Comparison of a Heuristic Dynamic Programming and a Dual Heuristic Programming Based Adaptive Critics Neurocontroller for a Turbogenerator

Ganesh K. Venayagamoorthy
Missouri University of Science and Technology

Donald C. Wunsch
Missouri University of Science and Technology, dwunsch@mst.edu

Ronald G. Harley

Comparison of a Heuristic Dynamic Programming and a Dual Heuristic Programming Based Adaptive Critics Neurocontroller for a Turbogenerator

[1]Ganesh K Venayagamoorthy *MIEEE*, [2]Ronald G Harley *FIEEE* and [3]Donald C Wunsch II *SMIEE*

[1]School of Electrical and Electronic Engineering
University of Natal, Durban 4041,
South Africa
gkumar@ieee.org

[2]School of Electrical and Computer Engineering
Georgia Institute of Technology,
Atlanta, GA 30332-0250, USA
ron.harley@ece.gatech.edu

[3]Department of Electrical and Computer Engineering
University of Missouri-Rolla, Rolla,
Missouri 65409-0040, USA
dwunsch@ece.umr.edu

Abstract: This paper presents the design of a neurocontroller for a turbogenerator that augments/replaces the conventional Automatic Voltage Regulator (AVR) and the turbine governor. The neurocontroller uses a novel technique based on the Adaptive Critic Designs (ACDs) with emphasis on Heuristic Dynamic Programming (HDP) and Dual Heuristic Programming (DHP). Results are presented to show that the DHP based neurocontroller is robust and performs better than the HDP based neurocontroller, as well as the conventional controller, especially when the system conditions and configuration changes.

INTRODUCTION

Turbogenerators are highly complex, non-linear, fast acting, multivariable systems with dynamic characteristics that vary as operating conditions change. As a result, the generator voltage and delivered power have to be coordinated to satisfy the requirements of the rest of the power system. The effective control of turbogenerators is important since these machines are responsible for ensuring the stability and the security of the electrical network. Conventional AVRs and turbine governors (called *conventional controllers* in the rest of this paper) are designed (using linearized mathematical models of the nonlinear turbogenerator) to control, in some optimal fashion, the turbogenerator around one operating point; at any other operating point the generator's performance is degraded [1].

Artificial neural networks (ANNs) are good at identifying and controlling complex nonlinear systems [2]. In general they are particularly suitable to identify the interactions between the inputs and outputs of multivariable systems. In the specific case of a turbogenerator it been shown that one multilayer feedforward neural network using deviation signals as inputs can *identify* [3] the complex and nonlinear dynamics of a single-machine-infinite-bus (SMIB) configuration with sufficient accuracy and pass this information to a second neural network which acts as multiple-input-multiple-output (MIMO) *controller*. The combination of identifier and controller neural networks is called a *neurocontroller* in the rest of this paper. Unlike the conventional controller, the neurocontroller therefore does not require any mathematical model, linear or nonlinear, for the SMIB system.

A number of publications have reported on the design of such neurocontrollers for turbogenerators, and presented both simulation and experimental results to show that they have the potential to replace conventional controllers [4,5,6]. However all these neurocontrollers require continual online training of their neural networks after commissioning. In closed loop control systems with relatively short time constants the computational time required by frequent online training could become the factor that limits the maximum bandwidth of the controller. This paper extends earlier work and presents a new and novel technique of designing a turbogenerator neurocontroller, which avoids the computational load of online training. In this new technique the neurocontroller uses a so-called *Adaptive Critic* and this avoids the need for online training. All training is done offline prior to commissioning. Two different types of Adaptive Critic are considered namely an *HDP* and a *DHP*. Results are presented, showing that the DHP critic produces the best results.

ADAPTIVE CRITIC DESIGNS

A. Background

Adaptive Critic Designs (ACDs) are neural network designs capable of optimization over time under conditions of noise and uncertainty. A family of ACDs was proposed by Werbos [7] as a new optimization technique combining concepts of reinforcement learning and approximate dynamic programming. For a given

0-7695-0619-4/00 $10.00 © 2000 IEEE

series of control actions that must be taken in sequence and not knowing the quality of these actions until the end of the sequence, it is impossible to design an optimal controller using the traditional supervised learning.

Dynamic programming prescribes a search which tracks backward from the final step, rejecting all suboptimal paths from any given point to the finish, but retains all other possible trajectories in memory until the starting point is reached. However, many paths that are extremely unlikely to be important are nevertheless also retained until the search is complete. The result of this is that the procedure is too computationally expensive for most real problems. In supervised learning, an ANN training algorithm utilizes a desired output and, having compared it to the actual output, generates an error term to allow the network to learn. The backpropagation algorithm is typically used to get the necessary derivatives of the error term with respect to the training parameters and/or the inputs of the network. However, backpropagation can be linked to reinforcement learning via a network called the ***Critic*** network, which has certain desirable attributes.

Critic methods remove the learning process one step from the control network (traditionally called the "***Action*** network" or "***actor***" in ACD literature), so the desired trajectory or control action information is not necessary. The critic network learns to approximate the cost-to-go or strategic utility function (the function *J* of Bellman's equation in dynamic programming) and uses the output of an action network as one of its inputs, directly or indirectly. When the critic network learns, backpropagation of error signals is possible along its input pathway back to the action network. To the backpropagation algorithm, this input pathway looks like just another synaptic connection that needs weight adjustment. Thus, no desired signal is needed. All that is needed is a desired cost function *J*.

Watkins [8] developed the system known as Q-learning explicitly based on dynamic programming. Werbos developed a family of systems for approximating dynamic programming [7]. His approach generalizes previously suggested designs for continuous domains. For example Q-learning becomes a special case of an ***action-dependent heuristic dynamic programming*** (ADHDP) in his family of systems. Werbos went further from a critic approximating the *J* function, to a critic approximating the derivatives of the function *J* with respect to its states called the ***dual heuristic programming*** (DHP), and to a critic approximating both *J* and its derivatives, called the ***globalized dual heuristic programming*** (GDHP). It should be noted that these systems do not require exclusively neural network implementations since any differentiable structure is suitable as a building block for the systems. The interrelationships between members of the ACD family have been explained in detail by Prokhorov [9]. This paper compares HDP and DHP ***model dependent*** designs of the neurocontroller against the results obtained using a conventional PID controller [10] for a turbogenerator plant.

B. Heuristic Dynamic Programming

The critic network estimates the function *J* (cost-to-go) in the Bellman equation of dynamic programming, expressed as follows:

$$J(t) = \sum_{k=0}^{\infty} \gamma^k U(t+k) \qquad (1)$$

where γ is a discount factor for finite horizon problems ($0 < \gamma < 1$), and $U(.)$ is the utility function or local cost. The critic network is trained forward in time, which is of great importance for real-time operation. The critic network tries to minimize the following error measure over time

$$\|E_1\| = \sum_t E_1^2(t) \qquad (2) \qquad\qquad E_1(t) = J(\Delta Y(t)) - \gamma J(\Delta Y(t+1)) - U(t) \qquad (3)$$

where $\Delta Y(t)$ stands for either a vector of observables of the plant (or the states, if available). The expression for the weights' update for the critic is as follows:

$$\Delta W_c = -\eta (J(\Delta Y(t)) - \gamma J(\Delta Y(t+1)) - U(t)) \frac{\partial J(\Delta Y(t))}{\partial W_c}$$ (4), where η is a positive learning rate.

The configuration for training the critic network according to eq. (3) is shown in Fig. 1. The same critic network is shown in two consecutive moments in time. The critic network's output $J(t+1)$ is necessary in order to provide the training signal $\gamma J(t+1) + U(t)$, which is the target value for $J(t)$. The objective here is to minimize *J* in the immediate future, thereby optimizing the overall cost expressed as a sum of all $U(t)$ over the horizon of the problem. This is achieved by training the action network with an error signal $\partial J/\partial A$. The gradient of the cost function J with respect to the outputs, A, of the action network, is obtained by backpropagating $\partial J/\partial J$ (i.e. the constant 1) through the critic network and then through the model to the action network as shown in Fig. 2.

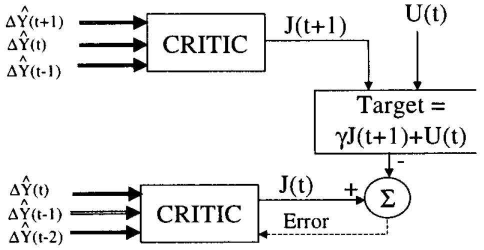

Fig. 1 Critic Adaptation in HDP

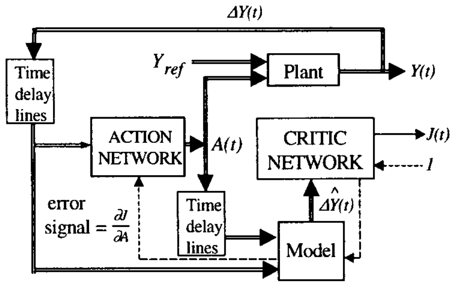

Fig. 2 Action Adaptation in HDP

This gives $\partial J/\partial A$ and $\partial J/\partial W_A$ for all the outputs of the action network and all the action network's weights W_A, respectively. Hence the expression for the weights' update (applying the Least Mean Squares (LMS) for the action network) is as follows:

$$\Delta W_A = -\alpha \frac{\partial A(t)}{\partial W_A} \frac{\partial J(t)}{\partial A(t)}$$ (5), where α is a positive learning rate.

C. Dual Heuristic Programming

DHP has a critic network that estimates the derivatives of J with respect to the vector ΔY. The critic network learns minimization of the following error measure over time:

$$\|E_2\| = \sum_t E_2^T(t)E_2(t)$$ (6), where $$E_2(t) = \frac{\partial J[\Delta Y(t)]}{\partial \Delta Y(t)} - \gamma \frac{\partial J[\Delta Y(t+1)]}{\partial \Delta Y(t)} - \frac{\partial U(t)}{\partial \Delta Y(t)} \quad (7)$$

and, $\partial J/\partial \Delta Y(t))$ and $\partial U/\partial \Delta Y(t)$ are each vectors containing partial derivatives of scalars J and U respectively with respect to the components of the vector ΔY. The critic network's training is more complicated than in HDP since there is a need to take into account all relevant pathways of backpropagation as shown in Fig. 3, where the paths of derivatives and adaptation of the critic are depicted by dashed lines.

In DHP, application of the chain rule for derivatives yields

$$\frac{\partial J(t+1)}{\partial \Delta Y_j(t)} = \sum_{i=1}^{n} \lambda_i(t+1) \frac{\partial \Delta Y_i(t+1)}{\partial \Delta Y_j(t)} + \sum_{k=1}^{m} \sum_{i=1}^{n} \lambda_i(t+1) \frac{\partial \Delta Y_i(t+1)}{\partial A_k(t)} \frac{\partial A_k(t)}{\partial \Delta Y_j(t)} \quad (8)$$

where $\lambda_i(t+1) = \partial J(t+1)/\partial \Delta Y_i(t+1))$, and n, m are the numbers of outputs of the model and the action networks, respectively. By exploiting eq. (8), each of n components of the vector $E_2(t)$ from eq. (7) is determined by

$$E_{2j}(t) = \frac{\partial J(t)}{\partial \Delta Y_j(t)} - \gamma \frac{\partial J(t+1)}{\partial \Delta Y_j(t)} - \frac{\partial U(t)}{\partial \Delta Y_j(t)} - \sum_{k=1}^{m} \frac{\partial U(t)}{\partial A_k(t)} \frac{\partial A_k(t)}{\partial \Delta Y_j(t)} \quad (9)$$

The action network is adapted in Fig. 4 by propagating $\lambda(t+1)$ back through the model to the action. The goal of such adaptation can be expressed as follows:

$$\frac{\partial U(t)}{\partial A(t)} + \gamma \frac{\partial J(t+1)}{\partial A(t)} = 0, \forall t \quad (10)$$

The weights' update expression when applying the LMS training algorithm is as follows:

$$\Delta W_A = -\alpha \left[\frac{\partial U(t)}{\partial A(t)} + \gamma \frac{\partial J(t+1)}{\partial A(t)} \right]^T \frac{\partial A(t)}{\partial W_A}$$ (11), where α is a positive learning rate.

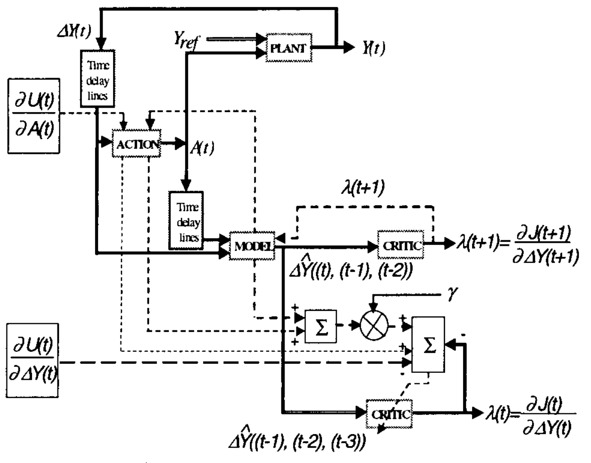

Fig. 3 Critic Adaptation in DHP

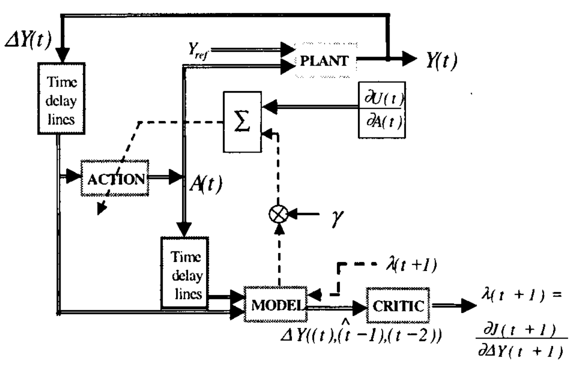

Fig. 4 Action Adaptation in DHP

TURBOGENERATOR MODELLING

This paper considers a 3 kW micro-alternator with per-unit parameters typical of those expected of 30 – 1000 MW generators [11], with conventional governor and excitation controls and connected to an infinite bus through a transmission line as shown in Fig. 5. The conventional AVR and excitation system, the turbine simulator and governor system and the micro-alternator with transmission line are represented in state space by a second, fourth and seventh order equations respectively. The mathematical implementations of these state space equations are carried out in the MATLAB/SIMULINK environment [4].

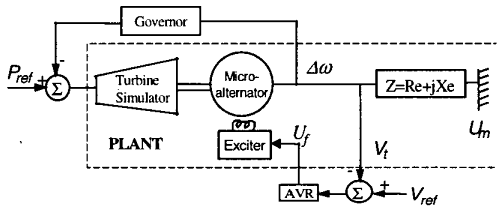

Fig. 5 The Single Machine Infinite Bus Configuration

GENERAL TRAINING PROCEDURE FOR THE CRITIC AND ACTION NETWORKS

The training procedure is that suggested in [14] and it is applicable to any ACD. It consists of two training cycles: that of the critic, and that of the action. The action network is pretrained on a linearized model of the plant. The critic's adaptation is done initially with the pretrained action network to ensure that the whole system consisting of the ACD and the plant remains stable. Then the action network is trained further while keeping the critic network weights fixed. This process of training the critic and the action one after the other is repeated until an acceptable performance is reached. It is assumed that there is no concurrent adaptation of the model network, which is previously trained offline, and W_C and W_A are initialized to any reasonable values.

In the critic's training cycle, an incremental optimization of eq. (2) and/or eq. (6) is carried out using a suitable optimization technique (e.g. LMS). The following operations are repeated N_I times:

1. Initialize $t = 0$ and $\Delta Y(0)$
2. Compute output of the critic network at time t, $J(t) = f_C(\Delta Y(t), W_C)$
3. Compute output of the action network at time t, $A(t) = f_A(\Delta Y(t), W_A)$
4. Compute output of the model at time $t+1$, $\Delta Y(t+1) = f_M(\Delta Y(t), A(t), W_M)$
5. Compute the output of the critic network at time $t+1$, $J(t+1) = f_C(\Delta Y(t+1), W_C)$
6. Compute the critic network error at time t, $E(t)$ from eq. (3)/eq. (7).

7. Update the critic network's weights using the backpropagation algorithm.
8. Repeat steps 2 to 7.

The functions $f_C(Y(t), W_C)$, $f_A(Y(t), W_A)$, and $f_M(Y(t), W_M)$ represent the critic, the action and the model networks with their weights W_i, respectively.

In the action network's training cycle, an incremental learning is also carried out using the backpropagation algorithm, as in the critic network's training cycle above. The list of operations for the action network's training cycle is almost the same as that for the critic network's cycle above (steps 1 to 7). However, instead of using eq. (2) and/or eq. (6) and $\partial J/\partial W_C$, $\partial J/\partial A$ and $\partial A/\partial W_A$ are used for updating the action network's weights. The action's training cycle is repeated N_2 times while keeping the critic's weights W_C fixed. N_1 and N_2 are the lengths of the corresponding training cycles. It is very important that the whole system consisting of ACD and the plant remains stable while both of the networks of ACD undergo adaptation.

HDP VERSUS DHP

The use of derivatives of an optimization criterion, rather than the optimization criterion itself, is known as being the most important information to have in order to find an acceptable solution. In HDP, this information is obtained indirectly: by backpropagation through the critic network. It has a potential problem of not being too smooth since the critic network in HDP is not trained to approximate derivatives of J directly.

DHP has an important advantage over HDP since its critic network builds a representation for the derivatives of J by being explicitly trained on them through $\partial U(t)/\partial R(t)$ and $\partial U(t)/\partial A(t)$. For instance, in the area of model-based control, a sufficiently accurate model and well-defined $\partial U(t)/\partial R(t)$ and $\partial U(t)/\partial A(t)$ exist. To adapt the action network, only the derivative $\partial J/\partial R$ or $\partial J/\partial A$ is required, rather than the J function itself. But the approximation of these derivatives is already a *direct* output of the DHP critic.

RESULTS WITH THE TRAINED ACTION NETWORK

A discount factor γ of 0.5 and the utility function in eq. (12) are used in the Bellman equation (eq. (1) and in the training of the critic network (eqs. (2) and (6)) and of the action network (eq. (10)). Once the critic network's and action network's weights have converged, the action network is connected to the plant (Fig. 5).

$$U(t) = [4\Delta V(t) + 4\Delta V(t-1) + 16\Delta V(t-2)]^2 + [0.4\Delta\omega(t) + 0.4\Delta\omega(t-1) + 0.16\Delta\omega(t-2)]^2 \quad (12)$$

The dynamic and transient operation of the action network (neurocontroller) is compared with the operation of a conventional PID controller (AVR and turbine governor) under two different conditions: ± 5% desired step changes in the terminal voltage setpoint, and a three phase short circuit on the infinite bus. Each of these is investigated for two different values of line impedance.

Figs. 6 and 7 show the performance of the different controllers for ± 5% desired step changes in the terminal voltage with the turbogenerator operating at 1 pu real power (P) and 0.85 lagging power factor (pf) (at the generator terminals), with the transmission line impedance Z = 0.02 + j 0.4 pu. Fig. 8 shows a turbogenerator operating under the same conditions but experiencing a 50 ms three phase short circuit on the infinite bus. Fig. 9 shows a turbogenerator operating under the same terminal conditions and short circuit as in Fig. 8, but with an increase in line impedance to Z = 0.025 + j 0.6 pu. The results with the conventional AVR and governor controllers, the HDP and the DHP neurocontrollers are shown in the diagrams below as CONV, HDP and DHP respectively.

Figs. 6 and 7 show that with step changes in the terminal voltage, the DHP based neurocontroller has a faster rise time than the HDP based neurocontroller. However, for this disturbance both neurocontrollers react slower than the conventional controller. The response of the DHP based neurocontroller can be improved by using a different utility function and discount factor in the Bellman equation. On the other hand, Figs. 8 and 9 show that for short circuit disturbances, the DHP based neurocontroller has the best damping compared to both the HDP neurocontroller and the conventional controller. The designer therefore has the final choice on whether terminal voltage or rotor angle damping is more important. These significant results have been reached with offline training of the neurocontroller and continually online training has been avoided.

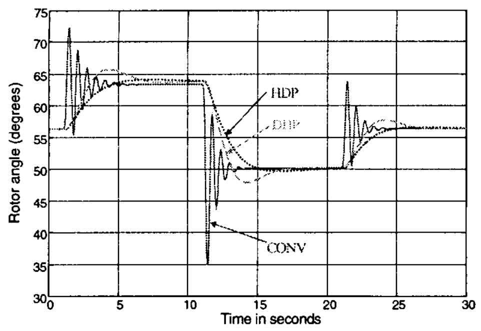

Fig. 6 Rotor angle variation for ±5% step changes in the desired terminal voltage

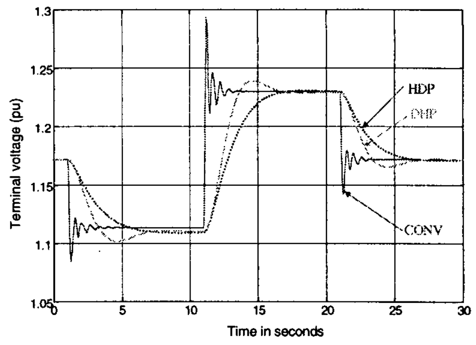

Fig. 7 ±5% Step changes in the desired terminal voltage

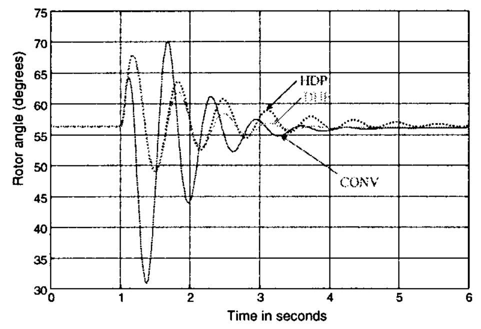

Fig. 8 Rotor angle variation for a 50 ms short circuit on the infinite bus (P = 1.0 pu, pf = 0.85 lagging, Z = 0.02 + j 0.4 pu)

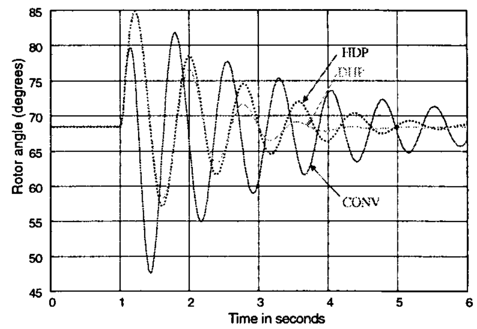

Fig. 9 Rotor angle variation for a 50 ms short circuit on the infinite bus (P = 1.0 pu, pf = 0.85 lagging, Z = 0.025 + j 0.6 pu)

CONCLUSIONS

This paper has presented a new and novel technique in which adaptive critic designs based neurocontrollers can be in the feedback loops of turbogenerators without needing continually online training. The dual heuristic programming type (DHP) of the ACD based neurocontroller has performed excellently during the short circuit tests compared to the HDP neurocontroller and the conventional controller. This paper has therefore proved that there is a potential for *adaptive critic designs* based neurocontrollers for real time control of turbogenerators.

REFERENCES

[1] Adkins B and Harley RG, "The general theory of alternating current machine", Chapman and Hall, London, 1975, ISBN 0-412-15560-5.

[2] Hunt KJ, Sbarbaro D, Zbikowski R and Gawthrop PJ, "Neural networks for control systems – a survey", Automatica, Vol. 28, No. 6, 1992, pp 1083 – 1112.

[3] Venayagamoorthy GK and Harley RG, "A continually online trained artificial neural network identifier for a turbogenerator", Proceedings of IEEE International Electric Machines and Drives Conference IEMDC' 99, Seattle, USA, 9 – 12 May, 1999, pp 404 - 406.

[4] Venayagamoorthy GK and Harley RG, "Simulation studies with a continuously online trained artificial neural network controller for a micro-turbogenerator", Proceedings of IEE International Conference on Simulation, University of York, UK, 30 September – 2 October 1998, pp 405 – 412.

[5] Venayagamoorthy GK and Harley RG, "Experimental studies with a continually online trained artificial neural network controller for a turbogenerator", Proceedings of International Joint Conference on Neural Networks, IJCNN' 99, Washington, DC USA, 10 – 16 July 1999, paper no. 2098.

[6] Flynn D, McLoone S, Irwin GW, Brown MD, Swidenbank E and Hogg BW, "Neural control of turbogenerator systems", Automatica, Vol 33, No 11, 1997, pp 1961 – 1973.

[7] Werbos P, "Approximate dynamic programming for real-time control and neural modeling", in Handbook of Intelligent Control, White and Sofge, Eds., Van Nostrand Reinhold, ISBN 0-442-30857-4, 1992, pp 493 – 525.

[8] Watkins C and Dayan P, "Q-learning", Machine Learning, Vol. 8, 1992, pp. 279-292.

[9] Prokhorov D and Wunsch D, "Adaptive Critic Designs", IEEE Trans. on Neural Networks, Vol. 8, No. 5, 1997, pp 997-1007.

[10] Ho WK, Hang CC and Cao LS, "Tuning of PID controllers based on gain and phase margin specifications", Proceedings of the 12th Triennial World Congress on Automatic Control, Sydney, Australia, July 1993, pp 199-202.

[11] Limebeer DJN, Harley RG and Lahoud MA, "A laboratory system for investigating subsynchronous resonance", Paper A80-0190-0, IEEE PES Winter Power Meeting, New York, USA, Feb 4 - 8, 1980.