Learning Type PID Control System Using Input Dependence Reinforcement Scheme

Hideharu Sawada* (Waseda University), Ji-Sun Shin (Waseda University)

Fumihiro Shoji (Fukuoka Institute of Technology), Hee-Hyol Lee (Waseda University)

Abstract : PID control has widely used in the field of process control and a lot of methods have been used to design PID parameters. When the characteristic values of a controlled object are changed due to a change over the years or disturbance, the skilled operators observe the feature of the controlled responses and adjust the PID parameters using their knowledge and know-how, and a lot of labors are required to do it. In this research, we design a learning type PID control system using the stochastic automaton with learning function, namely learning automaton, which can autonomously adjust the control parameters updating the state transition probability using relative amount of controlled error. We show the effectiveness of the proposed learning type PID control system by simulations.

Keywords : Learning automaton , Learning control , PID control , State transition probability

I. INTRODUCTION

PID control has been widely used in the field of process control and a lot of methods have been used to design PID parameters. When the characteristic values of a controlled object is changed due to a change over the years or disturbance, the skilled operators observe the feature of the control responses and adjust the PID parameters using their knowledge and know-how, and a lot of labors are required to do it. In this research, we design a learning type PID control system using stochastic automaton with learning function, namely learning automaton, which can autonomously adjust the control parameters updating the state transition probability a using relative amount of control error. We show the effectiveness of the proposed learning type PID control system.

II. Learning Automaton

The Learning Automaton (*LA*) is composed of a Stochastic Automaton (*SA*) and an unknown random environment illustrated in Fig.1.

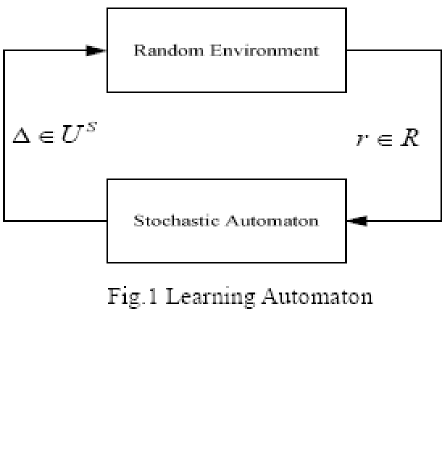


Fig.1 Learning Automaton

The stochastic automaton is a kind of discrete stochastic system. The stochastic automaton SA^S for each PID parameter is defined as follows:

$$SA^s = \langle E, Q^s, U^s, F^s, O^s \rangle \tag{2.1}$$

m-dimensional finite input set:

$$E = \{ e_1, e_2, \dots e_m \} \tag{2.2}$$

n-dimentional finite state set:

$$Q^s = \{ q_1^s, q_2^s, \dots q_n^s \} \quad (s = P, I, D) \tag{2.3}$$

n-dimensional finite output set:

$$U^s = \{ \Delta_1^s, \Delta_2^s, \dots \Delta_n^s \} \tag{2.4}$$

F^S is a function that decides the following state :

$$q^s(k) = F^s[q^s(k-1), e(k-1)] \tag{2.5}$$

where the function F^S is formed by a state transition probability:

$$p_{ij}^l = P\left\{ q^s(k) = q_j^s \mid q^s(k-1) = q_i^s, e(k-1) = e_l \right\} \tag{2.6}$$

the function O^S is an output function:

$$\Delta^s(k) = O^s[q^s(k)] \tag{2.7}$$

III. Learning Scheme of Learning Automaton

1. Learning Algorithm

A. Performance Criterion

The optimal control value $\Delta^{*s} \in \{ \Delta_1^s, \Delta_2^s, \dots \Delta_n^s \}$ satisfies the following performance criterion,

$$E\{ \xi \mid e_l, \Delta_i^{*s} \} = \min_{\Delta_h^s} [E\{ \xi \mid e_l, \Delta_h^s \}] \tag{3.1}$$

where $\xi = \varphi(e_l, \Delta_i^s, e_g)$, Δ_h^s is all of the outputs, e_l is an input to SA^S at the time $(k-1)$, Δ_i^s is output at the time $(k-1)$, and e_g is an input at the time k.

B. Estimation of Performance Criterion

Based on $e(k-1) = e_l$ and an increment of the parameter $\Delta^s(k-1) = \Delta_i^s$ at the time $(k-1)$, the performance criterion is estimated as:

$$\begin{cases} \hat{E}_k\{\,\xi \mid e_l, \Delta_i^s\,\} \triangleq \dfrac{N_{li} \cdot \hat{E}_{k-1}\{\,\xi \mid e_l, \Delta_i^s\,\} + \xi}{N_{li}+1} & (3.2) \\ \hat{E}_k\{\,\xi \mid e_l, \Delta_i^{'s}\,\} \triangleq \hat{E}_{k-1}\{\,\xi \mid e_l, \Delta_i^{'s}\,\} & (3.3) \end{cases}$$

where Δ_i^s is an output at the time $(k-1)$ and $\Delta_i^{'s}$ is the $(m-1)$th output set expect Δ_i^s. N_{li} is the number of appearances of input e_l and output Δ_i^s in the past. The estimation of the performance criterion for output Δ_i^s is updated by eq. (3.2) and the estimation for another output $\Delta_i^{'s}$ is leaved as eq. (3.3). The important function of the stochastic automaton is to receive little penalty response from random environment. The stochastic automaton approaches the optimum value by updating own state transition probability obtaining the penalty level based on the response of own output from random environment.

C. Update of State Transition Probability

The state $p_{ij}^l(k)$ at the time k is selected as:

$$p_{ij}^l(k) = \alpha^s p_{ij}^l(k-1) + (1-\alpha^s)\lambda^s(e_l, \Delta_i^s) \quad (3.4)$$

$$\lambda^s(e_l, \Delta_i^s) \begin{cases} 1: \hat{E}_k\{\xi|e_l, \Delta_i^s\} = \min\limits_{\Delta_i^s} \left[\hat{E}_k\{\xi|e_l, \Delta_i^{'s}\}\right] \\ 0: \hat{E}_k\{\xi|e_l, \Delta_i^s\} \neq \min\limits_{\Delta_i^s} \left[\hat{E}_k\{\xi|e_l, \Delta_i^{'s}\}\right] \end{cases} \quad (3.5)$$

The state $p_{ij}^l(k)$ at the time k is updated whether the estimation of performance criterion approaches optimal value (reward) or not (penalty). If the stochastic automaton chooses appropriate output, it can receive a little penalty response from random environment in average significance. Consequently, most little penalty response of the stochastic automaton output is known to satisfy.

$$p\left[\lim_{k\to\infty} p[\Delta^{*s}] = 1\right] = 1 \quad (3.6)$$

On the other hand, another state transition probability updated as follow.

$$\sum_{*=1}^{m} p_{i*}^l(k) = 1 \quad (3.7)$$

D. Selection of State and Output

The state $q^s(k)$ based on $e(k-1) = e_l$ is selected as follows:

$$p_{i,jm}^l(k) = \max_j p_{ij}^l(k) \quad (3.8)$$

$$q^s(k) = q_{jm}^s(k) \quad (3.9)$$

Using of these equations, the state $q^s(k-1)$ is shifted to the state $q^s(k) = q_{jm}^s$, and the output $\Delta^s(k) = \Delta^{*s}$ from the stochastic automaton is added to the PID controller as their increment.

$$\Delta^{*s} = O^s\left[q_{jm}^s(k)\right] \quad (3.10)$$

These procedures are repeated, and then the probability of obtaining the optimal output Δ^{*s} approaches to 1 by the self-learning of the *SA*.

2. Reinforcement Schemes

A. Linear Reward-Inaction Scheme

The linear reward-inaction scheme has the most simple reinforcement scheme.

Reward:

$$\begin{cases} p_{ij}^l(k) = (1-\alpha^s)\cdot p_{ij}^l(k-1) + \alpha^s \\ p_{i*}^l(k) = (1-\alpha^s)\cdot p_{i*}^l(k-1) \end{cases} \quad (3.11)$$

Penalty:

$$\begin{cases} p_{ij}^l(k) = p_{ij}^l(k-1) \\ p_{i*}^l(k) = p_{i*}^l(k-1) \end{cases} \quad (3.12)$$

$$* \neq j\,, \quad * = 1,2,\ldots,n$$

where α^s is learning parameter, $0 < \alpha^s < 1$.

The linear reward-inaction scheme has been proven to be absolute expediency. It does not update state transition probability when the response from random environment is penalty. In construct, it increase state transition probability p_{ij}^l and it decrease $p_{i*}^l(* \neq j)$ same amount when response from random environment is reward.

B. Linear Reward-Penalty Scheme

The reinforcement schemes were needed to update state transition probability on penalty response and to satisfy the absolute expediency condition.

The linear reward-penalty scheme is the most famous reinforcement scheme, which updates state transition probability as follows:

Reward:

$$\begin{cases} p_{ij}^l(k) = p_{ij}^l(k-1) + \alpha^s\cdot\left[1 - p_{ij}^l(k-1)\right] \\ p_{i*}^l(k) = (1-\alpha^s)\cdot p_{i*}^l(k-1) \end{cases} \quad (3.13)$$

Penalty:

$$\begin{cases} p_{ij}^l(k) = (1-\beta^s)\cdot p_{ij}^l(k-1) \\ p_{i*}^l(k) = (1-\beta^s)\cdot p_{i*}^l(k-1) + \frac{\beta^s}{n-1} \end{cases} \quad (3.14)$$

$$* \neq j\,, \quad * = 1,2,\ldots,n$$

where β^s is learning parameter, $0 < \beta^s < 1$.

C. Input Dependence Reinforcement Scheme

The input dependence reinforcement scheme can update the state transition probability using relative amount of control error.

The input dependence reinforcement scheme updates the state transition probability as follows:

Reward:

$$\begin{cases} p_{ij}^l(k) = p_{ij}^l(k-1) + \left\{\alpha^s + \left|g - \frac{m+1}{2}\right| \cdot \alpha^{s'}\right\} \cdot \left[1 - p_{ij}^l(k-1)\right] \\ p_{i*}^l(k) = \left[1 - \left\{\alpha^s + \left|g - \frac{m+1}{2}\right| \cdot \alpha^{s'}\right\}\right] \cdot p_{i*}^l(k-1) \end{cases} \tag{3.15}$$

Penalty:

$$\begin{cases} p_{ij}^l(k) = \left[1 - \left\{\beta^s + \left|g - \frac{m+1}{2}\right| \cdot \beta^{s'}\right\}\right] \cdot p_{ij}^l(k-1) \\ p_{i*}^l(k) = \left[1 - \left\{\beta^s + \left|g - \frac{m+1}{2}\right| \cdot \beta^{s'}\right\}\right] \cdot p_{i*}^l(k-1) \\ \qquad + \left\{\beta^s + \left|g - \frac{m+1}{2}\right| \cdot \beta^{s'}\right\}/(n-1) \\ \qquad\qquad * \neq j,\ * = 1,2,\dots,n \end{cases} \tag{3.16}$$

where $p_{ij}^l(k-1)$ is transition probability of state j from state i at the time $(k-1)$, and g is in the $e(k) = e_g$ at the time k, which denotes relative position from the middle value $(m+1)/2$ of the error signal.

IV. Composition of Learning Type PID Control System

The input to the stochastic automaton is the controlled error e, and the output from the stochastic automaton operates on three control operations of the PID control system, and the stochastic automaton updates those PID parameters by the reinforcement algorithm. The stochastic automaton carries out the tuning of the PID parameters as shown in Fig.2 according to the learning function of stochastic automaton.

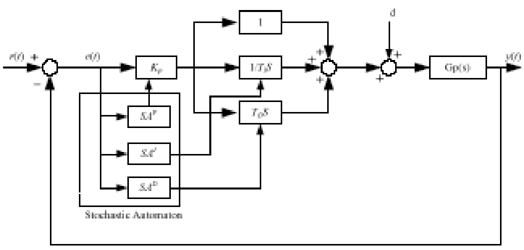


Fig.2 Configuration of Learning Type PID Control System

The updating equation of the PID parameters K_P, T_I, T_D are shown in eq (3.17).

$$\begin{cases} K_P = K_P^0 \cdot \left(1 + \gamma^P \Delta^P(k)\right) \\ T_I = T_I^0 \cdot \left(1 + \gamma^I \Delta^I(k)\right) \\ T_D = T_D^0 \cdot \left(1 + \gamma^D \Delta^D(k)\right) \end{cases} \tag{3.17}$$

$$0 < \gamma^s < 1$$

where initial optimal values K_P^0, T_I^0, T_D^0 of the PID parameters are designed for nominal values of the controlled object, and $\Delta^s(k)$ is the incremental quantity of each parameters, which is an output of stochastic automaton. PID parameters K_P, T_I, T_D are adjusted autonomously by the learning automaton.

On the stochastic automaton with the finite state, the number of finite input set e is m =11 and the number of finite output set Δ^s is n = 21, respectively, in this simulation:

$E = \{e_1, e_2, \dots, e_{11}\}$

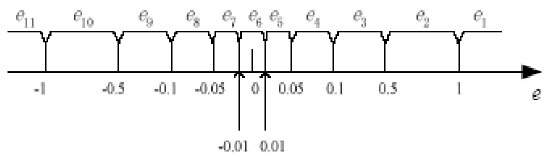


Fig.3. Finite Input-Set of Stochastic Automaton SA^s

$U^s = \{\Delta_1^s, \Delta_2^s, \dots, \Delta_{21}^s\}$

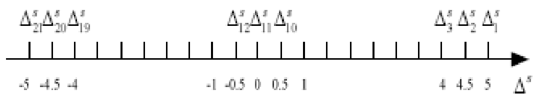


Fig.4. Finite Output-Set of Stochastic Automaton SA^s

1. Controlled object with the characteristic values changed

The characteristic values K, T, L of the controlled object $G_P(s) = Ke^{-Ls}/(Ts+1)$ are changed at the time shown in Fig.5. As an example, the characteristic values of a controlled object are changed as that; $K: 5 \to 9.2$, $T: 10 \to 9.2$, $L: 2 \to 4.5$; so that $G_P(s) = 5e^{-2s}/(10s+1)$ is changed to $G_P(s) = 9.2e^{-4.5s}/(9.2s+1)$.

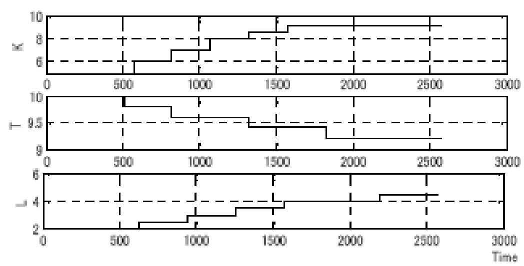


Fig.5 Variation Characteristic of Controlled System

2. Simulation of Stochastic Automaton by Input Dependence Reinforcement Scheme

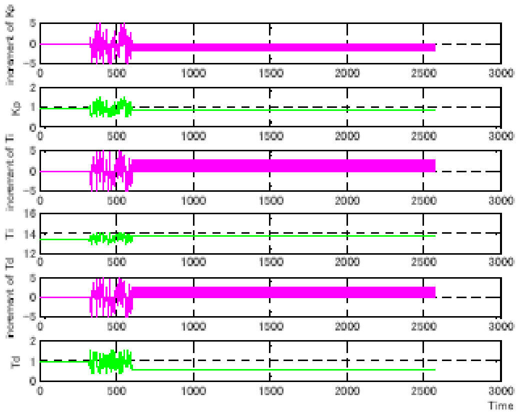
The learning parameters of reward α^s and penalty β^s are setting as follows:

$$\alpha^s = 0.08, \alpha^{s'} = 0.008, \beta^s = 0.03, \beta^{s'} = 0.003$$

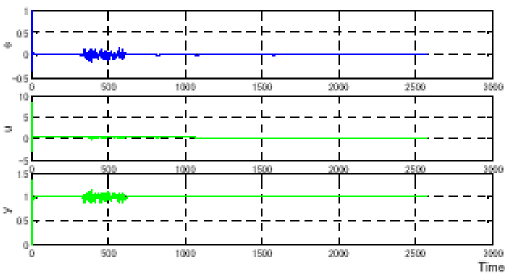
and parameters γ^s for K_P, T_I, T_D are setting as follows:

$$\gamma^P = 0.0975,\ \gamma^I = 0.008,\ \gamma^D = 0.115$$

The simulation results are shown in Fig.6. Fig.6(a) is output of the stochastic automaton, and then, PID parameters K_P , T_I , T_D are changed by the stochastic automaton. The controlled error input for stochastic automaton, manipulated variable *u*, and output *y* are shown in Fig.6(b).



(a) Output of Stochastic Automaton



(b) Controlled error input, manipulated variable, and output

Fig.6 Results by Stochastic Automaton with the Input Dependence Reinforcement Scheme

Fig.7 shows a comparison of the output of learning type PID control system using input dependence reinforcement scheme, linear reward-inaction scheme and linear reward-penalty scheme.

Each parameter of the *SA* is as follows:

- Linear Reward-Inaction Scheme

$$\alpha^P = 0.8,\ \alpha^I = 0.8,\ \alpha^D = 0.8$$
$$\gamma^P = 0.159,\ \gamma^I = 0.008,\ \gamma^D = 0.115$$

- Linear Reward-Penalty Scheme

$$\alpha^P = 0.8,\ \alpha^I = 0.8,\ \alpha^D = 0.8$$
$$\gamma^P = 0.159,\ \gamma^I = 0.008,\ \gamma^D = 0.115$$

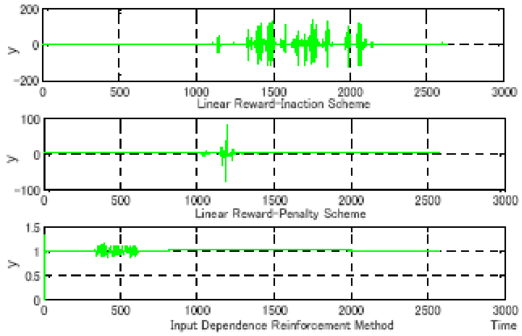


Fig.7 Comparing of Learning Scheme

The upper is the linear reward-inaction scheme, the middle is linear reward-penalty scheme, and the lower is input dependence reinforcement scheme. The linear reward-penalty scheme is unstable from 1000 to 1500. In the linear reward-inaction scheme, there are continuous vibrations from 1000 to 2000, and it takes time to convergence. On the other hand, the proposed method has very little vibration and stable for characteristic values changed.

V. CONCLUSION

This paper discussed about the linear reward-penalty scheme and the linear reward-penalty scheme, and the proposed input dependence reinforcement scheme which updates the state transition probability using amount of controlled error. And we showed the implementation of the stochastic automaton for the PID control system would enable self-modification of the PID parameter when the characteristic values are changed. Through the simulations, the effectiveness of the proposed method was shown.

REFERENCES

[1] M.A.L.Thathachar : "Varieties of Learning Automata : An Overview", IEEE Transactions On Systems,Men and Cybernetics-Part B : Cybernetics, Vol.32, No.6.(2002)

[2] M. Hara, K. Abe : "Decentralized Learning Control of Markov Decision Processes", IEICE D-2, Vol.J80-D-2, No.1, p.363-366 (1997)

[3] Y. Mogami, N. Baba : "Variable Hierarchical Structure Learning Automata Operating in a Non-stationary Multiteacher Environment", T IEE Japan, D-2, Vol.J74-D-I, No.7 pp.401-408 (1991)

[4] X.Zeng, J.Zhou, C.Vasseur : " A strategy for controlling nonlinear systems using learning automaton", Automatica, Vol.36, No. pp.1517-1524 (2000)