simulations done using several Matlab programs. The algorithm is focused on an easy numerical implementation and ignores a mathematical background or demonstration of its convergence. The algorithm is different from the fuzzy model reference adaptive control presented in textbooks in the adaptation part. Here, the adaptation law is not based on fuzzy inference and the fuzzy feature of the system is kept only for the controller.*

Key words: *fuzzy control, adaptive control, fuzzy model reference adaptive control, numerical implementation of fuzzy controllers.*

1. Introduction

Fuzzy logic controllers have always been a less preferred solution in control systems mainly due to the lack of analytical design methods. Usually, fuzzy controllers are obtained from experience or by following a set of guidelines, drawn from examples. All textbooks in fuzzy control theory present applications of conventional fuzzy control systems and explain the choices made, without stating a strict method.

Some may argue that, because a fuzzy inference system has a lot of parameters, having a strict method to determine all of them is at least useless, if not impossible. In the end, if the controller is "fuzzy" then why a "fuzzy" design method wouldn't be good enough.

There are at least two drawbacks in this approach. First, there is never any prove to say that an efficient, good performing fuzzy controller can obtained only by using the simple textbook solutions for a certain plant in a closed-loop control system. Or, in other words, when a fuzzy controller could not be determined to obtain any given control performance, it does not necessarily mean that the fuzzy logic solution is wrong or inappropriate.

Second, the classical fuzzy controllers from most textbook examples have a (very close to) linear behaviour. But this is in contradiction with the main advantage of a fuzzy control algorithm, namely the nonlinearities that may improve performance.

Over the years, the adaptive fuzzy control theory has been developed as solutions to these drawbacks. The model reference adaptive control (MRAC) theory [1] has been adapted to fuzzy systems, leading to notions as fuzzy model reference learning controllers introduced in [5], fuzzy model reference learning controllers, proposed by [7], or the self-

[1] Dept. of Automation and Information Technology, *Transilvania* University of Braşov.

organizing controllers, related to the original work of Mamdani [4]. For the basics and detailed examples please refer to [4] and [6]. The subtle differences are rarely explained, but some clues can be found in textbooks or comparative studies, as [3].

2. Table-based Fuzzy Controllers

In a table-based fuzzy inference system, the rules are represented by an array of numerical values for the output variable [2], [4], [6]. Each dimension of the array is associated to an input variable, and each row on that dimension is associated to a certain value of that variable. The range and values of each input variable are predefined. When the system runs, in any particular operating conditions (meaning that numerical values for input variables are available) the fuzzy inference algorithm computes the output from a few values picked from the table, weighted by the activation level of corresponding rules.

For clarity, let's consider a proportional-derivative (PD) fuzzy controller, which has two input variables, the control error - $e(t)$ and its discrete derivative - $d(t)$, and one output variable, the command action - $u(t)$. For this type of fuzzy controller, the rules-table is a two-dimensional array (table) of values for the command action, $\mathbf{M} = (u_{i,j})$, with $i = \overline{1, Ne}$ and $j = \overline{1, Nd}$. The Ne rows of the table are associated to preselected values of control error from its covering range $e_i \in [-e_{max}; e_{max}]$.Also, the Nd columns are associated to values of error derivative from a covering range, $d_j \in [-d_{max}; d_{max}]$. The details are more clearly presented in Figure 1.

		error derivative, $d(t)$					
		d_1	...	d_j	d_{j+1}	...	d_{Nd}
control error, $e(t)$	e_1	$u_{1,1}$	...	$u_{1,j}$	$u_{1,j+1}$	...	$u_{1,Nd}$
	...	...	...	...	...	...	...
	e_i	$u_{i,1}$	...	$u_{i,j}$	$u_{i,j+1}$	...	$u_{i,Nd}$
	e_{i+1}	$u_{i+1,1}$	...	$u_{i+1,j}$	$u_{i+1,j+1}$	...	$u_{i+1,Nd}$
	...	...	...	...	...	...	...
	e_{Ne}	$u_{Ne,1}$	...	$u_{Ne,j}$	$u_{Ne,j+1}$	...	$u_{Ne,Nd}$

Fig. 1. *Generic rules-table for a proportional-derivative fuzzy controller*

Let us consider the conventional closed-loop control system with a working table-based PD fuzzy controller (see Figure 2). If the ranges of the variables are correctly set, then, for any operating conditions defined by the current input values $e(t)$ and $d(t)$, there are no more than four adjacent active cells in $\mathbf{M}$. We will further refer them with the pairs: (i, j), $(i+1, j)$, $(i, j+1)$ and $(i+1, j+1)$. The row and column indexes are obtained from the conditions $e_i \le e(t) \le e_{i+1}$ and $d_j \le d(t) \le d_{j+1}$. The values in the active cells are $u_{i,j}$, $u_{i+1,j}$, $u_{i,j+1}$, $u_{i+1,j+1}$.

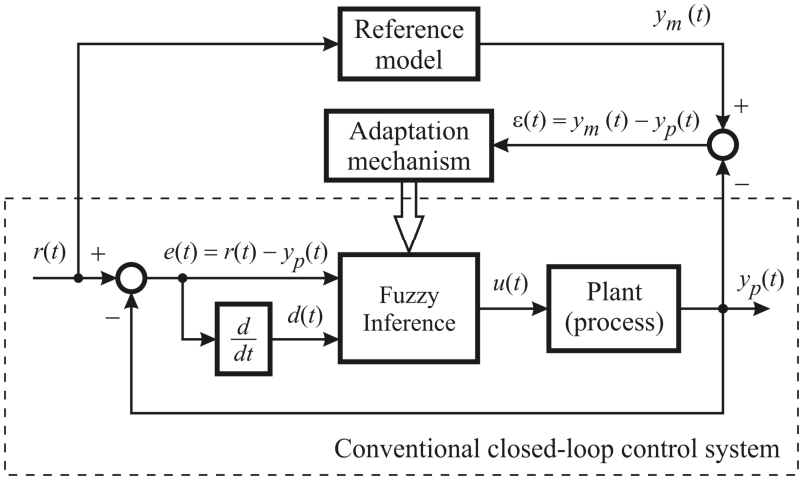

Fig. 2. *The model reference adaptive control system with proportional-derivative fuzzy controller*

In fuzzy logic, the active cells (which are fuzzy rules in another representation) have different activation intensities (or firing strengths [6]), calculated with:

$$\begin{aligned} w_{i,j}(t) &= (e_{i+1} - e(t))(d_{j+1} - d(t)) / (e_{i+1} - e_i)(d_{j+1} - d_j), \\ w_{i+1,j}(t) &= (e_i - e(t))(d_{j+1} - d(t)) / (e_{i+1} - e_i)(d_{j+1} - d_j), \\ w_{i,j+1}(t) &= (e_{i+1} - e(t))(d_j - d(t)) / (e_{i+1} - e_i)(d_{j+1} - d_j), \\ w_{i+1,j+1}(t) &= (e_i - e(t))(d_j - d(t)) / (e_{i+1} - e_i)(d_{j+1} - d_j). \end{aligned} \tag{1}$$

These values are the weights of the cells in the computation of output variable. (Note the indexes in the left-side and those in the nominators of the right-side are.) Finally, having the values in the active cells and their weights, the outputted command action is:

$$u(t) = w_{i,j}(t)u_{i,j} + w_{i+1,j}(t)u_{i+1,j} + w_{i,j+1}(t)u_{i,j+1} + w_{i+1,j+1}(t)u_{i+1,j+1}. \tag{2}$$

For the inference algorithm described here, there are a few elements to be set when defining the rules-table. First, the number of values of each input variable, *Ne* and *Nd*, has to be set. These are the numbers of rows and columns of **M**, so they have important influence on the efficiency of a real-time implementation. Then, the sets of numerical values of input variables, e_i and d_j, have to be set, within predefined covering ranges. So far, having the uniformly distributed points from $[-e_{max}; e_{max}]$ and $[-d_{max}; d_{max}]$ is a good setting. Finally, appropriate values for $u_{i,j}$ have to be found.

3. Adding the Adaptive Feature

Conventionally, the values of the output variable in the **M** table are taken from other examples or based on experience. Of course, this approach leads back to the drawbacks mentioned in the Introduction. A different solution is sought. Many text-books in fuzzy

control show that the table-based fuzzy inference algorithm brings the possibility of adding the adaptive feature, as further described.

In adaptive control theory, the desired performance is defined through a reference model [1]. In direct MRAC, the designer determines the model which assures the desired dynamic and steady-state responses of the control system. Later, any adaptable controller parameter is modified based on the difference between the output of the reference model $y_m(t)$ and the output of the plant $y_p(t)$ namely on the error $\varepsilon(t) = y_m(t) - y_p(t)$. Figure 2 shows the MRAC system, with the PD controller from the previous chapter.

In fuzzy MRAC, as presented in textbooks, the parameters are adapted through a method that implies the use of a second fuzzy rules-table, called *performance table* [6]. Although it is a proven efficient method, a simplification is possible to will ease the implementation. Since the command action is calculated from the values in the inference table based on how active they are, we may also say that the contribution of $u_{i,j}$ to the error $\varepsilon(t)$ is determined by the firing strength of the (i, j) cell, namely $w_{i,j}$. Therefore, it is reasonable to iteratively modify $u_{i,j}$ in strict relation with how much contribution it brought to the error at current iteration. Therefore, the adaptation rule would be:

$$u_{i,j}[k+1] \leftarrow u_{i,j}[k] + \gamma w_{i,j}[k]\varepsilon[k]. \tag{3}$$

We introduced here the specific notations used for discrete-time domain, with the square brackets and the index *k* for the current iteration. The constant coefficient γ is the algorithm convergence speed rate. The initial values in **M** are set to zero: $u_{i,j}[0] = 0$.

4. Simulation Results for a Few Generic Examples

To demonstrate the convergence of the algorithm for the adaptive system with the non-linear fuzzy controller is very difficult. So, in order to validate the method, we run a set of simulations with generic examples, using a custom Matlab program. The basic idea of the program was to avoid any high-level Matlab functions and to keep it as close as possible to a program written for a numerical device (as a microcontroller).

The fuzzy controller is a PD type. Since there are numerical simulations, the control error derivative can be approximated with $d[k] = (e[k] - e[k-1])/T_s$, with T_s being the sampling time. The size of **M** table is $Ne = 11$ rows by $Nd = 5$ columns.

Let's consider that the transient response to a step type command action applied to the plant is available and the plant is a first-order element (PT1). The suggested reference model would also be a PT1. The plant's and model's parameters are listed in Table 1. Of course, the plant and its parameters are considered unknown or uncertain, which is the main reason of the method itself. The discrete-time recurrent equation for the PT1 reference model is:

$$y_m[k] = \frac{T_m}{T_m + T_s} y_m[k-1] + \frac{k_m T_s}{T_m + T_s} r[k], \tag{4}$$

where k_m and T_m are the parameters that assures the desired system performance.

For each simulated case, 10 consecutive trials were done, keeping the values in the **M** table from one trial to the next. If the adaptation works, then the response of the control system should get closer to the one of the reference model trial by trial. A trial consists in applying the step-unit reference signal, $r(t)=1$, $t \geq 0$, which is similar to restarting the system without resetting **M**. For the slow systems, trial time was set to $T_t = 100$ seconds with $T_s = 0.1$ seconds sampling time. For the fast systems, trial time is 10 seconds, with 0.01 seconds sampling time. For every example, a trial has $N_s = 1000$ samples. The value of the convergence speed rate γ was kept to 0.1 for all examples. So far, there is no method to determine its value, neither any recommendation about how to choose it.

To check the algorithm convergence or the adaptation mechanism efficiency, we introduce a performance measure, which is defined for a trial as:

$$\eta_q = \sum_{k=1}^{N_s} \left| \frac{\varepsilon[k]}{y_m[k]} \right| 100\%, \tag{5}$$

where q is the trial index. This is the integral of absolute error relative to the output of the reference model, so the measure will be expressed as percentage of desired output.

Table 1 presents the results for two examples, while Figures 3 and 4 show the output of the control system compared to the reference model for the first and for the last trials.

Simulation results for a few parameter configurations Table 1

Reference model	**Plant's model**	**Performance criterion**	
		first trial	last trial
slow PT1: $k_m = 1; T_m = 10$	slow PT1: $k_p = 0.8; T_p = 20$	12.14	0.51
fast PT1: $k_m = 1; T_m = 0.2$	fast PT1: $k_p = 1.2; T_p = 0.5$	6.92	0.21

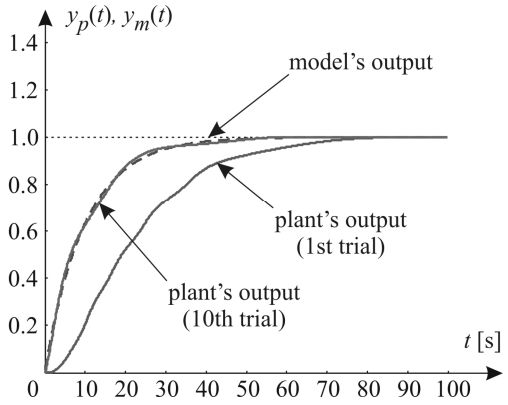

Fig. 3. *The slow plant's output for the 1st and the 10th adaptation trials, compared to the reference model*

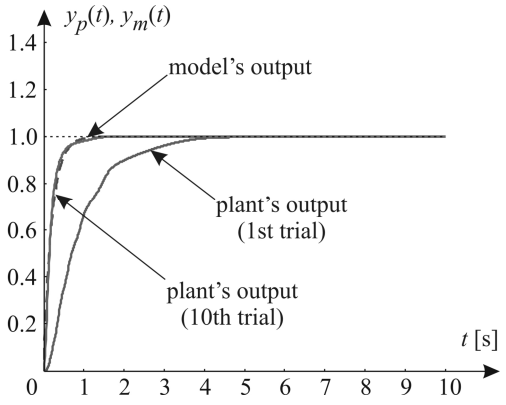

Fig. 4. *The fast plant's output for the 1st and the 10th adaptation trials, compared to the reference model*

5. Conclusions and Future Work

To overcome the drawbacks of the usual design methods for fuzzy controllers, a parameter adaptation mechanism is added to the conventional fuzzy control system. The extended scheme could be kept as an adaptive control system, or it can be used to derive the rule-base of a fixed controller.

The method is a simplified fuzzy model reference adaptive control system, focused on the easy numerical implementation. The adaptation mechanism stands on a simple and intuitive modification rule that uses the measure of how 'active' is the modified parameter at the current iteration. This is the firing strength of the rule containing the modified parameter.

The simulations proved that the method is efficient for a class of applications at least: the first-order systems with slow or with fast dynamics. However, for a better validation, more simulations should be done: second-order underdamped or over-damped plants with first-order or second-order reference models, higher-order linear plants, non-linear plants, unstable plants, different sizes for the rules table, different convergence speed rates, etc. Then, for the refinement of the method, implementations with microcontroller are to be done, to tackle other programming specific issues.

Acknowledgements

We hereby acknowledge the structural founds project PRO-DD (POS-CCE, 0.2.2.1., ID 123, SMIS 2637, ctr. No. 11/2009) for providing the infrastructure used in this work.

References

1. Astrom, K., Wittenmark, B.: *Adaptive Control*. Addison-Wesley, 1989.
2. Boldişor, C., Comnac, V., Coman, S.: *A Practical Review of a Design Method for Fuzzy Controllers Based On Self-Learning Algorithm*. In: Bulletin of the *Transilvania* University of Braşov, Series I, Vol. 4 (53), No. 1, 2011, p. 93-98.
3. Duka, A.V., Oltean, S.E., Dulău, M.: *Model Reference Adaptive vs. Learning Control for the Inverted Pendulum. A Comparative Case Study*. In: J. of Control Engineering and Applied Informatics **9** (2007), No. 3-4, p. 67-75.
4. Jantzen, J.: *Foundations of Fuzzy Control*. UK. John Wiley & Sons, 2007.
5. Layne, J.R., Passino, K.M.: *Fuzzy Model Reference Learning Control for Cargo Ship Steering*. In: IEEE Control Systems Magazine **13** (1993), p. 23-34.
6. Passino, K.M., Yurkovich, S.: *Fuzzy Control*. Addison Wesley, 1998.
7. Yin, T.K., Lee, C.S.G.: *Fuzzy Model-Reference Adaptive Control*. In: Proc. of the 33rd IEEE Conf. on Decision and Control, USA, 1994.