Memoirs of the Faculty of Engineering, Okayama University, Vol.37, No.2, pp.29-44, March 2003

Gain Tuning Model of Human Expert for Looper Height Control in Hot Strip Rolling

Shuya Imajo *
Division of Electronic and
Information System Engineering
Graduate School of Natural Science and Technology
Okayama University
3-1-1, Tsushima-Naka Okayama, 700-8530

Masami Konishi
Dept. of Electrical and Electronic Engineering
Okayama University
3-1-1, Tsushima-Naka Okayama, 700-8530

Jun Imai
Dept. of Electrical and Electronic Engineering
Okayama University
3-1-1, Tsushima-Naka Okayama, 700-8530

Tatsushi Nishi
Dept. of Electrical and Electronic Engineering
Okayama University
3-1-1, Tsushima-Naka Okayama, 700-8530

In hot strip rolling mills, the looper control system is automated. However, the looper's behavior tends to be unstable in threading. Therefore, human expert always intervenes and stabilizes the looper's behavior by tuning PID gains and interposing manipulation variable of looper control system. In this paper, we propose a method based on the recurrent neural network to express PID gains tuning action by human. Furthermore, we propose two methods to update the model by learning. To check the effectiveness of the proposed learning methods, numerical simulation applied to the looper height control is carried out.

1 Introduction

In these years, plant control systems are being highly automated. But both the performances of plant and controller change with the passage of time, so it is necessary to tune them. This is why human experts intervene and tune the control system to improve the total plant performances.

In this paper, it is targeted especially to tune looper control gain in threading of hot strip mills. In hot strip mills, the looper height control system is set up among stands. By use of them, the rolling operation are kept to be stable. The looper control system is governed by PID controller. Usually PID gains are determined by gain table according to a variety of strip kinds, but the optimal gain varies with time due to the deterioration of plant characteristics. Now, it is often the case that the skillful operators tune these gains. It is necessary to tune the PID gains optimal in order to stabilize the looper's behavior. In the past, the looper control in the steady state has been widely built up as the control technique [1]. Contrary to this, in threading condition of the looper's behavior tends to unstable, and then human expert never fail to intervene and stabilize the looper's behavior by tuning PID gains and interposing manipulated variable of looper height control system.

In general, it is difficult to automatize human's action because of flexibility and diversity of human operation not only in hot strip rolling but also in other objects. In this paper, we try to automatize PID gains tuning action of human. That is, the method of using recurrent neural networks (RNN) [2][3][4] is proposed. Because human can memorize the past tuning whether old or new, human's tuning action depends on past tuning experiences. RNN is a simple mechanism generating its outputs depending on the past data. Firstly, hu-

*imajo@cntr.elec.okayama-u.ac.jp

man tuned PID gains of looper control system. These data are stored in the data base as the training data of RNN model. Then RNN is trained by the tuning data, and the results are shown. Furthermore, we propose methods to update the RNN model by using learning algorithm. The algorithm is checked by the numerical examples.

2 Looper Height Control in Hot Strip Rolling

The looper angle is determined by the difference of strip velocities between adjacent rolling stands. In the following, the mathematical model of looper control in hot strip rolling, strip thickness control and looper control are described.

2.1 Mathematical Model for Hot Rolling Mills

The fundamental equations for rolling phenomenon are described as follows

$$P_i = \sigma\sqrt{R_i'(H_i - h_i)}Q_i \quad (1)$$

$$Q_i = \sqrt{\frac{h_i}{H_i - h_i}}\left\{\frac{\pi}{2}\arctan\sqrt{\frac{H_i - h_i}{h_i}} - \sqrt{\frac{R_i'}{h_i}}\ln\frac{h_{\phi i}}{h_i} + \frac{1}{2}\sqrt{\frac{R_i'}{h_i}}\ln\frac{H_i}{h_i}\right\} - \frac{\pi}{4} \quad (2)$$

$$h_{\phi i} = \frac{h_i}{\cos^2 F_i} \quad (3)$$

$$F_i = \frac{1}{2}\arctan\sqrt{\frac{H_i - h_i}{h_i}} - \frac{\pi}{8}\sqrt{\frac{h_i}{R_i'}}\ln\frac{H_i}{h_i} \quad (4)$$

$$R_i' = R_i\left(1 + \frac{cP_i}{H_i - h_i}\right) \quad (c = 0.214 \times 10^{-3}) \quad (5)$$

$$\phi_i = \sqrt{\frac{h_i}{R_i'}}\tan F_i \quad (6)$$

$$f_i = \frac{1}{2}\phi_i^2\left(\frac{2R_i'}{h_i} - 1\right) \quad (7)$$

$$b_i = \frac{h_i}{H_i}(1 + f_i) - 1 \quad (8)$$

$$h_i = s_i + \frac{P_i}{M_i} \quad (9)$$

where P_i is rolling force, σ is flow stress, H_i is entry strip thickness, h_i is exit strip thickness, R_i is roll radius, s_i is roll gap, f_i is forward slip, b_i is backward slip, M_i is mill modulus, and subscript i denotes i^{th} stand.

Using f_i and b_i led by solving above equations, exit strip speed v_{fi} and entry strip speed v_{bi} can be obtained as follows

$$v_{fi} = V_{Ri}(1 + f_i) \quad (10)$$

$$v_{bi} = V_{Ri}(1 + b_i) \quad (11)$$

where V_{Ri} is the rolling velocity of i^{th} mill.

2.2 Mathematical Model of Looper System

As shown in Fig.1, the distance between mills is $L = L_1 + L_2$, exit strip speed of i^{th} mill is v_{fi} ,and entry strip speed of $(i+1)^{th}$ mill is v_{bi+1}.

The strip loop angle θ can be obtained as follows

$$\sec\theta = 1 + \frac{1}{L}\int_0^t (v_{fi} - v_{bi+1})dt \quad (12)$$

where the integral is started when the strip gets into $(i+1)^{th}$ mill.

The strip loop figure can be approximated as isosceles triangle. That is, if $\theta \simeq \theta'$, and looper angle α is given by

$$\sin\alpha = \frac{1}{l}\left(\frac{L}{2}\tan\theta + \delta - r\right) \quad (13)$$

The dimensions of looper are shown in Table.1.

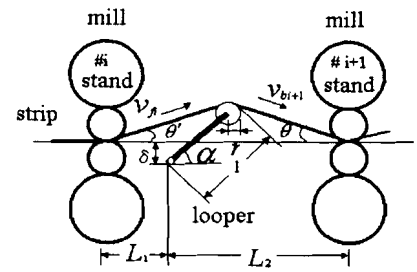


Fig.1: Looper

Table.1: Dimensions of Looper

symbol	dimension [mm]
l	600
δ	150
L_1	2100
L_2	3400
r	100

2.3 Mass Flow Rule

Mass flow rule is used in order to determine set point of rolling velocity. Mass flow rule can be described as

$$h_1^o V_{R1}(1+f_1) = h_2^o V_{R2}(1+f_2) = h_3^o V_{R3}(1+f_3) \quad (14)$$

where h_i^o is the desired value of strip thickness at i^{th} stand.

2.4 Strip Thickness Control

Gauge meter equation is written by

$$h_i = s_i + \frac{P_i}{M} \quad (15)$$

where h_i, s_i, P_i and M are exit strip thickness, roll gap, rolling force and mill modulus.

The control scheme of Gauge Meter AGC(Automatic Gauge Control) is given by

$$\Delta h_i = h_i - h_i^o \quad (16)$$

$$\Delta s_i = -k_i \Delta h_i \quad (17)$$

where k_i is AGC gain. AGC is started to operate when head end of strip arrives at the next stand.

2.5 PID Control of Looper Angle

PID control is described as

$$m(t) = K_p\left(e(t) + \frac{1}{T_i}\int e(t)dt + T_d\frac{de(t)}{dt}\right) \quad (18)$$

where m(t) is manipulated variable and in this case V_{Ri}, e(t) is error and in this case looper deviation, K_p is proportional gain, T_i is integral time, and T_d is derivative time of PID control.

In simulator, PID control is used with velocity form algorithm

$$V_{Ri} = V_{Ri} + \Delta V_{Ri} \quad (19)$$

$$\Delta V_{Ri} = K_p\left\{(e_n - e_{n-1}) + \frac{T_s}{T_i}e_n + \frac{T_d}{T_s}(e_n - 2e_{n-1} + e_{n-2})\right\} \quad (20)$$

where T_s is sampling time interval, e_n is error at sampling time nT_s, e_{n-1} is error at sampling time $(n-1)T_s$, and e_{n-2} is error at sampling time $(n-2)T_s$.

In order to decouple two loopers, rolling velocity at 1^{st} stand is express as follows

$$V_{R1} = V_{R1} + \Delta V_{R1} + \Delta V_{R2}\frac{1+\epsilon_2}{1+f_1} \quad (21)$$

3 Simulator of Looper Control System

The simulator used in this paper consists of three stands and two loopers between them. The looper angle is controlled by changing rolling speed of the upstream stand. Simulation program of looper height control system will be stated below.

3.1 Simulation of Rolling Characteristics

Using Eqs.(1) to (5) rolling force P is calculated. Then exit strip thickness h is calculated by Eq.(9). The method of the calculation is described in the flow chart of Fig.2. Table.2 summarizes the parameters used in the calculation.

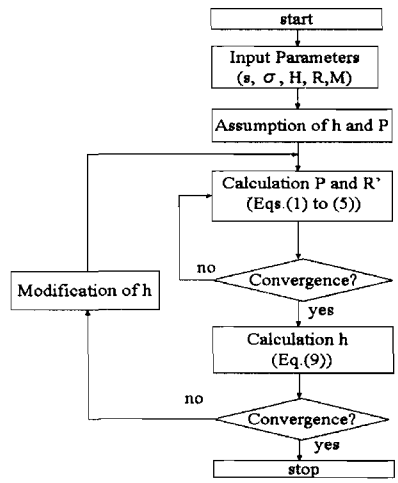


Fig.2: Flow Chart for Calculation P

Table.2: Parameters for Calculation

symbol	scale
M	400t/mm
σ	10t/mm^2
R	400mm
H_1	30mm
h_1^o	18mm
h_2^o	12mm
h_3^o	7mm

3.2 Simulation of Inter Stands Looper Dynamics

Fig.3 shows three stands hot strip mill system with controller. The block diagram showing dynamic characteristics of inter stands looper are given as one in Fig.4.

Firstly, rolling velocity and roll gap is determined initial setting value. Where V_{R3} is set at 2m/s and rolling velocity is determined by mass flow rule, and initial setting value of roll gap is calculated as follows

$$s_i = h_i^o - \frac{P_i}{M} \tag{22}$$

When the strip arrives at 1^{st} stand, rolling characteristics of the stand is started to calculate by flow in Fig.2. After $\frac{L}{v_{f1}}$ [sec], the strip arrives at 2^{nd} stand. Then rolling characteristics of the stand is started to calculate and the looper angle between #1 and #2 is started to calculate and PID control and AGC of stand #1 are started too. In the same way, after $\frac{L}{v_{f2}}$ [sec], the strip arrives at 3^{rd} stand. Then rolling characteristics of the stand is started to calculate and the looper angle between #2 and #3 is started to calculate and PID control and AGC of stand #2 are started too. And when the strip arrives at X-ray AGC after $\frac{L_X}{v_{f3}}$ [sec], AGC of stand #3 is started.

The rolling characteristics at i^{th} stand calculated by flow of Fig.2, is named G_i and used in the block diagram abbreviatedly. The looper angle at j^{th} looper by Eq.(13), is named A_j and used in the block diagram.

Table.3 shows simulation parameters about base rolling condition. Where α^o is desired value of looper angle, T_a, T_v and T_h are time constant of first order lag element. They are expressed as time-lag by looper angle sensor, time-lag by mill motor, and time-lag by rolling of AGC respectively. L_X is distance from stand #3 to X-ray.

Table.3: Simulation Parameters

symbol	scale
α^o	30deg
V_{R3}	2m/s
k_1	0.05
k_2	0.05
k_3	0.01
T_a	0.05sec
T_v	0.05sec
T_h	0.05sec
L_X	2000mm
T_s	0.05sec

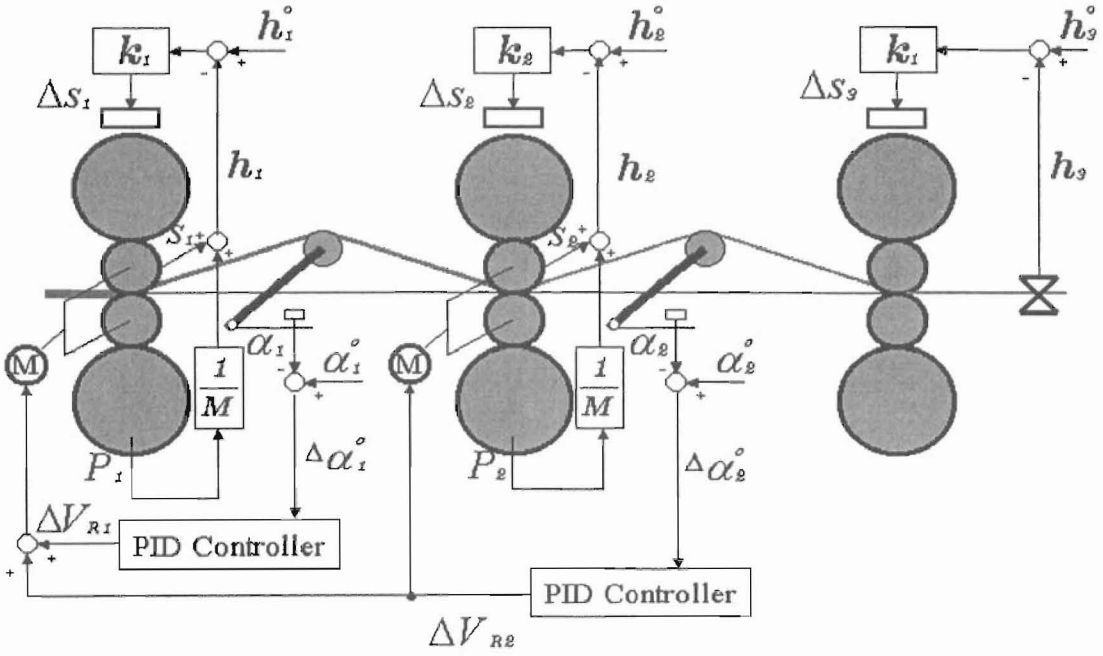


Fig.3: Hot Strip Mill

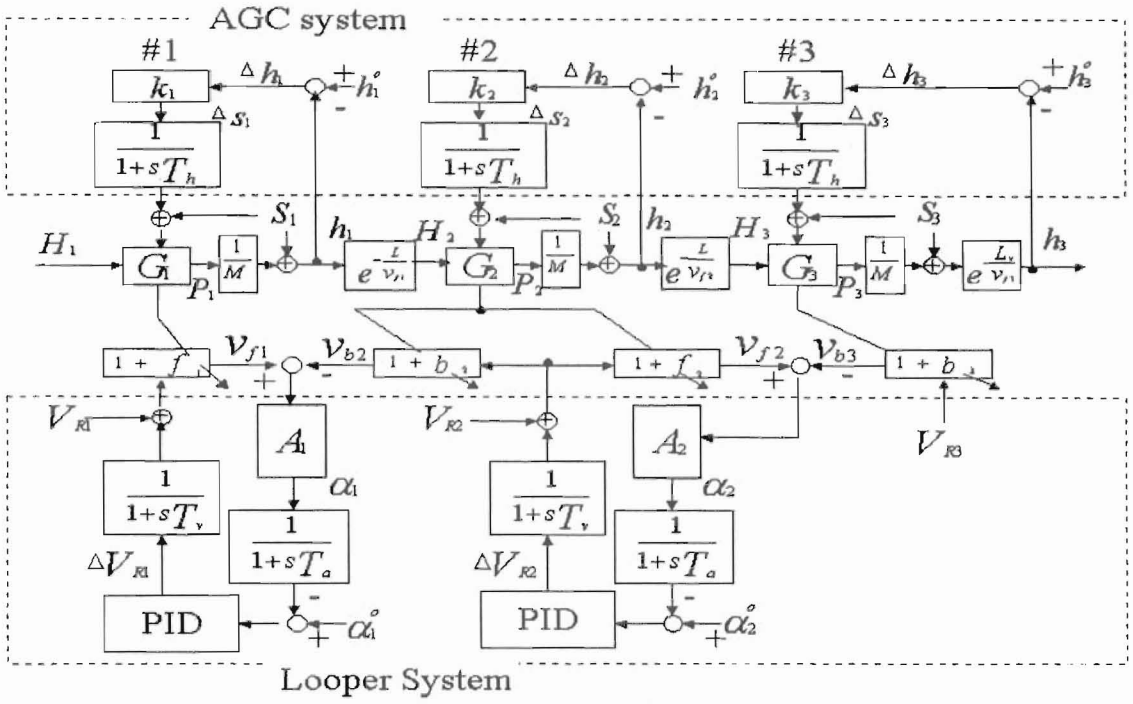


Fig.4: Block Diagram of Hot Strip Mill

Using these mechanisms shown in Figs. 2 to 4, simulator of looper control has been developed. Fig.5 shows display of simulated results describing dynamic behavior of rolling variables such as looper angle, rolling force, rolling speed and roll gap.

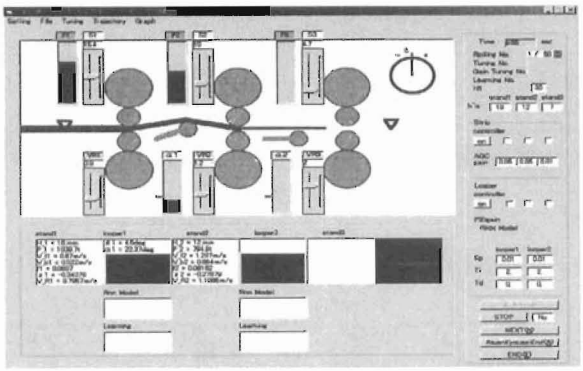


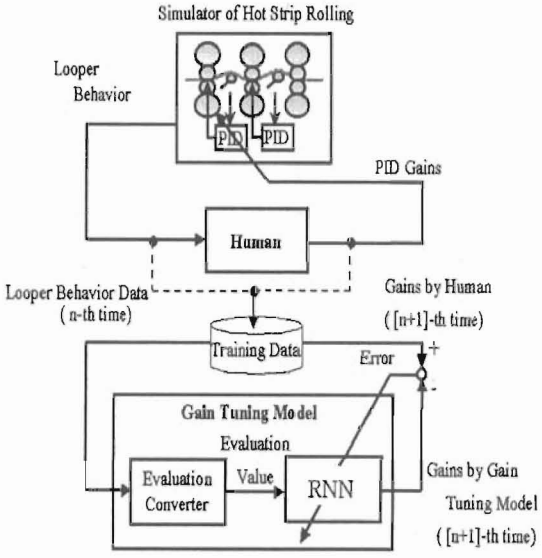
Fig.5: Display Screen of Simulator

4 Gain Tuning Model by Neural Network

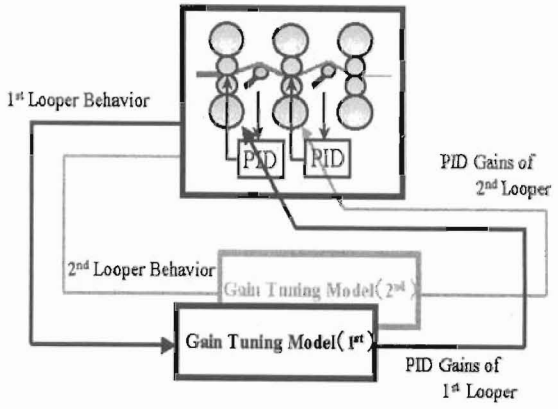
In the following, formation procedure for gain tuning model by neural network is described. The training procedure of PID gains tuning model is shown in Fig.6(a). The detail of the procedure is described as follows.

1. Human makes PID gains tuning watching looper simulator in Fig.5 by trial and error.
2. The tuned data sequence of PID gains are stored in data base and used in the next step.
3. The RNN model is trained using the tuning data stored in data base.
4. The tuned data is prepared for one inter stand looper between #1 and #2 stands.

Then RNN model tune PID gains in two inter stand looper systems between #1 and #2 stands and #2 and #3 stands simultaneously as shown in Fig.6(b).



(a) Training for Gain Tuning Model

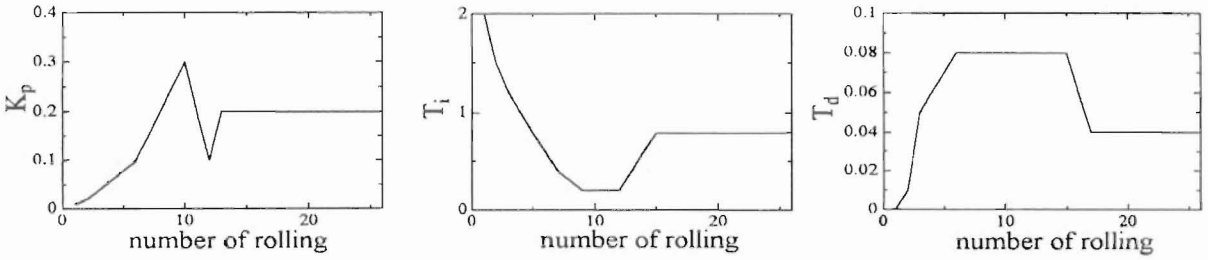


(b) Tuning by Gain Tuning Model

Fig.6: Gain Tuning Model by Neural Network

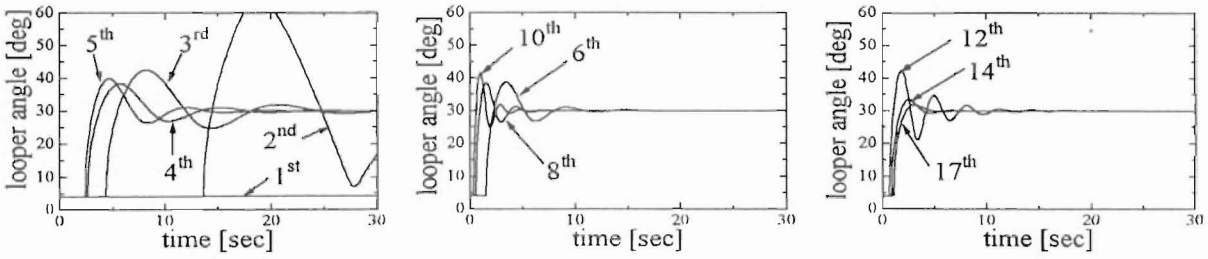
4.1 Generation of Training Data

The training data is generated under the condition of ΔV_{R1} is -0.05m/s. It becomes harder condition that 1^{st} looper is gotten up. Human made PID gains tuning using simulator in Fig.5 watching looper's behavior. The results are shown in Fig.7 and Fig.8. Human began to tune PID gains until number of rolling 17. RNN is trained by the data and constant data until number of rolling 26 in order to make output of RNN being convergent. That is, the training data set is 25.



(a) K_p (b) T_i (c) T_d

Fig.7: PID gains (Human)



(a) 1^{st}~5^{th} (b) 6^{th}~10^{th} (c) 11^{th}~17^{th}

Fig.8: Looper Angle (Human)

4.2 RNN Model

4.2.1 Structure of RNN Model

The structure of RNN model is shown in Fig.9. The network has three layers. The first layer is the input layer, which consists of two parts. The one is the plan layer, which is input evaluation value changed the performance index to the normalized value by preprocessing. The preprocessing is calculated by the linear transformation function as follows

$$p_i = \frac{I_{i_{[n]}} - I_{i_{min}}}{I_{i_{max}} - I_{i_{min}}} \quad ; i = 1 \sim 4 \qquad (23)$$

where $I_{i_{[n]}}$ is the evaluation value at number of rolling of n, and $I_{i_{max}}$ and $I_{i_{min}}$ are the maximum value and the minimum value respectively, which are set in advance.

The other is the context layer at number of rolling of n, which is expressed by

$$c_k = \sum_{m=1}^{n} d^{n-m} g_{k[m]} \quad ; k = 1 \sim 3 \qquad (24)$$

g_1: K_p, g_2: T_i, g_3: T_d

where d is a positive constant and less than 1, which means forgetting effect and m is the rolling number indicating m^{th} preceding instant. As shown in Fig.9, the context layer is the layer that the past output data of RNN is stored.

Then the output of input layer is

$$x_i^{(1)} = \begin{cases} p_i & 1 \le i \le 4 \\ c_{i-4} & 5 \le i \le 7 \end{cases} \qquad (25)$$

The second layer is the hidden layer, which can be expressed as

$$s_j^{(2)} = \sum_{i=0}^{7} w_{ji}^{(1)} x_i^{(1)} \quad ; j = 1 \sim 5 \qquad (26)$$

$$x_0^{(1)} = -1,\ w_{j0}^{(1)} = \theta_j^{(1)} \quad ; j = 1 \sim 5 \qquad (27)$$

$$x_j^{(2)} = f(s_j^{(2)}) = \frac{1}{1 + e^{-s_j^{(2)}}} \quad ; j = 1 \sim 5 \qquad (28)$$

where $w_{ji}^{(1)}$ is the weight for the connection between i^{th} neuron in the input layer and j^{th} neuron in the hidden layer.

The third layer is the output layer, which is descrived as follows.

$$s_k^{(3)} = \sum_{j=0}^{5} w_{kj}^{(2)} x_j^{(2)} \quad ; k = 1 \sim 3 \qquad (29)$$

$$x_0^{(2)} = -1,\ w_{k0}^{(2)} = \theta_k^{(2)} \quad ; k = 1 \sim 3 \qquad (30)$$

$$g_{k[n+1]} = f(s_k^{(3)}) = \frac{1}{1 + e^{-s_k^{(3)}}} \quad ; k = 1 \sim 3 \qquad (31)$$

where $w_{kj}^{(2)}$ is the weight for the connection between j^{th} neuron in the hidden layer and k^{th} neuron in the output layer.

Finally, it is postprocessed and the gains are output from RNN are converted to its original data ranges.

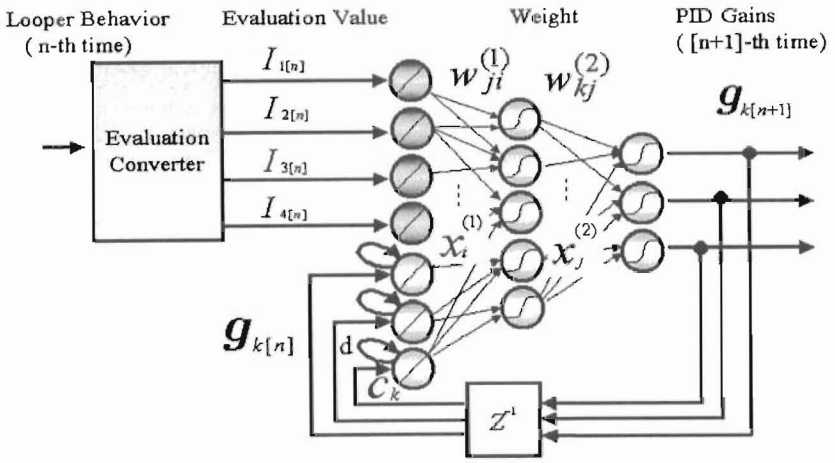


Fig.9: Structure of RNN Model

4.2.2 Input Data of NN

As for the input of Neural Network, the n^{th} time looper behavior is converted into evaluation value by following performance index.

As the performanece index of stability, the difference between max α_{max} and desired value α^o is written by

$$I_1 = \left|\frac{\alpha_{max}}{\alpha^o} - 1\right| \tag{32}$$

As the performance index of accuracy, the integral of absolute value of error is described as

$$I_2 = \frac{1}{T}\int_0^T \left|\frac{\alpha(t)}{\alpha^o} - 1\right| dt \tag{33}$$

where $\alpha(t)$ is the looper angle at time t, and T is the measurement time of performance index.

As the performance index of accuracy, the integral of time multiplied by absolute value is given by

$$I_3 = \frac{1}{T^2}\int_0^T t\left|\frac{\alpha(t)}{\alpha^o} - 1\right| dt \tag{34}$$

As the performance index of rapid response, the peak time, which is time when the looper angle has max value, is described by

$$I_4 = \frac{t_p}{T} \tag{35}$$

These performance index approach its minimum value as the looper behavior becomes desired, that is, human tune the gain. The measurement time in Eqs.(33) ~ Eq.(35) is fixed to $T = 30sec$.

4.2.3 Training of RNN Model

The RNN model is trained by the back propagation algorithm (BP). BP algorithm is described as follows.

step1 ; initialization

The end condition of learning is determined, and initial value of weight parameters are determined by generating random number($0 \leq w < 1$).

step2 ; input and preprocessing

The evaluation value $I_{i[n]}$ at number of rolling n are input to the plan layer, PID gains before number of rolling n are input to the context layer. Where the context layer is input 0 when $n = 1$. $I_{i[n]}$ as preprocessing when input to plan layer are normalized 0~1. And PID gains as postprocessing are normalized 0~1 too. The normalization is used linear transformation by Eq.(23). And their maximum value and minimum value are shown in Table.4 and Table.5.

Table.4: preprocessing

	$I_{i_{min}}$	$I_{i_{max}}$
I_1	0	2
I_2	0	0.7
I_3	0	0.05
I_4	0	0.5

Table.5: postprocessing

	$g_{k_{min}}$	$g_{k_{max}}$
K_p	0	1.5
T_i	0	3
T_d	0	0.5

step3 ; forward propagation

By Eqs. (25) ~ (31) , the output $x_j^{(2)}$ of hidden layer and the output $g_{k[n+1]}^{(3)}$ of output layer is calculated.

step4 ; back propagation of output error to hidden layer

w_{kj} is upadated by

$$\begin{cases} \delta_k^{(2)} = (t_{k[n+1]} - g_{k[n+1]}^{(3)})g_{k[n+1]}^{(3)}(1 - g_{k[n+1]}^{(3)}) \\ \Delta w_{kj}^{(2)} = \epsilon\delta_k^{(2)}x_j^{(2)} \\ w_{kj}^{(2)} \leftarrow w_{kj}^{(2)} + \Delta w_{kj}^{(2)} \end{cases} \tag{36}$$

where $t_{k[n+1]}$ is teaching data. That is, it is the gain that human tuned at number of rolling $n+1$ when human watches the looper behavior at number of rolling n.

step5 ; back propagation of output error to input layer

By using $w^{(2)}$ updated by above equations, $w_{ji}^{(1)}$ is updated as follows

$$\begin{cases} \delta_j^{(1)} = (\sum_k \delta_k^{(2)} w_{kj}^{(2)})x_k^{(2)}(1 - x_k^{(2)}) \\ \Delta w_{ji}^{(1)} = \epsilon\delta_j^{(1)}x_i^{(1)} \\ w_{ji}^{(1)} \leftarrow w_{ji}^{(1)} + \Delta w_{ji}^{(1)} \end{cases} \tag{37}$$

in steps 4 and 5, ϵ is the learning rate, which determines updating amount of w.

step6 ; condition of terminate

If $n = 25$(number of teaching data set) and the end condition of learning is satisfied, obtained weight parameter are adopted and stop. If n is smaller than 25, $n = n + 1$ and go to step. If end condition of learning is not satisfied and $n = 25$, $n = 1$ and go to step2.

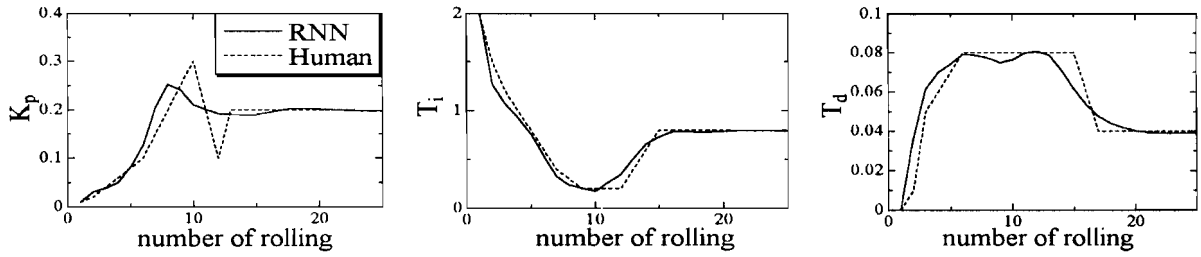
In order to speed up learning, momentum term is introduced. Updating weight Δw_{n-1} at one step ago are added as follows

$$w_n \leftarrow w_n + \Delta w_n + \zeta \Delta w_{n-1} \tag{38}$$

where ζ is positive constant.

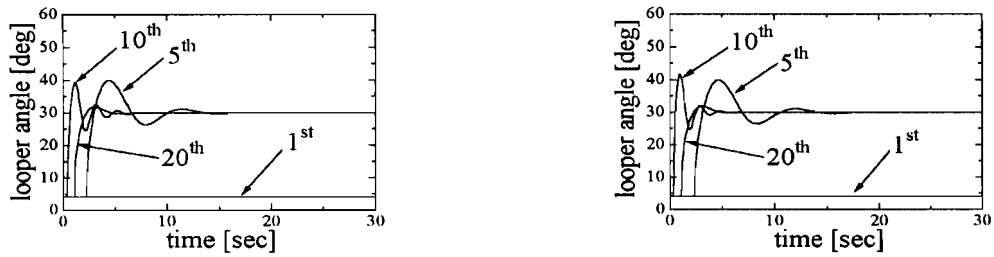
4.3 Performance of The Identified RNN Model

Performance of the identified RNN model are shown in Fig.10 and Fig.11, where $d = 0.6$. It can be seen that in the same rolling condition which generated training data by human the RNN model behaves approximately in the same way.



(a) K_p (b) T_i (c) T_d

Fig.10: PID gains (RNN Model)



(a) RNN Model (b) Human

Fig.11: Looper Angle (RNN Model)

4.4 Extrapolation of Identified RNN

To check the generalization property of NN, simulation is carried out in the different conditions of generated teaching data.

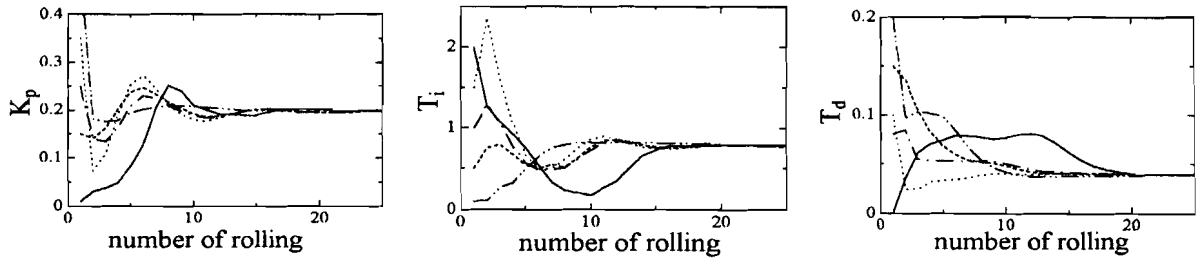
4.4.1 Change of Initial Gain

In the case that initial gains are different from supervised learning condition as shown in Table.6. The results about 1^{st} looper are shown in Fig.12.

It is shown that the tuning value by RNN model is not changed, even if the initial value of gain is changed,

Table.6: Initial Gain

	K_p	T_i	T_d
full line	0.01	2	0
dashed line	0.15	0.5	0.15
dotted line	0.35	1.5	0.1
chain line	0.25	1	0.08
chain double-dashed line	0.5	0.1	0.2

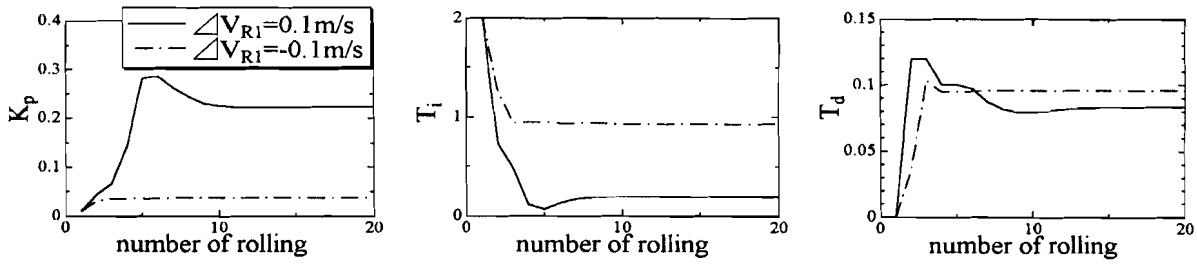


(a) K_p (b) T_i (c) T_d

Fig.12: PID gains (Change of Initial Gain)

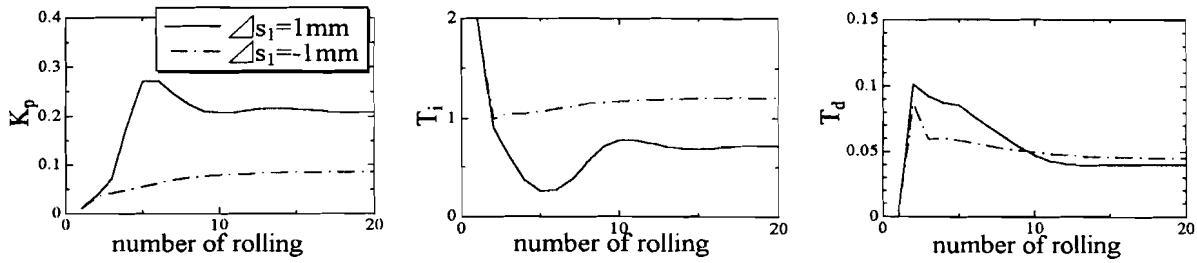
4.4.2 Change of Rolling Condition

Under rolling conditions ΔV_{R1} is $\pm$ 0.1m/s, Δs_1 is $\pm$ 1mm, and T_a are 0.25sec, 0.5sec, the tuning results about 1^{st} looper by RNN are shown in Fig.13~Fig.16. The tuned value are different according to rolling conditions. However, looper does not behave ideally as shown in Fig.16.



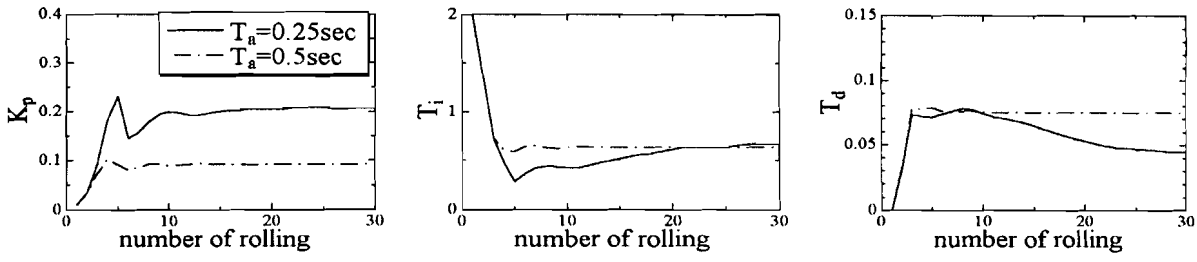
(a) K_p (b) T_i (c) T_d

Fig.13: PID gains ($\Delta V_{R1} = \pm$ 0.1m/s)



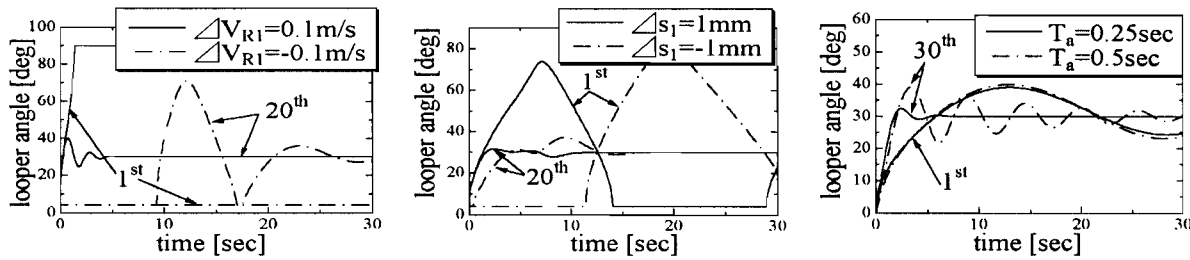
(a) K_p (b) T_i (c) T_d

Fig.14: PID gains ($\Delta s_1 = \pm$ 1mm)



(a) K_p (b) T_i (c) T_d

Fig.15: PID gains (T_a=0.25sec, 0.5sec)



(a) $\Delta V_{R1} = \pm 0.1 m/s$ (b) $\Delta s_1 = \pm 1 mm$ (c) $T_a = 0.25sec, 0.5sec$

Fig.16: Looper Angle (Change of Rolling Condition)

5 Learning of RNN Model

It is important for RNN model to be tuned at optimal gain as rapidly as possible in case that rolling condition is different from condition which the training data is generated.

In the following, we discuss two learning methods in order to improve the RNN model continuously.

5.1 Gain Tuning by Learning

This method is to tune the gain by RNN model as shown in Fig.17.

Firstly, the differential $\frac{\partial I_i}{\partial g_k}$ where evaluation value I_i is input to RNN model and gain g_k can be calculated by past two rolling data. It can be approximated by $\frac{I_{i[t]} - I_{i[t-1]}}{g_{k[t]} - g_{k[t-1]}}$. The value is changed to $I_{g_{ki}}$ by the model which has dead band and saturation as shown in Fig.18. The value of I_M and I_m are shown in Table.7.

The gain tuning value Δg_k is calculated as follows

$$\Delta g_k = \sum_{i=1}^{4} \eta_{ki} I_{g_{ki}} \quad (k = 1 \sim 3) \tag{39}$$

where η_{ki} is positive constant, which is weight of performance index.

Then PID gains output from RNN model are tuned as follows

$$g_k \leftarrow g_k - \Delta g_k \tag{40}$$

If the difference between previous and this gain is less than δ_g, the method is not used in order to converge gain tuning by RNN model. In numerical experiment the value of δ_g is set to 0.005.

Table.7: Value of I_M / I_m

	K_p	T_i	T_d
I_1	50/10	5/1	100/20
I_2	10/5	1/0.2	30/10
I_3	1/0.1	0.2/0.05	10/1
I_4	20/5	1/0.2	20/5

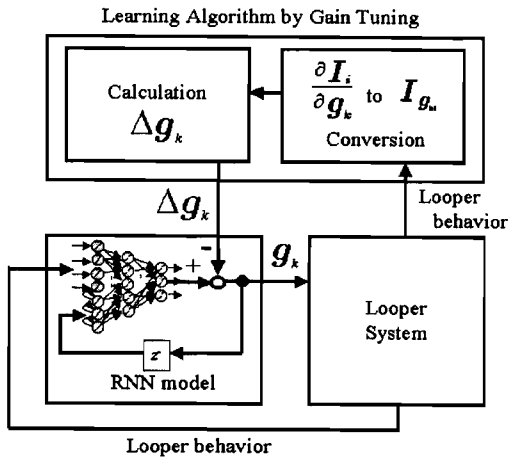


Fig.17: Gain Tuning

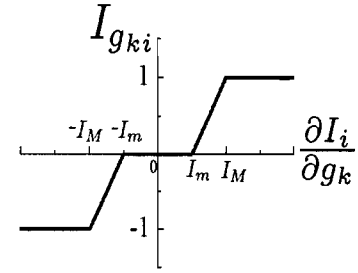


Fig.18: $\frac{\partial I_i}{\partial g_k}$ to $I_{g_{ki}}$ Conversion

5.1.1 Effectiveness of Learning

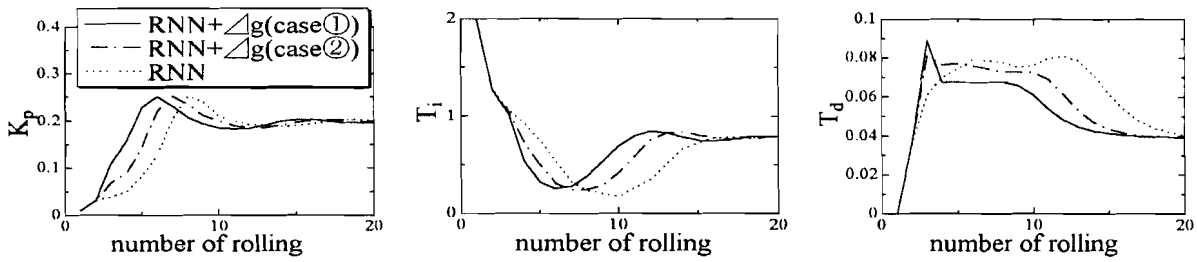
To check the performance of the proposed learning method, numerical experiment is carried out.

Under the generated teaching data condition, the results about 1^{st} looper are shown in Fig.19 and Fig.20. Where case ① is $\eta_{k1} = \eta_{k2} = 0.05$, $\eta_{k3} = \eta_{k4} = 0.01$, and case ② is $\eta_{ki} = 0.01$. That is, case ① is biased according to the performance index, and case ② is equal weight free from the performance index.

In both cases, the results can be obtained faster convergence in tuning than only RNN. It is shown that case ① can be obtained the fastest convergence.

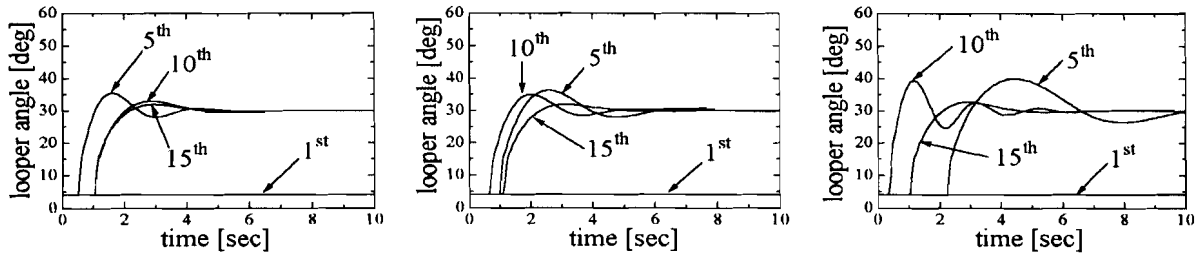
Next, the results about 1^{st} looper under the different conditions are shown in Fig.21 and Fig.22. In both cases, the results can be obtained faster convergence.

These results shows that this learning method make convergence tune faster by RNN, but the tuned value is not varied.



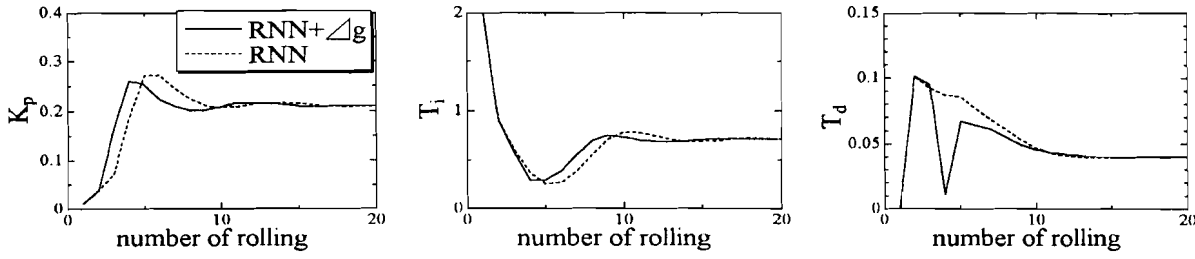
(a) K_p (b) T_i (c) T_d

Fig.19: PID gains (Gain Tuning by Learning; $\Delta V_{R1} = -0.05$m/s))



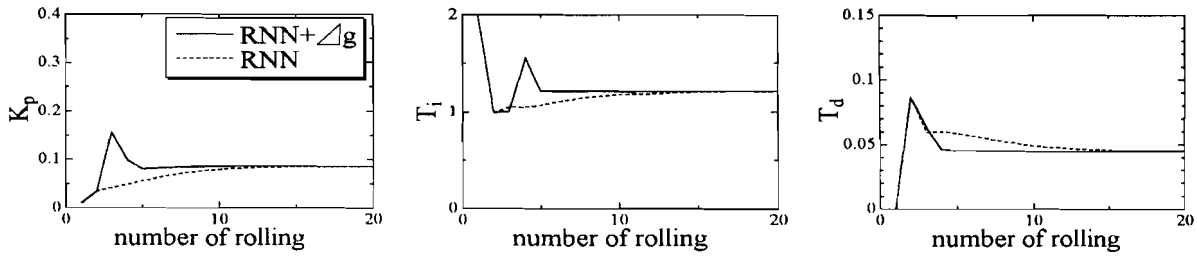
(a) RNN + Δg(case ①) (b) RNN + Δg(case ②) (c) RNN

Fig.20: Looper Angle (Gain Tuning by Learning; $\Delta V_{R1} = -0.05$m/s)



(a) K_p (b) T_i (c) T_d

Fig.21: PID gains (Gain Tuning by Learning ; $\Delta s_1 = 1mm$)



(a) K_p (b) T_i (c) T_d

Fig.22: PID gains (Gain Tuning by Learning ; $\Delta s_1 = -1mm$)

5.2 Update of RNN Weight by BP

The gain tuning is expected to tune the optimal gain. But the tuning value is not changed by the aforementioned learning method. Then the RNN model has to be updated. However, it is difficult to know optimal gains in unknown environment. It is means that there is no teaching signal for gain.

Hence the error function is defined as follows

$$E_{[l]} = \sum_{i=1}^{5} \frac{1}{2}(I_{i[l]} - I_i^o)^2 \tag{41}$$

where I_i^o is the evaluation value for ideal looper behavior. The error function is contained the performance index $I_{i[l]}$ input RNN model.

Furthermore, in order to settle down the vibrations of looper, the performance index is added in error function as follows

$$I_5 = \frac{1}{T}\int_0^T \left| \frac{d\frac{\alpha(t)}{\alpha^o}}{dt} \right| dt \tag{42}$$

The objective of learning is to minimize E as shown in Fig.23. In other words, $-\frac{\partial E}{\partial w}$ is added weight parameter. And the algorithm is led from BP algorithm.

The weight parameters of RNN model is updated in the following BP steps:

step1 : The initial value of gain $g_{k[1]}$ is set, and then simulation is carried out by the gain. The performance index $I_{i[1]}$ are calculated.

step2 :The performance index $I_{i[1]}$ are input the RNN model, and the model output next gain $g_{k[2]}$. It is $l = 2$.

step3 : The simulation is carried out the gain $g_{k[l]}$, and the performance index $I_{i[l]}$ are calculated.

step4 : The weight parameter $w_{kj}^{(2)}$ between hidden layer and output layer are updated by past two times data for rolling results by the following equations.

$$w_{kj}^{(2)} \leftarrow w_{kj}^{(2)} + \Delta w_{kj}^{(2)} \tag{43}$$

where

$$\begin{aligned}\Delta w_{kj}^{(2)} &= \epsilon_{[l]}\delta_k^{(2)} x_{j[l]}^{(2)} \\ &= -\epsilon_{[l]}\left\{\sum_i (I_{i[l]} - I_i^o)\frac{I_{i[l]} - I_{i[l-1]}}{g_{k[l]} - g_{k[l-1]}}\right\} \times \\ &\qquad g_{k[l]}(1 - g_{k[l]})x_{j[l]}^{(2)}\end{aligned} \tag{44}$$

where $x_{j[l]}^{(2)}$ is output of j^{th} neuron in the hidden layer, and $\epsilon_{[l]}$ is learning rate at number of learning l.

The learning rate is given by

$$\epsilon_{[l]} = \epsilon_{[0]}\left(1 - \frac{l}{T_L}\right) \tag{45}$$

where T_L is learning times, which is set in advance.

step5 : Next, the weight parameter $w_{ji}^{(1)}$ between input layer and hidden layer are updated by the following equations.

$$w_{ji}^{(1)} \leftarrow w_{ji}^{(1)} + \Delta w_{ji}^{(1)} \tag{46}$$

where

$$\begin{aligned}\Delta w_{ji}^{(1)} &= \epsilon_{[l]}\delta_j^{(1)} x_{i[l]}^{(1)} \\ &= \epsilon_{[l]}\left(\sum_k \delta_k^{(2)} w_{kj}^{(2)}\right) x_{j[l]}^{(2)}(1 - x_{j[l]}^{(2)})x_{i[l]}^{(1)}\end{aligned} \tag{47}$$

where $x_{i[l]}^{(1)}$ is output of i^{th} neuron in the input layer at number of learning l.

step6 : The next gain $g_{k[l+1]}$ is determined the previous performance index $I_{i[l]}$ by the RNN model.

step7 : If the end condition is satisfied, it is finished. But if it is not satisfied, $l \leftarrow l+1$ is carried out and go back to step3.

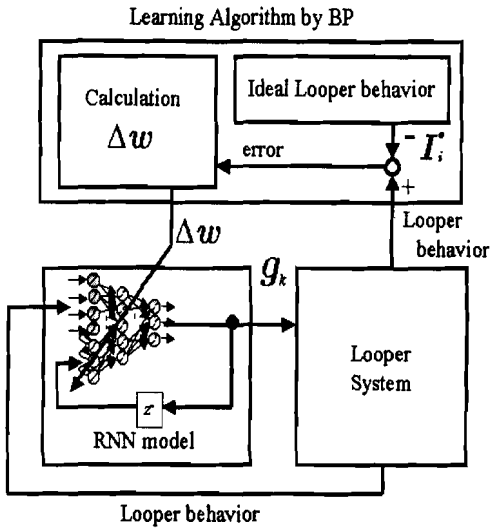


Fig.23: Update of RNN Weight by BP

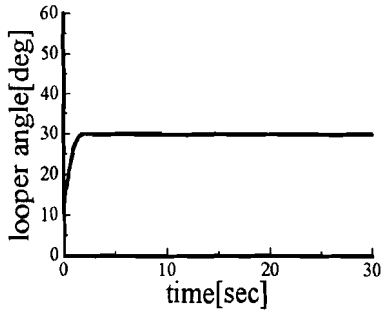


Fig.24: Ideal Looper Behavior (K_p=0.4, T_i=0.6, T_d=0 under base condition)

5.2.1 Numerical Test of BP Algorithm

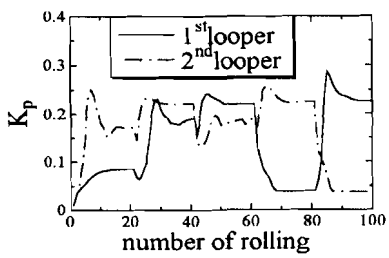
To check the performance of the learning algorithm, numerical experiment is carried out. The rolling conditions are changed at every twenty rollings as shown in Table.8.

The results by RNN only are shown in Fig.25 and Fig.27, and the results by RNN and gain tuning method and learning using BP are shown in Fig.26 and Fig.28. Where $\epsilon_{[0]}$ is 0.1, T_L is 10 from number of tuning 5, the gain tuning method is same parameters in case ①, and the context layer is initialized 0 when the rolling conditions change. If the gain variation is less than 0.001, the weight parameters are not updated. The objective of learning is to minimize the error against ideal looper behavior as shown in Fig.24.

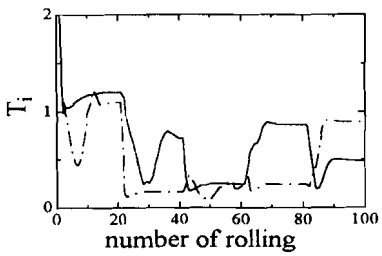
The results show that converged value of gain tuning by RNN can be changed. However, both improvement and corruption results can be obtained by learning.

Table.8: Rolliing Conditions

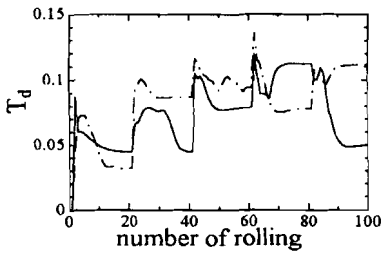
	Number of Rolling	$\Delta s_1[mm]$	$\Delta V_{R2}[m/s]$
I	1~20	-1	0
II	21~40	1	0.1
III	41~60	1	-0.1
IV	61~80	-1	0.1
V	81~100	-1	-0.1



(a) K_p

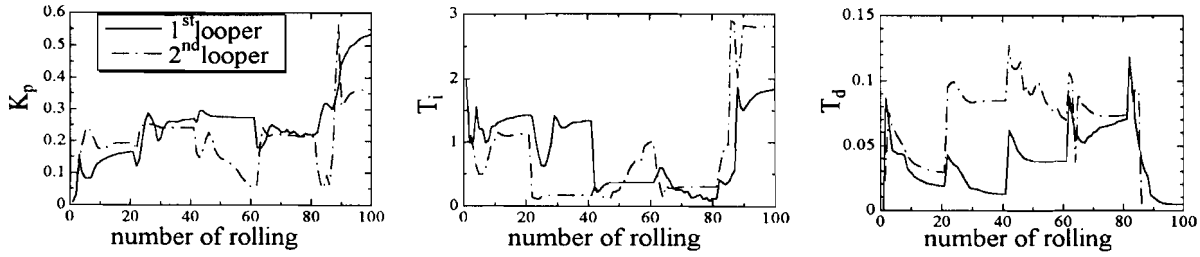


(b) T_i



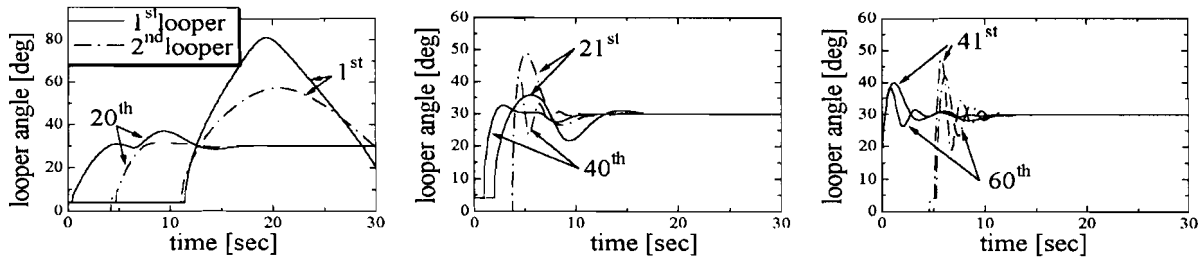
(c) T_d

Fig.25: PID gains (RNN)

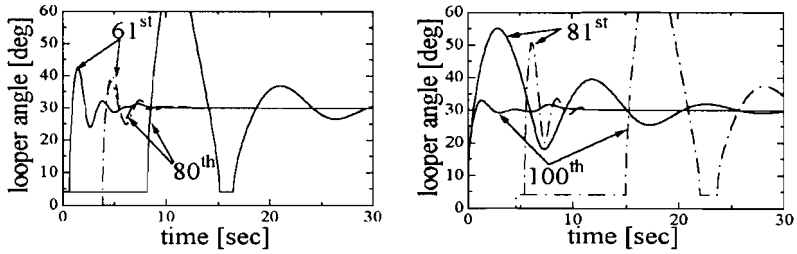


(a) K_p (b) T_i (c) T_d

Fig.26: PID gains (RNN+Learning)

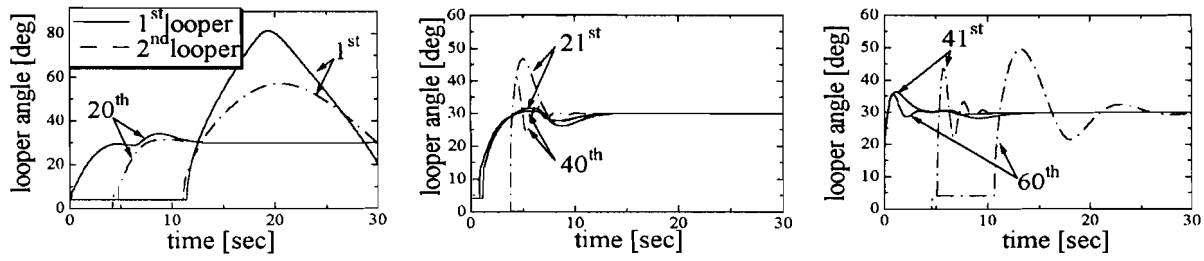


(a) rolling condition I (b) rolling condition II (c) rolling condition III

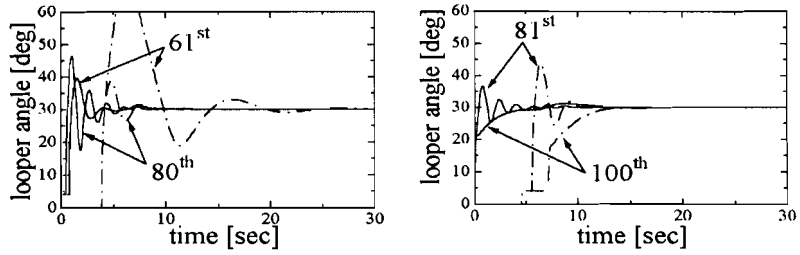


(d) rolling condition IV (e) rolling condition V

Fig.27: Looper Angle (RNN)



(a) rolling condition I (b) rolling condition II (c) rolling condition III



(d) rolling condition IV (e) rolling condition V

Fig.28: Looper Angle (RNN+Learning)

6 Conclusion

In this paper, RNN model is introduced to express gain tuning action of human for looper height control system in hot strip mills. The possibility of realizing tuning action of human by the model has been shown by numerical experiments. Also, the extrapolation property of the NN have has been checked numerically. Furthermore in order to improve the model, we proposed two learning methods. One has showed improvement of convergence by learning, and the other has showed improvement of converged value of gain tuning by the weight learning of NN model. The effectiveness of these two methods have been shown numerically.

In this paper, the gain tuning of looper height control system was dealt with. In the future, the proposed method is expected to be applied to other objects with more general manner.

Bibliography

[1] H.Asada et al, "Adaptive and Robust Control Method with Estimation of Rolling Characteristics for Looper Angle Control at Hot Strip Mill", ISIJ International, Vol.43, No.3(2003), pp358-365

[2] I.M.Mujtaba et al, "Application of Neural Networks and Other Learning Technologies in Process Engineering", Imperial College Press, 2001

[3] Lakhmi C.Jain et al, "INDUSTRIAL APPLICATIONS of NEURAL NETWORKS", The CRC Press, 1999

[4] Cihan H. Dagli, "ARTIFICIAL NEURAL NETWORKS for Intelligent Manufacturing", Chapman & Hall, 1994

[5] S.Imajo et al, "Human Model and its Learning for Setting of Looper Control Gain in Threading of Hot Tandem Mills," Proc. of the 46th Annual Conf. of the Institute of Systems, Control and Information Engineers (2002), pp.521-522

[6] S.Imajo et al, "Human Model for Gain Tuning of Looper Control in Hot Strip Rolling," CAMP-ISIJ, Vol.15(2002), p931