ESANN'2003 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 23-25 April 2003, d-side publi., ISBN 2-930307-03-X, pp. 173-178

Online Identification And Control of A PV-Supplied DC Motor Using Universal Learning Networks

Ahmed Hussein* Kotaro Hirasawa** Jinglu Hu**

* Graduate School of Information Science & Electrical Eng., Kyushu University
812-8581, 6-10-1, Hakozaki, Higashi-Ku, Fukuoka, JAPAN
** Graduate School of Information, Production and Systems, Waseda University
808-0135, 2-2, Hibikino, Wakamatsu-Ku, Kitakyushu, Fukuoka, JAPAN
Tel.: (+81) 92-642-3799 Fax: (+81) 92-642-3962
ahmed@cig.ees.kyushu-u.ac.jp

Abstract

This paper describes the use of Universal Learning Networks (ULNs) in the speed control of a separately excited DC motor drives fed from Photovoltaic (PV) generators through intermediate power converters. Two ULNs-based identification and control are used. Their free parameters are updated online concurrently by the forward propagation algorithm. The identifier network is used to capture and emulate the nonlinear mappings between the inputs and outputs of the motor system. The controller network is used to control the converter duty ratio so that the motor speed can follow an arbitrarily reference signal. Moreover the overall system can operate at the Maximum Power Point (MPP) of the PV source. The simulation results showed a good performance for the controller and the identifier during the training mode and the continuous running mode as well.

1. Introduction

Universal Learning Networks (ULNs) have been proposed to provide a universal framework for the class of NNs and to unify their learning algorithms [1]. ULNs can be also used to study the stability of any dynamical system by calculating the first and higher order derivatives and checking if they converge to zero or not. Therefore, this research is considered as an essential study that precedes the higher order derivatives-based stability analysis, which is not addressed in this paper.

For several adaptive control schemes proposed in the literature, the state estimation and parameter identification are based on linear models. But neglecting the nonlinearities of the motor system make it unrealistic and put its stability in risk. To overcome this problem, Weerasooriya and Sharkawi [2-3], have proposed the use of artificial Neural Networks (NNs) for system identification of the nonlinear motor dynamics, followed by the trajectory control of a DC motor using the direct model reference adaptive control. In their work, the training of the neural network is achieved using the offline static backpropagation algorithm. Rubaai and Kotaru [4], have also proposed a similar use of NNs as in [2-3]. In their work, the motor/load dynamics are modeled online and controlled using the dynamic backpropagation algorithm with adaptive learning rate. In [2-3] and [4], the DC motor is supplied directly from a conventional DC supply.

In this paper, two ULNs are used and trained online for identification and control of a photovoltaic (PV) supplied, separately excited DC motor loaded with a centrifugal pump via a DC-DC buck-boost converter. Two modes of operation are obtained: the training mode, in which the forward propagation algorithm updates the free parameters of both networks concurrently every window-size instants until their training errors reach almost a minimum value. The continuous running mode, in which both networks use the weight matrices obtained at the end of the training mode. This paper is organized as follows: In the next section, the dynamics of the DC motor system are described. The online identification and control algorithm is given in section3. After that in section 4, the simulation results for the online training algorithms of both networks are discussed. Finally, the major conclusion of this paper and some thoughts of the future research are summarized in section 5.

2 System Dynamics

The DC motor system mainly consists of PV generator, DC-DC converter and DC motor coupled to a centrifugal pump as shown in Fig. 1. This system is not experimentally installed yet. Therefore, to generate input/output training data, a mathematical model is needed. In the following subsections, a mathematical model for each device is developed and combined together to form the complete model, which used in the simulation studies.

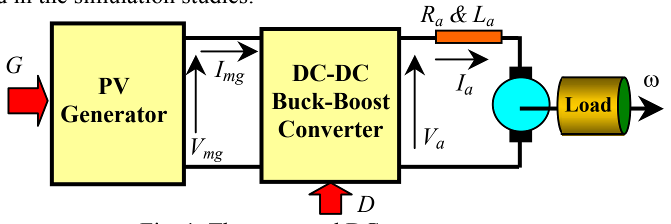


Fig. 1. The proposed DC motor system

2.1 PV Generator Model

The PV generator consists of solar cells connected in series and parallel fashion to provide the desired voltage and current required by the DC motor system. This PV generator exhibits a nonlinear voltage-current characteristic that depends on the insolation (solar radiation), as (1).

$$V_g = \frac{1}{\Lambda_g} \ln\left(\frac{G \times I_{phg} + I_{og} - I_g}{I_{og}} \right) - I_g R_{sg} \tag{1}$$

where V_g is the PV generator voltage; I_g is the PV generator current; $\Lambda_g = \Lambda / Ns$ is the PV generator constant; $\Lambda = q/(\varepsilon \times Z \times U)$, is the solar cell constant; $q = 1.602 \times 10^{-19}$C. is the electric charge; $Z = 1.38 \times 10^{-23}$ J/K is Boltzman constant; U= 298.15 °C is the absolute temperature; ε =1.0 is the completion factor; N_s=360 is the series-connected solar cells; N_p=3 is the parallel paths; $R_{sg} = R_s \times (Ns/Np)$ is the PV generator series resistance; R_s= 0.0152 Ω is the series resistance per cell; $I_{phg} = I_{ph} \times Np$ is the insolation-dependent photo current of the PV generator; I_{ph}=4.8 A is the photo current per cell; $I_{og} = I_o \times Np$ is the PV generator reverse saturation current; $I_o = 3.0797 \times 10^{-10}$ A is the reverse saturation current per cell; G is the solar insolation in per unit, and 1.0 per unit of G = 1000 W/m^2.
The PV generator Voltage-Current and Voltage-Power characteristics at five different values of G are shown in Fig. 2. From which, at any particular value of G, there is only one point at which the PV generator power is maximum. This point is called the Maximum Power Point (MPP). To locate its position, the corresponding voltage (V_{mg}) and current (I_{mg}) must be determined first

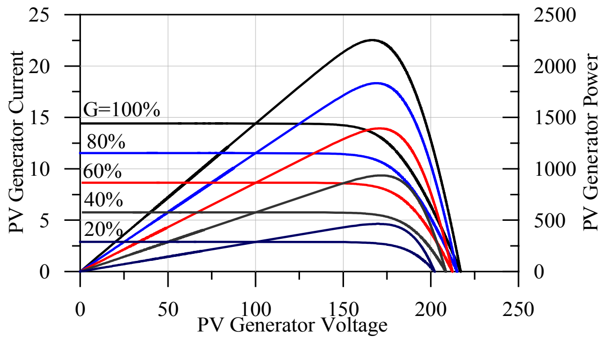


Fig. 2. PV generator characteristics at five different values of G.

2.2 DC Motor Model

The dynamics of the separately excited DC motor and its load are represented by the following set of differential equations with constant coefficients:

$$v_a(t) = R_a i_a(t) + L_a \frac{di_a}{dt}(t) + K\,\omega(t) \tag{2}$$

$$K\,i_a(t) = A_1 + B\,\omega(t) + J_a \frac{d\omega}{dt}(t) + T_L(t) \tag{3}$$

$$T_L(t) = A_2 + \xi\,\omega^{1.8} \tag{4}$$

where the name-plate parameters are: Voltage V_a=120 volt; Current I_a = 9.2 A.; Speed ω = 1500 rpm; Inertia J_a = 0.02365 Kg.m^2; Resistance R_a =1.5 Ω; Inductance L_a=0.2 H; Torque & back emf constant K = 0.67609 Nm.A^{-1}; Motor friction A_1 = 0.2 Nm; Load friction A_2 = 0.3 Nm; damping B = 0.002387 Nm.s.rad^{-1}; Load torque constant ξ = 0.00059 Nm.s.rad^{-1}.
Using a sampling time interval ΔT of 0.001s, and a first-order finite-difference approximation for the motor speed and current, the finite difference equation that governs the discrete-time dynamics of the DC motor is given by (5).

$$\omega(k) = \alpha\,v_a(k-1) - \beta - \gamma\,\omega(k-1) - \theta\,\omega^{1.8}(k-1) + \sigma\,\omega(k-2) + \psi\,\omega^{1.8}(k-2) \tag{5}$$

where α, β, γ, θ, σ and ψ are constants depending on ΔT as well as the motor/load parameters.

2.3 DC-DC Converter Model

The most important parameter of the converter is its chopping ratio $Y(k)$ that depends on the duty ratio $D(k)$ through a nonlinear relation given by (6).

$$Y(k) = \frac{D(k)}{1 - D(k)} \tag{6}$$

This converter is inserted between the PV generator and the DC motor to match the PV generator output characteristics to the DC motor input characteristics. Assuming the converter is ideal, then its input and output powers are equal resulting in the following relation:

$$\frac{v_a(k)}{V_{mg}(k)} = \frac{I_{mg}(k)}{I_a(k)} = Y(k) \tag{7}$$

3. Online Identification and Control Algorithms

The block diagram explaining the online identification and control of the DC motor system is shown in Fig. 3. Both the controller and the identifier are feed forward NNs with architecture 5-8-1 and 5-12-1, respectively. Each neuron in the hidden and output layers uses a sigmoidal activation function. The inputs to the controller network are: the MPP voltage $V_{mg}(k)$, the reference speed $\omega_{ref}(k)$, the previous values of the identified speed $\hat{\omega}(k\text{-}1)$, $\hat{\omega}(k\text{-}2)$ and the previous value of the controller output $D(k\text{-}1)$. Whereas the only output is the converter duty ratio $D(k)$. Also the inputs to the identifier network are: the previous values of the motor speed $\omega(k\text{-}1)$ and $\omega(k\text{-}2)$, $D(k)$, $D(k\text{-}1)$ and $V_{mg}(k)$. Whereas the only output is the identified motor speed $\hat{\omega}(k)$.
To obtain efficient performance for the identifier and the controller networks, their error functions $E_I(k)$ and $E_C(k)$ are calculated considering not only the current difference between the desired and actual outputs but also considering the past differences given by (8) and (9), respectively.

$$E_I(k) = \frac{1}{2T_I} \sum_{s \in T_I} \left[\omega(k-s) - \hat{\omega}(k-s)\right]^2 \tag{8}$$

$$E_C(k) = \frac{1}{2T_C} \sum_{s \in T_C} \left[\omega_{ref}(k-s) - \omega(k-s)\right]^2 \tag{9}$$

where T_I and T_C are the set of updating window-size instants of the identifier and the controller parameters, respectively.
Both the identifier and controller free parameters (λ_I) and (λ_C) are updated every window size based on the gradient method given by (10) and (11), respectively.

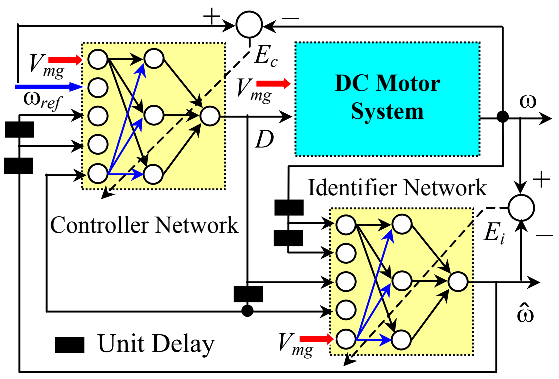


Fig. 3. Online identification and control scheme of the DC motor system.

$$\lambda_I \leftarrow \lambda_I - \eta_I \frac{\partial^\dagger E_I(k)}{\partial \lambda_I} + \mu_I \Delta\lambda_I \tag{10}$$

$$\lambda_C \leftarrow \lambda_C - \eta_C \frac{\partial^\dagger E_C(k)}{\partial \lambda_C} + \mu_C \Delta\lambda_C \tag{11}$$

where η_I & μ_I and η_C & μ_C are the learning rate and the momentum coefficient of the identifier and the controller networks, respectively. $\partial^\dagger E_I(k)/\partial\lambda_I$ and $\partial^\dagger E_C(k)/\partial\lambda_C$ are the ordered derivatives of the identifier and controller errors w.r.t. their free parameters.
The training algorithm of the controller network is based on the information transferred from the identifier network. Initially, this information may be incorrect. So that it is necessary to speed up the training ability of the identifier w.r.t the controller. This can be achieved by setting $\eta_I > \eta_C$ and this guarantees correct information flow from the identifier network to the controller network.
The value of the learning rate and the length of the window size instants are the most important parameters that play vital role in the training of both networks. Their values should be adjusted carefully to achieve an efficient training algorithm. Actually when dealing with dynamical systems, η must be as small as possible. Decreasing the window size instants has the same effect as decreasing η. Based on trial-and-error, it is found that $\eta_I = 10^{-6}$, $\mu_I = 0.2$, $T_I = 5$ instants (current instant plus 4 previous instants), $\eta_C = 10^{-9}$, $\mu_C = 0.08$ and $T_C = 10$ instants, (current instant plus 9 previous instants).
The ordered derivative $\partial^\dagger E_I(k)/\partial\lambda_I$ is given by (12) in which, both the direct and indirect relation between E_I and λ_I are considered. The direct relation is calculated by $\partial E_I(k)/\partial\lambda_I$. In this application, the direct relation always equals zero. But the indirect relation is calculated by considering $\hat{\omega}(k)$, which is the identifier output that directly influence $E_I(k)$, as intermediate variables.

$$\frac{\partial^\dagger E_I}{\partial \lambda_I}(k) = \frac{1}{T_I} \sum_{s \in T_I} \left[\{\hat{\omega}(k-s) - \omega(k-s)\} \cdot \frac{\partial^\dagger \hat{\omega}}{\partial \lambda_I}(k-s) \right] \tag{12}$$

Let us introduce the notation $P(j, t, \lambda_I)$ to represent the ordered derivative $\partial^\dagger h_j(t)/\partial\lambda_I$, where h_j is the output of node j. The calculation of $P(j, t, \lambda_I)$ also requires the help of some intermediate variables. This leads to the forward propagation algorithm give by (13). Where $J_F(j)$ is the set of suffixes of the nodes connected to node j, J is the set of suffixes of the total nodes, τ_{ij} is the time delay of the branch from node i to node j, and h_i is the output of the node i.

$$P(j,t,\lambda_I) = \sum_{i \in J_F(j)} \left[\frac{\partial h_j(t)}{\partial h_i(t-\tau_{ij})} P(i, t-\tau_{ij}, \lambda_I) \right] + \frac{\partial h_j(t)}{\partial \lambda_I} \quad (j \in J,\ t \in T_I) \tag{13}$$

If the branch connecting nodes i and j has no time delay, then τ_{ij} is set to zero. The order derivative $\partial^\dagger E_I(k)/\partial\lambda_I$ can be calculated using both (12) and (13) iteratively.
By the same way, the controller ordered derivative $\partial^\dagger E_C(k)/\partial\lambda_C$ can be calculated by using (14) and (15) iteratively.

$$\frac{\partial^\dagger E_C}{\partial \lambda_C}(k) = \frac{1}{T_C} \sum_{s \in T_C} \left[\{\omega(k-s) - \omega_{ref}(k-s)\} \cdot \frac{\partial^\dagger \hat{\omega}}{\partial \lambda_C}(k-s) \right] \tag{14}$$

$$P(j,t,\lambda_C)=\sum_{i\in J_F(j)}\left[\frac{\partial h_j(t)}{\partial h_i(t-\tau_{ij})}P(i,t-\tau_{ij},\lambda_C)\right]+\frac{\partial h_j(t)}{\partial \lambda_C} \quad (j\in J,\, t\in T_C) \tag{15}$$

Interested readers can get more details about the forward propagation algorithms for dynamical and static networks in [5].

Although the calculation of $\partial^{\dagger}E_I/\partial\lambda_I$ and $\partial^{\dagger}E_C/\partial\lambda_C$ are systematic and simple, the ordered derivatives of all nodes during window-size instants must be calculated first. This resulting in a little bit longer training time of both networks.

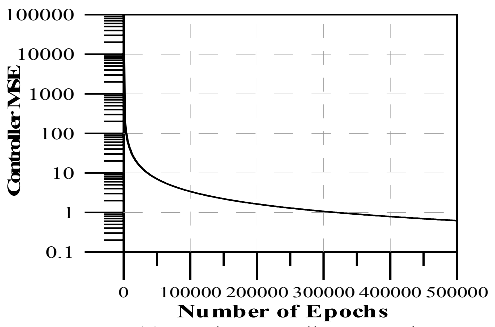
The controller performance is investigated for different shapes of the reference signal and showed a good tracking capability. For brevity, the only results addressed here are those related to the reference speed given by (16). The value of $G(k)$ is also assumed to be given by (17).

$$\omega_{ref}(k)=110+15\left[\sin(2\pi k\,\Delta T)+\cos(2\pi k\,\Delta T)\right] \tag{16}$$

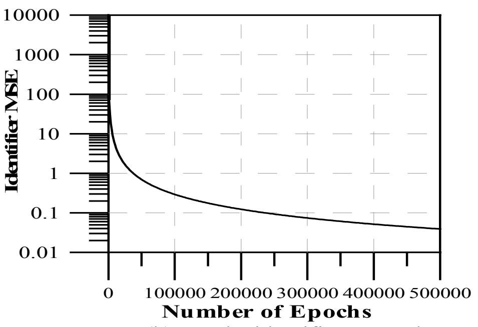
$$G(k) = 0.2 + 0.8\sin(\pi k\Delta T) \tag{17}$$

4. Simulation Results

Both the identifier and the controller networks are trained online concurrently by the forward propagation algorithm. The learning curves of the both networks are given in Fig. 4.



(a) For the controller network



(b) For the identifier network

Fig. 4. Training error vs number of epochs

From these figures, the controller error reached a value of 0.56 after 5×10^5 training epochs. After this point, no improvement in the controller performance is achieved. Therefore, the free parameters of the controller and the identifier are saved, and the training algorithm is terminated. But the controller is designed to operate continuously during the daytime. For that reason the saved free parameters can be used as initial values for the re-operation. As a conclusion, there are two modes of operation: *The training mode*, in which the weights and biases of both networks are initialized randomly and updated every window size instant based on the forward propagation algorithm until no improvement in the controller performance is achieved. The training algorithm during this mode is given in Table 1. *The continuous running mode*, in which the operation of both networks is based on the pre-saved weights and biases that are not updated during this mode of operation.

Table 1. Training algorithm for the identifier and the controller networks during the training mode.

Step 1: The free parameters of both networks are randomly initialized with small values.
Step 2: The controller output $D(k)$ is calculated.
Step 3: The actual motor speed is calculated based on (5).
Step 4: The Identifier output $\hat{\omega}(k)$ is calculated.
Step 5: Calculate the error functions $E_i(k)$ and $E_c(k)$ based on (8) and (9).
Step 6: The ordered derivatives $\partial^{\dagger}E_I(k)/\partial\lambda_I$ and $\partial^{\dagger}E_C(k)/\partial\lambda_C$ are calculated based on (12) ~ (15).
Step 7: Update the identifier and the controller free parameters based on (10) and (11).
Step 8: Repeat from step 2 to step 7 until the controller training error reaches a minimum value.

The motor dynamics are represented over 1000 sampling instants. During the continuous running mode, the motor actual and identified speeds are compared to the reference speed as shown in Fig.5. Where Fig. 5-a shows the starting characteristics of the DC motor throughout the first 1000 sampling time. And Fig. 5-b shows the steady-state operation of the DC motor throughout the next 1000 sampling instants. Similar curves to those displayed in Fig. 5-b are obtained at the end of the training mode. From these figures, both ω and $\hat{\omega}$ are very close to each other and also close to the reference speed indicating a good performance for the controller and the identifier networks as well.

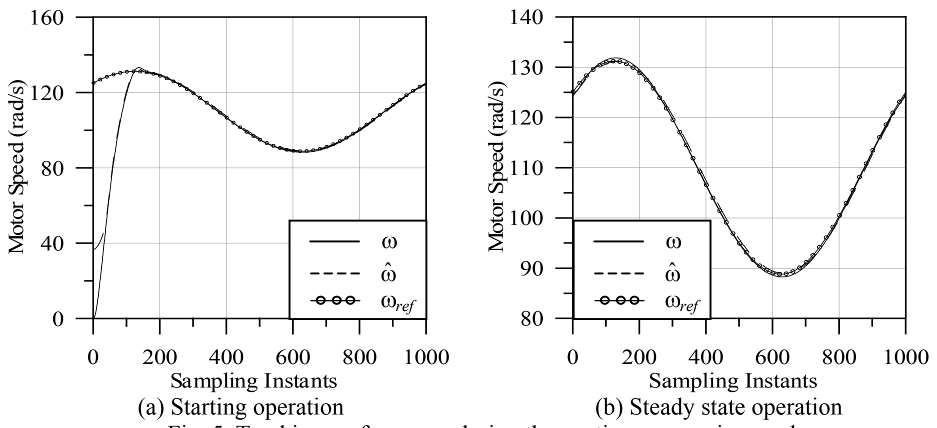


Fig. 5. Tracking performance during the continuous running mode

5. Conclusion

In this paper, a separately excited DC motor loaded with a centrifugal pump and supplied from PV generator via DC-DC buck-boost converter is successfully controlled using neural networks. Two neural networks are used for identification and control of the DC motor system. These networks are trained online using the forward propagation algorithm for calculating the ordered derivatives of the universal learning networks. The simulation results indicated a good performance for the controller and the identifier networks. The overall system stability was not discussed in this paper even it is a unique point. But it will be discussed later using the higher ordered derivatives of the ULNs. Also the control algorithm of the DC motor system will be verified experimentally. Therefore the system identification will be based on real data, and the mathematical model used in this paper will not be used any more.

References

[1] K. Hirasawa, X. Wang, J. Murata, J. Hu, and C. Jin: Universal learning network and its application to chaos control. Neural Network, Vol. 13, pp. 239-253, 2000.

[2] S. Weerasooriya and M. A. El-Sharkawi: Identification and control of a DC motor using back propagation neural networks. IEEE Trans. Energy Conversion, Vol. 6, pp. 663-669, 1991.

[3] S. Weerasooriya and M. A. El-Sharkawi: Laboratory implementation of a neural network trajectory controller for a DC motor. IEEE Trans. Energy Conv., Vol. 8, pp. 107-113, 1993.

[4] A. Rubaai and R. Kotaru: Online identification and control of a DC motor using learning adaptation of neural networks. IEEE Trans. Industry Application, Vol. 36, pp. 935-942, 2000.

[5] K.Hirasawa, J.Hu, M.Ohbayashi, and J.Murata: Computing higher order derivatives in universal learning networks. Journal of Advanced Computational Intelligence, Vol. 2, pp. 47-53, 1998.