01 Jan 2002

Comparison of Heuristic Dynamic Programming and Dual Heuristic Programming Adaptive Critics for Neurocontrol of 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

Recommended Citation

G. K. Venayagamoorthy et al., "Comparison of Heuristic Dynamic Programming and Dual Heuristic Programming Adaptive Critics for Neurocontrol of a Turbogenerator," *IEEE Transactions on Neural Networks*, Institute of Electrical and Electronics Engineers (IEEE), Jan 2002.
The definitive version is available at https://doi.org/10.1109/TNN.2002.1000146

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

Ganesh K. Venayagamoorthy, Ronald G. Harley, and Donald C. Wunsch

***Abstract*—This paper presents the design of an optimal neurocontroller that replaces the conventional automatic voltage regulator (AVR) and the turbine governor for a turbogenerator connected to the power grid. The neurocontroller design uses a novel technique based on the adaptive critic designs (ACDs), specifically on heuristic dynamic programming (HDP) and dual heuristic programming (DHP). Results show that both neurocontrollers are robust, but that DHP outperforms HDP or conventional controllers, especially when the system conditions and configuration change. This paper also shows how to design optimal neurocontrollers for nonlinear systems, such as turbogenerators, without having to do continually online training of the neural networks, thus avoiding risks of instability.**

***Index Terms*—Adaptive critics, artificial neural networks (ANNs), neurocontrol, optimal control, turbogenerator control.**

I. Introduction

TURBOGENERATORS are highly nonlinear, 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. Effective control of turbogenerators is important, since these machines are responsible for ensuring the stability and security of the electric power grid. Conventional automatic voltage regulator (AVR) and turbine governors (called *conventional controllers* in the rest of this paper) are designed (using linearized mathematical models) to control, the turbogenerator optimally around one operating point; at any other operating point, the generator's performance is degraded [1].

In recent years, there has been considerable research in the use of artificial neural networks (ANNs) for identification and control of nonlinear systems [2], [3]. An increasing demand in the performance specifications and the complexity of dynamic systems mandate the use of sophisticated information processing and control in almost all branches of engineering systems. The promise of fast computation, versatile representational ability of nonlinear maps, fault tolerance, and the capability to generate quick, robust, suboptimal solutions from neural networks, make the latter an ideal candidate for carrying out such a sophisticated identification or control task. Problems in nonlinear identification and control can be seen as the determination of the interactions between the inputs and outputs of multivariable systems.

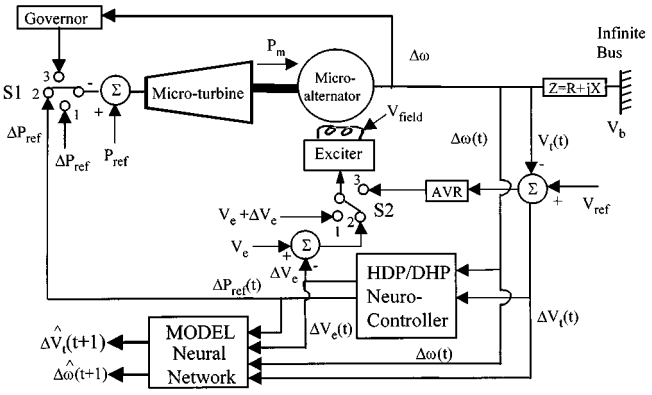

Fig. 1. The single machine infinite bus configuration with the conventional AVR and governor controllers, and neurocontroller. The micro-alternator, AVR and exciter, and governor and microturbine parameters are given in Tables I–III, respectively.

In the specific case of a turbogenerator, a multilayer feedforward neural network using deviation signals as inputs, can *identify* or estimate the complex and nonlinear dynamics of a single machine infinite bus (SMIB) (see Fig. 1) configuration with sufficient accuracy [4], and pass this information to a second neural network which acts as a multiple-input–multiple-output (MIMO) *controller*. The combination of the identifier and the controller neural networks is called a *neurocontroller* in the rest of the paper. Unlike the conventional controllers, 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 [5]–[7]. However, all these neurocontrollers require continual online training of their neural networks after commissioning. In most of the above results, an ANN is trained to approximate various nonlinear functions in the nonlinear system. The information is then used to adapt an ANN controller. Since an ANN identifier is only an approximation to the underlying nonlinear system, there is always residual error between the true plant and the ANN model of the plant. Stability issues arise when the ANN identifier is continually trained online and simultaneously used to control

Manuscript received November 29, 2000; revised October 26, 2001. This work was supported by the National Science Foundation, USA, National Research Foundation, South Africa, and the University of Natal, Durban, South Africa.

G. K. Venayagamoorthy is with the Department of Electrical and Computer Engineering, University of Missouri-Rolla, MO 65409-0040 USA. He is also with the School of Electrical and Electronic Engineering, University of Natal, Durban 4041, South Africa.

R. G. Harley is with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0250 USA. He is also with the School of Electrical and Electronic Engineering, University of Natal, Durban 4041, South Africa.

D. C. Wunsch is with the Department of Electrical and Computer Engineering, University of Missouri-Rolla, MO 65409-0040 USA.

Publisher Item Identifier S 1045-9227(02)04441-7.

the system. Furthermore, to update weights of the ANN identifier online, gradient descent algorithms are commonly used. However, it is well known in adaptive control that a brute force correction of controller parameters, based on the gradients of output errors, can result in instability even for some classes of linear systems [8], [9]. Hence, to avoid the possibility of instability during online adaptation, some researchers proposed using ANNs such as radial basis functions, where variable network parameters occur linearly in the network outputs, such that a stable updating rule can be obtained [10]. To date, the development of nonlinear control using ANNs is similar to that of linear adaptive control. Unfortunately, unlike linear adaptive control, where a general controller structure to stabilize a system can be obtained with only the knowledge of relative degrees, stabilizing controllers for nonlinear systems are difficult to design. As a result, most research on ANN based controllers has focused on nonlinear systems, whose stabilizing controllers are readily available once some unknown nonlinear parts are identified, such as

$$x^n = f(x^{n-1}, \ldots, x) + bu \tag{1}$$

with full state feedback, where f is to be estimated by an ANN. Even though some methods have been suggested for using ANNs in the context of a general controller structure [11], [12], the stability implication of updating a network online is unknown. Furthermore, since an ANN controller can have many weights, it is questionable whether the network can converge fast enough to achieve good performance. Besides, 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 technique for designing a turbogenerator neurocontroller, which overcomes the stability issues [13], the problem of residual error in the system identification [14], input uncertainties [15], and the computational load of online training. In this new technique the neurocontroller uses a so-called adaptive critic based on reinforcement learning and dynamic programming. The reasons behind these good and important features are discussed in Section IV. The neurocontroller is trained in an offline mode prior to commissioning. Two different types of adaptive critics are discussed, heuristic dynamic programming (HDP) and dual heuristic programming (DHP). Results are presented, showing that DHP produces the best results.

II. 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 [16] as a new optimization technique combining concepts of reinforcement learning and approximate dynamic programming. For a given series of control actions that must be taken sequentially, and not knowing the effect of these actions until the end of the sequence, it is impossible to design an optimal controller using the traditional supervised learning neural network. The adaptive critic method determines optimal control laws for a system by successively adapting two ANNs, namely, an action neural network (which dispenses the control signals) and a critic neural network (which "learns" the desired performance index for some function associated with the performance index). These two neural networks approximate the Hamilton–Jacobi–Bellman equation associated with optimal control theory. The adaptation process starts with a nonoptimal, arbitrarily chosen, control by the action network; the critic network then guides the action network toward the optimal solution at each successive adaptation. During the adaptations, neither of the networks need any "information" of an optimal trajectory, only the desired cost needs to be known. Furthermore, this method determines optimal control policy for the entire range of initial conditions and needs no external training, unlike other neurocontrollers.

Dynamic programming prescribes a search which tracks backward from the final step, retaining in memory all suboptimal paths from any given point to the finish, until the starting point is reached. 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 obtain 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 the critic network which has certain desirable attributes.

The technique of using a critic, removes the learning process one step from the control network (traditionally called the "action network" or "actor" in ACD literature), so the desired trajectory 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 the action network as one of its inputs, directly or indirectly. Different types of critics have been proposed. For example, Watkins [17] developed a system known as Q-learning, explicitly based on dynamic programming. Werbos, on the other hand, developed a family of systems for approximating dynamic programming [16]; his approach subsumes other designs for continuous domains. For example, Q-learning becomes a special case of action-dependent heuristic dynamic programming (ADHDP), which is a critic approximating the J function (see Section II-B below), in Werbos' family of adaptive critics. A critic which approximates only the derivatives of the function J with respect to its states, called the DHP, and a critic approximating both J and its derivatives, called the globalized dual heuristic programming (GDHP), complete this ACD family. These systems do not require exclusively neural network implementations, since any differentiable structure is suitable as a building block. The interrelationships between members of the ACD family have been generalized and explained in detail by Prokhorov [18], [19], whose results have been modified for the study in this paper as shown in Sections II-B, II-C and IV. This paper compares HDP and DHP types of critics for neurocontroller implementations, against the results obtained using conventional PID controllers [20] for a turbogenerator plant.

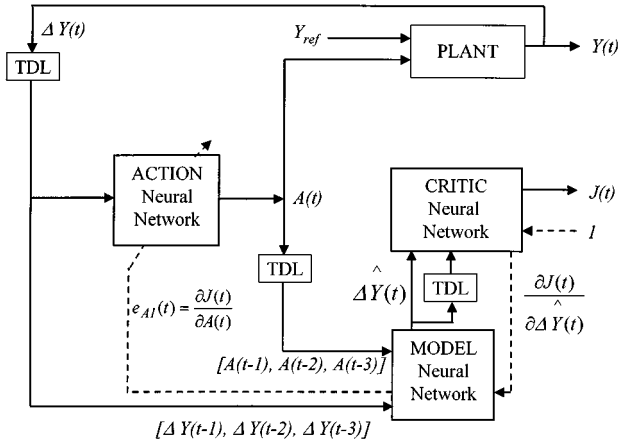

Fig. 2. Action adaptation in HDP. $A(t)$ is the control vector. The constant $\partial J/\partial J = 1$, is the error signal for training the action network in order to minimize J. The backpropagation path is shown by dashed line.

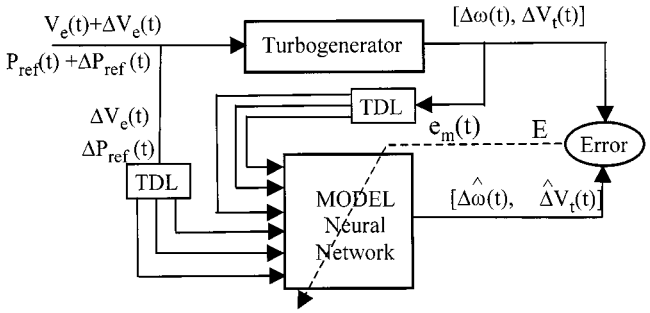

Fig. 3. Neural-network modeling of the plant in Fig. 1, using the backpropagation algorithm.

B. Heuristic Dynamic Programming Neurocontroller

Fig. 2 shows a HDP critic neural network connected to the action neural network through a neural network model of the plant, and is therefore called a model-dependent critic design. All three these different neural networks are described in the following sections.

1) Model Neural Network: Fig. 3 illustrates how the ANN model/identifier is trained to identify the dynamics of the plant. The ANN identifier structure is a three-layer feedforward neural network with 12 inputs, a single hidden layer with 14 sigmoidal neurons, and two linear output neurons as shown in Fig. 4. The inputs are the *actual* deviation in the input to the exciter ΔV_e, the *actual* deviation in the input to the turbine ΔP_{ref}, the *actual* terminal voltage deviation ΔV_t and the *actual* speed deviation of the generator $\Delta\omega$. These four inputs are time delayed by a sample period of 20 ms and together with the eight previously delayed values form the 12 inputs to the ANN identifier. The ANN identifier outputs are the one step ahead *estimated* terminal voltage deviation $\Delta\hat{V}_t$ and *estimated* speed deviation $\Delta\hat{\omega}$ of the turbogenerator. Pseudorandom signals are applied to the exciter and the microturbine of the plant with the switches S1 and S2 in position 1 in Fig. 1, in order to train the ANN model/identifier, for a period of time at different operating conditions until satisfactory identification results are obtained. The input and output weights W_M, of the ANN model are then fixed during the further development of the critic and the action neural

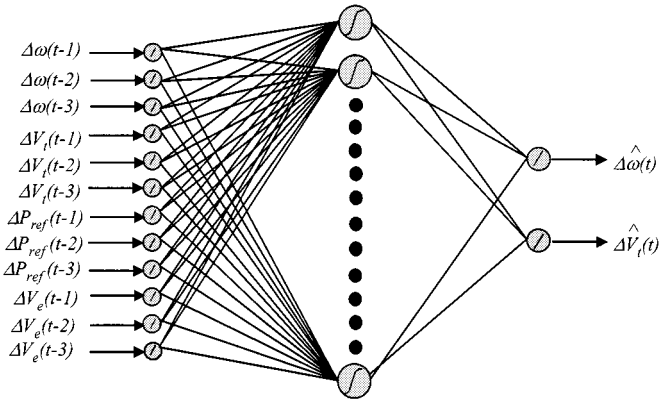

Fig. 4. Model neural-network structure with 12 inputs, 14 sigmoidal hidden layer neurons, and two linear output neurons.

networks. The backpropagation algorithm [21] is used for updating W_M in the ANN model based on the error $e_M(t)$ at E in Fig. 3 given in (2)

$$e_M(t) = \left\{ \left[\Delta V_t(t) - \Delta \hat{V}_t(t) \right], \left[\Delta\omega(t) - \Delta\hat{\omega}(t) \right] \right\}. \quad (2)$$

The training is carried out to minimize (3). The change in the weights is calculated using the backpropagation algorithm based on a gradient descent method and is given in (3). The ANN identifier weight update equation is given in (4) [21]. More details on the ANN model developments can be found in [4]

$$E_M(t) = \tfrac{1}{2} \sum_t e_M^2(t) \quad (3)$$

$$\Delta W_M = -\eta_1 e_M(t) \frac{\partial e_M(t)}{\partial W_M} \quad (4)$$

where η_1 is a positive learning rate.

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

$$J(t) = \sum_{k=0}^{\infty} \gamma^k U(t+k) \quad (5)$$

where γ is a discount factor for finite horizon problems $(0 < \gamma < 1)$, and $U(.)$ is the utility function or local cost. The configuration for training the critic network is shown in Fig. 5. The critic network is a neural network trained forward in time, which is of great importance for real-time operation. The structure of the critic neural network is shown in Fig. 6, and consists of a three-layer feedforward network with six inputs, a single hidden layer with 13 sigmoidal neurons, and a single linear output neuron. The inputs to the critic are the *estimated* speed deviation and *estimated* terminal voltage deviation (outputs of the model neural network), and their two time-delayed values, respectively, forming the six inputs. The critic network tries to minimize the following error measure over time:

$$\|E_{C1}\| = \tfrac{1}{2} \sum_t e_{C1}^2(t) \quad (6)$$

$$e_{C1}(t) = J[\Delta Y(t)] - \gamma J[\Delta Y(t+1)] - U(t) \quad (7)$$

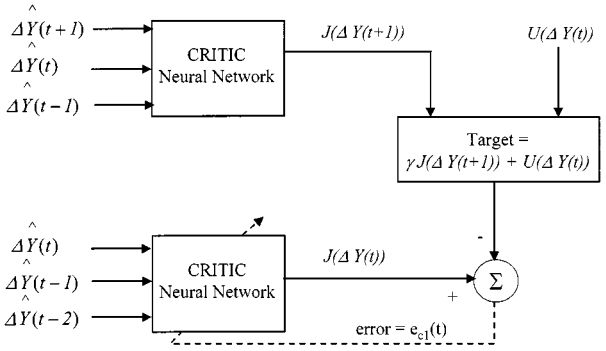

Fig. 5. Critic adaptation in HDP. The same critic network is shown for two consecutive times, t and $t+1$. The critic's output $J(t+1)$ at time $t+1$, is necessary to generate a target signal $\gamma J(t+1)+U(t)$, for training the critic network. The discount factor γ is chosen to be 0.5. The backpropagation path is shown by the dashed line.

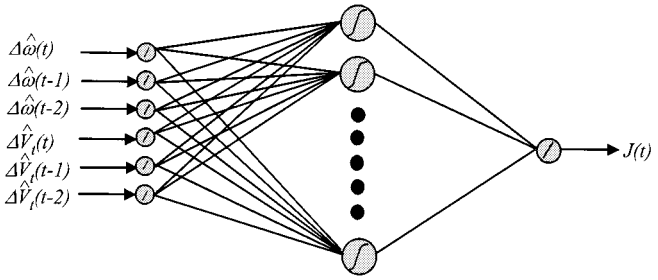

Fig. 6. HDP critic neural network structure with six inputs, ten sigmoidal hidden layer neurons, and one linear output neuron.

where $\Delta Y(t)$ is a vector of observables of the plant (or the states, if available). The necessary condition for (6) to be minimum is given in (8)

$$\frac{1}{2}\frac{\partial}{\partial W_{C1}}\langle e_{C1}^2(t)\rangle = \left\langle e_{C1}\frac{\partial e_{C1}(t)}{\partial W_{C1}}\right\rangle = 0. \tag{8}$$

The expression for the weights' update for the critic neural network is as follows:

$$\Delta W_{C1} = -\eta_2 e_{C1}(t)\frac{\partial e_{C1}(t)}{\partial W_{C1}} \tag{9}$$

$$\Delta W_{C1} = -\eta_2\{J[\Delta Y(t)] - \gamma J[\Delta Y(t+1)] - U(t)\} \cdot \frac{\partial\{J[\Delta Y(t)] - \gamma J[\Delta Y(t+1)] - U(t)\}}{\partial W_{C1}} \tag{10}$$

where η_2 is a positive learning rate and W_{C1} is the weights of the HDP critic neural network.

The same critic neural network is shown in two consecutive moments in time in Fig. 6. The critic neural 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 $J(t)$.

3) Action Neural Network: The action neural network in Figs. 2 and 7 is a three-layer feedforward neural network with six inputs, a single hidden layer with ten sigmoidal neurons, and two linear output neurons. The inputs to the action neural network are the turbogenerator's *actual* speed deviation and *actual* terminal voltage deviation, and their two time-delayed values respectively, forming the six inputs. The two outputs

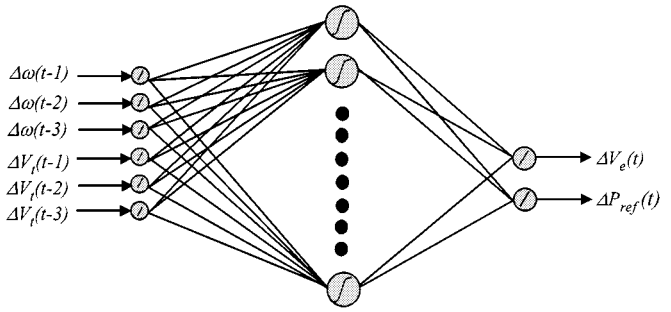

Fig. 7. Action neural network structure for both HDP and DHP schemes with six inputs, ten sigmoidal hidden layer neurons, and two linear output neurons.

of the action neural network, $A(t) = [\Delta V_e, \Delta P_{\text{ref}}]$, the *deviation* in the exciter voltage, augments the exciter voltage and *deviation* in the turbine power, augments the turbine input power. The objective of the action neural network in Fig. 2, 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 neural network with an error signal $\partial J(t)/\partial A(t)$. The gradient of the cost function $J(t)$, with respect to the outputs $A(t)$, of the action network, is obtained by backpropagating $\partial J/\partial J$ (i.e., the constant 1) through the critic neural network and then through the pretrained model neural network to the action neural network. This gives $\partial J(t)/\partial A(t)$ and $\partial J(t)/\partial W_{A1}(t)$ for all the outputs of the action neural network, and all the action neural network's weights W_{A1}, respectively. The action neural network is trained to minimize (11). The expression for the weights' update in the action neural network is based on an error feedback from the critic neural network backpropagated through the model neural network using the backpropagation algorithm, and is given in (13) and (14)

$$\|E_{A1}\| = \tfrac{1}{2}\sum_t e_{A1}^2(t) \tag{11}$$

$$e_{A1}(t) = \frac{\partial J(t)}{\partial A(t)} \tag{12}$$

$$\Delta W_{A1} = -\eta_3 e_{A1}(t)\frac{\partial e_{A1}(t)}{\partial W_{A1}} \tag{13}$$

$$\Delta W_{A1} = -\eta_3\frac{\partial J(t)}{\partial A(t)}\frac{\partial}{\partial W_{A1}}\left(\frac{\partial J(t)}{\partial A(t)}\right) \tag{14}$$

where η_3 is a positive learning rate and W_{A1} is the weights of the HDP action neural network.

C. Dual Heuristic Programming Neurocontroller

The critic neural network in the DHP scheme in Fig. 8, estimates the derivatives of J with respect to the vector ΔY, and learns minimization of the following error measure over time:

$$\|E_{C2}\| = \sum_t e_{C2}^T(t)e_{C2}(t) \tag{15}$$

where

$$e_{C2}(t) = \frac{\partial J[\Delta Y(t)]}{\partial \Delta Y(t)} - \gamma\frac{\partial J[\Delta Y(t+1)]}{\partial \Delta Y(t)} - \frac{\partial U(t)}{\partial \Delta Y(t)} \tag{16}$$

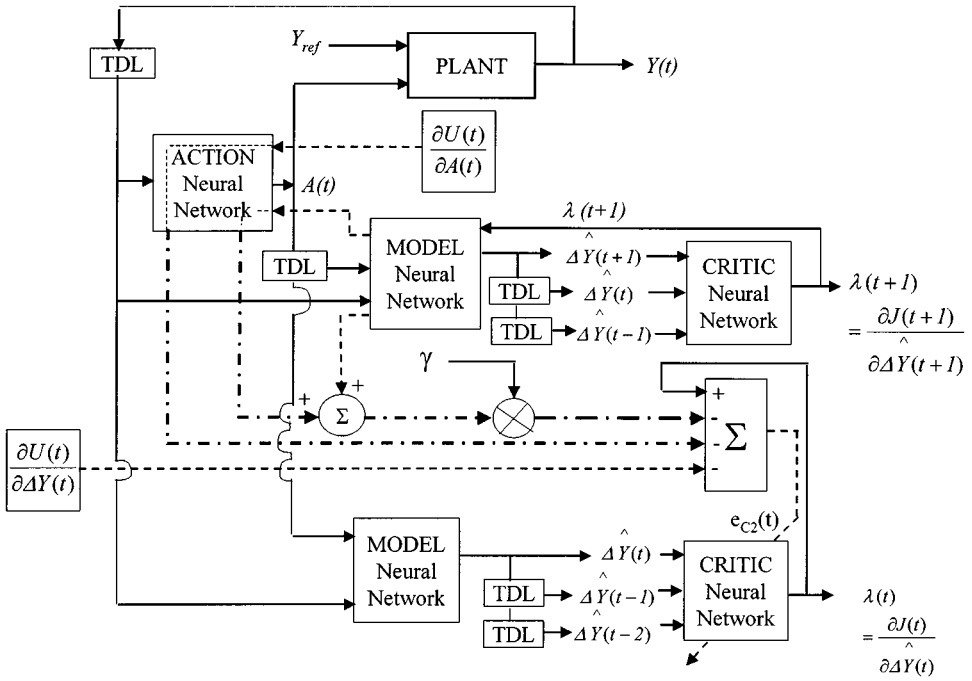

Fig. 8. Critic adaptation in DHP. This diagram shows the implementation of (10). The same critic network is shown for two consecutive times, t and $t+1$. The discount factor γ is chosen to be 0.5. Backpropagation paths are shown by dashed lines. The output of the critic network $\lambda(t+1)$ is backpropagated through the model network from its outputs to its inputs, yielding the first term of (9) and $\partial J(t+1)/\partial A(t)$. The latter is backpropagated through the action network from its outputs to its inputs forming the second term of (9). Backpropagation of the vector $\partial U(t)/\partial A(t)$ through the action network results in a vector with components computed as the last term of (10). The summer produces the error vector $E_2(t)$ used for training the critic network.

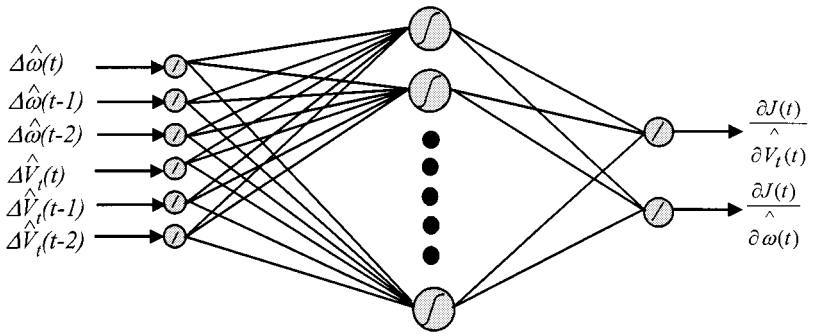

Fig. 9. DHP critic neural network structure with six inputs, ten sigmoidal hidden layer neurons, and two linear output neurons.

where $\partial(.)/\partial\Delta Y(t)$ is a vector containing partial derivatives of the scalar $(.)$ with respect to the components of the vector ΔY. The DHP critic neural network structure is similar to that of the HDP critic's, except that the DHP critic has two linear output neurons as shown in Fig. 9. The critic neural 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. 8, where the paths of derivatives and adaptation of the critic are depicted by dashed lines.

In the DHP scheme, application of the chain rule for derivatives yields

$$\frac{\partial J[\Delta Y(t+1)]}{\partial \Delta Y_j(t)} = \sum_{i=1}^{n} \lambda_i(t+1) \frac{\partial Y_i(t+1)}{\partial \Delta Y_j(t)} + \sum_{k=1}^{m}\sum_{i=1}^{n} \lambda_i(t+1) \frac{\partial \Delta Y_i(t+1)}{\partial A_k(t)} \frac{\partial A_k(t)}{\partial \Delta Y_j(t)} \quad (17)$$

where $\lambda_i(t+1) = \partial J[\Delta Y(t+1)]/\partial \Delta Y_i(t+1)$, and n, m are the numbers of outputs of the model and the action neural networks, respectively. By exploiting (17), each of n components of the vector $e_{C2}(t)$ from (16) is determined by

$$e_{C2}(t) = \frac{\partial J[\Delta Y(t)]}{\partial \Delta Y_j(t)} - \gamma \frac{\partial J[\Delta Y(t+1)]}{\partial \Delta Y_j(t)} - \frac{\partial U(t)}{\partial \Delta Y_j(t)} - \sum_{k=1}^{m} \frac{\partial U(t)}{\partial A_k(t)} \frac{\partial A_k(t)}{\partial \Delta Y_j(t)}. \quad (18)$$

The adaptation of the action neural network in Fig. 8, is illustrated in Fig. 10 which propagates $\lambda(t+1)$ back through the model network to the action network. The goal of such adaptation can be expressed as follows [18]:

$$\frac{\partial U(t)}{\partial A(t)} + \gamma \frac{\partial J(t+1)}{\partial A(t)} = 0 \qquad \forall t. \quad (19)$$

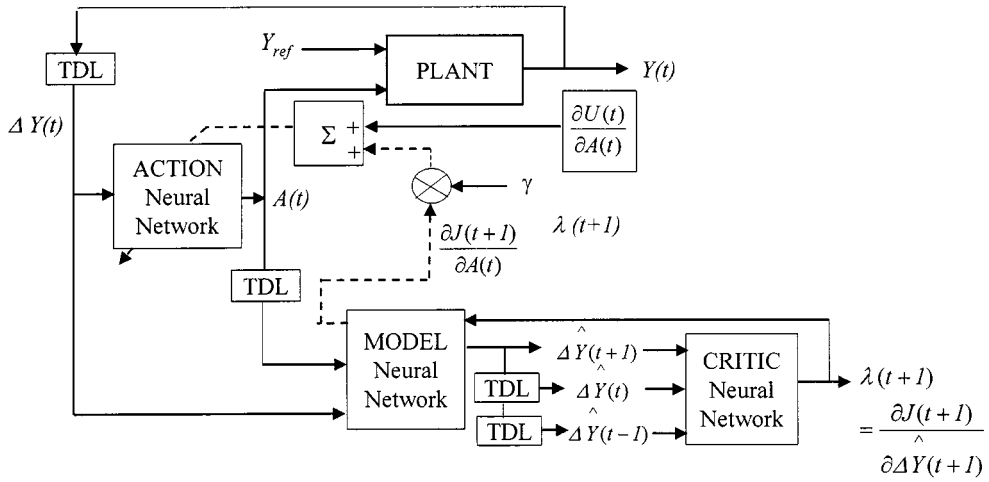

Fig. 10. Action adaptation in DHP. The discount factor γ is chosen to be 0.5. The backpropagation path is shown by dashed line. The output of the critic $\lambda(t+1)$ at time $(t+1)$ is backpropagated through the model network from its outputs to its inputs (output of the action network), and the resulting vector multiplied by the discount factor ($\gamma = 0.5$) and added to $\partial U(t)/\partial A(t)$. Then an incremental adaptation of the action network is carried in accordance with (11).

The weights' update expression [18], when applying backpropagation, is as follows:

$$\Delta W_{A2} = -\eta_4 \left[\frac{\partial U(t)}{\partial A(t)} + \frac{\partial J(t+1)}{\partial A(t)}\right]^T \frac{\partial A(t)}{\partial W_{A2}} \qquad (20)$$

where η_4 is a positive learning rate and W_{A2} is the weights of the action neural network in the DHP scheme. The structure of the action neural network is identical to that of the action network in the HDP scheme. The general derivation of the equations in this section are shown in [18] in detail.

The word "dual" is used to describe the fact that the target outputs for the DHP critic training are calculated using backpropagation in a generalized sense; more precisely, it does use dual subroutines (states and co-states) to backpropagate derivatives through the model and action neural networks, as shown in Fig. 8. The dual subroutines and more explanations are found in [16], [21].

D. Global Dual Heuristic Programming Neurocontroller

The GDHP critic minimizes the error with respect to both J and its derivatives. Training the critic neural network in GDHP utilizes an error measure which is a combination of the error measures of HDP and DHP [see (6) and (15)]. The training of GDHP critic is a complex task and the resulting behavior is expected to be superior. More detail on GDHP and its implementations can be found [18], [19], [22].

III. Practical Turbogenerator System

The neurocontroller is evaluated by applying it to a special scaled down laboratory power system of which the main components are described in this section. The power system in Fig. 1 consists of a micro-alternator, driven by a dc motor whose torque-speed characteristics are controlled by a power electronic converter to act as a micro-turbine, and a single short transmission line which links the micro-alternator to a voltage source which has a constant voltage and frequency, called an infinite bus. The 3 kW, 220 V, three phase micro-alternator was designed to have all its per-unit parameters, except the field winding resistance, the same as those normally expected of a 30–1000 MW alternator. The micro-alternator parameters, determined by the IEEE standards are given in Table I [23]. A time constant regulator is used to insert negative resistance in series with the field winding circuit [23], in order to reduce the actual field winding resistance to the correct per-unit value.

TABLE I
Micro-Alternator Parameters

T_{d0}' = 6.69 s	T_{q0}" = 0.25 s	X_d' = 0.205 pu
T_d' = 0.66 s	T_q" = 27 ms	X_d" = 0.164 pu
T_{d0}" = 33 ms	T_{kd} = 38 ms	X_q = 1.98 pu
T_d" = 26.4 ms	X_d = 2.09 pu	X_q" = 0.213 pu

The practical system uses a conventional AVR and exciter combination of which the transfer function block diagram is shown in Fig. 11, and the time constants and gain are given in Table II [6]. The exciter saturation factor S_e is given by

$$S_e = 0.6093 \exp(0.2165\, V_{\text{field}}). \qquad (21)$$

T_{v1}, T_{v2}, T_{v3}, and T_{v4} are the time constants of the PID voltage regulator compensator; T_{v5} is the input filter time constant; T_e is the exciter time constant; K_{av} is the AVR gain; V_{fdm} is the exciter ceiling voltage; and, V_{ma} and V_{mi} are the AVR maximum and minimum ceiling voltages.

A separately excited 5.6 kW thyristor controlled dc motor is used as a prime mover, called the micro-turbine, to drive the micro-alternator. The torque-speed characteristic of the dc motor is controlled to follow a family of rectangular hyperbola to emulate the different positions of a steam valve, as would occur in a real typical high pressure (HP) cylinder turbine. The three low pressure (LP) cylinders' inertia are represented by appropriately scaled flywheels attached to the microturbine shaft. The microturbine and governor combination transfer function block diagram is shown in Fig. 12, where, P_{ref} is the turbine input power set point value, P_m is the turbine output power, and

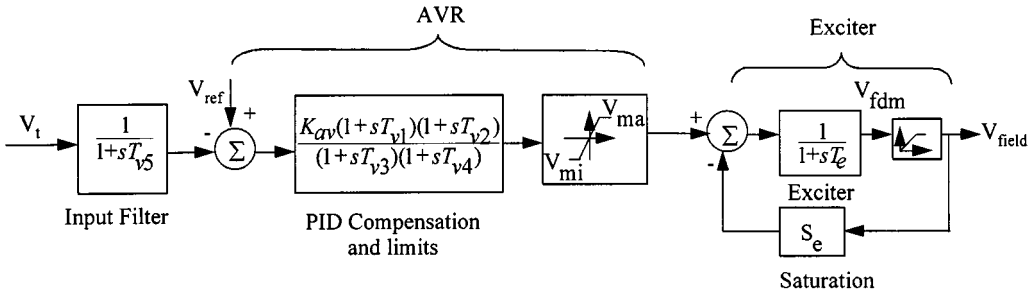

Fig. 11. Block diagram of the AVR and exciter combination. AVR and exciter parameters are given in Table II.

TABLE II
AVR AND EXCITER TIME CONSTANTS, AND GAIN

	Actual Value
T_{v1}	0.616 s
T_{v2}	2.266 s
T_{v3}	0.189 s
T_{v4}	0.039 s
T_{v5}	0.0235 s
T_e	0.47 s
K_{av}	0.003

$\Delta\omega$ is the speed deviation from the synchronous speed. The turbine and governor time constants and gain are given in Table III [6].

The values of the K_{av} and K_g in Tables II and III, respectively, were obtained by suitable choices of the gain and phase margins in each case, as described in [20]. Transmission lines are represented by using banks of lumped inductors and capacitors.

The nonlinear time-invariant equations for the block labeled plant in Fig. 1 are of the form

$$\dot{x} = f(x, u) + g(x) \tag{22}$$

where $g(x)$ contains the nonlinear terms. Equation (1) represents a 13-order nonlinear system and is developed from the two axis machine dq-equations [1] combined with those of Figs. 11 and 12, with the following selected states:

$$x = [\delta \quad \Delta\omega \quad i_d \quad i_f \quad i_{kd} \quad i_q \quad i_{kq}] \tag{23}$$

where the first two states are the rotor angle and the speed deviation, and the other states are the currents in the d, q, field, and damper coils. The plant is simulated in MATLAB/SIMULINK and details of this can be found in [5].

IV. General Training Procedure for the Critic and Action Networks

The training procedure is that suggested in [18] and it is applicable to any ACD. It consists of two separate training cycles: one for the critic, and the other for the action. An important measure is that the action neural network is pretrained with conventional controllers controlling the plant in a linear region. The critic's adaptation is done initially with the pretrained action neural network, to ensure that the whole system, consisting of the ACD and the plant remains stable. Then the action neural network is trained further while keeping the critic neural 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 pretrained model neural network, and W_C and W_A are initialized to small random values.

In the critic's training cycle, an incremental optimization of (6) and/or (15) is carried out using a suitable optimization technique. The following operations are repeated N_C times.

1) Initialize $t = 0$ and $\Delta Y(0)$.
2) Compute output of the critic neural network at time t, $J(t)$ or $\lambda(t) = f_C(\Delta Y(t), W_C)$.
3) Compute output of the action neural network at time t, $A(t) = f_A(\Delta Y(t), W_A)$.
4) Compute output of the model neural network at time $t+1$, $\Delta Y(t+1) = f_M(\Delta Y(t), A(t), W_M)$.
5) Compute output of the critic neural network at time $t+1$, $J(t+1)$ or $\lambda(t+1) = f_C(\Delta Y(t+1), W_C)$.
6) Compute the critic neural network error at time t, $e_C(t)$ from (7) or (16).
7) Update the critic neural network's weights using the backpropagation algorithm.
8) Repeat steps 2) to 7).

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

In the action neural network's training cycle, an incremental learning is also carried out using the backpropagation algorithm, as in the critic neural network's training cycle above, and the list of operations for the action neural network's training cycle is almost the same as that for the critic network's cycle above [steps 1) to 7)]. However, (12) and/or (19) are used for updating the action neural network's weights instead of using (6) and/or (15) and $\partial J/\partial W_C$. The action's training cycle is repeated N_A times while keeping the critic's weights W_C fixed. N_C and N_A are the lengths of the corresponding training cycles. It is important that the whole system consisting of the ACD and the plant remains stable while both of the critic and action neural networks undergo adaptation.

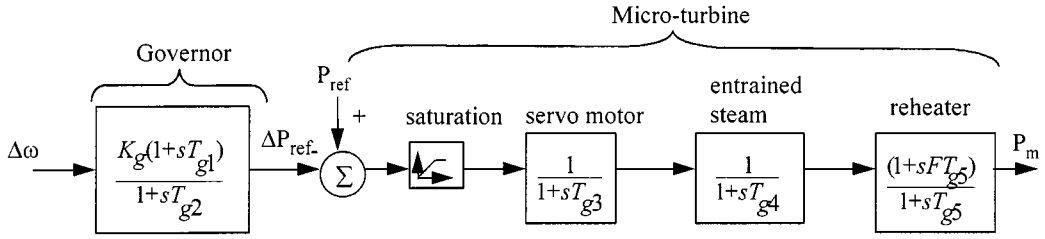

Fig. 12. Block diagram of the microturbine and governor combination. Governor and micro-turbine parameters are given in Table III.

TABLE III
MICROTURBINE AND GOVERNOR TIME CONSTANTS, AND GAIN

	Actual Value
Phase advance compensation, T_{g1}	0.264 s
Phase advance compensation, T_{g2}	0.0264 s
Servo time constant, T_{g3}	0.15 s
Entrained steam delay, T_{g4}	0.594 s
Steam reheat time constant, T_{g5}	2.662 s
pu shaft output ahead of reheater, F	0.322 s
Governor gain, K_g	0.05

V. 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 optimal trajectory. In HDP, this information is obtained indirectly by backpropagation through the critic neural network. It has a potential problem of not being too smooth since the critic neural network in HDP is not trained to approximate derivatives of J directly.

DHP has an important advantage over HDP since its critic neural network builds a representation for the derivatives of J directly by being explicitly trained on them through $\partial U(t)/\partial[\Delta Y(t)]$ and $\partial U(t)/\partial A(t)$. For instance, in the area of model-based control, as in the case of this paper, a pretrained model neural network and well-defined $\partial U(t)/\partial[\Delta Y(t)]$ and $\partial U(t)/\partial A(t)$ exist. To adapt the action neural network, only the derivatives $\partial J(t)/\partial[\Delta Y(t)]$ or $\partial J(t)/\partial A(t)$ are required, rather than the J function itself. However, the approximation of these derivatives is already a *direct* output of the DHP critic.

VI. RESULTS WITH THE TRAINED ACTION NETWORK

A discount factor γ of 0.5 and the utility function given in (24) are used in the Bellman equation [(5)] and in the training of the critic neural network [see (6) and (15)], and of the action neural network [eqs. (12) and (19)]. Once the critic and action neural networks' weights have converged, the training stops and the action neural network is connected to the plant (Fig. 1)

$$U(t) = [4\Delta V(t)+4\Delta V(t-1)+16\Delta V(t-2)]^2 + [0.4\Delta\omega(t)+0.4\Delta\omega(t-1)+0.16\Delta\omega(t-2)]^2. \quad (24)$$

The dynamic and transient operation of the action neural network (neurocontroller) is compared with the operation of a conventional PID controller (AVR and turbine governor), under two different conditions: $\pm 5\%$ step changes in the terminal voltage setpoint, and a three-phase short circuit at the infinite bus. Each of these is investigated for the turbogenerator operating at different power factors and transmission line impedances.

Figs. 13 and 14 show the performance of the different controllers for $\pm 5\%$ 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 $Z_1 = 0.02 + j0.4$ pu. Fig. 15 shows a turbogenerator operating under the same conditions but experiencing a temporary 50 ms three phase short circuit at the infinite bus. Fig. 16 shows a turbogenerator under the same terminal conditions as in Fig. 15 but experiencing a temporary 50 ms three phase short circuit at the infinite bus, with an increased transmission line impedance $Z_2 = 0.025 + j0.6$ pu. The results with the conventional AVR and governor controllers, and those of the HDP and the DHP neurocontrollers, are labeled as CONV, HDP, and DHP, respectively, in these figures.

Figs. 13 and 14 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 (5). On the other hand, Figs. 15 and 16 show that for the short circuit disturbances, the DHP-based neurocontroller has the best damping compared to both the HDP neurocontroller and the conventional controller. The designer/power station engineer has the final choice on whether terminal voltage or rotor angle damping is more important. It must be emphasized that these significant results have been obtained with training in an offline mode only, hence, avoiding continually online training.

The HDP and DHP neurocontroller tested above under the different tests all have fixed parameters for their neural networks which are trained off-line. This leads to the fact that there are no adaptive parameters with the neurocontroller and therefore avoids the risk of instability. The convergence guarantee of the critic and action neural networks during offline training has been shown in [24], [25]. In addition, the heavy computational load only arises during the offline training phase and therefore makes the online real-time implementation cost of the neurocontrollers cheaper. In this paper, HDP and DHP neurocontrollers, based on a model-based adaptive critic design approach, have been

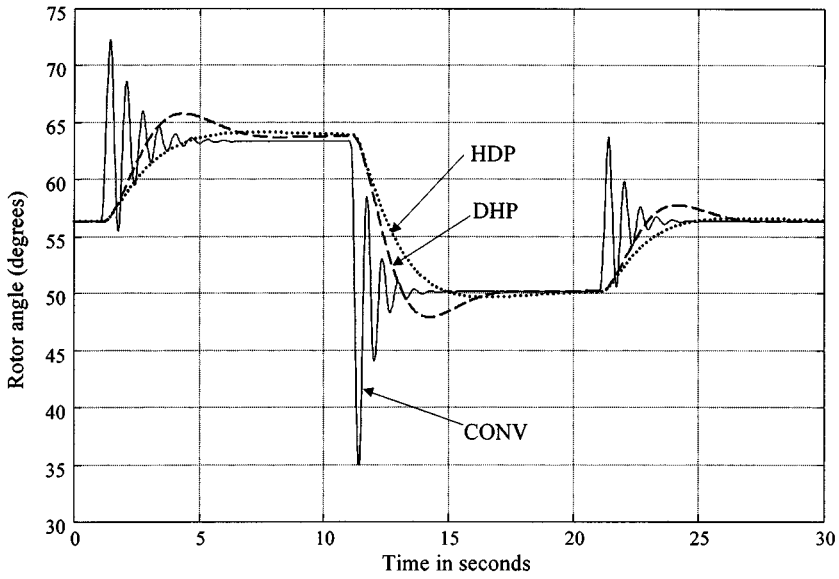

Fig. 13. Rotor angle variation for $\pm 5\%$ step changes in the desired terminal voltage.

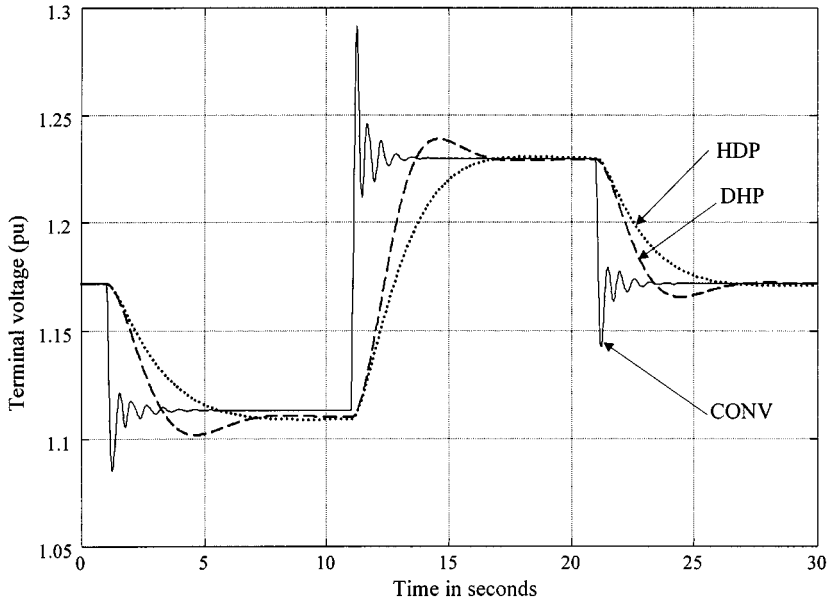

Fig. 14. Terminal voltage variations for $\pm 5\%$ step changes in the desired terminal voltage.

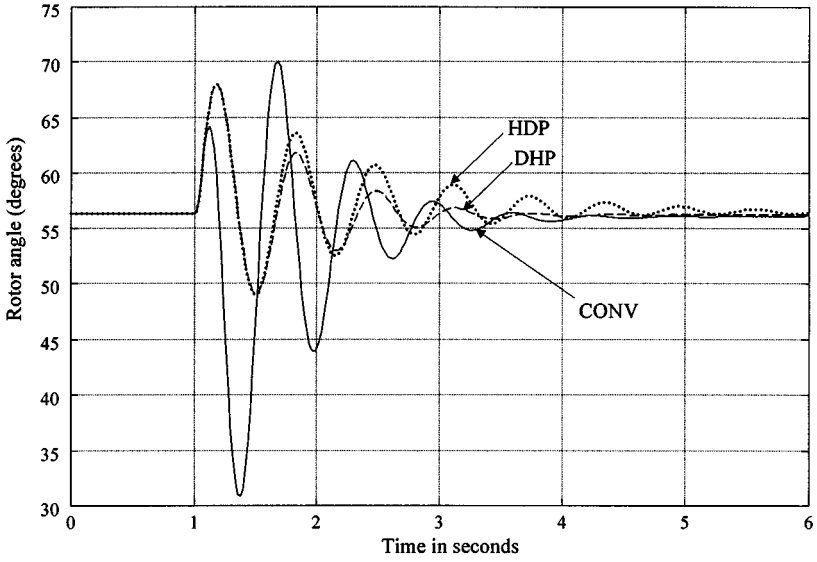

Fig. 15. Rotor angle variation for a temporary 50 ms three phase short circuit at the infinite bus ($P = 1$ pu, $pf = 0.85$ lagging, $Z = 0.02 + j0.4$ pu).

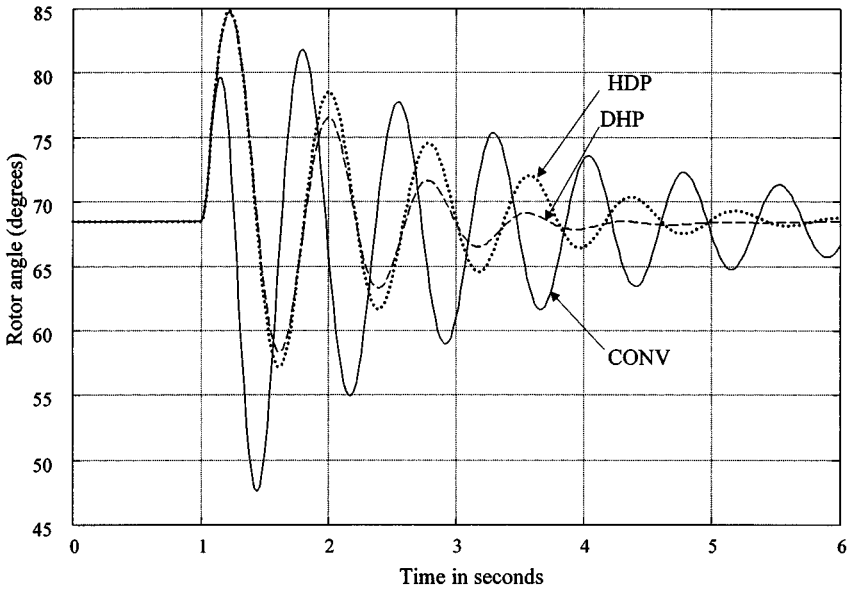

Fig. 16. Rotor angle variation for a temporary 50 ms three-phase short circuit at the infinite bus with increased transmission line impedance ($P = 1$ pu, $pf = 0.85$ lagging, $Z = 0.025 + j0.6$ pu).

demonstrated to have superior performance (compared to conventional controllers) with just an approximate model (using a neural network) of the plant being controlled. This benefit of a neural-network model agrees with the conclusions on the comparison of using exact and approximate models in adaptive critic designs which was explicitly shown in [14]. With regard to handling uncertainties, a Lyapunov based theory for robust stability of the adaptive critic design-based controllers with input uncertainty has been developed in [15]. All these features are desirable and important for industrial applications which require a neurocontroller technology that is nonlinear, robust, and stable.

VII. Conclusion

This paper has presented a new and novel technique in which adaptive critic design based neurocontrollers can be in the feedback loops of turbogenerators without needing continually online training. This avoids risks of instability due to continual online training. DHP performed excellently during the short circuit tests compared to HDP and the conventional controller. This paper has therefore demonstrated that there is a potential for adaptive critic designs for real-time optimal control of turbogenerators.

References

[1] B. Adkins and R. G. Harley, *The General Theory of Alternating Current Machines*. London, U.K.: Chapman and Hall, 1975.

[2] K. J. Hunt, D. Sbarbaro, R. Zbikowski, and P. J. Gawthrop, "Neural networks for control systems—A survey," *Automatica*, vol. 28, no. 6, pp. 1083–1112, 1992.

[3] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Trans. Neural Networks*, vol. 1, pp. 4–27, Mar. 1990.

[4] G. K. Venayagamoorthy and R. G. Harley, "Implementation of an adaptive neural network identifier for effective control of turbogenerators," in *Proc. IEEE Budapest PowerTech Conf.*, 1999, paper BPT99-431-23.

[5] ——, "Simulation studies with a continuously online trained artificial neural network controller for a micro-turbogenerator," in *Proc. IEE Int. Conf. Simulation*, vol. 457, 1998, pp. 405–412.

[6] ——, "Experimental studies with a continually online trained artificial neural network controller for a turbogenerator," in *Proc. Int. Joint Conf. Neural Networks (IJCNN)*, vol. 3, 1999, pp. 2158–2163.

[7] D. Flynn, S. MacLoone, G. W. Irwin, M. D. Brown, E. Swidenbank, and B. W. Hogg, "Neural control of turbogenerator systems," *Automatica*, vol. 33, no. 11, pp. 1961–1973, 1997.

[8] P. Parks, "Liapunov redesign of model reference adaptive control systems," *IEEE Trans. Automat. Contr.*, vol. AC-11, pp. 362–367, 1966.

[9] P. Osburn, H. Whitaker, and A. Kezer, "New developments in the design of model reference adaptive control systems," in *Proc. IAS 29th Annu. Meet.*, New York, 1961.

[10] R. M. Sanner and J. E. Slotine, "Gaussian networks for direct adaptive control," *IEEE Trans. Neural Networks*, vol. 3, pp. 837–863, Nov. 1992.

[11] M. I. Jordan and D. E. Rumelhart, "Forward models: Supervised learning with a distal teacher," *Cognitive Sci.*, vol. 16, pp. 307–354, 1992.

[12] A. U. Levin and K. S. Narenda, "Control of nonlinear dynamical systems using neural networks: Controllability and stabilization," *IEEE Trans. Neural Networks*, vol. 4, pp. 192–206, Mar. 1993.

[13] D. Prokhorov and L. A. Feldkamp, "Analyzing for stability with adaptive critics," in *Proc. Int. Conf. Syst., Man, Cybern.*, 1998, pp. 1658–1161.

[14] T. T. Shannon and G. G. Lendaris, "Qualitative models for adaptive critic neurocontrol," in *Proc. Int. Joint Conf. Neural Networks*, vol. 1, IJCNN 1999, pp. 455–460.

[15] Z. Huang and S. N. Balakrishnan, "Robust adaptive critic based neurocontrollers for systems with input uncertainties," in *Proc. Int. Joint Conf. Neural Networks (IJCNN)*, vol. 3, 2000, pp. 67–72.

[16] P. Werbos, "Approximate dynamic programming for real-time control and neural modeling," in *Handbook of Intelligent Control*, White and Sofge, Eds. New York: Van Nostrand Reinhold, pp. 493–525.

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

[18] D. Prokhorov and D. Wunsch, "Adaptive critic designs," *IEEE Trans. Neural Networks*, vol. 8, pp. 997–1007, Sept. 1997.

[19] D. Prokhorov, "Adaptive critic designs and their applications," Ph.D. dissertation, Texas Tech. Univ., Lubbock, 1997.

[20] W. K. Ho, C. C. Hang, and L. S. Cao, "Tuning of PID controllers based on gain and phase margin specifications," in *Proc. 12th Triennial World Congr. Automat. Contr.*, 1993, pp. 199–202.

[21] P. J. Werbos, *Roots of Backpropagation*. New York: Wiley, 1994.

[22] G. K. Venayagamoorthy, R. G. Harley, and D. C. Wunsch, "Adaptive critic based neurocontroller for turbogenerators with global dual heuristic programming," in *Proc. IEEE PES Winter Meet.*, vol. 1, 2001, pp. 291–294.

[23] D. J. N. Limebeer, R. G. Harley, and M. A. Lahoud, "A laboratory system for investigating subsynchronous resonance," in *Proc. 1980 IEEE PES Winter Meet.*, paper A80-0190-0.

[24] D. Prokhorov and D. C. Wunsch, "Convergence of critic-based training," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, vol. 4, 1997, pp. 3057–3060.

[25] D. Prokhorov and L. A. Feldkamp, "Analyzing for Lyapunov stability with adaptive critics," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, vol. 2, 1998, pp. 1658–1661.