1994 IEEE International Symposium on
Intelligent Control
16-18 August, 1994.
Columbus, Ohio, USA

Event-Based Intelligent Control of Saturated Chemical Plant Using Endomorphic Neural Network Model

Sung Hoon Jung, Tag Gon Kim, and Kyu Ho Park

Computer Engineering Research Laboratory
Department of Electrical Engineering
Korea Advanced Institutes of Science and Technology
373-1 Kusong-dong Yusong-gu, Taejon 305-701,
Korea.

Abstract

An event-based control system with an endomorphic neural network model is designed and realized to control a saturated nonlinear plant. This scheme is based on an event-based control paradigm. However, this scheme has some aspects different from the original scheme due to the saturation property of a plant. This new scheme may be viewed as a method combining a time-based diagnosis mechanism in event-based control with a state-based control mechanism in neural network control. A chemical plant having strong nonlinearity and complicated dynamics is controlled using the realized event-based control system. This paper discusses the structure of an event-based controller, the neural network modelling methodology, related problems, and experimental results.

1. Introduction

Recently, Zeigler [1], [2] introduced an intelligent control paradigm which provides a basis to construct a high autonomy. This paradigm, called *event-based intelligent control*, is based on the simulation theory for discrete event systems [3], [4]. In event-based control, the controller internally has an event-based model of a controlled plant, called an *endomorphic model* [2]. Using the endomorphic, the event-based controller can simulate the state changes of a plant and can diagnose the operation of the plant using time windows. Thus, the first step in event-based control is to develop an endomorphic event-based model of a controlled plant. Luh and Zeigler discuss this process in [5].

An endomorphic model is often managed by a tabular form. We call such a table a *table model*. A major limitation of a table model is that a table may not represent the complete plant dynamics in all operation points. To solve this problem, we take a neural network mapping model of a plant in our event-based control system. The neural network model approximately maps the parameters of a table model to automatically generate the parameters.

However, the mapped dynamics of a neural network model may not be exactly the same as those of the original continuous plant due to incomplete learning. When a plant has a saturation property, this incomplete learning results in a serious problem because the plant is very sensitive to the changes of input values.

To cope with this situation, every block between a current state and a target one is partitioned into several intermediate blocks and control inputs are repeatedly applied to the plant until a set point is reached. A state window is also employed to provides a tolerance of a set point due to incomplete learning. These two windows, namely a time window and a state window, make a cross area to check the state and time constraints. So, this scheme may be viewed as a combination of a time-based diagnosis mechanism in event-based control and a state-based control mechanism in neural network control [6].

We experiment a continuously stirred tank reactor (CSTR) plant using our realized event-based control system with a neural network mapping model. The CSTR plant having strong nonlinearity and complicated dynamics is a chemical process to produce chemical products. Experimental result shows relatively good control performance in spite of the strong nonlinearity and complicated dynamics.

This paper is organized as follows. Section describes outlines of the event-based control paradigm. The simulation environment are presented in section . The CSTR plant and neural network learning strategy are explained in section . Simulation results are given in section . Conclusions and further works are discussed in section .

2. Event-Based Control Paradigm

In event-based control, the endomorphic model of a plant is specified by an event-based-control DEVS [5]. The event-based-control DEVS is an extended version of the discrete event system specification (DEVS) formalism [3], [4], [2]. This section briefly discusses the event-based-control DEVS, an event-based control

0-7803-1990-7/94/$4.00 ©1994 IEEE

logic, and an event-based modelling methodology.

2.1 Brief Review of Event-Based-Control DEVS

An event-based-control DEVS offers a sound semantics for specification of discrete event systems. An event-based-control DEVS [5] is defined as a 7-tuples:

$$M = < X, S, Y, \delta_{int}, \delta_{ext}, \lambda, ta > \quad (1)$$

- X is the external input events set;
- $S = B \times X$, where B is a finite set of elements, each called a boundary;
- Y is a finite output events set;
- δ_{int} is the internal transition function;
- δ_{ext} is the external transition function;
- $\lambda : S \rightarrow Y$ is the output function;
- $ta : S \rightarrow R^{+}_{0,\infty} \times R^{+}_{0,\infty}$, i.e., $ta(s) = [r, r']$, where $r, r' \in R^{+}_{0,\infty}$ and $r \leq r'$;

subject to the constraints

1) $\delta_{int}(b, x) = (b', x)$;

2) $\delta_{ext}((b, x), 0, x') = (b, x')$;

In event-based control, an event-based model specified by the event-based-control DEVS is employed as an endomorphic model of a plant. Therefore, an input set, a state set, and an output set for the endomorphic model are control inputs, control points, and threshold outputs of the plant, respectively. The internal and external transition functions for the endomorphic model provide the behavioral characteristics of the plant. The time advanced function is slightly differ from the original DEVS formalism [4] due to the need of a time interval. These minimum and maximum times, called a *time window*, provide time tolerance as if a conventional control has state tolerance. That is, the arrival of a target state within the time interval is regarded as correct.

Using those three sets and four functions, an event-based controller simulates the operation of a plant, and the results of simulation enables the event-based controller to diagnose the plant. This event-based paradigm has several advantages over the traditional ones [1], [2]. The information produced by checking within the time window could be used for diagnostic purpose. Also the use of a time window provides the controller with robustness against expected values from sensors.

2.2 Event-Based Control Logic

This section briefly describes event-based control logic. We first describe the architecture of an event-based controller and the elements, and then their functions and relationships. Figure 1 shows the architecture of an event-based controller. An event-based controller is composed of two parts, i.e., a control part and a diagnosis one. The control part is composed of

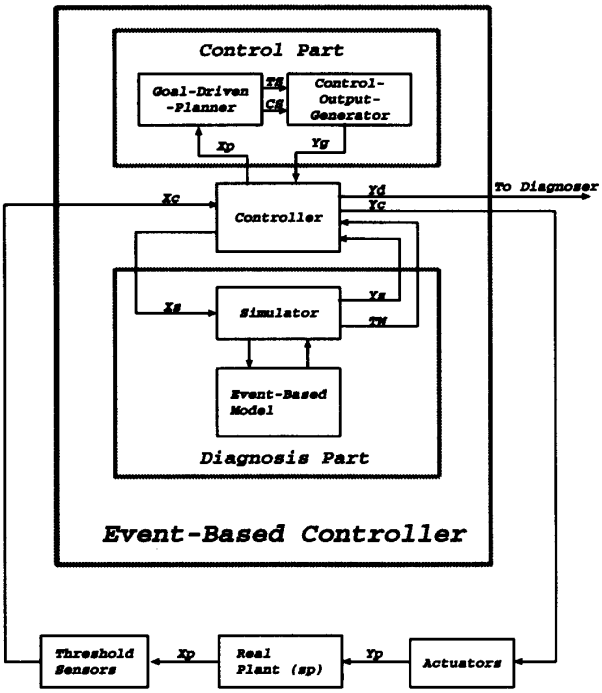

Fig. 1. Event-Based Control Environment

a *Goal-Driven-Planner* (GDP) and a *Control-Output-Generator* (COG); the diagnosis part a *simulator* and an *event-based model*. These two parts are managed by the central controller which provides a main control algorithm.

The control logic is as follows. The event-based controller receives an external input X_c from threshold sensors and sends control signals Y_c to a plant if no error is detected. Otherwise, the event-based controller sends diagnostic informations Y_d to a diagnoser. Internally, the central controller sends a *next-plan* signal to the GDP to get the next desired state and then receives the control output from the COG. Then the controller sends the received control output to the plant. At the same time, it sends the output to the simulator for simulation of the behavior of the plant. The simulator simulates the plant using an endomorphic event-based model and generates two outputs, i.e., expected outputs Y_s of the plant and time windows TW. With the outputs and time windows as reference, the controller diagnoses the state of the plant. If the expected state of a plant is sensed within the time window, then the event-based controller regards the current state of a plant as correct. Otherwise, an error is assumed and the controller invokes diagnoser functions to find where it occurs. The event-based controller repeatedly performs the described control logic as reported by sensor

readings.

2.3 Event-Based Modelling

An event-based controller should know an endomorphic model of a controlled plant. An event-based (or discrete event) model of a continuous plant reactions at discrete event points under observation. The endomorphic model is isomorphic to the continuous plant in the input-output. Luh and Zeigler [5] introduce two transformation steps, shown in Figure 2, to develop an event-based endomorphic model of a continuous plant. The first step is to transform a continuous plant into

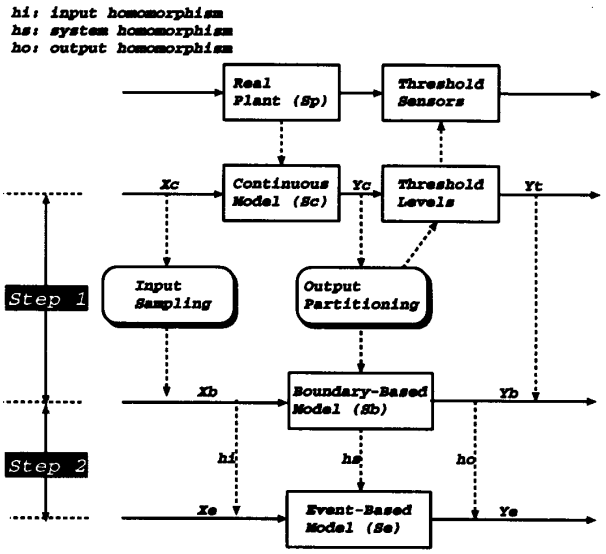

Fig. 2. Overall Sequence of Event-Based Modelling

a boundary-based DEVS model which composed of an *input sampling* and an *output partitioning*. The output partitioning partitions the outputs of the continuous model into several mutually exclusive blocks to quantize the output levels. The partitioning levels of continuous outputs are derived from the operational objectives of the plant under control. In event-based control, controlled point should be in such partitioned outputs. Therefore, we can view a control job as a tracking sequence of those output levels. Thus, control inputs should be selected such that a control input forces the plant to produce a desired output value. We call this work an *input sampling*. If n output states are existed, total nP_2 inputs could be selected. However, not all of these are required to be known for a control job. As a result of the first step, inputs, outputs, and states set necessary to construct a boundary-based model are determined.

The second step is an homomorphic mapping process which transforms a boundary based model into an event-based model. Luh and Zeigler [5] described the mapping by introducing two formalisms, i.e., boundary-based DEVS and event-based-control DEVS. See [5] for details about the homomorphic abstraction.

3. Neural Network Modelling Paradigm

An event-based model of a continuous plant described in the previous section is often managed by a tabular form. We call such a table a *table model*. However, the following problems may arise in event-based control. In event-based control, a controller asks the table model some information on the plant dynamics. A major limitation of a table model is that a table may not represent the complete plant dynamics in all operation points. Even though a table model can update its table during control, it cannot have capability to generate information which is not in table.

To provide our event-based control system with the capability, we take an artificial neural network as an endomorphic model of a plant [7], [8]. This section

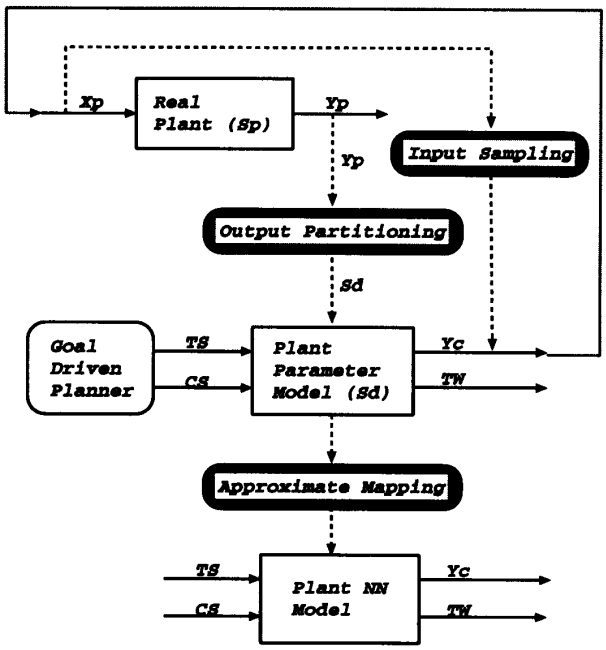

Fig. 3. Mapping Continuous Plant into NN Model

describes the neural network modelling methodology, the architecture of the event-based control system with the neural network model, and their related problems.

3.1 Neural Network Modelling

The versatile mapping capabilities of an artificial neural network (NN) provide a means for control of

nonlinear plants which cannot be well controlled by conventional controllers [7], [8]. The use of NN also makes it possible to model unknown nonlinear dynamic systems. To incorporate these capabilities, we take a NN as an endomorphic model of the controlled plant in our event-based control system. The NN model generates time windows and control inputs of the endomorphic model necessary for event-based control of the plant. In the previous event-based control scheme shown in Figure 1, those parameters are obtained by the COG and the event-based plant model. Figure 3 shows how the event-based plant model (plant parameter model) maps into a NN. The input sampling and output partitioning are already discussed in section . Note that the values of parameters for the plant parameter model are approximated mapped into a NN model. To cope with all set points in the operating range, the NN model should learn the dynamics of the plant. Data not learned are also generated by the NN model in approximate form. The structure of an event-based controller with a NN model is shown in Figure 4.

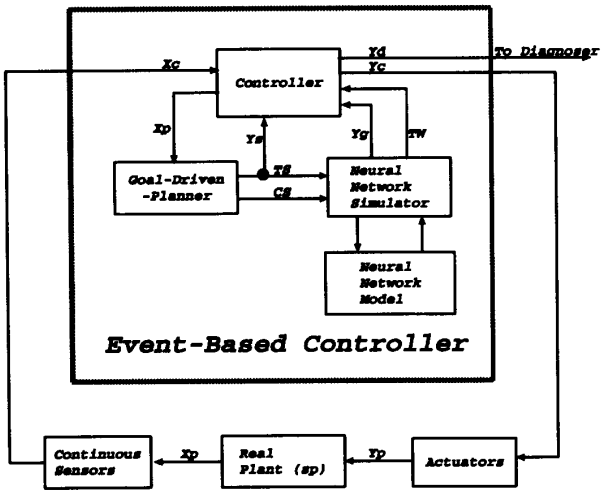

Fig. 4. Event-Based Control Environment with a NN Model

3.2 Model-Plant Mismatch Caused by Incomplete Learning

A back propagation algorithm, which is a typical supervised learning algorithm, is used for learning our NN model. The dynamics of a NN model may not be exactly the same as those of the original continuous system due to incomplete learning. This mismatching may cause the NN model to generate inaccurate values of parameters. However, in case of monotonically increasing/decreasing systems, the function of time window is to offer tolerance against inaccurate time of state changes in the plant and uncertainty in the environmental changes such as an initial state and plant parameter variations and etc. [5].

However, this incomplete learning results in a serious problem for a saturated plant. Because the output of a nonlinear saturated plant is very sensitive to the changes of input values. Therefore, using inaccurate control inputs generated by such a NN model may cause outputs of the plant to be unacceptably different from desired set points. To cope with this situation, we partition every block between a current state and a target one into several intermediate blocks. After that, we apply control inputs repeatedly to the plant until set points are reached. In this scheme, time windows are used to evaluate the saturated state of the plant, not used to diagnose the state. That is, even though the state of the plant is not reached to a set point within a time window, the controller does not generate the *LATE* error signal. Instead it applies new inputs to the NN model with a new current state and the target state. This operation is repeatedly applied to the plant until state of the plant reaches to the target state. Even though this partitioning method is applied, the plant output may not be bounded to a small variation of a set point due to incomplete learning. That is, the NN model may always generate the same control output Y_c for a specific current state and target state. This error can be seen as a steady state error which can be diminished by continuous learning of the dynamics of a plant during control.

For this situation, we adopt a state window, a interval of values of a state, which provides a state tolerance. These two windows, i.e., a time window and a state window make a cross area to check the state and time constraints with tolerance. Therefore, the event-based controller issues an error message only when the plant state is not within cross area even though n control operations are applied. The number n is set by a user to consider the control environment and the plant. This scheme may be viewed as a method combining a time-based diagnosis mechanism of event-based control with a state-based control mechanism of neural network control [6].

4. Simulation Environment

This section describes the controlled plant and the neural network learning strategy.

4.1 Continuously Stirred Tank Reactor (CSTR)

A CSTR is a chemical process which produces chemical products. This CSTR model is a part of a larger test system introduced by Williams and Otto [9], [10]. The CSTR system, shown in Figure 5, supports the

following multiple reactions:

$$\begin{aligned} A+B &\xrightarrow{k1} C \\ C+B &\xrightarrow{k2} P+E \\ P+C &\xrightarrow{k3} G \end{aligned} \qquad (2)$$

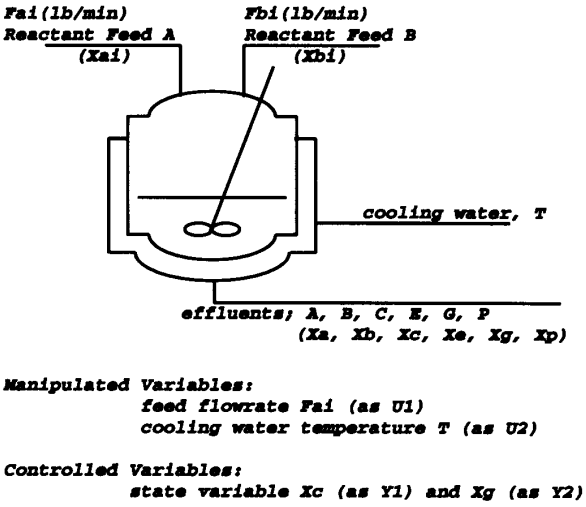

Fig. 5. CSTR Plant

The desired product is P, while G, C, and E are byproducts subject to quality and environmental constraints. Reactants A and B enter as pure components in separate streams with flow rates Fai and Fbi, respectively.

The manipulated variables are the flow rate Fai and cooling water temperature T. The input stream Fbi is considered as a disturbance variable. The equations describing the kinetic behavior of the above reactions and the dynamic mass balance for the CSTR are a coupled set of nonlinear algebraic and ordinary differential equations. A description of these equations has been provided by Williams and Otto [9]:

$$\begin{aligned} dXa/dt &= Fai/Fr - rx1 - Xa \\ dXb/dt &= Fbi/Fr - rx1 - rx2 - Xb \\ dXc/dt &= 2rx1 - 2rx2 - rx3 - Xc \\ dXe/dt &= 2rx2 - Xe \\ dXg/dt &= 1.5rx3 - Xg \\ dXp/dt &= rx2 - 0.5rx3 - Xp \\ rx1 &= 5.9755E9exp(-12000/T)XaXb\rho V/(60Fr) \\ rx2 &= 2.5962E12exp(-15000/T)XbXc\rho V/(60Fr) \\ rx3 &= 9.6283E15exp(-20000/T)XcXp\rho V/(60Fr) \end{aligned}$$

$$\begin{aligned} Fr &= Fai + Fbi \\ \rho &= 50lb/ft^3, V = 60ft^3, 580°R < T < 680°R \\ Xi &= mass\ fraction \end{aligned} \qquad (3)$$

The reaction constants, initial conditions, and constant parameters are as follows:

Xa = 0.075	Xe = 0.208	Fai = 170 lb/min
Xb = 0.57	Xg = 0.0398	Fbi = 679 lb/min
Xc = 0.015	Xp = 0.019	T = 645°R

The control objective of this system is to maximize the yield of the desired product P using regulation of two related state variables Xc and Xg. In this example, our neural network has 4-10-10-6 morphology as shown in figure 6.

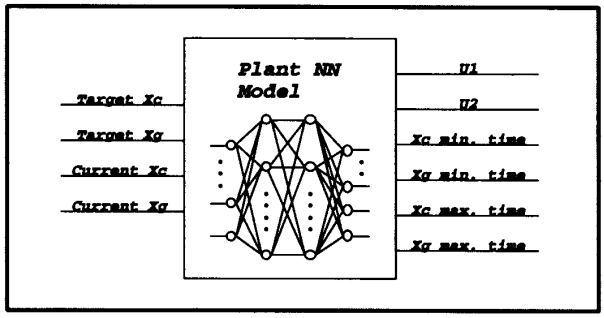

Fig. 6. Neural Network Structure for CSTR

4.2 Neural Network Learning Strategy

As shown in equation 1 of event-based DEVS definition, the states of a continuous system are represented by the cross product of boundaries and inputs. A number of initial values of control variables could also be employed as a initial states. If a system is represented by n boundaries, m inputs, and k initial states, then $n \times m \times k$ number of state space is necessary to completely represent the system at discrete levels. For completeness, the states in the state space should be mapped to the neural network. For simplicity, we take only the input variables as independent variables except for the initial states. We use a state space mapping algorithm which is outlined as follows. First, inputs within operating ranges are randomly selected by a random number generator and stored. Second, current plant outputs are stored as initial states. Third, the selected inputs are applied to the plant which is simulated with discrete time-based simulation. Finally, the time on which the output of the plant is saturated is measured and stored. Us-

ing those information, the state space of the plant is mapped into a neural network.

5. Simulation and Results

We realized an event-based control system, HICON (High-level Intelligent CONtroller), with a neural network model of a plant on an expert system ART-IM/windows [11]. In this experiment, the CSTR plant was simulated using the equation 3 with a discrete time based simulation method on every 50 msec. time interval. Before control, the neural network plant model is learned the parameters of the table model.

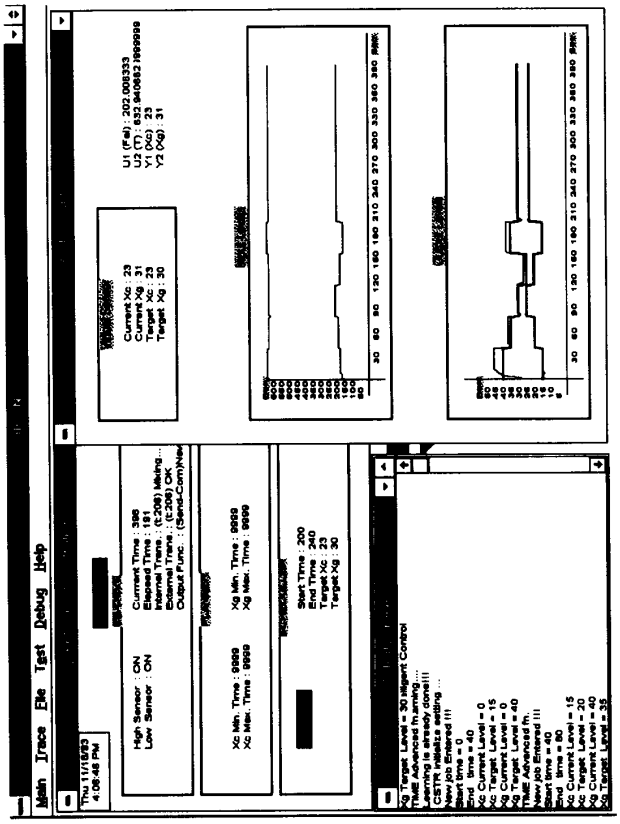

Fig. 7. Result of CSTR Plant Control

Using the neural network model, a neural network manager generates some parameters such as time windows and control outputs for a controller. A new data which is obtained by the control operation can also be learned to dynamically adapt to the control environment. This dynamic learning under control provides the flexible modelling capability to the event-based control system. Figure 7 shows the overall controlled situation. The input/output states of the controlled plant is respectively depicted in the middle/bottom boxes of "CSTR Plant" window. In the output state, the desired set points are represented by bold lines and the controlled output by thin lines. As previously mentioned, the steady state error of the result is caused by incomplete learning. This steady state error is generated only when the controlled plant has a saturation property. That is, when the plant output is saturated with respect to an input, model-plant mismatch causes the steady state error.

6. Conclusions

This paper discussed an event-based control paradigm with a neural network model for a saturated plant. We showed the usefulness of neural network modelling through an experiment of a chemical plant with the saturation property. The scheme may be viewed as a method combining a time-based diagnosis with a state-based control. Experimental result showed this scheme could be used to control a complex dynamic nonlinear plant and could be associated with the higher knowledge-based job management modules to construct more autonomous control system in further work [12], [13].

References

[1] B. P. Zeigler., "DEVS Representation of Dynamical Systems: Event-Based Intelligent Control," *Proceedings of the IEEE*, vol. 77, pp. 72–80, Jan. 1989.

[2] B. P. Zeigler., *Object Oriented Simulation with Hierarchical, Modular Models: Intelligent Agents and Endomorphic Systems.* Academic Press, 1990.

[3] B. P. Zeigler., *Theory of Modelling and Simulation.* John Wiley, 1976.

[4] B. P. Zeigler., *Multi-Faceted Modelling and Discrete Event Simulation.* Academic Press, 1984.

[5] C.-J. Luh and B. P. Zeigler., "Abstracting Event-Based Control Models for High Autonomy Systems," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 23, JANUARY/FEBRUARY 1993.

[6] A. Benveniste and P. L. Guernic, "Hybrid Dynamical Systems Theory and the SIGNAL Language," *IEEE Trans. on Automatic Control*, vol. 35, pp. 535–546, May 1990.

[7] K. S. Narendra and K. Parthasarathy, "Identification and Control of Dynamical Systems Using Neural Networks," *IEEE Trans. on Neural Networks*, vol. 1, Mar. 1990.

[8] Y. Ichikawa and T. Sawa, "Neural Network Application for Direct Feedback Controllers," *IEEE Trans. on Neural Networks*, vol. 3, Mar. 1992.

[9] W. T.I. and R. Otto, "A Generalized Chemical Processing Model for the Investigation of Computer Control," *Trans. Am. Inst. Elect. Engr.*, vol. 79, no. 458, 1960.

[10] C. McFarlane and D. Bacon, "Adaptive Optimizing Control of Multivariable Constrained Chemical Processes. 1. Theoretical Development, 2. Application Studies," *Ind. Eng. Chem. Res.*, vol. 28, no. 1828, 1989.

[11] "ART-IM/Windows Programming Language Reference." Inference Corporation, 1991.

[12] T. G. Kim and B. P. Zeigler., "AIDECS: An AI-Based, Distributed Environmental Control System for Self-Sustaining Habits," *Artificial Intelligence in Engineering*, vol. 5, no. 1, 1990.

[13] T. G. Kim, "Hierarchical Scheduling in an Intelligent Environmental Control System," *Journal of Intelligent and Robotic Systems*, vol. 3, pp. 183–193, 1990.