Proceeding of the 2004 American Control Conference
Boston, Massachusetts June 30 - July 2, 2004

Adaptive Critic Learning Techniques for Automotive Engine Control

Hossein Javaherian
Electrical & Controls Integration Lab.
General Motors R&D and Planning
Warren, MI 48090, USA
Tel: (586) 986-8734, Fax (586) 986-3003
hossein.javaherian@gm.com

Derong Liu, Yi Zhang, Olesia Kovalenko
Department of Electrical and Computer Engineering
University of Illinois at Chicago
Chicago, IL 60607, USA
Tel: (312) 355-4475, Fax (312) 996-6465
dliu@ece.uic.edu

***Abstract*— A new approach for automotive engine torque and air-fuel ratio control is presented in this paper. A class of adaptive critic designs that can be classified as (model-free) action-dependent heuristic dynamic programming is used in the present project. Adaptive critic designs are defined as designs that approximate dynamic programming in the general case, i.e., approximate optimal control over time in noisy, nonlinear environment. The present work uses a system, called "critic," to approximate the cost function in dynamic programming and thus to achieve optimal control. The goals of the present learning control design for automotive engines are emissions reduction and maintenance of optimal performance under various operating conditions. Using the data obtained from a test vehicle, we first develop a neural network model of the engine. A neural network controller is then designed based on the idea of approximate dynamic programming to achieve optimal control. In the simulation studies, the initial controller is trained using the neural network engine model developed rather than the actual engine. We have developed and tested self-learning neural network controllers for both engine torque and exhaust air-fuel ratio control. For both control problems, good transient performance of the neural network controller has been observed. A distinct feature of the present technique is the controller's real-time adaptation capability based on real vehicle data which allows the neural network controller to be further refined and improved in real-time vehicle operation through continuous learning.**

I. Introduction and Background

Dynamic programming is a theory developed back in the 1950's [3] for optimal control of nonlinear systems with the objective of minimizing a performance index that is defined as a summation of a utility function from the present time into the future. In general, using dynamic programming, such an optimal control design for nonlinear systems is only theoretically possible. Moreover, in practice, it has been known for years that due to the so-called "curse of dimensionality" [13], dynamic programming can only be applied to simple, small-scale control problems.

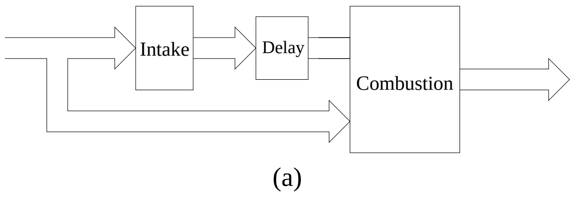
Automotive engines are known to be complex nonlinear dynamical systems. The control problem of automotive engines has been investigated for many years by researchers (see, e.g., [1], [6], [7], [15], [18], [20], [24], [28], [36]).

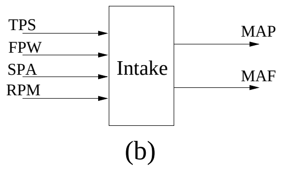
The present work will use neural networks as a tool for adaptive learning in order to obtain an optimal engine control algorithm. We emphasize that our neural network engine controller will be obtained using a specially designed learning process that performs approximate dynamic programming. Once a controller is obtained, it will be applied to perform the task of engine control. The performance of the controller will be further refined and improved through continuous learning in real-time vehicle operation. We note that continuous learning and adaptation to improve controller's performance is one of the key promising attributes of the present approach. Due to vehicle-to-vehicle variations, vehicle aging, environmental changes, etc., each individual engine may require a slightly different initial controller calibration to achieve its design objectives. For practical reasons, during the initial stage of the controller neural network learning it is preferable to use off-line engine data for initial simulation studies. We will therefore first develop a model of the engine for the purpose of initial neural network controller learning but such a model is not necessary for the real-time engine controller operation.

In the real-time implementation of the control algorithms the neural network model that is treated as plant will be replaced by the actual process. Due to the high level of accuracy achieved between the neural network model and the data collected from a real vehicle over a wide range of operating conditions, we expect the designed controller will require minor adjustments through further vehicle on-board learning to work effectively in the real-time operation of the vehicle.

II. Neural Network Modeling of the Test Engine

A test vehicle with a 5.3L V8 engine and 4-speed automatic transmission is instrumented with engine and transmission torque sensors, wide-range air-fuel ratio sensors in the exhaust pipe located before and after the catalyst on each bank, as well as exhaust gas pressure and temperature sensors. Data is collected at each engine





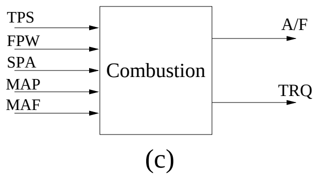


Fig. 1. The model structure of the test engine

event under various driving conditions, such as Federal Test Procedure (FTP cycles), as well as more aggressive driving patterns, for a length of about 95,000 samples during each test. The engine is run under closed-loop fuel control using switching-type oxygen sensors. In this study, engine control variables are throttle position (TPS), electrical fuel pulsewidth (FPW), and spark advance (SPA) while the output variables are engine torque (TRQ) and air-fuel ratio (A/F).

We consider a model of the test engine consisting of two submodules as shown in Figure 1. The model structure chosen here is compatible with the mathematical engine model developed by Dobner [11], [12] and others.

Due to the complexity of modern automotive engines, in the present work, we use the time-lagged recurrent neural networks (TLRNs) (cf. [25], [33], [35]).

A. The Intake Manifold Module

For the intake manifold module, we choose two output variables as intake manifold pressure (MAP) and mass air flowrate (MAF), we choose a reference input as engine speed (RPM), and we choose three control inputs as throttle position (TPS), electrical fuel pulsewidth (FPW), and spark advance (SPA). The time-lagged recurrent neural network used for the intake manifold module has 4 input neurons, 7 hidden layer neurons, and 2 output neurons, with recurrent connections.

Figures 2 and 3 show the comparison between the outputs of the neural network model for the intake manifold and the data collected. In these figures and all displays to follow, all values are normalized to a range between -1 and 1 for convenience in the neural network training. Figures 2 and 3 demonstrate excellent match between the model outputs and the validation data for the MAP and

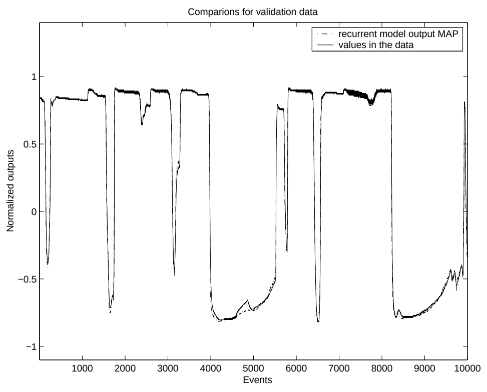


Fig. 2. Illustration of the intake manifold pressure (MAP) validation data

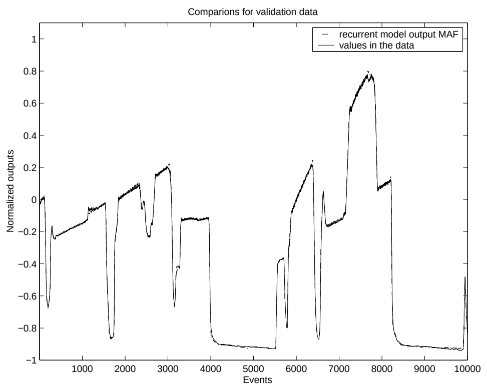


Fig. 3. Illustration of the intake manifold mass air flowrate (MAF) validation data

MAF, respectively. Therefore, we feel confident that the time-lagged recurrent model structure is quite suitable for the prediction of intake manifold output variables.

B. The Engine Combustion Module

For the engine combustion module, we choose two outputs as air-fuel ratio (A/F) and engine torque (TRQ). The exhaust air-fuel ratio, an engine output variable, is measured using wide-range A/F sensors and is usually controlled at the stoichiometric value. We choose two reference inputs as MAP and MAF generated using the intake manifold model. We choose three control inputs as TPS, FPW, and SPA. These are input signals to be generated using our new adaptive critic learning control algorithm. The time-lagged recurrent neural network used for the engine combustion module has 5 input neurons, a single hidden layer with 8 neurons, and 2 output neurons, with recurrent connections.

Validation results for the outputs A/F and TRQ of the neural network engine combustion module are shown in Figures 4 and 5, respectively. These figures indicate very good match between the real vehicle data and the neural network model outputs during the validation phase.

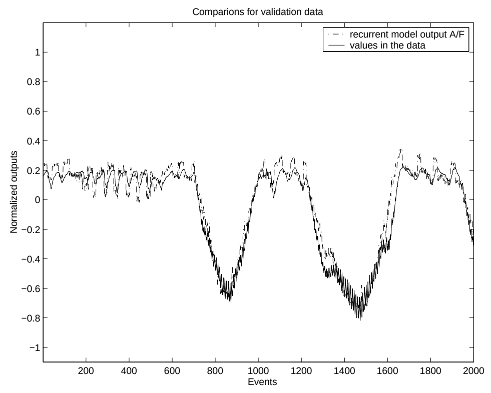


Fig. 4. Illustration of the exhaust A/F validation data

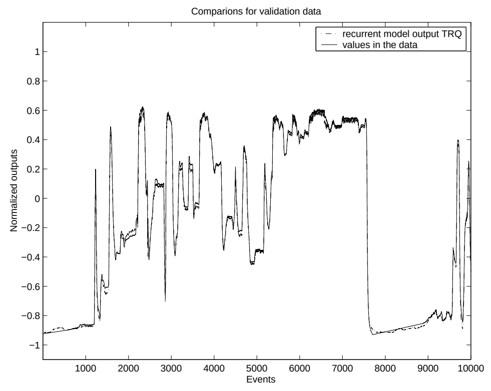


Fig. 5. Illustration of the engine torque TRQ validation data

III. Tracking Control Using Adaptive Critic Designs

In this section we discuss the fundamental approach to the problem of tracking control in the adaptive critic design methodology. We assume that the engine under control is characterized by the following discrete-time nonlinear dynamical system

$$x(t+1) = F[x(t), u(t), t] \tag{1}$$

where $x \in R^n$ represents the state vector of the system states and $u \in R^m$ denotes the control action. Suppose that one associates with this system the performance index (or cost)

$$J[x(i), i] = \sum_{k=i}^{\infty} \gamma^{k-i} U[x(k), u(k), k] \tag{2}$$

where U is called the utility function or local cost function and γ is the discount factor with $0 < \gamma \leq 1$. Note that J is dependent on the initial time i and the initial state $x(i)$, and it is referred to as the cost-to-go of state $x(i)$. The objective is to choose the control sequence $u(k)$, $k = i, i+1, \cdots$, so that the function J (i.e., the cost) in (2) is minimized. Dynamic programming is based on Bellman's principle of optimality [3], [4], [13], [17]: An optimal (control) policy has the property that no matter what previous decisions (i.e., controls) have been, the remaining decisions must constitute an optimal policy with regard to the state resulting from those previous decisions.

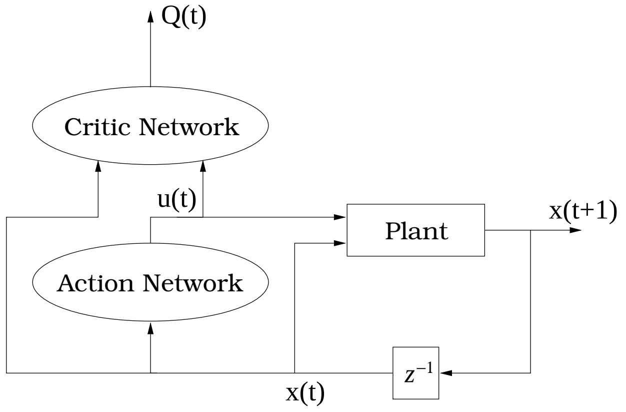


Fig. 6. A typical scheme of an action-dependent heuristic dynamic programming [19], also referred to as ADHDP

According to Bellman's principle of optimality, the optimal cost from time t on is equal to

$$J^*[x(t), t] = \min_{u(t)} \Big(U[x(t), u(t), t] + \gamma J^*[x(t+1), t+1] \Big). \tag{3}$$

The optimal control $u^*(t)$ at time t is the $u(t)$ that achieves this minimum. Equation (3) is the principle of optimality for discrete-time systems. Its importance lies in the fact that it allows one to optimize over only one control vector at a time by working backward in time. In other words, any strategy of action that minimizes the function J in the short term will also minimize the sum of U over all future times.

Adaptive critic designs (ACDs) are defined as designs that approximate dynamic programming in the general case, i.e., approximate optimal control over time in noisy, nonlinear environments [2], [5], [8], [9], [16], [19], [21]–[23], [26]–[34].

A typical ACD consists of three modules that can be implemented by using neural networks. These three modules provide functions of decision, prediction, and evaluation, respectively. In ACDs, when the critic network (i.e., the evaluation module) takes the action/control signal as part of its inputs, the designs are referred to as action-dependent ACDs.

In the present work we use an action-dependent version of ACDs that does not require the explicit use of the model network in the design. Consider the action-dependent heuristic dynamic programming (ADHDP) shown in Figure 6 (cf. [19]). The critic network in this case will be trained by minimizing the following error measure over time,

$$\begin{aligned}\|E_q\| &= \sum_t E_q(t) \\ &= \sum_t [Q(t-1) - U(t) - \gamma Q(t)]^2\end{aligned} \tag{4}$$

where $Q(t) = Q[x(t), u(t), t, W_C]$ and W_C represents the weight vector of the critic network obtained through training. When $E_q(t) = 0$ for all t, (4) implies that

$$\begin{aligned}Q(t-1) &= U(t) + \gamma Q(t) \\ &= U(t) + \gamma[U(t+1) + \gamma Q(t+1)] \\ &= \cdots \\ &= \sum_{k=t}^{\infty} \gamma^{k-t} U(k).\end{aligned} \tag{5}$$

We can see that by minimizing the error function in (4), we have a neural network trained so that its output becomes an estimate of the cost function (2) defined in dynamic programming.

After the critic network's training is finished, the action network's training starts with the objective of minimizing $Q(t)$. The goal of the action network training is to minimize the critic network output $Q(t)$, i.e., we will train the action network so that the output of the critic network becomes as small as possible. In general, a good critic network should not output negative values if $U(t)$ is nonnegative. This is particularly true when $U(t)$ is chosen as the square error function in tracking control problems [29].

After the action network's training cycle is completed, one may check the system's performance, then stop or continue the training procedure by going back to the critic network's training cycle again, if the performance is not acceptable yet.

IV. Adaptive Critic Learning Control of the Test Vehicle

The objectives of the present engine controller design are to provide control signals so that the generated engine torque will follow the commanded torque, and the exhaust air-fuel ratio is regulated at the desired setpoints (e.g., stoichiometric, rich, or lean A/F values). In simulations, our controller is trained using randomly generated target signals for both the torque and the air-fuel ratio quantities. Controllers trained through learning based on randomly generated target signals will in general have a large dynamic tracking range and will be able to track almost any type of signals after they are trained. If desired, further refinement in training can be performed using specific desired training signals as the tracking targets in the controller network training.

The measured values of the engine torque and air-fuel ratio in the test vehicle, an SUV equipped with a 5.3L engine and 4-speed automatic transmission, are obtained using commercial engine controllers under warmed-up conditions (a coolant temperatures of roughly 90 °C). Our learning controller will assume no knowledge about the control signals provided by the existing controller. It will generate a set of control signals that are independent of the control signals in the measured data. Based on the vehicle data collected, we use our learning controller to generate three control signals (i) throttle position (TPS), (ii) electrical fuel pulsewidth (FPW), and (iii) spark advance (SPA), with the goal of producing exactly the same torque as commanded, and achieving the same air-fuel ratio as desired. Part of our simulation experiments will use the values of measured torque and air-fuel ratio as the commanded and desired values. In these simulations, our controller will be built to provide control signals which achieve the required torque and air-fuel ratio control performance. The controller performance is represented by deviations in the generated torque and air-fuel ratio values from the corresponding commanded/desired values. The environment will be represented by MAP and MAF which are predicted using the intake manifold module based on the same control input signals. Other environmental variables which are not used in the present models such as oil and coolant temperature, barometric pressure and temperature, etc. are assumed to remain essentially the same during the test as the warmed-up conditions are established before data collection.

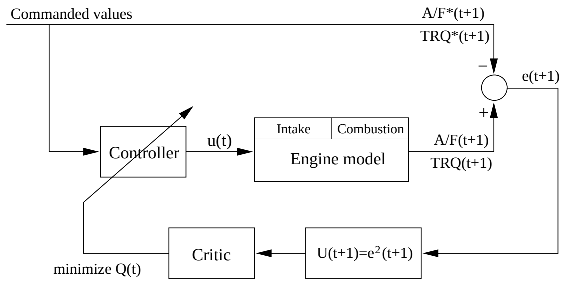


Fig. 7. Structure of adaptive critic learning engine controller

In the rest of this section we provide detailed description of the neural network controller for engine torque control (TRQ) and air-fuel ratio (A/F) control.

A. Torque and Air-Fuel Ratio Tracking Control

Our controller design follows the diagram shown in Figure 7. Initially, the target signals for our tracking control are chosen as randomly generated signals. Generally speaking, a controller trained for tracking randomly generated signals can track almost any well-behaved target signals including periodic, step, and impulse functions.

The local cost function U in this case is defined as

$$U(t) = \frac{1}{2}\,[\mathrm{TRQ}(t) - \mathrm{TRQ}^*(t)]^2 + \frac{1}{2}\,[\mathrm{A/F}(t) - \mathrm{A/F}^*(t)]^2 \tag{6}$$

where TRQ and A/F are the engine torque and air-fuel ratio generated using the proposed controller, respectively, and TRQ* and A/F* are the demanded TRQ value and the desired A/F value, respectively. Both TRQ* and A/F*

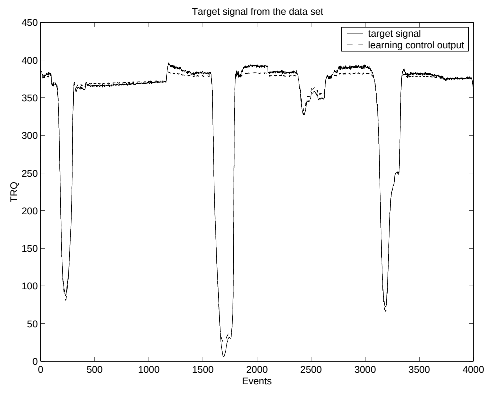


Fig. 8. Torque output generated by the neural network controller

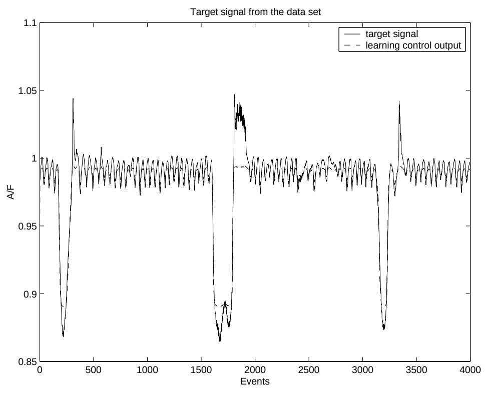


Fig. 9. Air-fuel ratio output generated by the neural network controller

are taken from the actual measured data in the present case. The utility function chosen in this way will lead to a control objective of TRQ following TRQ* and A/F following A/F*.

The structure of the controller/action network is chosen as 4–7–3 with 4 input neurons, 7 hidden layer neurons, and 3 output neurons. The 4 inputs to the action network are TRQ, TRQ*, A/F, and A/F*. Both the hidden layer and the output layer use the sigmoidal function `tansig` (cf. [10]). The outputs of the action network are TPS, FPW, and SPA. All other variables, such as intake manifold pressure (MAP), mass air flowrate (MAF), and engine speed (RPM), remain the same as in the data. The training algorithm we choose to use is `traingdx` (gradient method with momentum and adaptive learning rate, cf. [10]). We employ batch training for the action network.

The critic network is chosen as a 7–13–1 structure with 7 input neurons and 13 hidden layer neurons. The 7 inputs to the critic network are TRQ, TRQ*, A/F, A/F*, TPS, FPW, and SPA. Both the hidden layer and the output layer use the sigmoidal function `tansig`. The output of the action network is Q (the estimated value of the cost function J). The critic network is trained with a controller that has randomly chosen initial weights. The action network is trained after the critic network training. We then start the critic network training again after the action network training. This procedure is repeated until a satisfactory controller (action network) is obtained. We note that the goal of our neural network learning is to obtain an optimal controller. The present optimal controller is obtained by training an action network using the output signal provided by the critic network (to minimize the critic network output).

B. Simulation Results

Figures 8 and 9 show the TRQ and A/F output when TRQ* and A/F*, respectively, are chosen as the measured values in the data set. The neural network controller in this case is trained for 15 cycles. Figure 8 and 9 show that very good tracking control of the commanded torque signal (TRQ) and the exhaust A/F are achieved. We note that at the present stage of the research we have not attempted to regulate the A/F at the stoichiometric value but to track a given command. In these experiments we simply try to track the measured engine-out A/F values so that the control signal obtained can directly be validated against the measured control signals in the vehicle. In Figure 9, it appears that better tracking of A/F was achieved on the rich side of stoichiometric value possibly due to more frequent rich excursions encountered during model training. This could also have been caused by intentional fuel enrichments (i.e., wall-wetting compensation) during vehicle accelerations.

We have observed very clear convergence in the training process starting from very large errors between the observed (TRQ, A/F) and the commanded signal (TRQ*, A/F*), and are gradually reduced to very small values.

Figures shown in this section indicate that the present learning controller design based on approximate dynamic programming (adaptive critic designs) is effective in training a neural network controller to track the desired TRQ and A/F sequences through proper control actions.

V. Concluding Remarks

Our initial research shows that adaptive critic techniques can form the basis for a radically different approach to engine control. In this approach, we develop neural network learning using approximate dynamic programming. The method begins with minimal a priori information about the system and after the network is fully trained the proposed controller may have the potential to outperform existing controllers with respect to the following features: (1) The proposed techniques will automatically learn the inherently complex dynamic and nonlinearities associated with an engine from real vehicle data and therefore do not require a mathematical model of the system. (2) The proposed techniques will automatically adapt to uncertain changes in environmental and vehicle conditions. This is by default a feature of the present learning controller. Most conventional vehicle controllers are either non-adaptive or they possess only a limited range of adaptation capabilities. (3) The proposed controller has some degree of self-learning capability. It can learn to improve its performance in real-time as it gains more experience during the actual vehicle operations. As such, these techniques may offer promises for use as engine calibration tools.

ACKNOWLEDGMENTS

This research was conducted under a contract from General Motors Corporation. We would like to thank Dr. Man-Feng Chang for technical discussions and support, and Alan W. Brown and Michael P. Nolan (all of General Motors R&D and Planning) who helped us in the experimental vehicle set up and algorithm implementation and evaluation.

REFERENCES

[1] C. Alippi, C. de Russis, and V. Piuri, "A neural-network based control solution to air-fuel ratio control for automotive fuel-injection systems," *IEEE Transactions on Systems, Man, and Cybernetics–Part C: Applications and Reviews*, vol. 33, pp. 259–268, May 2003.

[2] S. N. Balakrishnan and V. Biega, "Adaptive-critic-based neural networks for aircraft optimal control," *Journal of Guidance, Control, and Dynamics*, vol. 19, pp. 893–898, July-Aug. 1996.

[3] R. E. Bellman, *Dynamic Programming*, Princeton, NJ: Princeton University Press, 1957.

[4] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, Belmont, MA: Athena Scientifi c, 1995.

[5] D. P. Bertsekas and J. N. Tsitsiklis, *Neuro-Dynamic Programming*, Belmont, MA: Athena Scientifi c, 1996.

[6] F. E. Coats, Jr. and R. D. Fruechte, "Dynamic engine models for control development - Part II: Application to idle speed control," *International Journal of Vehicle Design*, Special Publication SP4, pp. 75–88, 1983.

[7] J. A. Cook and B. K. Powell, "Modeling of an internal combustion engine for control analysis," *IEEE Control Systems Magazine*, vol. 8, pp. 20–26, Aug. 1988.

[8] C. Cox, S. Stepniewski, C. Jorgensen, R. Saeks, and C. Lewis, "On the design of a neural network autolander," *International Journal of Robust and Nonlinear Control*, vol. 9, pp. 1071–1096, Dec. 1999.

[9] J. Dalton and S. N. Balakrishnan, "A neighboring optimal adaptive critic for missile guidance," *Mathematical and Computer Modeling*, vol. 23, pp. 175–188, Jan. 1996.

[10] H. Demuth and M. Beale, *Neural Network Toolbox User's Guide*, Natick, MA: MathWorks, 1998 (Version 3).

[11] D. J. Dobner, "A mathematical engine model for development of dynamic engine control," *SAE Paper No. 800054*, 1980.

[12] D. J. Dobner, "Dynamic engine models for control development - Part I: Non-linear and linear model formation," *International Journal of Vehicle Design*, Special Publication SP4, pp. 54–74, 1983.

[13] S. E. Dreyfus and A. M. Law, *The Art and Theory of Dynamic Programming*, New York, NY: Academic Press, 1977.

[14] Y. Fukuoka, H. Matsuki, H. Minamitani, and A. Ishida, "A modifi ed back-propagation method to avoid false local minima," *Neural Networks*, vol. 11, no. 6, pp. 1059–1072, 1998.

[15] N. V. Kulkarni and K. KrishnaKumar, "Intelligent engine control using an adaptive critic," *IEEE Transactions on Control Systems Technology*, vol. 11, pp. 164–173, Mar. 2003.

[16] G. G. Lendaris and C. Paintz, "Training strategies for critic and action neural networks in dual heuristic programming method," *Proceedings of the 1997 IEEE International Conference on Neural Networks*, Houston, TX, June 1997, pp. 712–717.

[17] F. L. Lewis and V. L. Syrmos, *Optimal Control*, New York, NY: John Wiley, 1995.

[18] X. Li and S. Yurkovich, "Sliding mode control of delayed systems with application to engine idle speed control," *IEEE Transactions on Control Systems Technology*, vol. 9, pp. 802–810, Nov. 2001.

[19] D. Liu, X. Xiong, and Y. Zhang, "Action-dependent adaptive critic designs," *Proceedings of the INNS-IEEE International Joint Conference on Neural Networks*, Washington, DC, July 2001, pp. 990–995.

[20] J. J. Moskwa and J. K. Hedrick, "Nonlinear algorithms for automotive engine control," *IEEE Control Systems Magazine*, vol. 10, pp. 88–93, Apr. 1990.

[21] D. V. Prokhorov, *Adaptive Critic Designs and Their Applications*, Ph.D. Dissertation, Texas Tech University, Lubbock, TX, Oct. 1997.

[22] D. V. Prokhorov, R. A. Santiago, and D. C. Wunsch, "Adaptive critic designs: A case study for neurocontrol," *Neural Networks*, vol. 8, pp. 1367–1372, 1995.

[23] D. V. Prokhorov and D. C. Wunsch, "Adaptive critic designs," *IEEE Transactions on Neural Networks*, vol. 8, pp. 997–1007, Sept. 1997.

[24] P. F. Puleston, S. Spurgeon, and G. Monsees, "Automotive engine speed control: A robust nonlinear control framework," *IEE Proceedings–Control Theory and Applications*, vol. 148, pp. 81–87, Jan. 2001.

[25] G. V. Puskorius, L. A. Feldkamp, and L. I. Davis, "Dynamic neural network methods applied to on-vehicle idle speed control," *Proceedings of the IEEE*vol. 84, pp. 1407–1420, Oct. 1996.

[26] J. Si and Y.-T. Wang, "On-line learning control by association and reinforcement," *IEEE Transactions on Neural Networks*, vol. 12, pp. 264–276, Mar. 2001.

[27] R. S. Sutton, "Learning to predict by the methods of temporal differences," *Machine Learning*, vol. 3, pp. 9–44, 1988.

[28] G. J. Vachtsevanos, S. S. Farinwata, and D. K. Pirovolou, "Fuzzy logic control of an automotive engine," *IEEE Control Systems Magazine*, vol. 13, pp. 62–68, June 1993.

[29] N. A. Visnevski and D. V. Prokhorov, "Control of a nonlinear multivariable system with adaptive critic desings," *Proceedings of the Artificial Neural Networks in Engineering Conference*, St. Louis, MO, Nov. 1996, pp. 559–565.

[30] P. J. Werbos, "Advanced forecasting methods for global crisis warning and models of intelligence," *General Systems Yearbook*, vol. 22, pp. 25–38, 1977.

[31] P. J. Werbos, "Building and understanding adaptive systems: A statistical/numerical approach to factory automation and brain research," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-17, pp. 7–20, Jan./Feb. 1987.

[32] P. J. Werbos, "A menu of designs for reinforcement learning over time," in *Neural Networks for Control* (Chapter 3), Edited by W. T. Miller, R. S. Sutton, and P. J. Werbos, Cambridge, MA: The MIT Press, 1990.

[33] P. J. Werbos, T. McAvoy, and T. Su, "Neural networks, system identifi cation, and control in the chemical process industries," in *Handbook of Intelligent Control: Neural, Fuzzy, and Adaptive Approaches* (Chapter 10), Edited by D. A. White and D. A. Sofge, New York, NY: Van Nostrand Reinhold, 1992.

[34] P. J. Werbos, "Approximate dynamic programming for real-time control and neural modeling," in *Handbook of Intelligent Control: Neural, Fuzzy, and Adaptive Approaches* (Chapter 13), Edited by D. A. White and D. A. Sofge, New York, NY: Van Nostrand Reinhold, 1992.

[35] P. J. Werbos and X. Pang, "Generalized maze navigation: SRN critics solve what feedforward or Hebbian nets cannot," *Proceedings of the SPIE*, vol. 2878, pp. 2–9, 1996.

[36] M. Won, S. B. Choi, and J. K. Hedrick, "Air-to-fuel ratio control of spark ignition engines using Gaussian network sliding control," *IEEE Transactions on Control Systems Technology*, vol. 6, pp. 678-687, Sept. 1998.