A BP Neural Network Controller for Magnetic Suspended Flywheel System

Chen Xiaofei[1,a], Liu Kun[1,b]

[1]Coll. of Aerospace and Material Eng., National Univ. of Defense Tech., Changsha, China
[a]areo1981@hotmail.com, [b]liukun@nudt.edu.cn

Abstract: A BP neural network controller is proposed for direct suspending control for Magnetic Suspended Flywheel System (MSFS) that is supported by Active Magnetic Bearings (AMB). A one hidden layer configuration is adopted in the BP neural network, and the back propagated algorithm for network weights updating is derived based on AMB's linear model. The discussed controller is implemented in the MSFS with random initial network weights, and it is trained online as the whole system operated. Simulations show the proposed BP neural network controller is apt to succeed in suspending the flywheel, and better performances such as precise position control, disturbance rejection, vibration suppression and quiet control are achieved under power consumption limitation. The results validate the feasibility and effectiveness of the presented BP neural network controller.

Keywords: Magnetic Suspended Flywheel System, Active Magnetic Bearing, BP Neural Network, Disturbance Rejection, Vibration Suppression

Introduction

The discussed Magnetic Suspended Flywheel (MSFS) is used as attitude control actuator in spacecraft [1, 2]. Unlike conventional flywheel that is supported by contact bearing, flywheel of MSFS is suspended by Active Magnetic Bearings (AMB), so that friction and abrasion can be avoid, long life cycle is guaranteed and the problem of rotation speed crossing zero is eliminated too. Moreover, active vibration control is feasible for AMB to suppress MSFS' vibration.

MSFS or AMB is a typical system of nonlinearity. For example, magnetic force of AMB is proportional to the square of current in electromagnets, while it is inversely proportional to displacement square of flywheel. In addition, there are many coupling effects in MSFS and AMB such as gyroscopic torque, magnetic circuit coupling, and so on. Because of nonlinearity, the suspending control is difficult, although the flywheel may be suspended by a simple PD controller, however, advanced method is required for better performances, such as H∞ synthesis [3], Linear Parameter Varying (LPV) method [4], Linear Matrix Inequality (LMI) method [5], neural network [6, 7]and so on.

A robust controller for AMB is modeled in [3], the most often used H∞ synthesis is discussed. However, the synthesized controller is usually in high order and time consumption for digital application. In [4], a LPV model is discussed with respect to rotation speed, and uncertainty weights are identified via neural network. A LMI method is proposed to set parameters of PD controller by Bian in [5], in this work, asymptotically stability, H∞ robust synthesis and LQR are unified into a problem of inequality. The methods in [4] and [5] are quite good, however, the calculation is rather complex.

Neural network is good at nonlinear mapping, self learning and concurrent information processing, thus provide a good choice for nonlinear system control. Usually, add and multiply are basic operation of neural network, and make the neural network suitable for digital application. Moreover, controller modeled based on neural network is adaptive to disturbance as the neural network has the ability of self learning.

In [6], an adaptive neural network is introduced to overcome disturbance and system nonlinearity. However, neural network in [6] is much like a filter that provides rotor's position signal for PD controller, besides, an adaptive subsystem is needed, and complexity of the control system is increased. In [7], a multi-layer perceptron controller was proposed for AMB by Roger while the network is trained offline.

Aimed at the direct neural network control and online training, a 2 layers BP neural network controller will be presented in the following part. First, single freedom model of AMB is discussed, then the BP network controller is introduced and the learning algorithm is derived based on the AMB model, next, simulations are performed to validate the performance of the proposed BP neural network controller.

Model of MSFS

To decrease the power consumption, the bias magnetic field of the discussed MSFS is generated by permanent magnets, and inner stator - outer rotor configuration is taken [1, 2]. Cross section of structure and magnetic circuit of MSFS is illustrated in Fig. 1.

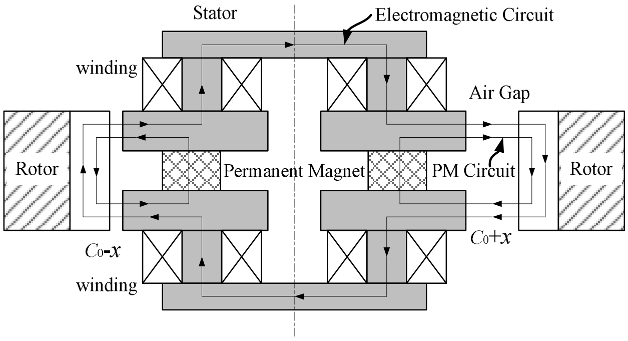


Fig. 1. Cross Section of Structure and Magnetic Circuit of MSFS

Here in Fig. 1, boxes filled with lattice are permanent magnets, boxes with two cross lines are windings of electromagnets, the box hatched with oblique lines is flywheel rotor which is shown as two cross sections, and boxes in gray are magnetic conductive components of the stator.

In Fig. 1, permanent magnetic circuit is generated by permanent magnets, and the magnetizing direction of permanent magnets is vertically upward. When the rotor is moved to the right and no electromagnetic circuit is taken effect, denotes x as the displacement, the thickness in right airgap is $C_0 + x$, and $C_0 - x$ in left airgap, a right direction force is provided by permanent magnets since magnetic flux density in left airgap is bigger than the one in right airgap. Here C_0 is the nominal thickness of airgap.

Hence, electromagnetic circuit is generated clockwise to increase magnetic flux density in right airgap and decrease correspondingly in left airgap, then, a left direction force can be provided to drag the rotor back. In a similar way, an anti-clockwise electromagnetic circuit is generated when the rotor is moved to the left.

Therefore, by balancing the magnetic force, the displacement x is kept near to zero and the rotor is suspended. Note that, if $x = 0$, the flywheel is suspending in equilibrium point, or operation point; if $x = C_0$, the rotor is in contact with auxiliary bearings which support the flywheel in case of no suspending.

Generally, the magnetic force F generated by AMB is given as

$$F = \frac{\mu_0 S_g N^2}{2}\left[\left(\frac{I_m - i}{C_0 - x}\right)^2 - \left(\frac{I_m + i}{C_0 + x}\right)^2\right], \tag{1}$$

here in Eq. 1, μ_0 is magnetic permeability in vacuum, S_g is the area of magnetic pole at the airgap, N is the turns of winding, I_m is bias current (provided by permanent magnets), i is control current. Equation Eq. 1 is the single freedom nonlinear model of AMB. According to magnetic circuit theorem, bias current can be given as

$$I_m = fC_0 B_g / \mu_0 N\ , \tag{2}$$

here, f is magnetic reluctance, B_g is magnetic flux density generated by permanent magnets in airgap. Expanding Eq. 1 in Taylor series at operation point, ignoring high order items, we get the magnetic force equation in linear form

$$m\ddot{x} = k_x x + k_i i\,, \tag{3}$$

here, m is the mass of flywheel rotor, k_x is negative displacement stiffness, and k_i is current stiffness. It is shown in Eq. 3 that stable suspension cannot be obtained by negative stiffness without control force.

BP Neural Network Controller

Principle of BP Neural Network. BP neural network is actually multilayer perceptron network updated in the way of back propagation, and nonlinear mapping ability is enhanced by the multilayer construction. A 2 layers BP neural network is shown in Fig.2.

Here in Fig. 2, **p** is network input vector, **w** is layer weight matrix, **b** is layer bias, **f** is transfer function vector with respect to input vector **n**, superscript ‘(1)’ denotes hidden layer, and superscript ‘(2)’ denotes output layer, $\mathbf{a}^{(2)}$ is control signal output by the network.

The input and output relationship of BP neural network is given as

$$\begin{aligned} &\mathbf{n}^{(1)} = \mathbf{w}^{(1)}\mathbf{p} + \mathbf{b}^{(1)}, &&\mathbf{a}^{(1)} = \mathbf{f}^{(1)}(\mathbf{n}^{(1)}) \\ &\mathbf{n}^{(2)} = \mathbf{w}^{(2)}\mathbf{a}^{(1)} + \mathbf{b}^{(2)}, &&\mathbf{a}^{(2)} = \mathbf{f}^{(2)}(\mathbf{n}^{(2)}) \end{aligned}, \tag{4}$$

in Eq. 4, given m inputs, l outputs and n neurons in hidden layer, then, **p** is column vector with m rows, $\mathbf{w}^{(1)}$ is matrix with n rows and m columns, $\mathbf{b}^{(1)}$ is column vector with n rows, $\mathbf{w}^{(2)}$ is matrix with l rows and n columns, $\mathbf{b}^{(2)}$ is column vector with l rows. The element of function vector $\mathbf{f}^{(1)}$ and $\mathbf{f}^{(2)}$ is usually sigmoid transfer function, e.g., $f(x) = (e^x - e^{-x})/(e^x + e^{-x})$, or pureline function $f(x) = x$.

Assume the expected network output is **d**, there exist energy function $E = (\mathbf{d} - \mathbf{a}^{(2)})^T(\mathbf{d} - \mathbf{a}^{(2)})$, a minimum E must be guaranteed by the learning algorithm, thus according to Least Mean Square (LMS) algorithm, the weights should be updated as

$$w_{i,j}^{(h)}(k+1) = w_{i,j}^{(h)}(k) - \alpha\frac{\partial E}{\partial w_{i,j}^{(h)}} + \beta\Delta w_{i,j}^{(h)}(k-1), \qquad b_i^{(h)}(k+1) = b_i^{(h)}(k) - \alpha\frac{\partial E}{\partial b_i^{(h)}} + +\beta\Delta b_i^{(h)}(k-1)\,,\quad h = 1,2\,, \tag{5}$$

here α is learning speed, β is momentum factor. The weights are updated from output layer to hidden layer sequentially, i.e., back propagation. Momentum factor is introduced here to avoid local minimization and to ensure a global minimization.

BP Neural Network Controller. BP neural network controller in control system diagram is illustrated in Fig. 3.

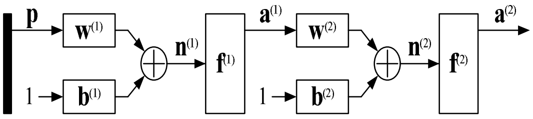


Fig. 2. A 2 Layers BP Neural Network

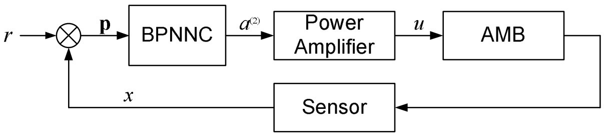


Fig. 3. BP Neural Network Controller in Control System Diagram

The controller is implemented into the system, and $\mathbf{a}^{(2)}$ is set to one scalar value $a^{(2)}$ for direct control (can also be set as vector with 3 elements for PID control or parameters tuning). Here, r is the reference signal setting rotor's position, network input vector $\mathbf{p}$ is generated after the sum point for calculating network output $a^{(2)}$, the control signal u is then output by power amplifier according to $a^{(2)}$, and rotor displacement x is controlled by u finally.

For discrete time control system, $\mathbf{p}$ is usually set as $\mathbf{p}(k) = [x(k), \Delta x(k), \Delta\Delta x(k)]^T$, here, $\Delta x(k) = x(k) - x(k-1)$ and $\Delta\Delta x(k) = x(k) - 2x(k-1) + x(k-2)$. All the layer bias are set to zero for simplifying network calculation, and the output is set to one scalar value, i.e., l=1.

Expected output $\mathbf{d}$ is set as the reference signal r, then the energy function will be

$$E = (r-x)^2 , \tag{6}$$

and weights update algorithm is given as

$$w_{1,i}^{(2)}(k+1) = w_{1,i}^{(2)}(k) - \alpha \frac{\partial E}{\partial w_{1,i}^{(2)}} + \beta \Delta w_{1,i}^{(2)}(k-1), \qquad w_{i,j}^{(1)}(k+1) = w_{i,j}^{(1)}(k) - \sigma \frac{\partial E}{\partial w_{i,j}^{(1)}} + \eta \Delta w_{i,j}^{(1)}(k-1) , \tag{7}$$

here in Eq. 7, i=1,2,…,n, is the sequence of hidden layer neurons, j=1,2,…,m, is the sequence of input neurons, α and σ are learning speed, β and η are momentum factor. Partial derivatives in Eq. 7 are written as

$$\frac{\partial E}{\partial w_{1,i}^{(2)}} = \frac{\partial F}{\partial x}\frac{\partial x}{\partial u}\frac{\partial u}{\partial a^{(2)}}\frac{\partial a^{(2)}}{\partial n^{(2)}}\frac{\partial n^{(2)}}{\partial w_{1,i}^{(2)}} = \frac{\partial x}{\partial u} x\dot{f}^{(2)}(n^{(2)})a_i^{(1)}, \qquad \frac{\partial E}{\partial w_{i,j}^{(1)}} = \frac{\partial x}{\partial u} x\dot{f}^{(2)}(n^{(2)})w_{1,i}^{(2)} \dot{f}_i^{(1)}(n_i^{(1)})p_j \text{ ,} \tag{8}$$

in Eq. 8, $\partial x / \partial u$ is derived based on Eq. 3 Rewrite Eq. 3 into difference equation with respect to time series, we have

$$x(k) = ax(k-1) - x(k+2) - bu(k) - 2bu(k-1) - bu(k-2) , \tag{9}$$

here in Eq. 9, a and b are coefficients, take careful when deriving Eq. 9 from Eq. 3, since the proportional gain of sensors must be considered. Assuming the sampling time is small enough, and according to the definition of partial derivative, we get

$$\frac{\partial x}{\partial u} = \lim_{\Delta u \to 0} \frac{x(t, u+\Delta u) - x(t,u)}{\Delta u} \approx \frac{x(k, u(k+1)) - x(k, u(k))}{u(k+1) - u(k)} , \tag{10}$$

and finally, we have

$$\frac{\partial x}{\partial u} = \frac{-bu(k) - bu(k-1) + bu(k-2) + bu(k-3)}{u(k) - u(k-1)} = -b + b\frac{\Delta u(k-2)}{\Delta u(k)} , \tag{11}$$

The program of network control is a matter of work combining the above equations. The network starts with random weights, after reading input vector $\mathbf{p}$, the output of BP neural network controller is calculated according to Eq. 4, then, weights are updated according to equations from Eq. 7 to Eq. 11, at the next time, network output is calculated with the updated weights and new input vector $\mathbf{p}$, the weights are further updated according to the new output, and so on. Usually, weights updating should be terminated when energy function is small enough.

Simulation and Discussion

Simulation Settings. Simulation is performed in MATLAB Simulink. MSFS is modeled according to Eq. 1. Rotor displacement and output of power amplifier are constrained to actual values. The sampling time Ts is set to 0.0001s, overall simulation time is 0.4s. Some related parameters of MSFS are listed in Table 1.

It is assumed that the MSFS is operated at rotation speed of 4800RPM, vibrations in synchronous, subharmonic (120Hz) and ultraharmonic (40Hz) are considered, the vibration forces correspondingly are 200N, 100N and 80N, all of the forces are much bigger than actually measured values. Moreover, noise and disturbance are introduced into modeled MSFS system. The variance of force noise is 100N, noise magnitude of sensor's output is 0.1V. (Sensor's output range is -2.4V to 2.4V and the p-p value of rotor vibration is usually smaller than 0.2V.) A step disturbance of 200N at time 0.1s is applied, and at time 0.2s, reference position signal is change from 0 to 1V.

Table 1. Parameters of MSFS

Parameter Name	**Value**
Turns of winding N	150
Magnetic pole area S_g	2×10^{-4}[m^2]
Bias current I_m	3[A]
Nominal airgap C_0	0.5[mm]
Rotor displacement range x	-0.3~0.3[mm]
Initial rotor position x_0	0.3[mm]
Mass of flywheel m	5[Kg]
Gain of power amplifier	1
Time constant of power	0.0003
Gain of sensor	8000[V/m]

An elaborately tuned PD controller is selected to compare the performance. The transfer function is given as

$$C(s) = 2\square\frac{0.0030s+1}{0.0005s+1}\square\frac{0.0030s+1}{0.0005s+1}. \tag{12}$$

Besides, an integrator is used to eliminate the steady state error of the PD controller.

Power consumption index is introduced here to compare power consumption approximately. It is calculated as accumulated absolute values of control current at every time. Meanwhile, magnetic forces are recorded and compared, for smaller magnetic force usually means that the system is running in a quiet way, which is useful for spacecraft.

Network Training. The network is implemented in MSFS directly and trained online. The initial weights of BP neural network controller are random numbers between 0 and 1, the learning process is performed according to equations from Eq. 7 to Eq. 11, it is noticed that, the network weights are limited between 0.1 and 0.9 for the stability of updating.

The start position of the rotor is set to 0.3mm away from working point (that is, in contact with auxiliary bearings), the input vector is set as $\mathbf{p}=[x,\Delta x,\Delta x]^T$, the controller gain is 50, select sigmoid function as hidden layer transfer function and pureline function as output layer transfer function, the success rate of training is almost as high as 100%. The historical diagram of rotor displacement during network training is shown in Fig. 4, the historical diagram of weights of hidden layer during network training is illustrated in Fig. 5, and the historical diagram of weights of output layer during network training is presented in Fig. 6. In Fig. 5, 'w1-xy' is the element in x-th row and y-th column of hidden layer weight matrix $\mathbf{w}^{(1)}$. In Fig. 6, 'w2-z' is the z-th element in output layer weight matrix $\mathbf{w}^{(2)}$. In Fig. 5 and Fig. 6, the value of weight is measured by y-axis.

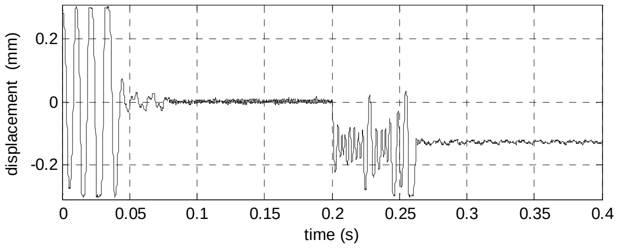


Fig. 4, Historical Diagram of Rotor Displacement during Network Training

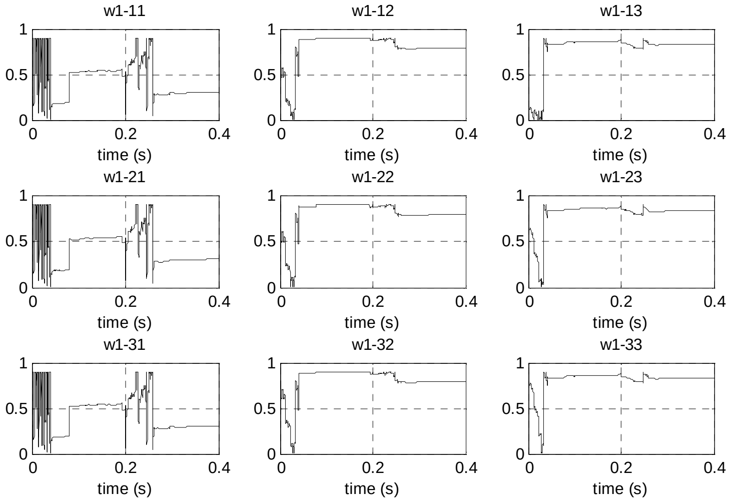


Fig. 5, Historical Diagram of Weights of Hidden Layer during Network Training

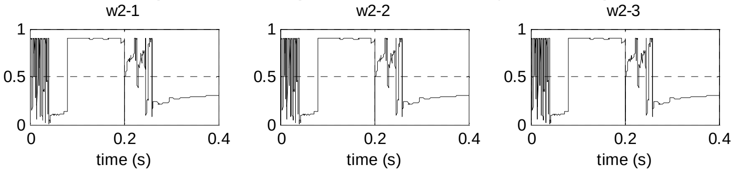


Fig. 6, Historical Diagram of Weights of Output Layer during Network Training

In Fig. 4, vibration is observed at the beginning of suspending as the flywheel need to be suspended form initial position; the controller 'learns' how to suspend the flywheel at time 0.05s, and then does the work better as time goes on; the reference position is set to a step change at 0.2s, as a result, the rotor oscillates around the new reference position for a while, however, the controller starts a new 'learning' process and finally turns out to be able to stabilize the rotor again after the time 0.26s.

Different learning processes of weights are shown in Fig. 5 and Fig. 6. There are some common phenomena during the training process, and further discussion can be addressed on learning process. Yet, what the important is, visualized supports are provided in Fig. 5 and Fig. 6 for initial weights selection.

Note that all the disturbances are acted in the system, it can be seen form Fig. 4 to Fig. 6, the proposed learning algorithm derived based on Eq. 3 is apt to success and is robust to disturbance and vibration.

Performance Discussion. Further simulation is performed based on the above trained weights leaving other initial condition unchanged.

Power consumption index of BP neural network controller is shown in Fig. 7. The dashed line stands for power consumption index of PD controller with respect to a fixed gain. Generally, by increasing the gain of PD controller, high frequency vibration will be introduced to AMB-rotor system, and power consumption will also be increased remarkably. So, the gain can only varied in a small domain and has little effect on power consumption index. The solid line is the power consumption index of BP neural network controller with the gain varying from 5 to 65. Obviously, the power consumption index of proposed BP neural network is lower than that of the PD controller when the network gain varying between 10 and 60. Here, the network gain is fixed at 30 for the following simulation and discussion.

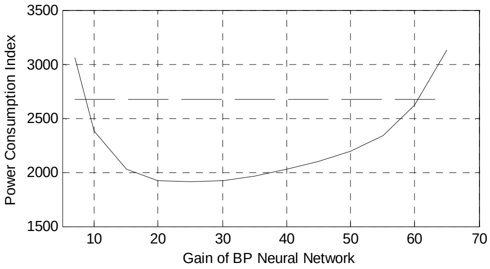
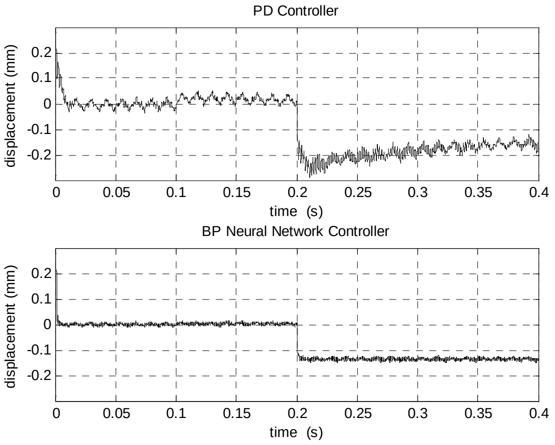


Fig. 7. Power Consumption Index of BP Neural Network Controller

Flywheel’s displacement and vibration with PD control and BP neural network control are presented in Fig. 8. In the first subfigure, the process of suspending from initial position is almost one rotation period, the flywheel synchronous vibration is obvious with p-p value about 50 μm, and a jump in displacement is occurred when force disturbance is applied at time 0.1s, a long time tracking process (more than 0.2s) is needed after a step change in reference position. However, in the second subfigure, compared with PD control, the rotor is suspended immediately with BP neural network controller, the displacement signal is quite flat and insensitive to force disturbance, the synchronous vibration is small and unapparent, and the reference position signal is tracked better. Fig. 8 indicate that the proposed network controller yields better performance such as precise position tracking, vibration suppression and disturbance rejection.

Magnetic forces with PD control and BP neural network control are shown in Fig. 9. The observation noise in simulation is removed to give a clearly view of magnetic forces.
In the first subfigure, magnetic force is sharply increased and last for a long time in initial suspending process or when there is a step change in reference position. In the second subfigure, the magnetic force is smaller than that of the first subfigure, although there are still two sharply increased magnetic forces, however, just lasting for a short time. Apparently, magnetic force is smaller using BP neural network controller than that of PD controller, and a quiet control is guaranteed.

Note that the above results are constrained by power consumption, thus, the benefits of BP neural network can be summarized as disturbance rejection, vibration suppression, precise position tracking and quiet control.



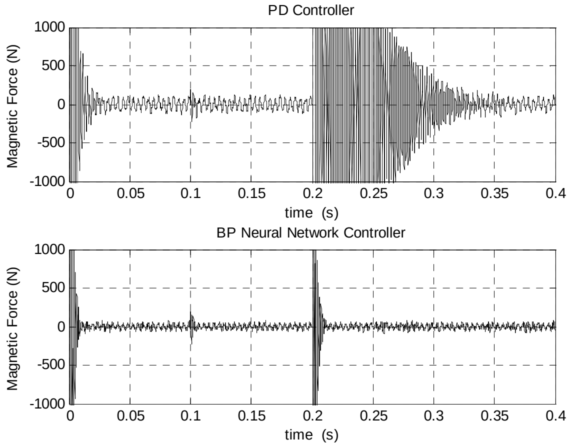


Fig. 8, Flywheel’s Displacement and Vibration with PD Control and BP Neural Network Control

Fig. 9. Magnetic Forces with PD Control and BP Neural Network Control

Conclusion

A BP neural network controller for MSFS is addressed in this work. The direct suspending control and online training are achieved toward real application. The performed simulations indicate that the proposed BP neural network controller is apt to succeed in suspending the flywheel, and precise position control is guaranteed also. Moreover, better performances such as disturbance rejection, vibration suppression and quiet control are obtained under power consumption limitation.

The next effort should be dSPACE implementation of the discussed BP neural network controller. 4 axes decentralized control or centralized control would be discussed, and gyroscopic torque of flywheel must be taken into consideration.

References

[1] Xiao Kai: Control on Hybrid Magnetic Bearings and Flywheel. (Graduate School of National Univ. of Defense Tech., Changsha, 2006)

[2] Wu Gang: System Design and Controller of Hybrid Magnetic Bearing. (Graduate School of National Univ. of Defense Tech., Changsha, 2006)

[3] Zdzislaw G, Arkadiusz M: Robust control of active magnetic suspension: Analytical and experimental results. Mechanical Systems and Signal Processing. Vol. 22(2008), p.1297-1303.

[4] Bei L, Heeju C, Gregory D B: Linear Parameter-Varying Techniques for Control of a Magnetic Bearing System. Control Engineering Practice. Vol. 16(2008), p.1161-1172.

[5] Bian Bin, Liu Shuqin: Design of Robust Optimal PID Control for Magnetic Bearing via LMI Approach. Proc. of 3rd China Symp. of Magnetic Bearing (CSMB3), August, 2009, p. 40-44.

[6] Wang Jieqiong, Fang Jiancheng, Liu Gang: Adaptive Neural Network Control of Magnetically suspended Reaction-Wheel with High Accuracy. Journal of System Simulation. Vol. 20(2008), p.85-88.

[7] Roger A, Chaiban N, Jerome D M, in: A New Method of Controlling Active Magnetic Bearing through Neural Network. Proc. of the 2006 IEEE International Conference on Control Applications. IEEE Press, 2006, p. 2842-2847.