Self-learning Fuzzy Controller with a Fuzzy Supervisor

Chiy-Ferng Perng, National Taiwan University, Taiwan, R.O.C.
Yung-Yaw Chen, National Taiwan University, Taiwan, R.O.C.

Abstract

Through the observation of the learning process of human, we can find punishments and rewards playing a somewhat important role. The judgment of whether the action should be punished or rewarded and the decision of the degree of punishments or rewards are two major problems in such a scheme. In this paper, we propose a fuzzy rulebase to provide the guiding criterion for the self-learning fuzzy controller to construct its fuzzy control rulebase.

1. Introduction and Motivation

Since Prof. L. A. Zadeh proposed the concept of fuzzy sets in 1965 [1], fuzzy sets theory has become a popular and fast-growing research topic. Fuzzy sets theory is used in many fields to solve practical problems in the real world. The applications of fuzzy sets theory range from consumer electronics to medical diagnosis. One of the most successful applications is fuzzy control. The first fuzzy control steam engine was produced in 1975 [2]. Since then, in literature, one can find many fruitful examples of fuzzy control applications. However, there exists one problem: construction of a workable fuzzy controller.

A fuzzy controller is composed of 4 elements: fuzzification, defuzzification, the inference engine, and the rulebase. Basically speaking, the processes of fuzzification, defuzzification, and inference engine are easily to establish because there are certain formulae to follow. Only the construction of the rulebase could be varied with different usage. The rulebase is usually derived from experts' experiences. Nevertheless, only in few control problems one can find so-called experts. Even though, the transformation of experts' experience into a linguistic rulebase is also a difficult job. Learning schemes seem a good way to solve such a problem.

Michie [3] divided the concerned space into several boxes. The learning machine helped the boxes to adjust their output according to the behaviors of the system. Barto and Sutton [4] used the similar structure as Michie's boxes system. They used associate critic elements (ACE) and associate search elements (ASE) to evaluate the effect of the default control effort and correct the output of each box. By the method called "reinforcement learning", the controller can learn how to control a complex plant. Anderson [5] used two neural nets to substitute the functions of ASE and ACE. He called them "evaluation network (EN)" and "action network (AN)". C. C. Lee [6] also used the scheme of the reinforcement learning. His controller's structure is the same as Barto's controller but

0-7803-4863-X/98 $10.00©1998 IEEE

he used fuzzy rules to control the plant. His ASE is the combination of a fuzzy controller and an associate learning neuron (ALN). ALN adjusted the membership function of the output of each rule. Berenji [7][8] combined Anderson's work with a fuzzy controller. His fuzzy controller is also a neural network. This neural net learned the behavior of the fuzzy controller. C. T. Lin and G. C.-S. Lee [9] also developed a hybrid fuzzy controller. This controller uses two learning algorithm, backpropagation and self-organization to construct a neural network. It succeeded to create a new rulebase and make a car pass a curve road through learning. Besides, several other methods have been proposed for the construction the controller by self-learning. [10][11]

These self-learning schemes indeed solve some problems caused by the construction of rulebases. However, the methods often entangle with complex computations or many meaningless weighting values. In fact, the learning process of a human is usually complex but meaningful and purposeful. The easiest learning process is punishments and rewards. Punishments make clear what should not be done. Rewards let one know what he or she could do or even do better. If the concept of punishments and rewards could be applied to the construction of a fuzzy controller's rulebase, it should make the construction of a rulebase reasonable and improve the efficiency of building the rulebase.

In this paper we propose a fuzzy supervisor to realize the concept of punishments and rewards. In section II, details about the fuzzy supervisor are given. How to add this fuzzy supervisor to a fuzzy controller is explained in Section III. Section IV contains the simulation results of the inverted pendulum. Our conclusions are given in the last section.

2. The Fuzzy Supervisor

Let's consider a plant, whose control purpose is to keep balance in zero state, position zero and velocity zero. When the previous states, velocity and position, are both positive large, the position of the plant's next states will become further from zero without any control signal. Obviously, the control force should let the velocity move towards negative to slow down the increase in position or even make the position moving reversely. If the controller-generating force makes velocity move toward negative, then this action should be encouraged, otherwise, and it should be punished..

At first, we have to define what the punishment is and what the reward is. The practical method to realize these concepts is to quantify them into small real number. Thus we define them in the domain [-1,1]. Positive number means a reward and negative number means a punishment.

For a second-order system, the concerned area is divided into four blocks: (position, velocity), (+,+), (+,-), (-,+), and (-,-). Then it is necessary to find some parameters of the system to be the reference for the evaluation or criterion. Since we tried to implement a fuzzy rulebase to be the criterion generator. Appropriate sets of parameters are important. In our system, the parameters chosen are not ordinary errors and error dots. What we used are shown in figure 1. At first, the error and error dot should be normalized. Normalization makes the values of the concerned states bounded by 1 and −1. The criterion given by the supervisor is concerned the system state at time t, $X(t)$, and the state at time t+1, $X(t+1)$. Ra_1 represent the distance between origin and $X(t)$. Ra_2 is the distance between $X(t)$ and $X(t+1)$. An-1 is the angle between e axis to $X(t)$. An-2 is the angle between the vertical line through $X(t)$ and ra_2. After all, the rule in the

fuzzy supervisor is like :

IF an_1 is *big*

and an_2 is *big*

and ra_1 is *small*

Then EV is a,

where EV is the criterion of the action from X(t) to X(t+1), which is given by the fuzzy supervisor. *a* is a constant, so the output is a singleton.

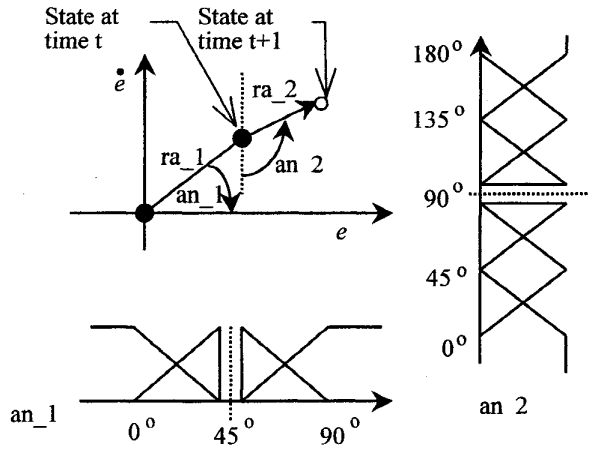

Figure 1. The parameters used by the fuzzy supervisor

Based on the above-mentioned thoughts, we develop a fuzzy rulebase, which is so-called fuzzy supervisor. The output *a* is a singleton. The domain (+,+) and (-,-) shares the same output function because the behavior are similar in these two domain. For example, both elements of $X(t)$ in the second order system are positive. If the control force makes the state toward more positive, the criterion should be negative. On the contrary, in the (-,-) domain, both elements of $X(t)$ are negative and the control force makes the state toward negative. It is obvious that criterion should be also negative. We can see that the criterions are symmetric in the two domain.

As the same reason, in the domain (-,+), the criterion functions are also symmetric to the ones in area(+,-). The strategy is to keep the state in this area and not move towards extreme situation. Compare $X(t)$ and $X(t+1)$, when the error dot of $X(t+1)$ moves towards positive, then the criterion mechanism will give more serious punishments and hope the controller to make the error dot move towards negative. According to the above description, we have construct total 144 rules. And the outputs of the rules of the fuzzy supervisor are shown as following.

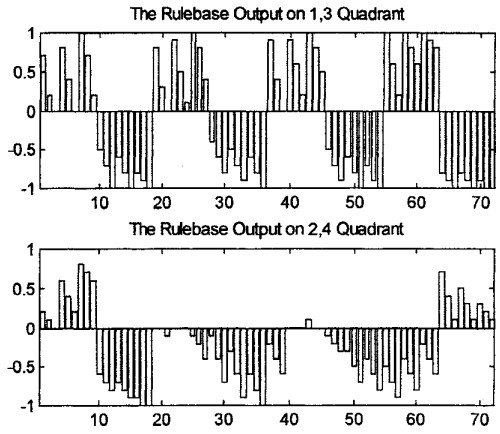

Figure 2. The output of the fuzzy supervisor

3. The Structure of the Heuristic Criterion Self-learning Fuzzy Controller

We have made some changes in the structure of the fuzzy controller in this paper. Every contol rule in the rule base will be simplified into one round cell in the concerned space. And the parameters of a rule are also simplified into center of the cell, radius of the cell, and output of the cell. Figure 3 shows the basic structure of this kind of cellular fuzzy controller.

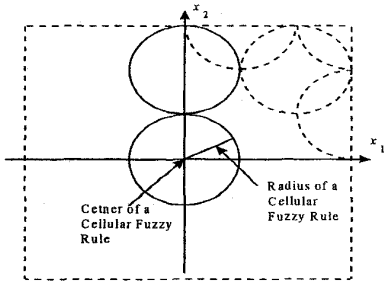

Figure 3. A cellular fuzzy rule

In the cellular fuzzy controller, the fuzzification process slightly differs from traditional fuzzy controllers. At first the fire strength depends on the distance between the rule center and the states :

$$f\_s = \frac{r - \sqrt{(x_1 - c_1^n)^2 + (x_2 - c_2^n)^2}}{r} \qquad (1)$$

where f_s means fire strength of the rule, and r is the radius of the rule. (x_1, x_2) is the current state. (c_1^n, c_2^n) is the center of the n-th rule. Of course if, the distance between the default states and the rule center is larger than the rule radius, then the fire strength of the rule will be zero. The states will be normalized before they are sent into the controller. The output of one rule is only one crisp number not a fuzzy set. The defuzzification we used is like weighting sum :

$$F = \frac{\sum_{n=1,\ldots,N} f\_s_n \times f_n}{\sum_{n=1,\ldots,N} f\_s_n} \quad , \qquad (2)$$

where F is the final output of the fuzzy controller. f_n means the nth rule output.

At the beginning of the time period (t-1), the system sensor will feed the plant states at the end of time period (t-2) back to the controller and the criterion mechanism. The fuzzy controller will send a control signal to the plant according to the states. The plant acts under this control signal at the time period (t-1). The criterion mechanism will receive states at the end of time period (t-1). Then it will evaluate the fitness of the control action adopted in time period (t-1) according to the plant states at the end of time period (t-2) and (t-1). It sends the β signal as the result of evaluation. The output of rules will be adjusted according to the following formula:

$$f(t+1) = f(t) + \alpha \times EV(t) \times h(t) \times ra\_1 \times sign(F(t) - F(t-1)) \qquad (3)$$

where α is the learning coefficient, EV is the evaluation from the fuzzy supervisor, and $h(t)$ is the fire strength history :

$$h(t) = 0.9 \times f\_s(t) + 0.1 \times h(t-1) \qquad (4)$$

F(t) is the output force at the time period (t). The adjusted rule outputs will be the bases of controller output at the time period (t). In (3), the sign function is used to express the moving direction of status' error dot. The structure of our self-learning fuzzy controller with a fuzzy supervisor is shown in figure 4.

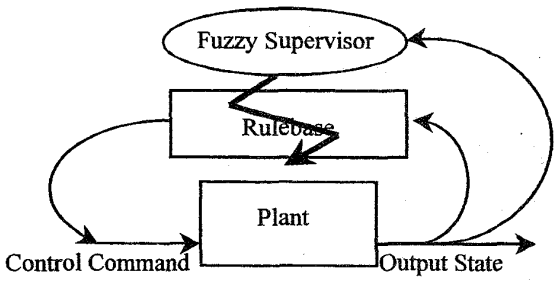

Figure 4.The structure of the fuzzy controller with a fuzzy supervisor

4. Simulation Results

We use the control of the inverted pendulum to test the effect of the self-learning fuzzy controller with a fuzzy supervisor. In the inverted pendulum problem, we will concentrate on the angle control. The basic data of our cart and pole are as follows: length of the pole is 0.5m; mass of

the pole is 0.3kg; mass of the cart is 0.5kg; friction coefficient of pole is 0.000002; friction coefficient of cart is 0.0005; The concerned area: the angle ranges from -12 deg to 12 deg and the angular velocity range from -50 deg/sec to 50 deg/sec.

The learning coefficient is :

$$20 \times \frac{10}{10 + \text{trial}}, \qquad (4)$$

where trial is the number of training process, trial = 1,...,48. The concerned area is divided into 5*5=25 cells.

In the learning process of the inverted pendulum, the time interval is 0.02 second. The initial outputs of control rules are generated at random and the amplitudes are limited between 10 Nts and -10 Nts. The number of learning processes is 48. The learning time steps in each learning process are 1000. The initial point of each learning process is also generated at random. That will reduce the dependence of a constant initial point.

Figure 5 shows the angle trajectory in the 27th trial. Figure 6 is the trajectory of angular velocity during the 27th process. Figure 7 shows the final output of rules after 48 trials. After the learning procedure, we choose four initial states to test the controllability of the newly generated fuzzy controller and take off the learning part of the controller. The result is satisfactory. This controller indeed can achieve the request. Figure 8 shows the angle trajectory of the four tests. And figure 9 shows the detail of the 3rd test at final 200 steps.

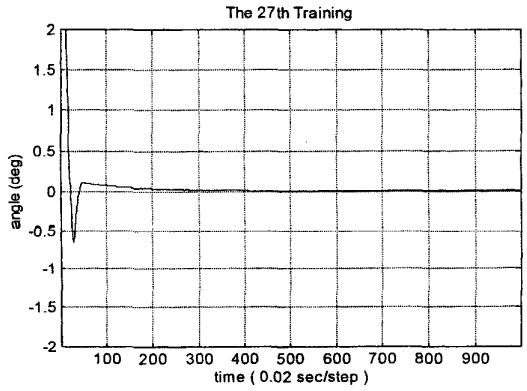

Figure 5. The angle trajectory of the 27th train trial

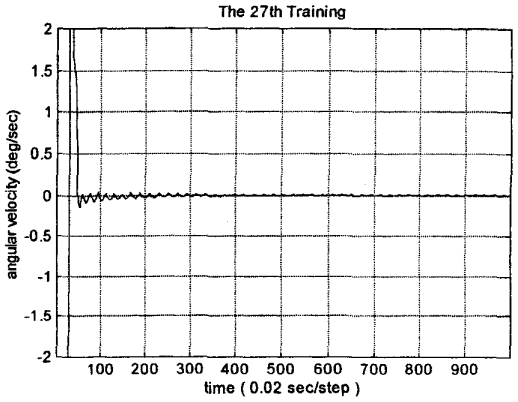

Figure 6. The trajectory of the angular velocity of the 27th training trial

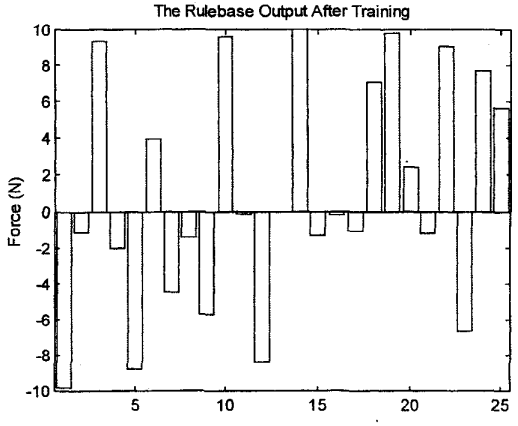

Figure 7. The outputs of rules after training

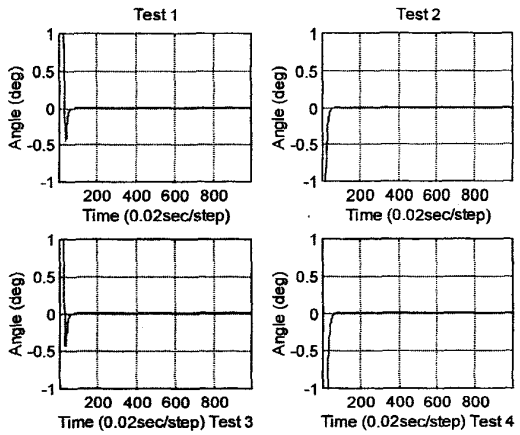

Figure 8. The angle trajectories in the four tests

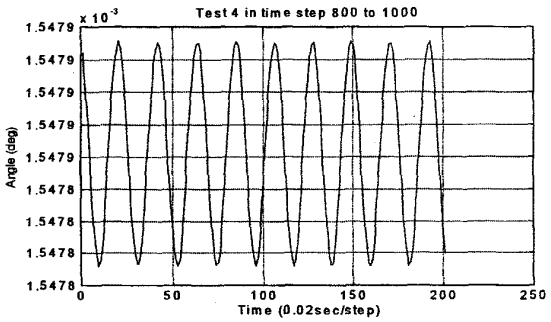

Figure 9. The final 200 steps of the 3rd test

V. Conclusions

The fuzzy supervisor indeed helps the self-learning fuzzy controller to establish a usable rulebase. The performance of the fuzzy controller is good. But there are no mechanisms to inform the controller when it should stop learning. We have to stop learning at a fixed time. So the oscillation around the equilibrium is possible.

The learning process is reasonable, so we can easily tune the learning object as we wish. Moreover, the radius and the location of the cell should be also adjustable. This will be our next object.

In our train, we introduce the history of punishments and rewards into the learning process. But the effects in not clear. We should think more about this in the future.

References

[1] L. A. Zadeh, 'Fuzzy sets', *Information and Control*, vol. 8, pp. 338-353, 1965

[2] E. H. Mamdani, 'Application of fuzzy algorithms for control of simple dynamic plants', *Proc. IEE* 121(12), pp.1585-1588, 1974

[3] D. Michie and R. A. Chambers, 'Boxes' as a model of pattern-formation', in *Toward a Theoretical Biology,* vol. 1, Prolegomena, C. H. Waddington, ed. Edinburgh Univ. Press, 1968, pp. 206-215

[4] A. G. Barto, R. S. Sutton, and C. W. Anderson, 'Neuronlike adaptive elements that can solve difficult learning control problems', *Trans. SMC,* vol. 13, no. 5, 1983, pp. 834-846

[5] C. W. Anderson, 'Strategy learning with multilayer connectionist representation', Tech. Rep. TR87-509.3, GTE Laboratories Inc., May 1988

[6] C. C. Lee, 'Self-learning rule-based controller employing approximate reasoning and neural-net concepts', *Int. J. Intelligent Systems,* vol. 6, pp. 71-93, 1991

[7] H. R. Berenji, 'A reinforcement learning-based architecture for fuzzy logic control', *Int. J. Approximate Reasoning,* no. 6, 1992, pp.267-299

[8] H. R. Berenji and P. Khedkar, 'Learning and tuning fuzzy logic controllers through reinforcements', *IEEE Trans. Neural Networks,* Sep. 1992, pp. 724-740

[9] C.-T. Lin and C.-S. G. Lee, 'Neural-network-based fuzzy control and decision system', *IEEE Trans. Computer,* Dec. 1991, pp.1320-1336

[10] Y.-Y. Chen, K.-Z. Lin, and S.-T. Hsu, 'A self-learning fuzzy controller', *IEEE Internationl Conference on Fuzzy Systems*, San Diego, USA, March 8-12, 1992

[11] C.-F. Perng and Y.-Y. Chen, 'A heuristic criterion algorithm of the self-learning fuzzy controller', NAFIPS '97, Syracuse, NY, USA, September 21-24, 1997