IFAC Symp. on Robot Control, SYROCO'91
16-18. Sept. 1991, Vienna, AUSTRIA,
Part 2, pp. 447-452

SELF-TUNING SELF-ORGANISING FUZZY ROBOT CONTROL

D. Stipaničev

Faculty of Electrical Engineering, Machine Engineering and Naval Architecture, University of Split, R. Boškovića bb, 58000 Split, Croatia, YU

M. De Neyer[1,2], R. Gorez[1]

[1]Laboratoire d'Automatique, Dynamique et Analyse des Systemes, Universite Catholique de Louvain, Place du Levant, 3, 1348 Louvain-La-Neuve, Belgium
[2]Laboratoire de Neurophysiologie, Universite Catholique de Louvain, Belgium

Abstract. The paper describes the idea of self-tuning self-organising fuzzy control. Simple fuzzy controller, introduced at the beginning of seventies as a rule-based controller, had two main disadvantages: difficulty with definition of good control rules and problems with tuning of controller parameters. Self-organising controller was developed to overcome the first problem. In this paper we propose a self-tuning procedure to overcome the second problem. That procedure is based on expert knowledge about the influence of the tuning parameters on the system response. Theoretical results are illustrated and tested by simulation a two link robot manipulator.

Keywords. Fuzzy control; self-organising control; tuning of controller parameters; robot dynamic control.

INTRODUCTION

Fuzzy controllers have been introduced more than fifteen years ago (Mamdani,1974) as a rule base controllers structured to synthesize the linguistic control protocol of a skilled human operator. Since that time many applications have been reported in different fields (Mayers, Scherif, 1965; Mamdani, Stipaničev, 1989). In the last couple of years some attempts have been made to apply ideas of fuzzy control in the field of robotics,too, for control of robot dynamic, for planning and control of robot motion, for analysis and interpretation of information from robot's sensors and for communication with robots using natural language (Stipaničev, Efstathiou, 1991). Here our particular interest is application of fuzzy logic for control of robot dynamics.

Let us first emphasize main advantages of fuzzy approach to control. These are:

(a) Fuzzy control does not required a detailed mathematical model of the controlled process to formulate the control algorithm.
(b) It has quite robust and adaptive capabilities.
(c) It is capable to operate for a large range of inputs.

Property of fuzzy controllers robustness and adaptivity was particularly interesting for their application in the field or robotics. Let us use as an example a robot system with revolute and prismatic joints (Scharf, Mandić, Mamdani, 1986).

Such robots are widely used for many tasks in industry because they are fast acting and they approach the flexibility of use which is usually ascribed to the human arm. Industry has successfully mastered the techniques for manufacturing such robot arms but it is now at the stage where their dynamic performances is often called into question.

The problem is that the moment of inertia of the arm links about their main control axes-rotation, shoulder and arm-can exhibit pronounced changes with change in disposition of the robot, as well as the load carried at its tip. This results in highly complex dynamics as Lee (1982) has emphasized.

Model-reference adaptive control (MRAC) is usual approach to control such systems Disadvantage of MRAC is that it generally requires a detailed knowledge of the system dynamics and assumes that
(a) these are linear, and,
(b) the dynamics of the actual system differ from the model only with respect to values of its coefficients.

Contrary to this approach, fuzzy control approach is based either on only approximate knowledge of the system behavior which need not be linear (in the case of non-learning rule-base fuzzy controller) or on very simple incremental system model (in the case of learning, self - organizing, rule - base fuzzy controller). Joint feature of both fuzzy control approaches to control of robot dynamics is the existence of knowledge base, or more preciously the existence of rule-base where rules about control procedures are stored. Because. of that fuzzy controller is a special case of *productional system*, where fuzzy set theory has been used for representation of knowledge (control rules) and for doing inferences with that knowledge.

Ordinary fuzzy controller has two main disadvantages: first it is quite difficult to find good control rules and second, the tuning of control parameters was a long procedure usually based on try-and-error. To overcome the first problem the self-organising controller (SOC) was developed. It was capable to create its own control rules automatically after a number of learning sessions. But problems with tuning were still present and due to even more tuning parameters, the tuning of controller was even more difficult. Our idea is to make this tuning procedure automatic, so in this paper we propose a self-tuning self-organising fuzzy controller. It has all features of self-organising fuzzy controller and additionally the controller parameters could be automatically tuned.

First, shortly the self-organising controller will be introduced, then tuning procedure will be analysed in details and last a numerical example of robot dynamic control will be presented to illustrate the proposed method. Example is based on a simulation model of two-link robot manipulator.

We want to emphasize that this study is in relation with a physiological research project aiming at the use of fuzzy model for the human behavior and their transpose to robotics applications.

PRINCIPLES OF A SELF-ORGANISING CONTROLLER

In this section, the underlying principles of a self-organising controller are given. A more detailed description can be found in (De Neyer et al, 90; Procyk, 79). Details of the implementation are given in Chapter 5.

A self-organising controller is a fuzzy controller which acquires its control rules through experience in order to obtain a predetermined closed-loop control performance. It consists in two hierarchical layers: the basic layer is a fuzzy controller and the second one is a learning module which generates and modifies control rules (Fig. 1).

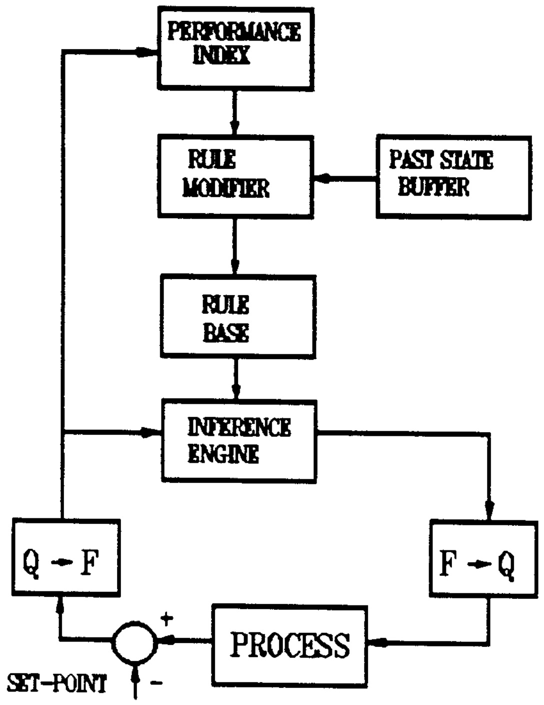


Fig. 1. Self-organising controller
Q→F quantitative to fuzzy interface
F→Q fuzzy to quantitative interface

The control strategy of the fuzzy controller is described by a set of linguistic rules , expressed as " if situation" then action" statements. An example is

If Error is A_i *and* Variation of error is B_i
Then Control = C_i

where A_i, B_i, C_i are linguistic-fuzzy values like big, small, etc. They are labels of fuzzy sets describing the meaning of those values. Fuzzy sets are characterized by a membership function defined on an ordinary set (called universe of discourse. Error and control are the controller fuzzy variables. For a given actual situation (error = A) each rules gives a contribution to the control variable by means of an inference law and all the contributions are gathered in one fuzzy value using an aggregation method.

In control applications, both values coming form measure devices and values entering the actuator are not fuzzy, they are precise numbers, so interfaces between the real-world and the controller are needed. They perform two functions: scaling and conversion from quantitative to fuzzy and vice-versa. Scaling is due to the boundness of the universe of discourse on which the linguistic values are defined. Quantitative to fuzzy conversion is an interpretation of the real value of the variable using linguistic values. The inverse conversion is performed using a defuzzification method, transforming a fuzzy value in a quantitative one.

The learning module generates and modifies the control rules in order to obtain a predetermined control performance. A two steps modification mechanism is used: first, a control performance assessment and then a modification of the control strategy if the performance is not satisfactory. Performance assessment is done by means of a performance index (PI). The performance index is a measure of the difference between the actual process output and the desired one. It is described by linguistic rules like that of fuzzy controller:

If Error is A_i *and* Variation of error is B_i
Then Performance Index = D_i

All the rules with a PI value equal to zero defines a tolerance band in witch the process has to be contained. A non null PI value is a measure of the correction which is required at the process output.
Rule modification method is based on the assumption that the controller output at n samples in the past is responsible for the present performance. If PI value is null, no modification is performed. Otherwise, the linguistic control value of the rule corresponding to the process state at time t-nT is changed (T is the sampling period, n is a delay). The new value is equal to the sum of PI value and control value at time t-nT.

TUNING OF CONTROLLER PARAMETERS

In synthesis of self-organising controllers one of difficulties is the tuning of controller parameters. SOC has a lot of parameters which have to be defined and tuned in order to obtain

(a) satisfactory system response, and
(b) convergence of self-organisation of control rules.

The most important parameters of SOC are

(a) scaling factors G_E, $G_{\Delta E}$, $G_{\Delta^2 E}$ and G_U,
(b) delay in rule modification n,
(c) performance index (PI) table,

but

(d) definition of fuzzy values in control rules,
(e) definition of inference procedure, and
(f) definition of defuzzyfication method,

also have influence on close loop behavior.

A lot of simulations and experimental trials have been made to compare different inference and defuz- zification methods and to find influence of fuzzy values definition on controller performances. Here we have used the most simple definition of fuzzy values by triangular fuzzy sets, plus-product inference and center of gravity for defuzzification without any comparison with other methods, because our interest was primarily in tuning of scaling factors.

Performance index table was taken from (Sugiyama,1986) with certain modifications, because we have used a real three-term controller and Sugiyama's controller was a pseudo three-term controller.

Delay in modification of control rules was determinate experimentally taking into account the behavior of the controlled system.

Main attention was given to tuning of scaling factors, or more preciously to tuning of input scaling factors. Output scaling factor G_U was assumed fixed and equal to 1.

Sugiyama (1986,1988) has proposed a method for choosing of scaling factors comparing the behavior of SOC with that of model-reference adaptive control (MRAC). In this sense the allocation of zero elements of PI table specifies the desired process behavior. Sugiyama made a linearisation of PI table, assuming that zero element are on diagonal, and after comparing that table with the second order system, he find a correspondence between scaling factors and behavior of reference model and expressed that with equations

$$\omega_n = (G_E/G_{\Delta E})^{0.5} \qquad (1)$$

$$\zeta = 0.5\ (G^2_{\Delta E}/G_E\ G_{\Delta^2 E})^{\ 0.5} \qquad (2)$$

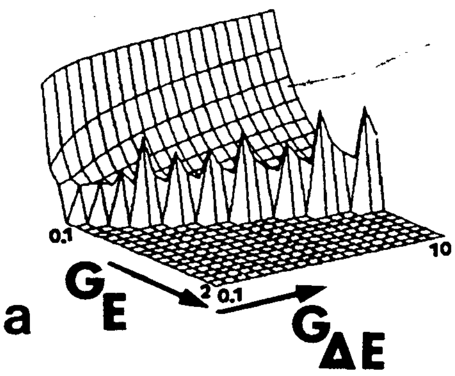
Using this equations we have calculated the dependence of scaling factors and maximal overshoot M_p and time of maximal overshoot T_p of the reference model. Fig.2. shows this dependence for $G_{\Delta^2 E} = 5$.

The control goal is usually quicker response and smaller overshoot. From Fig.2. it is possible to notice that both M_p and T_p have more than one local minimum. Sugiyama (1986) has proposed to use these approximate equations in tuning of SOC parameters. We did a lot of trials applying his methods, but without a real success. Sugiyama proposal was to tune scaling factors in order to define reference model which is supposed to be linear and of the second order. His method is successful for simple, linear processes but than we don't need fuzzy controller at all. For more complex nonlinear processes, as robot manipulator is, this method of tuning parameters does not give satisfactory results.

Our idea was to try to summarize the influence of different tuning parameters on system response in the form of tuning rules and than to use this approximate knowledge in tuning procedure. Four such rules could be defined: three about influence of each scaling factor to system response and one about relative values of scaling factors.

1. *Rule about G_E*

Decrease of G_E causes increase of system dumping (slower response), decrease of overshoot and increase of tolerant band around steady state value.



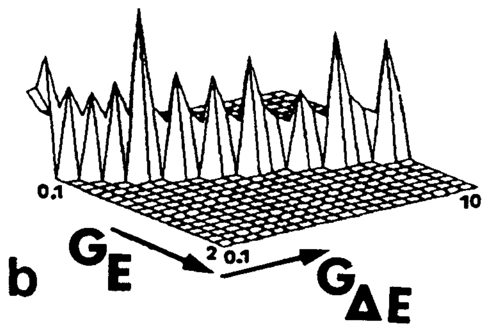


Fig.2. Behavior of the reference model for $G_{\Delta^2 E} = 5$
a) Maximum overshoot M_p
b) Time of maximal overshoot T_p

Note: Too big G_E causes limited cycle oscillations and instability.

2. *Rule about $G_{\Delta E}$*

Decrease of $G_{\Delta E}$ causes decrease of dumping (quicker response) and increase of overshoot.

Note: To small $G_{\Delta E}$ causes instability.

3. *Rule about $G_{\Delta^2 E}$*

Decrease of $G_{\Delta^2 E}$ for small values of $G_{\Delta E}$ causes decrease of dumping (quicker response) and increase of overshoot. For large values of $G_{\Delta E}$ influence is opposite.

Note: Too big $G_{\Delta^2 E}$ causes increase of overshoot and non regular response and after that limited cycle oscillations.

4. *Rule about relative values of scaling factors*

To have a satisfactory response an inequality $G_{\Delta E} > G_{\Delta^2 E} \geq G_E$ must be satisfied.

Important is to emphasize that usually local minima exist, so sometimes it is necessary to make few steps in one direction to find a better response.

This tuning strategy was used in simulation of control of robot dynamic. After each learning session only one gain was changed in view to obtain better performances. Gain were changed cyclically in following order: first $G_{\Delta E}$, then G_E and last $G_{\Delta^2 E}$, and than again from the beginning.

For example if speed is not satisfactory first $G_{\Delta E}$ is decreased, then G_E increased and last $G_{\Delta^2 E}$ increased or decreased depending of the value of $G_{\Delta E}$.

This tuning strategy is quite simple and it could be easily implemented as a automatic tuning procedure, so self-tuning self-organising fuzzy controller could be constructed. The main limitation of the tuning strategy is that it is procedure for improvement of response performances, so it is necessary to start with scaling factors which at least lead to a stable closed loop and too the convergence of the control strategy.

SIMULATION OF ROBOT DYNAMIC CONTROL

Simulations runs were performed on a PC 386 computer using a dedicated program written in C language. A two link robot manipulator structured according to Fig.3. is considered and described by equations:

$$u_1 = I_{11}\ddot{\theta}_1 + I_{12}\ddot{\theta}_2 + CC_1(\underline{\theta}, \dot{\underline{\theta}}) + G_1(\underline{\theta}) \qquad (3)$$

$$u_2 = I_{21}\ddot{\theta}_1 + I_{22}\ddot{\theta}_2 + CC_2(\underline{\theta}, \dot{\underline{\theta}}) + G_2(\underline{\theta}) \qquad (4)$$

where I_{ij} is the matrix of inertia, CC_i represents the Coriolis and centrifugal effects, G_i represents the gravity effects, and u_i is the applied torque. Those terms are given by

$$I_{11} = (0.125\, m_1 + m_2)\, l_1^2 + 0.125\, m_2 l_2^2 + m_2 l_1 l_2 \cos\theta_2 = p_1 + p_2 \cos\theta_2 \qquad (5)$$

$$I_{12} = I_{21} = 0.125\, m_2 l_2^2 + 0.5\, m_2 l_1 l_2 \cos\theta_2 = p_3 + 0.5\,\theta_2 \qquad (6)$$

$$I_{22} = 0.125\, m_2 l_2^2 = p_3 \qquad (7)$$

$$CC_1 = -m_2 l_1 l_2 \sin\theta_2\, \dot{\theta}_1 \dot{\theta}_2 - 0.5\, m_2 l_1 l_2 \sin\theta_2\, \dot{\theta}_2^2 = -p_2 \sin\theta_2 (\dot{\theta}_1 \dot{\theta}_2 + 0.5\, \dot{\theta}_2^2) \qquad (8)$$

$$CC_2 = 0.5\, m_2 l_1 l_2 \sin\theta_2\, \dot{\theta}_1^2 = 0.5\, p_2 \sin\theta_2\, \dot{\theta}_1^2 \qquad (9)$$

$$G_1 = (0.5\, m_1 g l_1 + m_2 g l_2)\sin\theta_1 + 0.5\, m_2 g l_2 \sin(\theta_1 + \theta_2) = p_4 \sin\theta_1 + p_5 \sin(\theta_1 + \theta_2) \qquad (10)$$

$$G_2 = 0.5\, m_2 g l_2 \sin(\theta_1 + \theta_2) = p_5 \sin(\theta_1 + \theta_2) \qquad (11)$$

Data of links 2 and 3 of Unimation PUMA 560 robot manipulator are used (Geng, Jamshidi, 1988):

$l_1 = l_2 = 0.432$ (m)	$p_2 = 2.12$
$m_1 = 15.91$ (kg)	$p_3 = 0.265$
$m_2 = 11.36$ (kg)	$p_4 = 81.82$
$p_1 = 3.82$	$p_5 = 24.06$

The task was to reach a desired position $\theta_{1,ref}$ starting from 0, where $\theta_{1,ref}$ is assumed to be in the instable part of the plain and the angle of the second link was fixed and equal to zero. Fuzzy controller was an incremental one. It had three inputs and one output and control rules of the form:

If (E is A_i and ΔE is B_i and Δ^2E is C_i)
Then ΔU = D_i

where E is the error, ΔE is the variation of the error Δ^2E is the variation of ΔE and ΔU is the increment of the control variable. A_i, B_i, C_i, D_i are their respective linguistic values modeled with simple triangular fuzzy sets. 13 linguistic values for E, ΔE, ΔU and only 3 for Δ^2E (positive, negative and zero) were used.

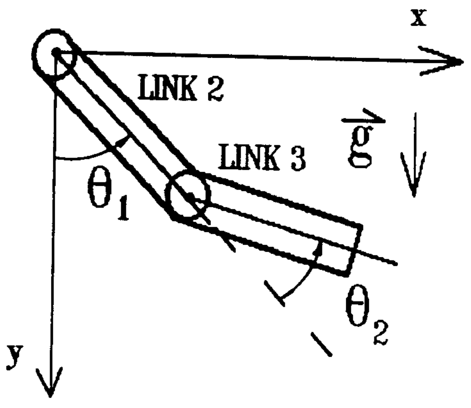


Figure 3: Two-link robot manipulator

Inference mechanism was defined by

$$\mu_D(z) = \sum_i (\mu_{A_i}(e_s) \cdot \mu_{B_i}(\Delta e_s) \cdot \mu_{C_i}(\Delta^2 e_s) \cdot \mu_{D_i}(z)) \qquad (12)$$

where D is the inferred fuzzy value from all the rules (aggregation being done by taking the sum of all the contributions), e_s, Δe_s, $\Delta^2 e_s$ are scaled values of the errors (Fuzzy values are in upper cases and quantitative ones are is lower cases). Also nonlinear scaling factors were used:

$$x_s = S(G_x \cdot x(k)) \qquad (13)$$

where x(k) is the actual value of sampled errors (e(k), Δe(k), Δ^2e(k)), G_x is a constant scaling factor and S is a piece wise linear function with saturation. The gain of S increases when x decreases (its shape is similar to an hyperbolic tangent function).

Fuzzy output value D was defuzzified by a center of gravity method

$$\Delta u_s = \frac{\sum_z z \cdot \mu(z)_D}{\sum_z \mu_D(z)} \qquad (14)$$

and the control variable was updated as:

$$u(k) = u(k-1) + G_U \cdot \Delta u_s(k) \qquad (15)$$

G_U being a constant scaling factor.

Performance index rules used in the learning procedure, have the same form of the controller ones

If (E is A_i and ΔE is B_i and Δ^2E is C_i)
Then PI = E_i

The rule R_c corresponding to the process state at sampling time k-n is updated if the present value of performance index value is different from zero as follows:

R_c: *If* (E is A_c and ΔE is B_c and Δ^2E is C_c)
Then ΔU = D_c(k-n)

$$D_c(k) = F(\Delta u(k-n) + PI(k))$$

$D_c(k)$ being the new linguistic value of the control variable in R_c, Δu(k-n), the incremental control output at k-n (k is the present sampling time, n is the delay in modification), PI(k), the present value of the performance index and F, a fuzzification function. Fuzzification consists in transforming a crisp value in a fuzzy value described by a fuzzy set having a given membership function. Process was simulated in continuous time whereas the controller has operated in discrete time with sampling period of 0.05).

Here below a tuning experience with the robot arm is presented using the tuning strategy proposed above (section 4). After each learning phase one gain was changed in view to obtain a better control performance until the desired performance is obtained. A learning phase consists in a number of learning experiences and ends when the control strategy converges or the maximum number of experiences is reached. In our case this maximum number was limited to 10.

The three inputs gains where changed in the order: first $G_{\Delta E}$ then G_E and last G_{Δ^2E}. The control performance was estimated by means of four ratios: the rise time from 10% to 90% of the steady value (RT), the 5% settling time (ST), the average absolute error over the last 80 samples (AAE), the integral of the square of the error (ISE). Experimental conditions were:

θ_{ref} = 2.3 , delay n = 2, G_U = 1., No. of samples = 400

Seven tuning steps are described, the last one being considered as having the desired performance. Those steps are summarized in Table 1 giving the values of the gains and of the ratios. First step consists in finding gains values which lead to a stable closed loop and to the convergence of the control strategy. The state 1 (Table 1 and Fig.4a) being satisfactory for the steady state error, an increase of the speed is tried by decreasing $G_{\Delta E}$. This increase of speed goes together with an increase of the steady state error (state 2). Thus next step is to decrease the steady state error by an increase of G_E. As a consequence, an increase of speed is also obtained (state 3). The goal of the following step is the same than the previous but by trying an increase of G_{Δ^2E}. Again this leads to smaller error in steady state and to a faster response. The three gains have been changed, the cycle begins again. In state 5, an increase of speed is obtained through a decrease of $G_{\Delta E}$ but with ,as a consequence, a greater steady state error and small oscillations. Next step is to decrease the error in steady state. In state 6, no improvements are observed by changing G_E. This decreasing is obtained in state 7 (Table 1 and Fig.5b)by an increase of G_{Δ^2E}.

At the end let us mention that another kind of SOC, introduced by, Sugiyama (1988), was also implemented for controlling the same robot manipulator but gains tuning appears to be impossible or at least very difficult. None convergence in the rule base was observed. This SOC is more complex than that one which is presented here. In peculiar, it does not use linguistic values for describing the control value of a rule. The control values can be any real value and the number of control value is not limited. Moreover the learning procedure allows at each sampling time the modification of several rules. That makes the convergence harder to obtain especially in the case of a nonlinear instable process as we considered.

TABLE 1 Tuning steps

state	G_E	$G_{\Delta E}$	G_{Δ^2E}	RT	ST	AAE	ISE
1	9	300	80	4.11	5.45	0.0003	6.86
2	9	200	80	3.29	4.25	0.07	5.68
3	12	200	80	2.42	3.5	0.027	5.23
4	12	200	90	2.28	3.1	0.0073	5.15
5	12	150	90	1.8	3.0	0.06	4.6
7	12	150	100	2.14	2.85	0.0045	4.8

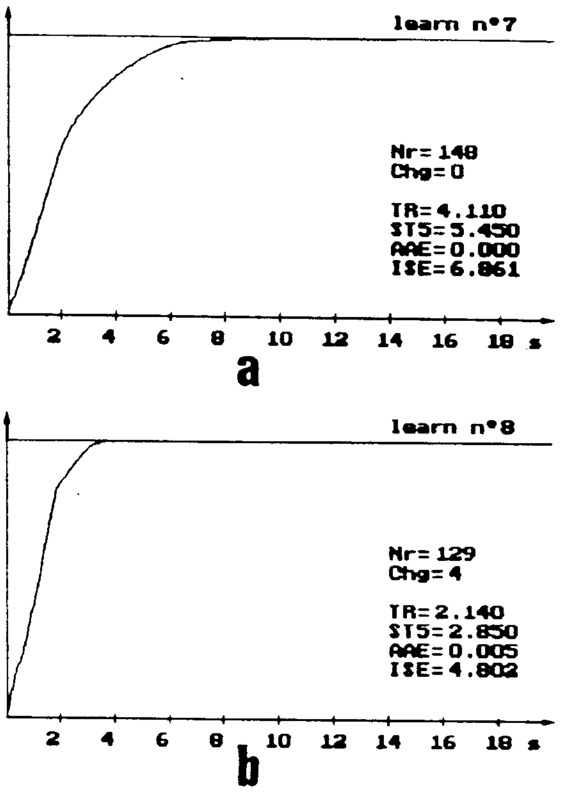


Figure 4: Process response
a) state 1
b) state 7

CONCLUSION

Principles of self-tuning self-organising fuzzy control is introduced and described. Self-tuning procedure was proposed to overcome the difficulty with tuning the parameters of ordinary self-organising fuzzy control. We have tried to make the tuning phase automatic using the heuristic knowledge about the influence of the tuning parameters on the system response. Simulations were performed on nonlinear model of two link robot manipulator and controller was used to control the robot dynamic.

Three main difficulties were met during implementation of self-tuning self-organising fuzzy procedure for control of robot arm. The first one was due to the instable characteristic of the system. The use of large values for the gains allows the process stabilisation but convergence was very hard. With small values of gains, several learning experiences were needed to obtain stability. Afterwards convergence was possible. This problem could be solved by including an "*a priori*" knowledge in the control rule base to avoid instability instead of starting the learning phase without rules as we did. That starting rule base could reproduce approximately the behavior of a PID controller such that $\Delta U \approx E + \Delta E + \Delta^2 E$.

The second difficulty is that several steps in the same direction of change (increase or decrease) are needed to observe the effects of these change in gains. In fact relationships linking the ratios and the gain have several local minima due to the nonlinearities of controller and process.

The third one is that it is necessary to start with scaling factors which at least lead to a stable closed loop and to the convergence of the control strategy. Our tuning strategy is good for improvement of response characteristics but not for find the starting combination.

Although this study was primarily in relation with the project aiming to use the fuzzy models of the human behavior and their transpose to robotics applications, we believe that such control principles could be quite useful also for real control of robot dynamic. Experimental studies has yet demonstrated advantages of self-organising fuzzy control for robot dynamic control and we hope that using self-tuning procedure it will be even easier to implement and apply such kind of controllers.

ACKNOWLEDGEMENTS. This research was supported by the Belgian National incentive-program for fundamental research in Artificial Intelligence, Prime Minister's Office - Science Policy Programming and Ministry of Science of Republic Croatia. The scientific responsibility is assumed by its authors.

REFERENCES

Geng,Z., Jamshidi,M. (1988) Expert self-learning controller for robot manipulator. Proc.of the 27th Conference on Decision and Control. Austin. Texas. 1090-1095

Lee,G. (1982). Robot arm kinetics, dynamics and control. IEEE Computer. 12. 62-80

Mamdani,E.H. (1974). Application of fuzzy algorithms for control of simple dynamic plant. Proc. IEE (London). 121. 1585-1588

Mamdani,E.H., Stipaničev,D. (1989). Fuzzy set theory and process control- past, present and future. Proc.of IFAC Symp. on Advanced Information in Autonmatic Control. Nancy. July. 269-272

Mayers,J., Scherif,Y.S. (1985). Application of fuzzy set theory IEEE Trans.Syst.Man Cybern.. 15. 175-189

De Neyer,M., Stipaničev,D. & Gorez,R. (1990). Intelligent self - organizing controllers and their application to the control of dynamic systems. Proceedings of the IMACS Symposium on Mathematical and Intelligent Models in System Simulation, Brussels. Sept.3-7. IV.B.V.1-6.

Procyck,T.J. (1979). A linguistic self - - organising controller. Automatica (U.K), 15, 15-30.

Scharf,E.H., Mandić,N.J., Mamdani,E.H. (1986) A self organising algorithm for the control of a robot arm. Int.J.of Robotics. 1. 33-41

Stipaničev,D., Efstathiou,J. (1991). Application of fuzzy reasoning in planning, decision making and control of intelligent robots. to be published in Fast, Invariant, Dynamic and Parallel Intelligence B.Souček and ISI group. J.Wiley and Sons.Inc. New York

Sugiyama,K. (1986). Analysis and synthesis of the rule - based self - organising controller. Ph.D. Thesis. Queen Mary College. University of London

Sugiyama,K. (1988). Rule-based self-organising controller. Fuzzy computing: theory of hardware and applications. M.M.Gupta & T.Yamakawa (ed). Elsvier Science Publishers. London. 341-353,