Research on Seeker Servo Platform Based on BP neural network controller

Wei Tang, huhai Jiang*, bin Pu, Tongtong Zhang, Rui Mao
South-west institute of technical physics, Chengdu, 610041, China

*Corresponding author Email: jhhqgj@163.com

Abstract. Seeker is the core of precision guided weapon. The performance of its servo mechanism has a decisive impact on the accuracy of guidance weapon. The controller design of seeker servo system is of great significance to improve the function of seeker servo system. The mathematical model of the seeker servo system in the form of transfer function is built. The traditional PID control and BP neural network PID control methods are used to design the controller respectively. Aiming at the limitation of BP neural network, the BP neural network is improved. Finally, the simulation experiment is carried out in Matlab / Simulink. Simulation results show that the neural network PID controller has better performance than the traditional PID controller, so that the system has better adaptability and robustness.

Keywords: Seeker; Neural network; PID control.

1. Introduction
With the development and continuous improvement of science and technology, precision guided weapons are the main weapons of modern warfare and an important part of information warfare. Guided weapons have become the world's military applications in the focus of research and the race to develop one of the key technologies. The guide head is the core of the precision guided weapons, its servo mechanism performance on the accuracy of the guided weapons has a decisive impact on [1], The PID controller has the advantages of simple structure, high reliability and flexible control, and is commonly used in different intelligent control areas. However, due to the actual industrial control process, the controlled object is often complex mechanism, there are non-linear and time-varying characteristics, the application of conventional PID controller is not only difficult to set parameters and often can not get the desired control effect [2].

Currently, neural networks with self-learning capabilities are proposed to be widely used in PID control research to solve the control problem of complex dynamically uncertain systems [3]. Through the characteristics of BP (Back Propagation) neural network, the three parameters of the PID controller are automatically adjusted, which has good adaptability compared to the traditional PID control algorithm. After the introduction of BP neural network control, the response speed and adaptive capability of the servo-stabilized platform are improved, which makes the system more adaptable and robust.
2. BP neural network PID control algorithm principle

2.1. Models of Neural Networks

BP neural network is an artificial neural network based on error back propagation algorithm. As shown in Figure 1, it is a three-layer BP neural network structure, which contains input layer, implicit layer and output layer, where the neuron state of the output layer corresponds to the PID controller parameters $K_p$, $K_i$ and $K_d$. When the external state changes, the neural network outputs the PID control parameters corresponding to the optimal control law through the error back propagation and weighting factor adjustment.

Traditional incremental digital PID control algorithms [4] are

$$u(k) = u(k - 1) + K_p[e(k) - e(k - 1)] + K_i e(k) + K_d[e(k) - 2e(k - 1) + e(k - 2)]$$

(1)

In the formula, $u(k)$ is the output of the PID controller, $K_p$, $K_i$ and $K_d$ are the proportional, integral and differential control parameters respectively, and $e(k)$ is the deviation between the system setting value and the actual output value.

The three-layer BP neural network shown in Figure 1 is introduced, and the input layer input to the network is:

$$O^{(1)} = x(j) \ (j = 1, 2 \ldots M)$$

(2)

Where the number of input variables $M$ depends on the complexity of the system under control.

![Figure 1 Structure of the BP neural network](image)

Implicit layer i neuron input:

$$net_i^{(2)}(k) = \sum_{j=0}^{M} w_{ij}^{(2)} O_j^{(1)}$$

(3)

Output of the i-th neuron of the implicit layer:

$$O_i^{(2)}(k) = f \left( net_i^{(2)}(k) \right) \ (i = 1, 2 \ldots m)$$

(4)

Where $w_{ij}^{(2)}$ is the implicit layer weighting factor, and superscript (1), (2), and (3) represent the input, implicit, and output layers, respectively.
Using a positively or negatively symmetric Sigmoid-type function as an activation function for implicit layer neurons $f(\cdot)$:

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$ (5)

Output layer I neuron input:

$$net_i^{(3)}(k) = \sum_{l=0}^{m} w_{il}^{(3)} O_l^{(2)}$$ (6)

Output layer I neuron output:

$$O_l^{(3)}(k) = g\left(net_i^{(3)}(k)\right) \quad (l = 1, 2, 3)$$

$$O_1^{(3)}(k) = K_p$$

$$O_2^{(3)}(k) = K_i$$

$$O_3^{(3)}(k) = K_D$$ (7)

Where $O_l^{(3)}(k)$ denotes the three output nodes of the neural network output layer, corresponding to the three parameters $K_p$, $K_i$, and $K_D$ of the PID controller, respectively.

The activation function $g(\cdot)$ of the output layer neurons takes a non-negative S-function:

$$g(x) = \frac{e^x}{e^x + e^{-x}}$$ (8)

The error back propagation algorithm is to start with the output layer of the neural network, and while back-calculating the output error of each layer of neurons in the network, the gradient descent learning algorithm corrects the thresholds and weights of each layer in the network to realize that the adjusted output value of the neural network can be close to the target value. The performance indicator function is taken as:

$$E(k) = \frac{1}{2} (rin(k) - yout(k))^2$$ (9)

Where $rin(k)$ and $yout(k)$ are the input and output of the system, respectively.

The neural network adjusts the search in the direction of the negative gradient of the weighting coefficients by $E(k)$, corrects the weighting coefficients of the network, and addresses the shortcoming that it is prone to falling into local minima by adding an inertia term that causes the search adjustment to converge quickly to global minima.

$$\Delta w_{li}^{(3)}(k) = -\eta \frac{\partial E(k)}{\partial w_{li}^{(3)}} + \alpha \Delta w_{li}^{(3)}(k - 1)$$ (10)

Where $\eta$ is the learning rate and $\alpha$ is the inertia coefficient. The $\frac{\partial E(k)}{\partial w_{li}^{(3)}}$ in Eq. (10) can be expressed as:

$$\frac{\partial E(k)}{\partial w_{li}^{(3)}} = \frac{\partial E(k)}{\partial y(k)} \cdot \frac{\partial y(k)}{\partial u(k)} \cdot \frac{\partial u(k)}{\partial O_l^{(3)}(k)} \cdot \frac{\partial O_l^{(3)}(k)}{\partial net_i^{(3)}(k)} \cdot \frac{\partial net_i^{(3)}(k)}{\partial w_{li}^{(3)}(k)}$$ (11)
Where $y(k)$ is the transfer function of the accused object.

And

$$\frac{\partial net_i^{(3)}(k)}{\partial w_{ii}} = O_i^{(2)}(k)$$  \hspace{1cm} (12)

Since $\frac{\partial y(k)}{\partial u(k)}$ is unknown, it is here approximated by replacing it with the symbolic function $\text{sgn}(\frac{\partial y(k)}{\partial u(k)})$, and the resulting effect of computational impreciseness is compensated for by adjusting the learning rate $\eta$.

From equation (1) and equation (7) we get.

$$\frac{\partial u(k)}{\partial o_1^{(3)}(k)} = e(k) - e(k - 1)$$  \hspace{1cm} (13)

$$\frac{\partial u(k)}{\partial o_2^{(3)}(k)} = e(k)$$  \hspace{1cm} (14)

$$\frac{\partial u(k)}{\partial o_3^{(3)}(k)} = e(k) - 2e(k - 1) + e(k - 2)$$  \hspace{1cm} (15)

The above analysis leads to the following learning algorithm for the network output layer weighting factor:

$$\Delta w_{ii}^{(3)}(k) = \alpha \Delta w_{ii}^{(3)}(k - 1) + \eta \delta_i^{(3)}(k)$$  \hspace{1cm} (16)

$$\delta_i^{(3)} = e(k) \text{sgn}\left(\frac{\partial y(k)}{\partial u(k)}\right) \frac{\partial u(k)}{\partial o_i^{(3)}(k)} g'\left(\text{net}_i^{(3)}(k)\right)$$

$$l = 1, 2, 3$$  \hspace{1cm} (17)

Where $g'(\cdot) = g(x)(1 - g(x))$

The learning algorithm for implicit layer weighting coefficients can be obtained by the same reasoning:

$$\Delta w_{ij}^{(2)}(k) = \alpha \Delta w_{ij}^{(2)}(k - 1) + \eta \delta_i^{(2)}(k)$$  \hspace{1cm} (18)

$$\delta_i^{(2)} = f'\left(\text{net}_i^{(2)}(k)\right) \sum_{l=1}^{3} \delta_l^{(3)} w_{il}^{(3)}(k)$$

$$l = 1, 2, \ldots, m$$  \hspace{1cm} (19)

Where $f'(\cdot) = (1 - f(x)^2)/2$

The above algorithm allows the three parameters of the PID corresponding to the network output layer to be adaptively adjusted.

2.2. Architecture of the BP Neural Network PID Controller

The structure of the BP neural network PID controller is shown in Figure 2[5]. It consists of two parts: the traditional PID controller and the BP neural network. The traditional PID controller carries out closed-loop control of the controlled object and is responsible for the positive conduction of the control signal, while the BP neural network corrects the weights of each layer of the neural network according to the error caused by the system operation and external state changes, adjusts the three PID parameters
corresponding to the output layer of the network, and adjusts the controller parameters $K_P$, $K_I$, $K_D$ online to achieve certain performance targets. The Optimization.

![Figure 2 Architecture of a BP neural network PID controller](image)

**Figure 2** Architecture of a BP neural network PID controller

### 2.3. Algorithmic flow

From the above analysis, the algorithmic flow of the BP neural network PID control is summarized in Figure 3.

![Figure 3 Flowchart of the BP neural network PID algorithm](image)

**Figure 3** Flowchart of the BP neural network PID algorithm
3. Servo system based on Matlab

3.1. General structure of the servo system
An example of a dual closed-loop guide head servo system is shown in Figure 4.

As can be seen in the diagram, the guide head stabilization system has two internal and external circuits, which form a double closed-loop cascade control system. The inner loop is the stabilization loop, which is used to isolate the disturbance of the carrier and achieve optic axis stability. The rate gyro is the core component of the stabilization loop, which constitutes the closed-loop feedback of the stabilization loop. The outer ring is the tracking loop, which mainly serves to track the target.

3.2. Servo System Matlab Simulation
According to the structure schematic and specific parameters, the Simulink model of the guide head servo platform was built in Matlab, and a PID controller based on BP neural network was introduced in the outer ring, and the simulation model is shown in Figure 5.

In order to verify the designed neural network controller, a step signal is input to the simulation model to obtain the step response of the system and compare it with the control system whose outer ring is a traditional PID controller. The step response curve is shown in Figure 6. It can be seen from the
figure that, compared with the traditional PID controller based on BP neural network, the response adjustment time $t_s$ is reduced from 0.62s to 0.12s, the rise time $t_r$ is reduced from 0.026s to 0.008s, and the overshoot of the step response $\sigma$ is reduced from 4\% to 1.5\%. It can be seen that the dynamic performance of the BP neural network based controller is superior to that of the traditional PID controller.

The adjustment of the BP neural network on the control parameters is shown in Figure 7, and the weighting coefficients are continuously adjusted in the beginning period of time, and after obtaining the optimal value of the control parameters, the step response curve of the system quickly reaches the steady-state value, and the output of the system is stabilized at the set value at 0.12s. Because the output layer of the BP neural network is a unipolar Sigmoid function, the controller parameters are fixed between (0,1), and to meet the demand of the stable system for the controller parameters, the controller parameters are amplified here.
4. Concluding remarks
In this paper, a PID controller based on BP neural network is designed for the guide head servo platform, and it can be seen from the simulation example that after the introduction of the BP neural network PID controller, the ability of online learning and adjusting the weighting factor of BP neural network can be used to adjust the three parameters of the PID controller online when the external environment changes, so that the control system can quickly reach a stable state. Compared with the traditional PID controller, the BP neural network based PID controller can effectively reduce the amount of system overshoot, reduce the adjustment time and rise time of the system, which is conducive to improving the dynamic performance of the servo platform.

References
[1] Zhu Huazheng. Research on some basic problems of imaging guide head servo mechanism [D]. National University of Defense Technology. 2012:12-14.
[2] Liu Di, Zhao Jianhua. An adaptive PID control algorithm based on BP neural network model [J]. Automation Technology and Applications, 2008, 7(8):8-10.
[3] Shu Hua, Zhao Jianhua. Design and simulation of a LabView-based PID neural network controller [J]. Journal of System Simulation, 2006, 18 (10): 2918-2920.
[4] Liu Jinkun. Advanced PID control and its MATLAB simulation [M]. Beijing: Electronic Industry Press, 2003.96-98.
[5] Hongbin Du, Huihe Shao. Adaptive neural network control for a class of nonlinear systems [J]. Control and Decision, 2005, (4): 455-48.