Networked Control System Based on RBF Neural Network

Haitao Zhang, Jinbo Hu and Wenshao Bu

Electronic and Information Engineering College,
Henan University of Science and Technology, Luoyang 471023, China

zhang_haitao@163.com

Abstract

In the networked control system, the control interference, measurement noise and time delay make traditional digital PID algorithm not reach stable state. On the basis of simple digital PID algorithm, Kalman filter is first introduced, the effect of the interference and noise is decreased, and stability is improved. Then RBF (Radial Basis Function) neural network is used, Jacobian array is computed, the three parameters of PID algorithm are adjusted. Furthermore, the resistance integral saturation is used to limit the size of control quantity. Finally the simulation research on a DC (Direct Current) motor is done, and the simulation results show the effectiveness of the proposed algorithm when time delay, noise and interference are all large.

Keywords: *networked control system; RBF neural network; Kalman filter; resistance integral saturation*

1. Introduction

In the networked control system (NCS), time delay not only makes real-time performance of the system become worse, but also affects the stability of the system. At present, for the random network time delay in NCS, the deterministic method is often used. It is to convert the random delay to fixed delay by introducing data buffer, and then use the existing method to design the controller [1-2]

In the networked control system with large time delay, the existence of interference and noise makes simple PID controller difficult to get better effect [3]. On the other hand, when we don't consider the interference and noise simple PID algorithm can be used to ensure the stability of system although the system has bigger adjusting time [4]. However, under larger size of noise and interference, simple PID algorithm even can not guarantee the stability of the system. Owing to the above reasons, it is difficult to get better effect to compensate the influence of the network time delay, interference and noise only utilizing simple PID algorithm [5].

In order to overcome the deficiencies of simple PID algorithm, it is necessary to utilize new effective control means. Simple PID control has limited functions, and its control effect can be improved by introducing other control algorithm. Many intelligent control algorithms including fuzzy control, BP neural network, genetic algorithm and fuzzy immune are combined with PID so as to make up for the deficiencies of basic digital PID algorithm [6-7].

In these common intelligent control algorithms, only BP neural network has nonlinear mapping ability of any function, but it has the following deficiencies: its Jacobian array is approximated by the value of sign function of Jacobian array; the online training of weight and threshold is slow and can't get consistent results.

Radial basis function (RBF) neural network may approximate any continuous function with arbitrary precision, so we introduce RBF neural network into PID controller, and some improvements are done considering the influence of interference and noise

2. System Model

The basic model of networked control system with time delay based on computer control is shown in Figure 1. *r(k)*, *u(k)*, *y(k)* and *e(k)=r(k)-y(k)* are the reference, control, output, and error signals in discrete time domain respectively. *D(z)* is the pulse transfer function of controller, *Gp(s)* is the transfer function of controlled object, and τ is the time delay which includes two parts. One is the time delay between sensor and controller; the other is the time delay between controller and actuator. *T* is the sample period. *H(s)* is zero-order hold. *w(t)* and *v(t)* are the control interference and measurement noise respectively.

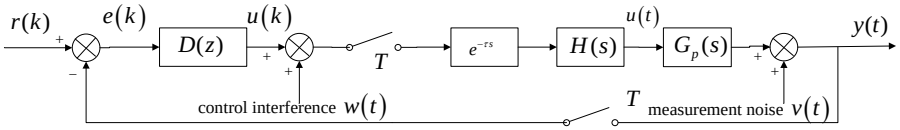

Figure 1. Basic model of networked control system

Regardless of interference and noise, the pulse transfer function of the closed-loop system shown in Figure 1 can be expressed as follows:

$$\Phi(z)=\frac{Y(z)}{R(z)}=\frac{D(z)G(z)z^{-N}}{1+D(z)G(z)z^{-N}} \tag{1}$$

In equation (1), $N=\tau/T$, *G(z)* is the pulse transfer function of generalized object and satisfy equation (2).

$$G(z)=(1-z^{-1})Z\left(\frac{G_p(s)}{s}\right) \tag{2}$$

If the interference and noise are considered, the pulse transfer function of the closed-loop system become more complex, and the stability of the system also becomes worse.

In order to solve the problem, combining simple PID control algorithm with the filter is still difficult to get better control effect. So RBF neural network and resistance integral saturation are introduced to improve the performance of the system.

3. Proposed Control Algorithm

3.1. PID controller and its deficiencies

In general, the controller uses positional digital PID algorithm, the control function is expressed as the following Equation (3) and (4).

$$\Delta u(k)=k_p(e(k)-e(k-1))+k_i e(k)+k_d(e(k)-2e(k-1)+e(k-2)) \tag{3}$$

$$u(k)=u(k-1)+\Delta u(k) \tag{4}$$

In equation (4), k_p, k_i and k_d are the proportional gain, integral time constant and differential time constant respectively. Using digital PID control, the networked control

system without interference and noise can be easily stabilized. However, simple PID control exists some deficiencies. First, the existence of interference and noise makes time delay system is difficult to be stabilized; secondly, the method has long adjusting time; thirdly, if time delay is changed, the PID controller doesn't get better results.

3.2. PID Controller Based on RBF Neural Network

In order to improve the control effect of simple PID control algorithm, the three parameters of PID controller may be adjusted by some intelligent algorithm. The rule library of fuzzy algorithm is difficult to be determined; BP neural network is difficult to get consistent results by online weight adjustment. So RBF neural network is introduced into digital PID control system to identify the Jacobian array. The array reflects the change of output relative to the control input, and its identification makes PID controller correctly adjust its three parameters.

Figure 2 is The system model of PID controller based on RBF neural network and Kalman filter.

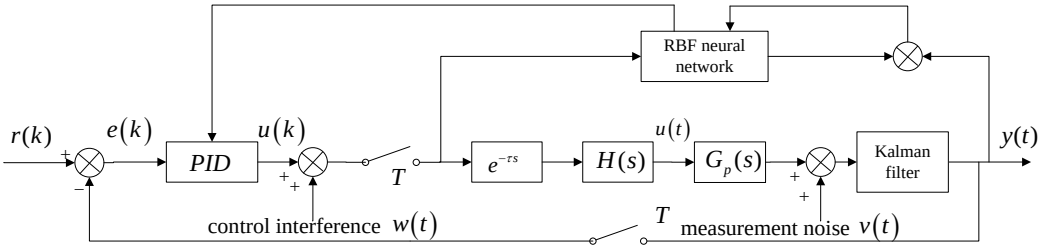

Figure 2. The system model of PID controller based on RBF neural network

The adjusting index of the network system is $E(k)=½e^2(k)$. The adjustment of k_p, k_i and k_d is done by using gradient descent method. First, the change rate of three parameters is computed as follows:

$$\begin{aligned}\Delta k_p(k) &= -\eta\frac{\partial E(k)}{\partial k_p(k)} = -\eta\frac{\partial E(k)}{\partial y(k)}\frac{\partial y(k)}{\partial \Delta u(k)}\frac{\partial \Delta u(k)}{\partial k_p(k)} \\ &= \eta e(k)\frac{\partial y(k)}{\partial \Delta u(k)}\big(e(k)-e(k-1)\big)\end{aligned} \tag{5}$$

$$\Delta k_i(k) = -\eta\frac{\partial E(k)}{\partial k_i(k)} = -\eta\frac{\partial E(k)}{\partial y(k)}\frac{\partial y(k)}{\partial \Delta u(k)}\frac{\partial \Delta u(k)}{\partial k_i(k)} = \eta e(k)\frac{\partial y(k)}{\partial \Delta u(k)}e(k) \tag{6}$$

$$\begin{aligned}\Delta k_d(k) &= -\eta\frac{\partial E(k)}{\partial k_d(k)} = -\eta\frac{\partial E(k)}{\partial y(k)}\frac{\partial y(k)}{\partial \Delta u(k)}\frac{\partial \Delta u(k)}{\partial k_d(k)} \\ &= \eta e(k)\frac{\partial y(k)}{\partial \Delta u(k)}\big(e(k)-2e(k-1)+e(k-2)\big)\end{aligned} \tag{7}$$

Then we can get:

$$k_p(k) = k_p(k-1) + \Delta k_p \tag{8}$$

$$k_i(k) = k_i(k-1) + \Delta k_i \tag{9}$$

$$k_d(k) = k_d(k-1) + \Delta k_d \tag{10}$$

In Equation (5)~(7), η is learning rate, and $\partial y(k)/\partial \Delta u(k)$ is Jacobian array. Once Jacobian array is gotten, Δk_p, Δk_i and Δk_d can all be computed. In PID controller based on BP neural network, Jacobian array is gotten by the approximation $\partial y(k)/\partial \Delta u(k)=sgn(\partial y(k)/\partial \Delta u(k))$. After RBF neural network is introduced to identify Jacobian array $\partial y(k)/\partial \Delta u(k)$, the control performance of system can be improved.

3.3. RBF Neural Network

RBF neural network is a three-layer feedforward network, its mapping from input layer to output layer is nonlinear, and the mapping from hidden layer to output is linear, so it has faster learning rate and avoids getting local minimum value.

The structure of RBF neural network is shown as Figure 3. $X=[x_1,x_2,...,x_n]^T$ is the input vector of network, n is the number of input variable. From Fig.3, the input vector directly reaches the hidden layer.

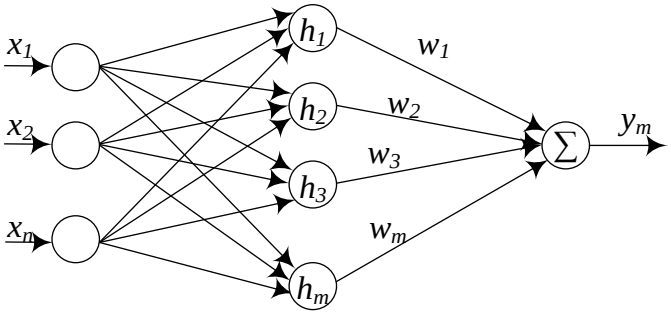

Figure 3. Structure of RBF neural network

The threshold function of hidden layer used radial basis function. Supposed that the radial basis vector is $H=[h_1,h_2,...,h_m]$, where the radial basis function $h_j=exp(-||X-C_j||^2/(2b_j^2))$ $(j=1,2,...,m)$, the center vector $C_j=[c_{j1},h_{j2},...,c_{jn}]$, and the basis width vector $B=[b_1,b_2,...,b_m]^T$. b_j is the basis width of node j.

The weight vector from the hidden layer to the output layer is $W=[w_1,w_2,...,w_m]^T$. The output of RBF neural network is $y_m(k)=w_1h_1+w_2h_2+...+w_mh_m$.

The performance index function of network is $J_1=½(yout(k)-y_m(k))^2$. Then using gradient descent method we can get the following results [8].

$$\Delta w_j(k)=-\eta\frac{\partial E(k)}{\partial w_j(k)}=-\eta\frac{\partial E(k)}{\partial y_m(k)}\frac{\partial y_m(k)}{\partial w_j(k)}=\eta\big(yout(k)-y_m(k)\big)h_j \tag{11}$$

$$\begin{aligned}\Delta b_j(k)&=-\eta\frac{\partial E(k)}{\partial b_j}=\eta\frac{\partial E(k)}{\partial y_m(k)}\frac{\partial y_m(k)}{\partial h_j(k)}\frac{\partial h_j(k)}{\partial b_j}\\&=\eta\big(yout(k)-y_m(k)\big)w_j(k)h_j(k)\frac{\|X-C_j(k)\|^2}{b_j^3}\end{aligned} \tag{12}$$

$$\begin{aligned}\Delta c_{ji}(k)&=-\eta\frac{\partial E(k)}{\partial c_{ji}(k)}=\eta\frac{\partial E(k)}{\partial y_m(k)}\frac{\partial y_m(k)}{\partial h_j(k)}\frac{\partial h_j(k)}{\partial c_{ji}(k)}\\&=\big(yout(k)-y_m(k)\big)w_j(k)h_j(k)\frac{x_i(k)-c_{ji}(k)}{b_j(k)^2}\end{aligned} \tag{13}$$

Then we may add two inertial terms so as to make the search for rapid convergence to the global minimum. Therefore, we can get the following results:

$$w_j(k) = w_j(k-1) + \eta\left(yout(k) - y_m(k)\right)h_j + \alpha\left(w_j(k-1) - w_j(k-2)\right) + \beta\left(w_j(k-2) - w_j(k-3)\right) \tag{14}$$

$$b_j(k) = b_j(k-1) + \eta\left(yout(k) - y_m(k)\right)w_j h_j \frac{\left\|X - C_j\right\|^2}{b_j^3} + \alpha\left(b_j(k-1) - b_j(k-2)\right) + \beta\left(b_j(k-2) - b_j(k-3)\right) \tag{15}$$

$$\Delta c_j = \left(yout(k) - y_m(k)\right)w_j h_j \frac{x_j - c_{ji}}{b_j^2} \tag{16}$$

$$c_{ji}(k) = c_{ji}(k-1) + \eta\left(yout(k) - y_m(k)\right)w_j h_j \frac{x_i - c_{ji}}{b_j^2} + \alpha\left(c_{ji}(k-1) - c_{ji}(k-2)\right) + \beta\left(c_{ji}(k-2) - c_{ji}(k-3)\right) \tag{17}$$

After getting the parameter $w_j(k)$, $b_j(k)$ and $c_{ji}(k)$, Jacobian array $\partial y(k)/\partial \Delta u(k)$can be computed using the following equation.

$$\frac{\partial y(k)}{\partial \Delta u(k)} \approx \frac{\partial y_m(k)}{\partial \Delta u(k)} = \sum_{j-1}^{m} \frac{\partial y_m(k)}{\partial h_j}\frac{\partial h_j}{\partial \Delta u(k)} = \sum_{j-1}^{m} \frac{\partial y_m(k)}{\partial h_j}\frac{\partial h_j}{\partial x_1} = \sum_{j=1}^{m} w_j h_j \frac{c_{j1} - x_1}{b_j^2} \tag{18}$$

In equation (18), x_1 is one input of RBF neural network. In our system, we let $x_1=\Delta u(k)$.

3.4. Kalman filter

In order to solve the problem caused by interference and noise, Kalman filter may be introduced.

We may transform continuous object model into discrete state space model. The discrete system model considering interference and noise is as follows:

$$\begin{cases} x(k) = Ax(k-1) + B\left(u(k) + w(k)\right) \\ y(k) = Cx(k) + v(k) \end{cases} \tag{19}$$

In equation (19), $x(k)$ represents discrete state of system in time k; A, B and C are state matrix, control matrix and output matrix. The recursive algorithm of discrete Kalman filter is as follows [9]:

$$M_n(k) = \frac{P(k) * C^T}{C * P_k * C^T + R} \tag{20}$$

$$P(k) = A * P(k-1) * A^T + B * Q * B^T \tag{21}$$

$$P(k) = \left(I_n - M_n(k) * C\right) * P(k) \tag{22}$$

$$x(k) = A * x(k-1) + M_n(k) * \left(y(k) - C * A * x(k-1)\right) \tag{23}$$

$$y_e(k) = C * x(k) \tag{24}$$

In the above equation (20)~(24), Q and R are respectively covariance of interference $w(t)$ and noise $v(t)$; P is initial error covariance.

In Matlab, in order to, we use the following code to get approximate transfer function of the time delay function $e^{-\tau s}$ [5].

[num,den]=pade(tau,3);

G1=tf(num,den);

In the above code, "tau" is time delay, "3" expresses the order of transfer function, and "G1" is the approximate transfer function of $e^{-\tau s}$.

3.5. Resistance integral saturation

After the system use RBF neural network to identify Jacobian array, we use the array to adjust the three parameters, and further to computer the control quantity. In this method, all the data will continue to increase, which leads to the control quantity increasing, the system generates the oscillation.

So the resistance integral saturation (RIS) is introduced to control the addition of control quantity, and ensure that the system reach stable state.

4. Simulation

In this paper, we use a DC motor as controlled object to analyze the system performance, the transfer function of the controlled object is expressed as follows [9]:

$$G_p(s) = \frac{2029.826}{(s+26.29)(s+2.296)} = \frac{2029.826}{s^2+28.586s+60.36184} \tag{25}$$

In order to verify the effectiveness of the proposed method, the larger range of time delay need to be used. In the following, all kinds of time delay systems are used to verify the control method using the RBF neural network and Kalman filter.

The simulation is done in Matlab environment. The parameters are reference value rin=50rad/s, sample period T=0.001s. We select two sizes of interference and noise so as to verify the performance of system. One is small interference and noise, and its $w(t)$=$v(t)$=0.002**rands*(1) which represents that -0.02≤$w(t)$=$v(t)$≤0.002; the other is large interference and noise, and its $w(t)$=$v(t)$=0.02**rands*(1). For Kalman filter, the parameters are Q=R=1. For PID algorithm with Kalman filter, PID parameters are gotten by trial and error method. For PID algorithm with RBF neural network, Kalman filter and RIS, the initial value of PID parameters is k_p=0.03, k_i=0.01 and k_d=0.03. In addition, the input layer selects three nodes, and three input variables are respectively the differential of control quantity, the system output and the system output of previous sampling period. The hidden layer selects six nodes, and the parameters are η=0.05, α=0.05, β=0.01, b_j=0(j=1,...,6), c_{ji}=0(i=1,...,3;j=1,...,6) and hj=0(j=1,...,6). Even if the interference and noise are small, simple PID algorithm doesn't reach stable state. So we use PID algorithm with Kalman filter, PID algorithm with RBF neural network, Kalman filter and RIS (resistance integral saturation) respectively, then the step responses are compared under conditions of different time delay. The results are shown in Figure 4 to Figure 15.

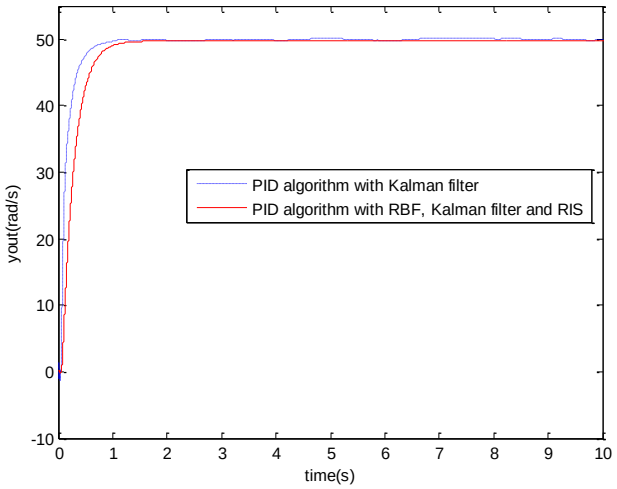

Figure 4. Step response when time delay is 0.1s, and interference and noise are small

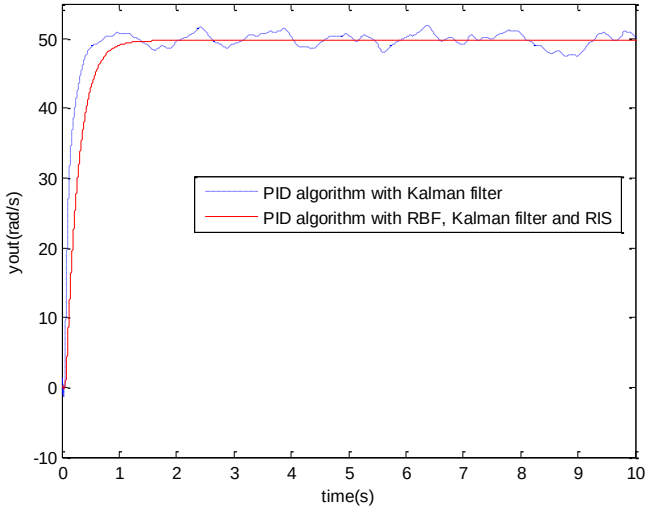

Figure 5. Step response when time delay is 0.1s, and interference and noise are large

In the simulation of Figure 4 and Figure 5, Kalman filter are both used in the two algorithms. In the two simulations only the size of interference and noise is different. In Figure 4, the interference and noise are small, i.e. $w(t)=v(t)=0.002*rands(1)$, while in Figure 5, the interference and noise are large, i.e. $w(t)=v(t)=0.02*rands(1)$. The simulation results show that, when time delay is small, the performance of simple PID algorithm is better than that of proposed algorithm. However, when the interference and noise are become larger, the performance of the proposed algorithm keeps unchanged, but the performance of simple PID algorithm becomes worse.

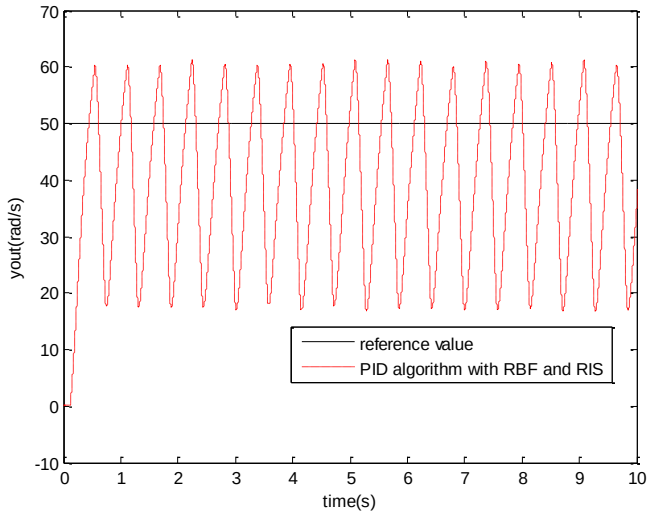

Figure 6. Step response when the proposed algorithm doesn't use Kalman filter

In Figure 6 and Figure 7, $\tau=0.1s$, $w(t)=v(t)=0.002*rands(1)$. For PID algorithm with RBF neural network and resistance integral saturation, the step response of system generates larger oscillation because of the lack of Kalman filter. It can be seen that Kalman filter is necessary for the proposed algorithm. In Fig.7, for PID algorithm with RBF neural network and Kalman filter, the step response of system generates greater oscillation because of the lack of resistance integral saturation. It can be seen that resistance integral saturation is also necessary for the proposed algorithm.

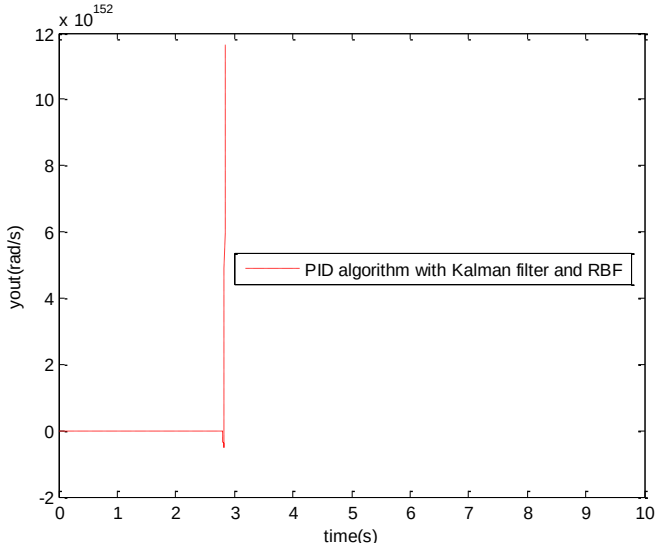

Figure7. Step response when the proposed algorithm doesn't use RIS

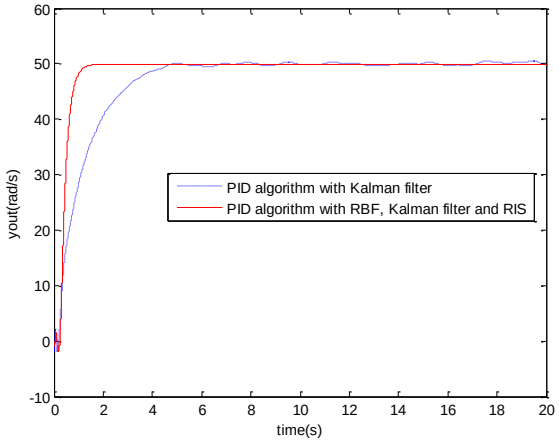

Figure 8. Step response when time delay is 0.5s, and interference and noise are small

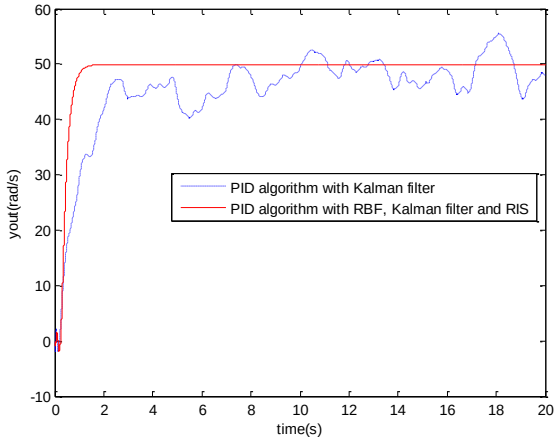

Figure 9. Step response when time delay is 0.5s, and interference and noise are large

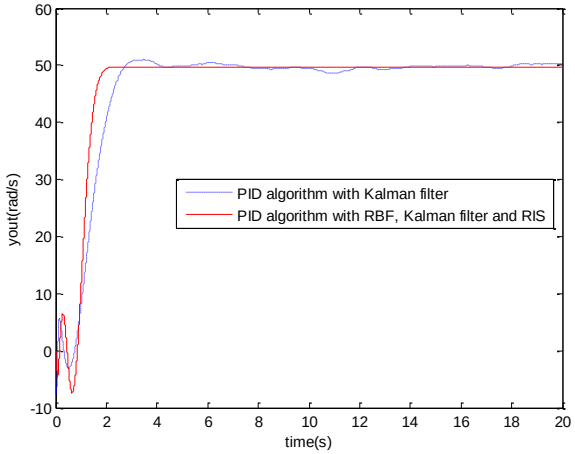

Figure 10. Step response when time delay is 2s, and interference and noise are small

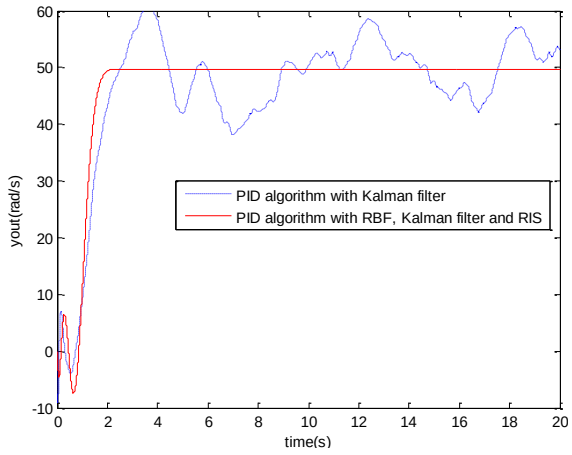

Figure 11. Step response when time delay is 2s, and interference and noise are large

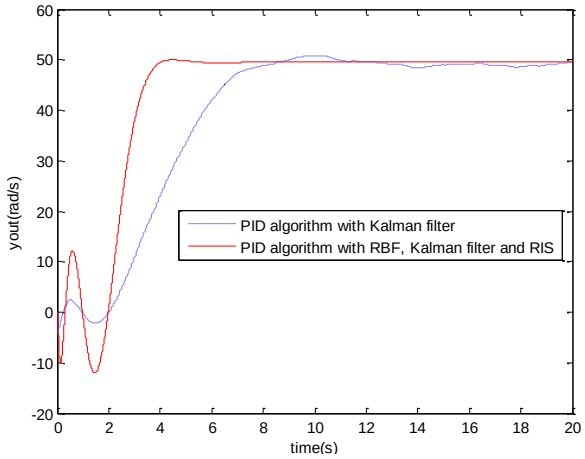

Figure 12. Step response when time delay is 5s, and interference and noise are small

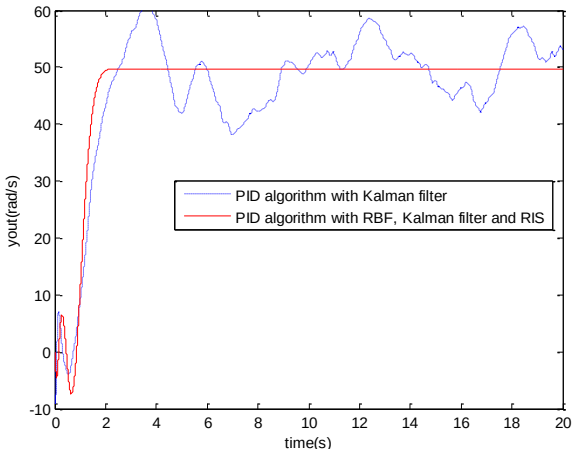

Figure 13. Step response when time delay is 5s, and interference and noise are large

With the increase of delay, the control effect of these methods differs significantly. In Figure 8 and Figure 9, the time delay of system is 0.5s. In Figure 10 and Figure 11, the time delay of system is 2s. In Figure 12 and Figure 13, the time delay of system is 5s. In Figure 14 and Figure 15, the time delay of system is 10s. No mater the size of interference and noise, the performance of proposed algorithm is better than that of PID algorithm with Kalman filter. Especially when the interference and noise are large, PID algorithm with Kalman filter is very poor, but the performance of the proposed algorithm basically remain unchanged.

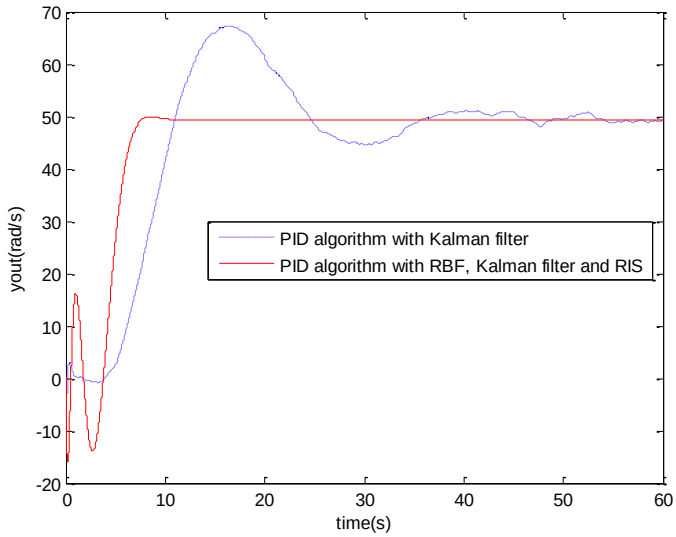

Figure 14. Step response when time delay is 10s, and interference and noise are small

In the following, we mainly compare the two algorithms under two size of interference and noise.

For PID algorithm with Kalman filter, when the interference and noise are small, the system may reach stable state though the step response has some fluctuation and long adjusting time. However, when the interference and noise are large, its fluctuation becomes larger, and the system is difficult to reach stable state. It can be seen that the algorithm can not be applied to the time delay system with large control interference and measurement noise.

For PID algorithm with RBF neural network, Kalman filter and resistance integral saturation, regardless of the size of interference and noise, the system may reach stable state, and the step response has not fluctuation. When the time delay is large, it generates very small overshoot and stable error. But the system reaches the stable state, and stability is not affect by time delay, interference and noise. On the other hand, with the increase of time delay, its adjusting time of becomes longer, it seems that the rapidity becomes lower; However, its adjusting time is basically equal to the time delay which actually indicates the rapidity of system. This proves the validity of the proposed algorithm.

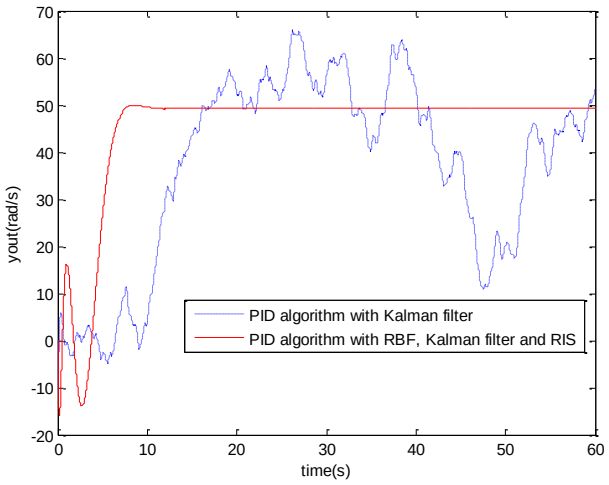

Figure 15. Step response when time delay is 10s, and interference and noise are large

References

[1] S. S. Hu and Q. X. Zhu, "Stochastic Optimal Control and Analysis of Stability of Networked Control Systems with long delay", Automatica, vol. 39, no. 11, **(2003)**, pp. 1877-1884.
[2] Z. X. Yu, H. T. Chen and Y. J. Wang, "Research on Markov Delay Characteristic-Based Closed Loop Network Control System", Control Theory and Applications, vol. 19, no. 2, **(2002)**, pp. 263-267.
[3] Y. Tipsuwan and M. Y. Chow, "Control Methodologies in Networked Control Systems", Control Engineering Practice, vol. 11, no. 10, **(2003)**, pp. 1099-1111.
[4] Z. Li, "Research on Delay Compensation of Networked Control System", Henan University of Science and Technology, Luoyang, China, **(2011)**.
[5] H. T. Zhang, M. M. Du and W. S. Bu, "Research of Networked Control System Based on Improved Smith Predictor and Kalman Filter", ICIC Express Letters, vol. 7, no. 10, **(2013)**, pp. 2765-2771.
[6] H. Y. Chen, Q. Guan and W. L. Wang, "Design of a fuzzy PI controller with Smith predictor for networked control systems with long time delay", vol. 33, no. 4, **(2005)**, pp. 418-420.
[7] R. Q. Lin and F. W. Yang, "Realization of a Class of Neuron Controller Based on Smith Predictor", Information and Control, vol. 33, no. 2, **(2004)**, pp. 137-140.
[8] J. K. Liu, "Matlab Simulation of Advanced PID Control", Publishing House of Electronics Industry, Beijing, China, **(2011)**.
[9] H. T. Zhang and Z. Li, "Simulation of Networked Control System Based on Smith Compensator and Single Neuron Incomplete Differential Forward PID", Journal of Networks, vol. 6, no. 12, **(2011)**, pp. 1675-1681.