Reinforcement Learning with Perturbation Method to Turn Unidirectional Linear Response Fuzzy Controller for Inverted Pendulum

Hang Yu[†], Yan Wang[†], Huiran Zhang[†] and Zheng Tang[†]

[†]Graduate School of Innovative Life Science, University of Toyama, Toyama, Japan

Summary
In this paper, we present a unidirectional linear response fuzzy controller (FC) to control the inverted pendulum system. The performance of turning fuzzy controller is defined as an evaluation function and our proposed technique, which is based on the integration of reinforcement learning and a perturbation method, is utilized to diversity the search of minimization of the evaluation function. Once the reinforcement learning method results in a local optimal solution for the problem, the perturbation method is implemented, actually a gradient ascent learning phrase is carried out. The two phrases are repeated until a better solution is found. The proposed hybrid learning method is utilized to adapt the parameters in the fuzzy control. Simulation results reveal that the proposed learning system can control the inverted pendulum system better than other traditional systems.

Key words:
Unidirectional linear response, fuzzy controller, reinforcement learning, perturbation

1. Introduction

In recent years, modern industrial plants increase their complexity and demand flexibility that makes the control design difficult. The conventional control system design requires an explicit mathematical model of the plant. Intelligent control can be a practical alternative since control design can be based on the knowledge and experience of human, such as fuzzy logic, neural network. A fuzzy controller is such a combination of neural network and fuzzy logic. It is knowledge-based or rule-based systems which provide a framework for formalising intuition and experiences of human experts and operators [1]. Based on Zadeh's theory of fuzzy sets [2], a typical fuzzy controller [3] contain descriptive if-then rules that are created from human knowledge and heuristics and associated fuzzy sets for mapping real-numbered inputs to outputs, and thus overcome the limitation that classical expert systems may meet because of their inflexible representation of human decision making.

Based on this understanding, fuzzy controllers have successfully been employed for various applications such as information systems, consumer products, industrial process control, medical instrumentation and decision analysis [4-8]. It has proven to be an effective tool for complicated and imprecise processes because it can easily approximate what human experts perform well under such ill-defined environments when no mathematical formulation is able to define [9]. Despite the advantages of fuzzy controllers, its limitation lies in its inability to tune the membership functions effectively, which can be well accommodated by the learning in neural network. The learning capability of fuzzy controller makes it possible to change the membership functions for the current control situation and at the same time retains the rule. Realizing this researchers have discussed the possible combinations of neural network learning in order to improve the fuzzy controller so that it is able to handle extreme situations where the non-learning controller fails [10]. Many learning algorithms have been proposed in order to construct fuzzy controllers automatically [11-14]. However, some of these learning algorithms have proved to be not so efficient such as the evolutionary algorithm because it results in a very long learning process [15].

Reinforcement learning (RL) addresses the problem of an agent that optimizes its reactive policy in a poorly structured and initially unknown environment [16]. Algorithms developed in this area can be viewed as computational processes that transform observations of states, actions and rewards into policy parameters. Several important RL algorithms, such as Q-Learning [17] and Actor critic methods [18], process the data sequentially. Each single observation is used for adjusting the algorithms' parameters and then becomes unavailable for further use, and in general call such methods "sequential". They are based on a common assumption that RL applications to real-world learning control problems require large amounts of data which cannot be kept in a limited amount of memory assigned to the algorithm.

In order to improve the performance of RL, some researchers applied other meta-heuristic methods to implement reinforcement learning in the design of fuzzy controllers. Lin et. al [19] proposed GA-based fuzzy reinforcement learning to control magnetic bearing systems. In the reference of [20], Juang et. al proposed genetic reinforcement learning in designing fuzzy

Manuscript received January 5, 2011
Manuscript revised January 20, 2011

controllers. The GA adopted was based upon traditional symbiotic evolution which, when applied to fuzzy controller design, complements the local mapping property of a fuzzy rule. In [21], Er and Deng proposed dynamic Q-Learning for on-line tuning the fuzzy inference systems. Kaya and Alhajj [22] proposed a novel multi-agent reinforcement learning approach based on fuzzy OLAP association rules mining. However, these approaches encountered one or more of the following major problems: (1) the initial values of the populations were generated randomly; (2) the mutational value was generated by the constant range while the mutation point is also generated randomly; (3) the population sizes always depend on the problem which is to be solved.

In this paper, we propose a permutation method to enhance the performance of RL. The performance of turning FC is defined as an evaluation function and our proposed technique is utilized to diversity the search of minimization of the evaluation function. Initially, RL is implemented to adapt the weights and fuzzy-rules in FC to achieve an ideal state for inverted pendulum. Once the reinforcement learning method results in a local optimal solution, (that is to say, the FC system is not able to control the inverted pendulum any longer) for the problem, the perturbation method is implemented. In fact, the perturbation method performs a gradient ascent learning phrase to enable RL to a new state for further well controlling. The above two phrases are repeated until a better solution is found. The proposed hybrid learning method is utilized to adapt the parameters in the fuzzy control. Simulation results reveal that the proposed learning system can control the inverted pendulum system better than other traditional systems.

The rest of this paper is organized as follows: the unidirectional linear response fuzzy controller is introduced in the next section. In Section 3, we present the inverted pendulum system and its corresponding representation in FC. The proposed hybrid reinforcement learning together with the perturbation method is illustrated in Section 4. Experimental results and the comparative analysis with other traditional learning methods are discussed in Section 5. Finally, we give some general remarks to conclude this paper.

2. Unidirectional Linear Response Fuzzy Controller

There are several types of fuzzy controller in the literature. The main difference during them is the fire function and definition of nodes in the network. In [23], the authors constructed a fuzzy logic controller using simplified table-lookup to build a set of fuzzy rules. Alata et. al [24] used adaptive neuro-fuzzy inference to construct the rules for the fuzzy gain schedule to control an inverted pendulum system. Sakai and his colleague [25] applied a nonlinear optimization method to learn fuzzy control rules for an inverted pendulum system by using the vertor simplex method. A fuzzy logic controller based on the single input rule modules (SIRM) to stabilize an inverted pendulum system can be referred to as in [26].

Different from the typical nodes in neural network which are primarily sigmoid or hard limiters, unidirectional linear response (ULR) works as well as the traditional nodes with some extra features. The advantage of ULR is that it is able to not only retain the significant behaviors of the traditional models, but gives powerful capability of the analog and continuous signal processing [27].

The unidirectional linear response fuzzy controller (ULRFC) is illustrated in Fig. 1. In this figure, the nodes N is defined as the input singles where x_1, x_2 …. x_n is each node as the input with weight of w_1, w_2 …. w_n and the threshold is set at θ. In general, the fire function in ULR can be defined mathematically as in the following.

$$f(u) = \begin{cases} u & u > 0 \\ 0 & u \le 0 \end{cases}$$

$$u = \sum_{i=1}^{n} w_i x_i - \theta$$

This representation of unidirectional can be achieved either in a single bipolar or a diode connected MOS transistor. However when one or more layers are present connecting hidden units or nodes, the ULR multilayer network can be represented as in Fig. 2. The figure shows a ULR multilayer fuzzy controller network with two inputs of x_1 and x_2, four fuzzy if-then rules, local mean-of-maximum (LMOM) and an output of F. It consists of four layers:

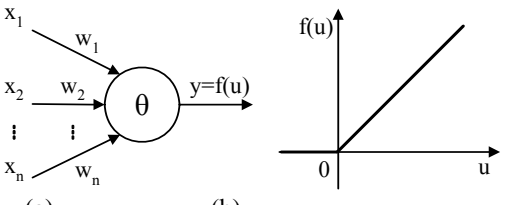


Fig. 1: Definition of unidirectional linear response model.

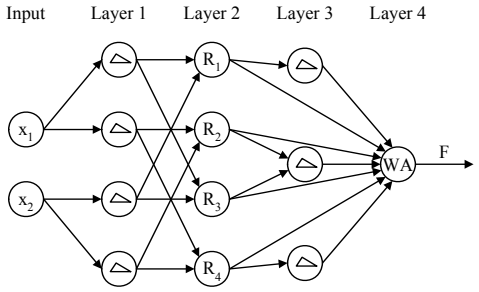


Fig. 2: An ULR multi-layered neural network.

Layer 1: The nodes in this layer share the membership function of $\mu_v(x)$ of V and V can be defined as linguistic label for the node function. V also defines the degree to which the given x satisfies it. The membership function used here is the triangular function (Fig. 3 and Fig. 4).

$$U_{C_V S_{VL} S_{VR}}(x) = \begin{cases} 1 - |x - C_V| / S_{VR} & x \in [C_V, C_V + S_{VR}] \\ 1 - |x - C_V| / S_{VL} & x \in [C_V - S_{VL}, C_V] \\ 0 & otherwise \end{cases}$$

where C_V denotes the center of triangle, S_{VR} represents the right of the center, and S_{VL} denotes the left of the center. Weights are in the center where else the threshold of the network is spread about URL triangular. As both values of this systems change, the membership function too varies accordingly resulting in different forms of membership function. The triangular membership function used in the two-layer ULR network can berepresented as below:

$$w_{A1} = \frac{1}{S_{VL}}$$

$$w_{A2} = \frac{S_{VL} + S_{VR}}{S_{VL} \cdot S_{VR}}$$

$$\theta_{A1} = -\frac{S_{VL} - C_V}{S_{VL}}$$

$$\theta_{A2} = \frac{C_V \cdot (S_{VL} + S_{VR})}{S_{VL} \cdot S_{VR}}$$

Layer 2: Inputs for this layer originate from the previous layer. Besides performing a minimum operation that has been proposed to have a continuous differentiable softmin operation, it is also involved in the if part of the rule. Fig. 5(a) shows how the network achieves the minimum operation in the case of $u_{V1} \le u_{V2}$ when the threshold for the

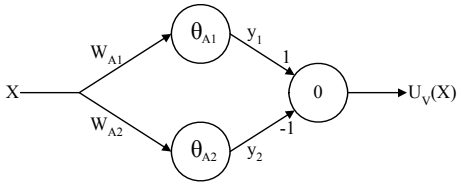


Fig. 3: The details in unidirectional linear response model.

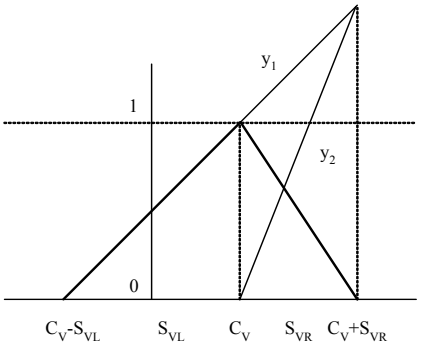


Fig. 4: The membership function.

network is zero and the outputs from layer one are:

$$f_1(u_{V1} - u_{V2}) = 0$$

$$f_2(u_{V1}) = u_{V1}$$

$$Z_r = f(0 + u_{V1}) = u_{V1}$$

For the case of $u_{V1} \ge u_{V2}$:

$$f_1(u_{V1} - u_{V2}) = u_{V1} - u_{V2}$$

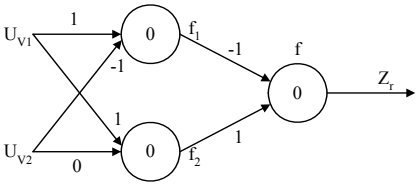
$$f_2(u_{V1}) = u_{V1}$$

$$Z_r = f(-(u_{V1} - u_{V2}) + u_{V1}) = u_{V2}$$

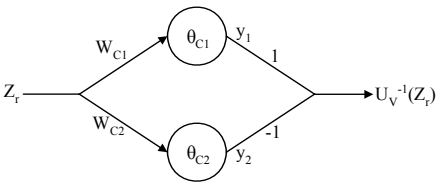
Layer 3: This layer uses a triangular membership function. The specified LMOM method is as such:

$$U^{-1}_{C_V, S_{VL}, S_{VR}}(Z_r) = C_V + \frac{1}{2}(S_{VR} - S_{VL})(1 - Z_r)$$

The output is a limited value of $\mu_v^{-1}(z_r \to 0^*)$ and the membership function is monotic with $\mu_v^{-1}(z_r)$ as the mathematical inverse. When the membership function of defuzzification used is the triangular function, the input output relationship can be represented as $y = ax + b$. Therefore it can be implemented by the network as shown in Fig. 5(b).



(a) Minimum operations



(b) Defuzzification

Fig. 5: ULR network: the definitions in Layer two and Layer three, respectively.

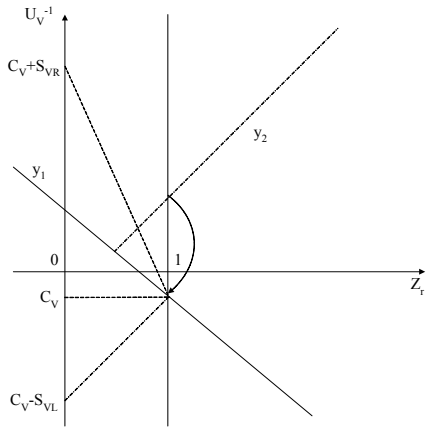


Fig. 6: Defuzzification network response.

Fig. 6 shows the input-output relation of the defuzzification is the inverse of the membership function whereby the output can either be positive or negative. The output will always remain positive as for the URL. Nevertheless, the output of the defuzzification will be positive if $y_1 > 0$ and $y_2 = 0$ and negative if $y_1 = 0$ and $y_2 > 0$ by multiplying y_2 with -1. The response of the defuzzification network is expressed by pulsing y_1 with $-y_2$ while the parameters of the system is determined with Z_r= 0 and 1. The following conditions apply when determining the values of w_{c1}, θ_{c1} for y_1:

$$Z_r = 0 \qquad U_V^{-1}(0) = \frac{1}{2}\left((C_V + S_{VR})(C_V - S_{VL})\right)$$

$$Z_r = 1 \qquad U_V^{-1}(1) = C_V$$

w_{c2}, and θ_{c2} for y_2 are inverse of w_{c1} and θ_{c1}:

$$w_{C1} = -\frac{S_{VR} - S_{VL}}{2}$$

$$\theta_{C1} = -\left(C_V + \frac{S_{VR} - S_{VL}}{2}\right)$$

In order to obtain the value for w_{c2}, and θ_{c2} for y_2:

$$w_{C2} = \frac{S_{VR} - S_{VL}}{2}$$

$$\theta_{C2} = C_V + \frac{S_{VR} - S_{VL}}{2}$$

Layer 4: This is where all the signals coming from layers 3 and 4 are summed as following and the rule is represented as r.

$$F = \frac{\sum_r Z_r U_V^{-1}(Z_r)}{\sum_r Z_r}$$

3. Inverted Pendulum System

The inverted pendulum problem is one of the most important problems in control theory and has been studied excessively in control literatures [28]. It is well established benchmark problem that provides many challenging problems to control design. The system is nonlinear, unstable, non-minimum phase and under-actuated. Because of their nonlinear nature pendulums have maintained their usefulness and they are now used to illustrate many of the ideas emerging in the field of nonlinear control [29]. The challenges of control made the inverted pendulum systems classic tools in control laboratories.

In this paper, the inverted pendulum system (IPS) considered is shown in Fig. 7, which consists of a straight-line rail, a cart, a pendulum and a driving unit. The cart can move left or right on the rail freely. The pendulum is hinged on the center of the top surface of the cart and can rotate around the pivot in the same vertical plane with the rail. It is a non linear and unstable system with one input

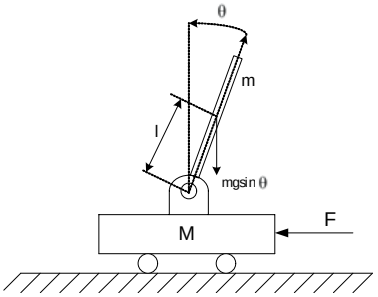


Fig. 7: Configuration of the inverted pendulum system.

and several output signals. The aim is to balance the pendulum vertically on a cart which moves in an uncontrolled state when an initial force is applied. The variables are set at zero when the cart at rest initially. The pendulum is set straight up and has only one degree of freedom. The primary task is to balance the pole and to keep the cart within boundaries by supplying the appropriate force to the cart. The aim is to move the wagon along the x direction to a desired point without the pendulum falling. At a pendulum of x_1 from vertical, gravity produces an angular acceleration equal to x_2, and cart acceleration.

Given that no friction exists in the system, the dynamic equation of the inverted pendulum system can be expressed as:

$$\ddot{\theta} = \frac{g\sin\theta + \cos\theta \dfrac{-F - ml\dot{\theta}^2 \sin\theta}{m_c + m}}{l\left(\dfrac{4}{3} - \dfrac{m\cos^2\theta}{m_c + m}\right)}$$

$$\ddot{x} = \frac{F + ml\left(\dot{\theta}^2 \sin\theta - \ddot{\theta}\cos\theta\right)}{m_c + m}$$

Here, the parameters m_c and m are the mass of the cart and the mass of the pendulum in the unit [kg], in the simulation, they equal to 1.0 and 0.1, respectively. g is the

gravity acceleration. The parameter l is the length from the center of the pendulum to the pivot in the unit [m] and equals to the half length of the pendulum. The variable F represents the driving force in the unit [N] applied horizontally to the cart. The variables θ, $\dot{\theta}$, $\ddot{\theta}$ represent, respectively, the angle of the pendulum from upright position, its angular velocity, its angular acceleration, and the clockwise direction is positive.

4. Reinforcement Learning combined with Perturbation Method

One of the most important breakthroughs in reinforcement

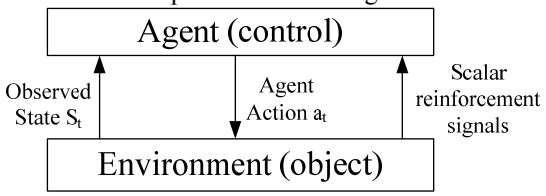


Fig. 8: The framework of reinforcement learning.

learning is the development of an off-policy TD control algorithm known as Q-learning. In reinforcement learning, Q-learning can discover the optimal policy. It is an incremental dynamic programming procedure that determines the optimal policy in a step-by-step manner. Q-learning is an on-line procedure for learning the optimal policy through experience gained solely on the basis of samples of the form: $s_n = (i_n, a_n, j_n, g_n)$, where n denotes discrete time, and each sample Sn consists of a four-tuple described by a trial action an on state in that results in a transition to state $j_n = i_n + 1$ (in denote state at time n) at a cost $g_n = g(i_n, a_n, j_n)$. And it is highly suited for solving Markovian decision problems without explicit knowledge of the transition probabilities. The requirement of using Q-learning successfully is based on the assumption that the state of the environment is fully observable, which in turn means that the environment is a fully observable Markov chain. Nevertheless, if the state of the environment is partially observable, such as the sensor device on the inverted pendulum may be imprecise, special methods are required for discovering the optimal policy. For the purpose of overcoming this problem, a utilization of perturbation method combined with Q-learning as a learning agent will be proposed.

An agent is connected to its environment via perception and action, as depicted in Fig. 8. On each step of interaction the agent receives as input, some indication of the current state, of the environment; the agent then chooses an action, a, to generate as output. The action changes the state of the environment, and the value of this state transition is communicated to the agent through a scalar reinforcement signal. The agent's behavior should choose actions that tend to increase the long-run sum of values of the reinforcement signal. It can learn to do this over time by systematic trial and error, guided by a wide variety of algorithms.

In order to realize an implementation of Q-learning, according to Bellman's optimality criterion combined with value iteration algorithm, the small step-size version formula of Q-learning is described by

$$Q(i,a) := (1-\eta)Q(i,a) + \eta\sum_{j=1}^{N} p_{ij}\left(g(i,a,j) + \gamma \max_{b\in A_j} Q(j,b) \right)$$

where η is a small learning-rate parameter that lies in the range $0 < \eta < 1$. The symbol i denotes sate at time n, j is state at time $n+1$, n is the action been taken at time n, p_{ij} is the transition probability defined as $p_{ij}(a) = P(X_{n+1} = j \mid X_n = i, A_n = a)$. By adjusting γ we are able to control the extent to which the learning system is concerned with long-term versus short-term consequences of its own actions. In the limit, when $\gamma = 0$ the system is myopic in the sense that it is only concerned with immediate consequences of its action. As γ approaches 1, future costs become more important in determining optimal actions. The action is computed as

$$action = \max\big(Q(i,a)\big) \quad (1)$$

And the averaging performed in an iteration over all possible states is replaced by a single sample, there by resulting in the following update for the Q-factor:

$$Q_{n+1}(i,a) = (1-\eta)Q_n(i,a) + \eta\left[\text{reinf} + \gamma \max_{b\in A_j} Q_n(j,b) \right] \quad (2)$$

The total flow chart for implementing Q-learning is illustrated in Fig. 9.

Procedure of reinforcement learning:
Start
Steps 1. Set Q-factor table all entries zero.
2. Initial cart and pole.
3. Get current state.
4. Determine a action according to equation (1).
5. Push cart and get current state.
6. If fail, reinforcement = -1 and reset cart, else reinforcement = 0.
7. Update Q according to equation (2).
8. Repeat 3-7, until the agent learns it to a steady state.
End

Fig. 9: The implementation flow of Q-learning.

In order to improve the control performance of Q-learning, a perturbation method is incorporated into it. There are several such methods to use. The basic idea is inspired from the guided local search [30], [31] which is one of the

methods to help local search escape local optima. The basic principle of this method is to penalize features of the candidate solutions when local search settles in a local optimum. Similar to guided local search, we propose an improved method called objective function adjustment algorithm, to further improve the performance of Q-learning as a perturbation mechanism.

The followings will explain the mechanism of the proposed method. Every term of the objective function multiplies a multiplier and the energy function E is described as

$$E = \lambda_1 f_1(a_1) + \lambda_2 f_2(a_2) + ... + \lambda_n f_n(a_n)$$

where $f_i(.)$ is a term of objective function E and λ_i is a multiplier for each function. In order to explain the adjustment process of this method, we use a conceptual

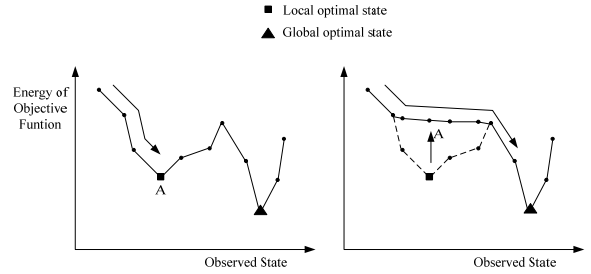


Fig. 10: The situation of escaping from the local optimum value by perturbation.

graph of the energy landscape of the objective function with a local optimum and a global optimum as shown in Fig. 10. The energy function is reflected in the height of the graph. Each position on the energy landscape corresponds to an observed state of the Q-learning. For instance, if the solution of Q-learning has steadily trapped on point A (left graph in Fig. 10), because of the updating procedure of Q-learning made the state move towards negative gradient direction. Then we change the multipliers in gradient ascent direction so as to increase the energy temporarily, and point A will be renewed to a higher energy position of the new energy landscape (right graph in Fig. 10). The gradient ascent learning phase is according to

$$\lambda_i = \lambda_i + \alpha \frac{\partial E}{\partial \lambda_i} \tag{3}$$

After updating E with the new multipliers in the updating phase again, the learning goes down the slope of the valley and reaches a new stable state. The all multipliers are reset to 1 and the newly gained state will decrease and reach a new state. The above search process continues until a new optimum is reached. The total implementation flow can be summarized in Fig. 11.

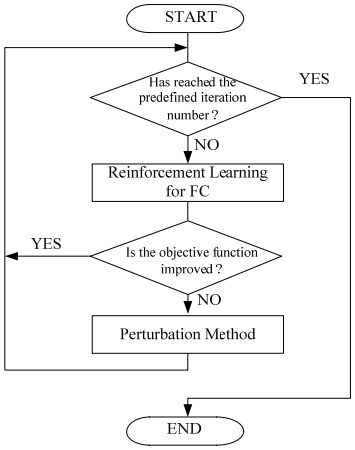


Fig. 11: The flowchart of reinforcement learning combined with perturbation method.

5. Simulation Results and Discussion

The effectiveness is verified by applying the proposed hybrid learning method to the inverted pendulum system. The ULR fuzzy controller is constructed with nine if-then rules as shown in Table 1, modeling after a skilled human operator's knowledge of handling the system. The horizontal axis represents the angle of the pole where else the vertical is the pole velocity with representations such as positive large (PL), positive medium (PM), positive small (PS), negative large (NL), negative medium (NM), negative small (NS) and zero (ZE).

Fig. 12 is the fuzzy controller for the specified system of inverted pendulum. The simulation is carried as offline. In offline state, the fuzzy controller runs for a specified period of time after which the learning is applied to it. When learning is applied, it chooses the next possible local minima and again repeats the cycle using the newly found solution. This cycle is carried out until the best solution is found and the each cycle time is set at 2000ms. The simulation of the problem is carried out in three methods of the temporal backpropagation (BP), the canonical local search (LS) and the proposed improved hybrid learning method. Fig. 13 clearly demonstrates how the member-ship functions have shifted after learning was applied.

Table 1: If-then rules used in ULR fuzzy controller.

θ \ $\dot{\theta}$	PO2	ZE2	NE2
PO1	PL	PM	ZE
ZE1	PS	ZE	NS
NE1	ZE	NM	NL

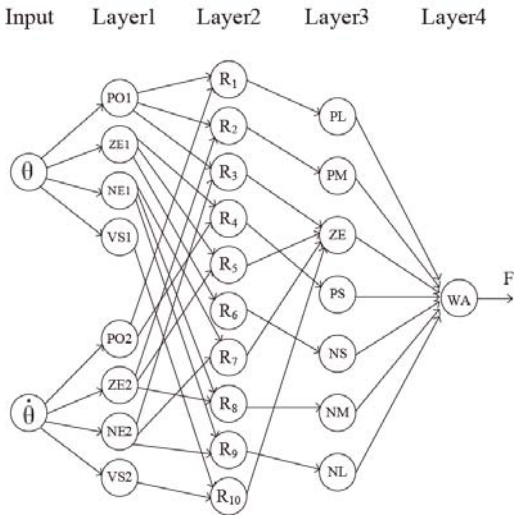
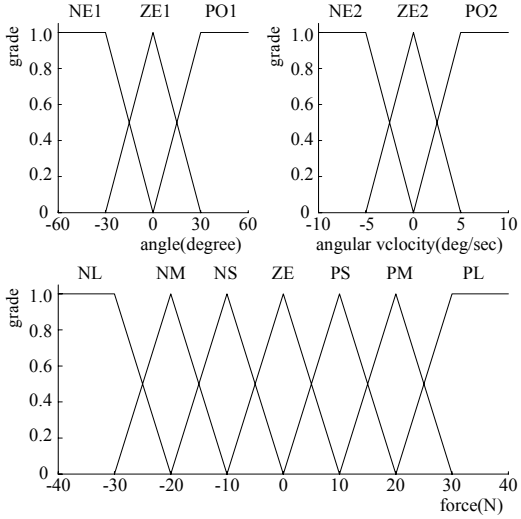
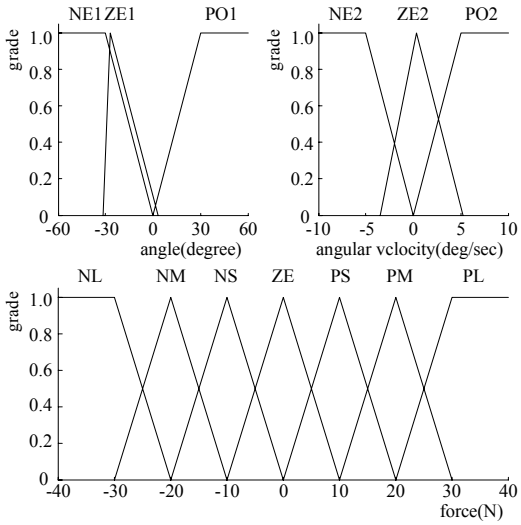


Fig. 12: Fuzzy controller used for the inverted pendulum system.



(a) Before learning state by the hybrid method.



(b) After learning state by the hybrid method.

Fig. 13: Membership functions before and after learning.

Table 2: Initial parameters used in the experiments.

Label	Center	Left Spread	Right Spread
PO1	30.0	30.0	5000.0
ZE1	0.0	30.0	30.0
NE1	-30.0	5000.0	30.0
PO2	5.0	5.0	5000.0
ZE2	0.0	5.0	5.0
NE2	-5.0	5000.0	5.0
PL	30.0	10.0	5000.0
PM	20.0	10.0	10.0
PS	10.0	10.0	10.0
ZE	0.0	10.0	10.0
NS	-10.0	10.0	10.0
NM	-20.0	10.0	10.0
NL	-30.0	10.0	10.0

Table 3: Final values of the system parameters after learned by the proposed method.

Label	Center	Left Spread	Right Spread
PO1	34.9	25.7	5040.0
ZE1	0.8	64.5	75.0
NE1	-35.0	5460.6	34.9
PO2	5.7	2.3	5049.0
ZE2	-0.3	3.8	0.9
NE2	-5.0	1233.1	3.2
PL	45.1	54.0	4940.0
PM	19.7	70.8	24.1
PS	-98.0	4537.0	11.2
ZE	0.5	3.2	12.3
NS	-24.1	7.9	6.5
NM	13.4	24.6	6.8
NL	-28.9	5001.3	21.9

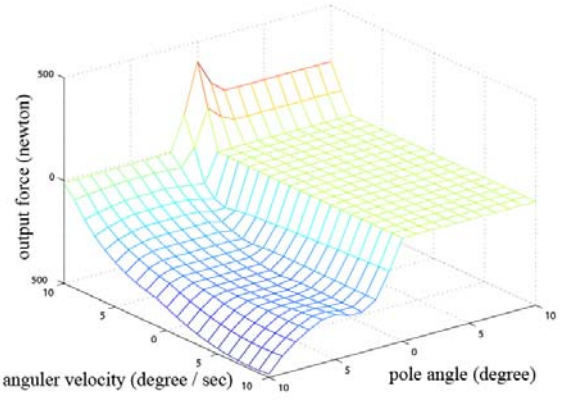


Fig. 14: The 3D surface of the learned parameters by proposed method.

The control parameters of the membership function are shown in Table 2 and Table 3. Table 2 shows the control parameters without any learning while as Table 3 is after learning process. It can be seen that there are some changes to the membership functions after the learning process covering the interval [-10, 10]. In Fig. 14 we showed the whole control performances of the proposed algorithm to demonstrate how the fuzzy controller responds at different initial angles. The simulation shows that the fuzzy controller has no problem in balancing the pole either in small or large angles and this clearly proves the control quality of the proposed learning.

The simulation is also repeated with other learning algorithm such as the conventional reinforcement learning method and the guided local search method. Comparison is also carried out with no learning fuzzy (indicated as

Table 4: Simulation results of the fuzzy controller for several initial angles.

Angle (degree)	Normal	Reinforce-ment Learning (RL)	Guided Local Search (GLS)	Proposed Hybrid Method
10	success	success	success	success
20	fault	success	success	success
30	fault	success	success	success
40	fault	success	success	success
50	fault	success	success	success
60	fault	success	fault	success
70	fault	fault	fault	success
80	fault	fault	fault	success

normal). The obtained results as listed in Table 4 show that among the four methods, the proposed hybrid reinforcement learning combined with perturbation method shows the best performance. By using the proposed learning method, the pole returns to the balance state for even the largest angle. This goes to show that by using the proposed learning, we can avoid the vigorous reinforcement learning and canonical guided local search. During the simulation, a temporary increase in error was noticed and this temporary observation was the result of the changes done to escape a local pit when it comes across.

6. Conclusion

In this paper, we proposed a hybrid learning method by combining the reinforcement learning and a perturbation method for controlling the unidirectional linear response fuzzy controller. Firstly, one of the reinforcement learning methods, i.e. Q-learning was implemented to achieve a steady state. Once the Q-learning method resulted in a local optimal solution for the problem, the perturbation method was implemented; actually a gradient ascent learning phrase was carried out. The two phrases were repeated until a better solution was found. The proposed hybrid learning method was utilized to adapt the parameters in the inverted pendulum system. Simulation results revealed that the proposed learning system could control the inverted pendulum system better than other traditional methods, such as the normal learning, the pure reinforcement learning and the guided local search.

In the future, we plan to apply the proposed hybrid method to other engineering problems, such as the job shop scheduling problems. Furthermore, some other heuristics involving the stochastic jump noisy, the chaotic dynamic mechanism, and the estimation distribution algorithm can also be incorporated into the reinforcement learning to achieve a better performance.

References

[1] H. R. Berenji, “A reinforcement learning-based architecture for fuzzy logic control,” International Journal of Approximate Reasoning, vol.6, no.1, pp.267-292, 1992.
[2] L. A. Zadeh, “Fuzzy sets,” Information and Control, vol.8, pp.338-353, 1965.
[3] S. Kawaji and T. Maeda, “Fuzzy serve control system for an inverted pendulum,” Proceedings of IFES’91, vol.2, pp.812-823, 1991.
[4] S. Shao, “Fuzzy self-organizing controller and its application for dynamic processes,” Fuzzy Sets and Systems vol.26, pp.151-164, 1988.
[5] T. Culliere, A. Titli and J. Corrieu, “Neuro-fuzzy modeling of nonlinear systems for control purposes,” Proceedings of the IEEE International Conference on Fuzzy Systems, pp.2009-2016, 1995.
[6] N. Bridgett, J. Brandt and C. Harris, “A neuro-fuzzy route to breast cancer diagnosis and treatment,” Proceedings of the IEEE International Conference on Fuzzy Svstems, pp. 641-648, 1995.
[7] R. Kruse, J. Gebhardt, R. Palm, Fuzzy systems in computer science, Vieweg Braunschweig, 1994.
[8] J. Hollatz, “Neuro-fuzzy in legal reasoning,” Proceedings of the IEEE International Conference on Fuzzy Systems, pp.655-662, 1995.
[9] E. H. Mandani, “Application of fuzzy algorithms for control of simple dynamic plant,” IEEE Proc. Control Sci. vol.121, no.12, pp.1585-1588, 1974.
[10] D. Nauck, R. Kruse, “Interpreting changes in the fuzzy sets of a self-adaptive neural fuzzy controller,” Proc. 2nd Int. Workshop in Industrial Fuzzy Control and Intelligent Systems College Station, pp.146- 152, 1992.
[11] C. T. Lin, C. S. G. Lee, Neural fuzzy systems: a neuro-fuzzy synergism to intelligent system, Prentice-Hall, NJ, 1996.

[12] J. S. R. Jang, "ANFIS: adaptive-network-based fuzzy inference system," IEEE Trans. Syst. Man Cybern., vol.23, pp.665-685, 1993.
[13] C. F. Juang, C. T. Lin, "A self-constructing neural fuzzy inference network and its applications," IEEE Trans. Fuzzy Syst., vol.6, no.1, pp.12-31, 1998.
[14] F.J. Lin, C.H. Lin, P.H. Shen, "Self-constructing fuzzy neural network speed controller for permanent-magnet synchronous motor drive," IEEE Trans. Fuzzy Syst., vol.9, no.5 pp.751-759, 2001.
[15] A. Homaifar, Ed. McCormick, "Simultaneous design of membership functions and rule sets for fuzzy controllers using genetic algorithms," IEEE Trans. Fuzzy Syst., vol.3, no.2, pp.129-139, 1995.
[16] R. S. Sutton, and A. G. Barto, Reinforcement learning: an introduction. Cambridge, MA, MIT Press, 1998.
[17] C. Watkins, and P. Dayan, "Q-learning," Machine Learning, vol.8, pp.279-292, 1992.
[18] V. Konda, and J. Tsitsiklis, "Actor critic algorithms," SIAM Journal on Control and Optimization, vol.42, no.4, pp.1143-1166, 2003.
[19] C. T. Lin, and C. P. Jo, "GA-based fuzzy reinforcement learning for control of a magnetic bearing system," IEEE Trans. Syst. Man Cybern. B, vol.30, no.2, pp.276–289, 2000.
[20] C. F. Juang, J. Y. Lin, and C. T. Lin, "Genetic reinforcement learning through symbiotic evolution for fuzzy controller design," IEEE Trans. Syst. Man Cybern. B, vol.30, no.2, pp.290–302, 2000.
[21] M. J. Er, and C. Deng, "Online tuning of fuzzy inference systems using dynamic Q-Learning," IEEE Trans. Syst. Man Cybern. B, vol. 34, no.3, pp.1478–1489, 2004.
[22] M. Kaya, and R. Alhajj, "Fuzzy OLAP association rules mining-based modular reinforcement learning approach for multiagent systems," IEEE Trans. Syst. Man Cybern. B, vol.35 , no.2, pp.326–338, 2005.
[23] Z. H. Xu, D. M. Jin, and Z. J. Li, "Using learning samples to construct fuzzy logic system with the application to inverted pendulum system," Proc. Int. Conf. on Machine Learning and Cyber., vol.2, pp.1085-1088, 2002.
[24] M. Alata, and K. Demirli, "Fuzzy control and gain scheduling-case study: robust stabilization of an inverted pendulum," IFSA world congress and 20th Int. Conf. NAFIPS, vol.5, pp.3015-3020, 2001.
[25] S. Sakai, and T. Takahama, "Learning fuzzy control rule by vertor simplex method," IFSA world congress and 20th Int. Conf. NAFIPS, vol.5, pp.2541-2546, 2001
[26] J. Yi, and N. Yubazaki, "Stabilization fuzzy control of inverted pendulum systems," AI Eng., vol.14, no.2, pp.153-163, 2000.
[27] Z. Tang, O. Ishizuka and H. Matsumoto, "A Model of Neurons with Unidirectional Linear Response," IEICE Trans. Fundamentals, vol.E76-A, no.9, pp.1537-1540, 1993.
[28] J. Yi, and N. Yubazaki, "Stabilization fuzzy control of inverted pendulum systems," Artificial Intelligence in Engineering, vol.14, pp.153-163, 2000.
[29] K. J. Astrom, and K. Furuta, "Swinging up a pendulum by energy control," Automatica, vol. 36, no.2, pp.287-295, 2000.
[30] E. Tsang, and C. Voudouris, "Fast local search and guided local search and their application to British Telecom's workforce scheduling problem," Oper. Res. Lett., vol.20, pp.119-127, 1997.
[31] C. Voudouris, and E. Tsang, "Guided local search and its application to the traveling salesman problem," Eur. J. Oper. Res., vol.113, pp.469-499, 1998.

Yu Hang received the B.S. degree from Jiangxi University of Finance and Economics, Jiangxi, China in 2006 and an M.S. degree from University of Toyama, Toyama, Japan in 2009. Now he is working towards the D.E. degree at University of Toyama, Toyama, Japan. His main research interests are intelligence computing and neural networks.