BACKPROPAGATION NEURAL NETWORK CONTROL FOR DC-MOTOR STICTION COMPENSATION

Anthony Tzes Pei-Yuan Peng
Polytechnic University
Department of Mechanical Engineering
333 Jay Street
Brooklyn, NY 11201

Abstract

The application of a backpropagation neural network controller for compensating the effects induced by the static friction in a DC-motor system is considered in this paper. The input vector to the neural network controller consists of the time history of the motor angular shaft velocity within a prespecified time window. The network weights are adapted through a learning process which minimizes a quadratic function of the motor system output error. The proposed neural network controller is demonstrated and contrasted with a PI controller through simulation studies on an example system.

I. Introduction

This study is motivated by the increasing interest in designing controllers for high-accuracy applications such as force control [1] and micropositioning of robot manipulators [2]. The DC-motor static friction possesses its greatest challenge when the motor is commanded to perform a tiny motion at a corresponding low velocity in which the minimum achievable displacement and sustained velocity arise from the stick-slip motion phenomenon [3]. The classical way to reduce the effects of friction on the system response is to apply either jittering (high frequency excitation signal) and a deadband in an integral control technique. Both methods counter to the demands of high-fidelity control and adaptive algorithms have recently appeared [4,5] in the literature.

In this paper, an adaptive scheme based on the backpropagation neural network algorithm [6,7] is proposed to compensate the effects of friction in a typical DC-motor micropositioning problem.

II. Problem Statement

A. DC-Motor Modeling

A representative block diagram of the DC-motor example system appears in Figure 1. If the motor and load inertia are reflected to the motor axis, the dynamic equations can be written as :

$$J\frac{d\omega}{dt} + [b + \frac{K_a}{R_a}(K_b + AK_g)]\omega = \frac{AK_a}{R_a}u - T_f \qquad (1)$$

where ω is the motor shaft angular velocity, J and b are the effective moment of inertia and viscous friction coefficients, K_a the torque constant, R_a the armature winding resistance, K_b the back-emf constant, K_g the tachometer constant, A the amplifier gain, and T_f the friction torque. The effects of Coulomb friction τ_c and static friction τ_s are included within T_f ($T_f = \tau_c + \tau_s$). The friction model [3] adopted in this study for the Coulomb and static friction is

$$\tau_c = \tau_c^m \text{sign}(\omega), \quad \tau_s = \tau_s^m \exp^{-\frac{\omega}{\omega_s}} \qquad (2)$$

B. Backpropagation Neural Networks

The elementary Backpropagation NN (BNN) consists of artificial neurons clustered in a hierarchical three layer configuration. Information signals flow in a feedforward direction from the input layer neurons to the output layer through the hidden layer neurons [7]. The backpropagation encoding algorithm performs the input to output mapping by minimizing a cost function. This results in interlayer connection weight adjustments according to the error between the computed and desired system response. A gradient algorithm is employed for weight adjustment by computing the effect on the cost function with respect to the weight variation.

At time k, the input vector $\Omega(k)$ to the BNN input layer neurons consists of the past system output values $\Omega(k) = [\omega(k-1), \omega(k-2), \ldots, \omega(k-I-1)]$ as illustrated in Figure 1. The BNN output is derived in such a way that the quadratic

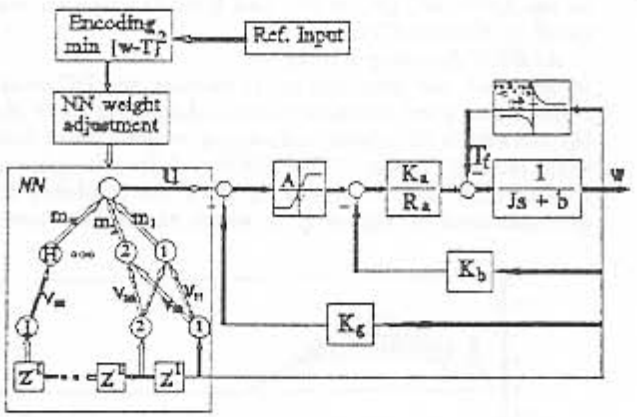

Figure 1: Backpropagation Neural Network Controller

cost function $E(k) = \frac{1}{2}[\omega^d(k) - \omega(k)]^2$ between the desired $\omega^d(k)$ and the actual system response $\omega(k)$ at every time instant is minimized. In this error minimization procedure the BNN output $u(k)$ does not emerge directly, since the desired BNN output is unknown.

1. Encoding BNN Algorithm

The backpropagation algorithm performs the input $\Omega(k)$ to output $u(k)$ mapping at every time instant k according to the following set of relationships.

Assume a BNN with I (H) neurons in the input(hidden) layer. The input layer neuron activations are filtered and propagated to the hidden layer neurons as :

$$p_h(k) = f(\sum_{i=1}^{I} \omega(k-i)v_{i,h}(k)) \quad h = 1, \ldots, H \qquad (3)$$

Similarly, the hidden layer neuron activations are filtered and propagated to the BNN output as:

$$u(k) = f(\sum_{h=1}^{H} m_h(k) p_h(k)) \quad . \tag{4}$$

2. Learning BNN Algorithm

During the learning process, the output error $E(k)$ is minimized through adaptation of the hidden layer neuron connection weights using the "sign gradient descent" algorithm [8] as :

$$\delta m_h(k+1) = -\alpha \frac{\partial E(k)}{\partial m_h(k)} = -\alpha \frac{\partial E(k)}{\partial \omega(k)} \frac{\partial \omega(k)}{\partial u(k)} \frac{\partial u(k)}{\partial m_h(k)} \quad , \tag{5}$$

where α is the learning coefficient (step size) along the learning surface. Although the term $\frac{\partial \omega}{\partial u}$ is unknown, it can be approximated by its sign. From equation (1), the phase response of the linear DC-motor transfer function is 0° at $\omega = 0$, and the aforementioned sign term can therefore be approximated as one. Moreover, if the hyperbolic tangent function ($f(x) = \tanh(x)$) is utilized for the nonlinear activating function, equation (5) can be written as

$$\delta m_h(k+1) = \alpha[\omega^d(k) - \omega(k)] p_h[1 + u(k)][1 - u(k)] \quad . \tag{6}$$

This process is repeated for modifying the input layer weights, and using the chain rule twice in succession the weight adjustment relationship is:

$$\delta v_{i,h}(k+1) = \alpha[\omega^d - \omega][1 - u^2] m_h[1 - (p_h)^2] \omega(k-i).$$

III. Simulation Case Study

The proposed scheme was applied in simulation studies on the E350-MG DC-Motor and E350-O amplifier manufactured by Electro-Craft Co.

A. BNN Learning Ability

In this case, the reference input exciting the DC-motor corresponds to a low frequency sinusoidal signal (0.66 Hz), and the network's interlayer connection weights were initialized with random values. The BNN's ability to generate the control input for the motor to track the reference signal is demonstrated in Figure 2, in which the upper (lower) segment of the figure corresponds to the system output during the initial 0-to-3 (6-to-9) second period. During the initial learning period, the BNN's weights are adapted to the nonlinear characteristics of the friction model. Having these weights converged to their "nominal" values, the DC-motor output is almost indistinguishable from the reference. The BNN weights' speed of convergence depends on the number of neurons in the input and hidden layers.

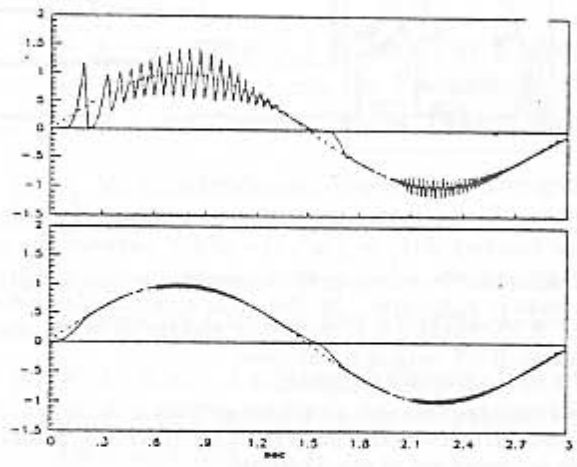

Figure 2: System Response with BNN Controller

B. BNN-PI controller comparative studies

To emphasize furthermore the robustness and the tracking ability of the BNN-controller we compared the BNN-compensated system's response shown in Figure 3, with the one when a Proportional Integral (PI) controller is utilized. The PI-controller gains were selected such that the performance index $J = \int_0^\infty (\omega - \omega^d)^2 + (\frac{du}{dt})^2 dt$ was minimized based on the assumption that there was no stiction in the DC-motor system. As anticipated, the system response suf-

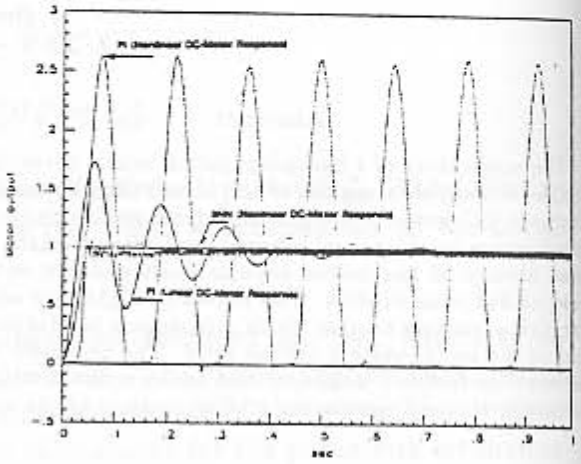

Figure 3: PI/BNN System Compensated Response

fered from oscillations (limit cycle), which the PI controller could not compensate. On the contrary, the BNN adjusted its weights to account for the stiction and overall manifested a remarkable system response.

References

[1] W. Townsend and J. Salisbury, "The Effect of Coulomb Friction and Stiction on Force Control," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 883-889, Raleigh NC, April 1987.

[2] C. Canudas, P. Noel, A. Aubin, B. Brogliato, and P. Drevet, "Adaptive Friction Compensation in Robot Manipulators: Low Velocities," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1352-1357, Scottsdale, AZ, May 1989.

[3] B. Armstrong-Helouvry, *Control of Machines with Friction*. Norwell, MA: Kluwer Academic Publs., 1991.

[4] C. Canudas, K. Astrom, and K. Braun, "Adaptive friction compensation in dc-motor drives," *IEEE Transactions on Robotics and Automation*, vol. 3, no. 6, pp. 681-685, December 1987.

[5] C. Canudas and V. Seront, "Robust Adaptive Friction Compensation," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1383-1389, Cincinatti, OH, May 1990.

[6] F. Chen, "Backpropagation neural networks for nonlinear self-tuning adaptive control," *Control Systems Magazine*, vol. 10, no. 2, pp. 44-48, April 1990.

[7] K.S. Narenda and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Transactions on Neural Networks*, vol. 1, pp. 4-27, March 1990.

[8] M. Saerens and A. Soquet, "A Neural Controller," in *Proceedings of the First IEE International Conference on Artificial Neural Networks*, pp. 211-215, London, UK, October 1989.