HICON: An Autonomous Control System

Sung Hoon Jung, Kwang-Hyun Cho, Tag Gon Kim, and Kyu Ho Park

Abstract

Development of an autonomous control system is one of the most interesting issues in control theory. To achieve autonomous control facilities, some high-level control modules and intelligent control methodologies should be incorporated into one system. This paper proposes a framework for autonomous control with three layers. Each layer has its own functions and its own modules which cooperate with one another. An abstracted control task in a topmost layer is practically realized with more detailed informations by control modules in lower layers. This framework is based on an event-based supervising control, a rule-based fuzzy logic control, and neural-network-based adaptive modelling. In this paper we deal with the architecture of proposed autonomous control system, functions of each module, a supervising and learning method, an adaptive modelling method. Based on these concepts, we introduce a realized autonomous control system, named HICON. Finally, we explain experimental results with a temperature plant.

I. Introduction

Nowadays, some researchers have studied intelligent control systems with a high degree of autonomy[1, 2, 3, 4].

A highly autonomous control system must have the ability to function as an independent unit or element over an extended period of time under significant uncertainties in the plant and the environment[1, 4]. This autonomy needs some capabilities such as self-diagnosing, self-learning, self-reasoning, and so on [5].

To equip with these capabilities, some intelligent control methodologies must be integrated into one unified control system. This paper proposes a framework for constructing an autonomous control system based on an event-based intelligent control, a rule-based fuzzy logic control, and neural network modelling. This framework has a hierarchical structure and its functional modularity. We employed three layers for information management---a task management layer, a supervising control layer, a continuous control layer.

Each layer deals with its own informations: the task management layer manages control tasks, the supervising control layer treats discrete events for a control task, the continuous control layer controls the real plant for an event with continuous data. A scheduled task is concretely planned in a task management layer; the planed detail objectives are managed by a supervising control layer; and each objective is practically realized by a continuous control layer.

An event-based control in this framework provides a main control algorithm. This paradigm, introduced by Zeigler [6, 7], is based on the simulation theory for discrete event systems [8, 9]. This intelligent control paradigm internally has knowledges of a plant with discrete levels. Using these knowledges, the controller can diagnose the plant operation by comparing the knowledges with the results of plant operation. These knowledges are formally represented as an *endomorphic plant model.*

However, a static plant model can not offer an adaptation of an environment change caused by a model-plant mismatch, parameter variations, and so on. We employed an artificial neural network model as an *endomorphic plant model.* This modelling offers two advantages: 1) adaptation of environment change, 2) on-line modelling. This ability, on-line modelling of an endomorphic plant model, provides a control system with high autonomy.

A nonlinear laboratory-scale temperature plant is employed to observe the control operations of our system. As we designed, our control system operates well with information hierarchy and high autonomy such as on-line modelling and diagnosis. This framework provides a basis to construct a highly autonomous control system and shall be extended to a more autonomous control system through incorporation of automatic rule generation capability for a fuzzy logic control and autonomous task generation and scheduling from control objectives given by users.

Manuscript receive August 19, 1995; revised November 17, 1995.
The authors are with the Department of Electrical Engineering Korea Advanced Institute of Science and Technology.

This paper is organized as follows. Section II briefly reviews the concept of event-based control and fuzzy logic control. Proposed event-based fuzzy control paradigm is presented in section III. The construction of an experimental framework is described in section IV. Experimental results with a laboratory-scaled temperature plant and discussion are given in section V. Section VI concludes this paper with future researches.

II. Brief Review of Basic Methodologies

Our realized system consists of some modules which take their own functions in the system. Before detailed description of the realized system, this section describes basic methodologies---an event-based intelligent control, neural network modelling, and a fuzzy logic control.

1. Event-Based Intelligent Control

Event-based intelligent control, introduced by [6, 7, 8], is a discrete event control paradigm based on the simulation theory for discrete event systems [8, 9]. Thus, all informations between the controller and a plant are finite event forms. Figure 1 shows the overall control modules of event-based control.

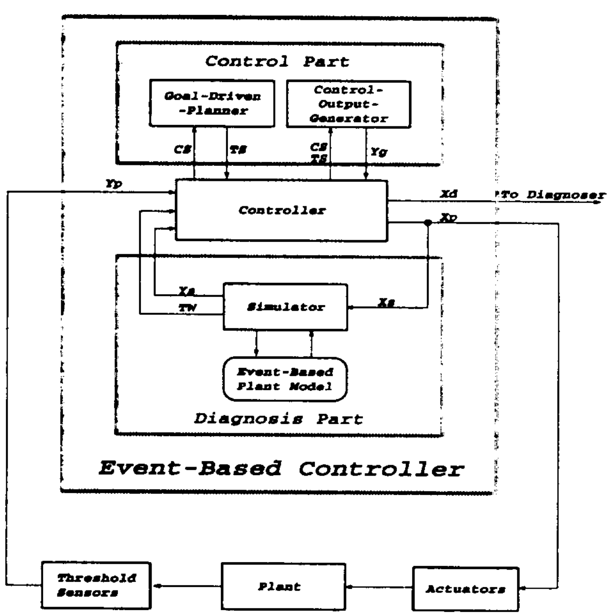


Fig. 1. Control Structure of Event-Based Control.

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

Overall control sequences are as follows. The event-based controller receives an external input Y_p from threshold sensors, and sends control signals X_p to a plant if no error is detected. If errors are detected, the event-based controller sends diagnostic informations X_d to a diagnoser and stops until the errors are recovered. The central controller internally sends a current state (CS) to the GDP to get the next target state (TS). The determined CS and TS are sent to the COG to receive control commands Y_g from the COG. The controller then sends the received control commands to the plant to control. At the same time, the controller sends these commands to the simulator to simulate the plant behavior. The simulator simulates the plant operation using an endomorphic event-based plant model and generates two outputs, i.e., expected outputs Y_s of the plant and time windows TW. This states that the output of the controlled plant will become an expected output Y_s within the time window TW. Thus using the two informations, the controller diagnoses the plant operation. If the expected state of a plant is sensed within the time window, then the event-based controller regards the plant operation as being correct. Otherwise, an error is assumed to be occurred and the controller invokes diagnoser functions to find where it had occurred. The event-based controller repeatedly performs the described control logic in accordance with the sensor readings.

Figure 2 shows all diagnostic situations which can be occurred during an event-based controller tries to move the state of a plant from a CP to a TP. The event-based controller issues a too-early error if an expected sensor signal is arrived prior to the minimum time of a time window. Similarly, the controller issues a too-late error if the expected sensor signal is arrived after the maximum time of the time window. Although a sensor signal is arrived within the time window, the controller issues an unexpected-state error if the signal is different from the expected sensor one.

The event-based controller regards the operation of the plant as correct only if the expected sensor signal is arrived within the time window. These three informations help a diagnoser find where errors occur [3].

In the control operation, the controller must know dynamics of the controlled plant to simulate the controlled plant. The dynamics can be managed by several methods such as differential equations, difference equations, modelling formalisms, and so on [6]. However, a discrete event modelling formalism is the most adequate method for the event-based controller because the controller needs discrete dynamics of the controlled plant. Thus, the authors in [6, 2] employed a discrete event system specification (DEVS) formalism to model the controlled plant.

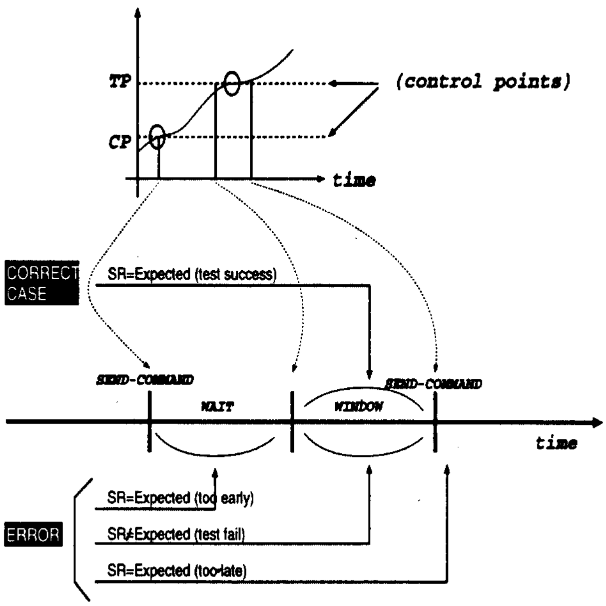


Fig. 2. Diagnosis of event-based control paradigm (SR: Sensor Reading).

The DEVS provides a mathematically sound semantics to specify operations of discrete event systems in a hierarchical, modular manner. The DEVS formalism is specified by a 7-tuple as:

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

where

- X is the input events set
- S is the sequential state set
- Y is the output events set
- $\delta_{int} : S \rightarrow S$ is the internal transition function
- $\delta_{ext} : Q \times X \rightarrow S$is the external transition function where $Q = \{(s, e) \mid s \in S, 0 \leq e \leq ta(s)\}$
- $\lambda : S \rightarrow Y$ is the output function
- $ta : S \rightarrow R^{+}_{0,\infty}$ is the time advanced function

Dynamics of a discrete event system is fully modeled with the formalism at finite discrete event levels. The inputs X, states S and outputs Y are static components to represent the dynamics of the system. On the other hand, the other four elements are dynamic functions to represent the behavioral characteristics of the system. The external transition function takes charge of the state transition of a system caused by external inputs. That is, when the function receives an external input, the function makes the state transition with the current states S and the elapsed time e.

Similarly, the internal transition function handles the state transition of a system after the schedule time, specified by the time advance function, is completely elapsed. The output function generates outputs in accordance with the current state.

The state transition time in DEVS is specified by a time point because the state transition of a system will occur on a time point. As already discussed, the state transition time in event-based control is determined by a time duration, *time windows*. Luh and Zeigler [2] devised a modified DEVS to fit in event-based control. The modified DEVS, named *event-based-control DEVS*, is defined as follows [2].

$$M = \langle X, S, Y, \delta_{int}, \delta_{ext}, \lambda, ta \rangle \quad (2)$$

where X, Y, δ_{int}, δ_{ext}, λ are as before, and

- S = B × X, where *B* is a finite set of elements, each called a boundary;
- *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) $\delta_{int} = (b', x)$;
2) $\delta_{ext}((b', x), 0, x') = (b, x')$;

The first constraint indicates that the boundary crossing is represented by the internal transition function. The second constraint states that the discontinuous change of outputs is impossible even though external inputs are changed. The main difference is that time advanced function generates a time window instead of a time point as already mentioned.

Using this event-based-control DEVS, an event-based plant model, often called an *endomorphic plant model*, is constructed and used to diagnose. The main advantage of the event-based control is that the error information can be effectively used to find where the errors occur and to correct them[3]. Moreover, an event-based controller does not require high precision of the output sensor in contrast to conventional sampled data systems[3, 6]. This event-based controller can also be used as a supervising controller for lower continuous controllers. In supervising control, main operations of the event-based controller are to schedule control objectives, to plan each objective, and to diagnose the plant operation.

2. *Endomorphic Neural Network Plant Model*

Development of an event-based plant model, however, would be extremely unwieldy and may lead to intractable computations in practice[2]. Moreover, the modelling has difficulty to express the change of the plant dynamics caused by environmental variation and time varying parameters of the plant. These problems can be overcome by taking artificial neural network modelling. An endomorphic neural network model can well adapt to the change of the plant dynamics by learning new dynamics[10]. The neural network model maps the parameters of an event-based plant model. Thus, the model can automatically generate parameters of an event-based plant model, and can learn new parameters when the parameters are changed in an on-line manner. Thus, this modelling method provides the event-based controller with self-learning capability. More detailed description of the neural network modelling is in[10].

3. Fuzzy Logic Control

Ever since Mamdani proposed it in 1974[11], fuzzy control has been one of the most successful application techniques in the fuzzy theory. Nowadays fuzzy control based on fuzzy logic resolves so many problems that are seemed to be impossible by conventional control methodologies. In particular, fuzzy control can be very useful when plants are too complex for analysis or when the available sources of information are inexact and uncertain[12].

A fuzzy logic control (FLC) can be regarded as a means of emulating a skilled human operation in the human-in-loop system [13] because many control strategies of skilled humans can be mapped into the FLC through the rules [12, 13]. The FLC is basically composed of several functional modules---fuzzification, defuzzification, knowledge base, and decision-making logic---as shown in Figure 3.

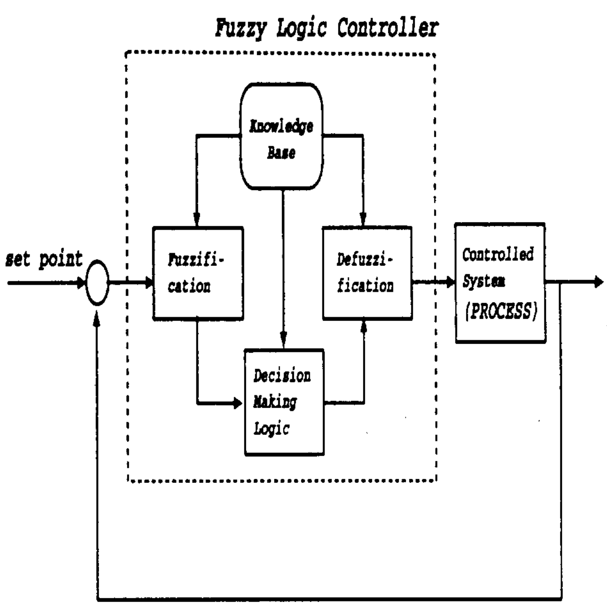


Fig. 3. A Systematic Overview of a Fuzzy Logic Control.

All informations treated in decision making logic are fuzzy numbers. Thus, external error informations must be converted to fuzzy numbers by the fuzzification module. On the other hand, the result outputs of decision making logic must be converted to crisp numbers through the defuzzification module.

Control dynamics in decision making logic is stored in knowledge base as an *If (antecedent)-Then (consequent)* rule form. Finally, the outputs of the decision-making logic controls the plant through the defuzzification module.. For input processing, singletons are often used directly without employing complicated fuzzification in many practical cases. Recently, researches of automatic rule generation for a FLC using neural networks have been intensively studied to provide more autonomous control [14, 15, 16].

III. Event-Based Fuzzy Control System

This section describes motivation, the architecture of an event-based fuzzy control system, and its control operation.

1. Motivation

To make a control system perform complex and laborious tasks currently done by people, some intelligent control methodologies must be integrated into an intelligent control system in a systematic manner with hierarchy and modularity. This integration of intelligent control methodologies helps a control system to be equipped with an autonomous control capability. An autonomous control system must perform well under significant uncertainties in the plant and the environment for extended periods of time. They must also be able to compensate for system failures without external intervention [4]. Methods to extract only the necessary information from the data are becoming essential in the quest for higher autonomy. The methods are related to the problem of extracting more abstract models with levels and to the more abstract control knowledges with levels. Thus, the autonomous control system must employ *a hierarchical control structure* in which a higher-level intelligent controller supervises a lower-level controller [17]. Normally, plants are too complex to describe them with conventional mathematical system models such as differential or difference equations. The control system must be able to deal effectively with significant uncertainties in models of increasingly complex dynamical systems.

Resolution directly determines the complexity of computations. In complex systems and situations one level of resolution is not sufficient because the total space of interest is usually large, and the final accuracy is usually high enough. This consecutive focusing of attention results in a multilevel task decomposition. Intelligence is oriented toward complexity reduction. Intelligence allows for an increase in functionality with a reduction of computational complexity. Thus instead of solving in one shot the whole problem with the maximum volume of the state space and with the amount of high resolution details one may choose to solve several substantially simpler problems nested one within another. The control intelligence is hierarchically distributed according to the Principle of Precision with Decreasing Intelligence (IPDI), evident in all hierarchical management systems. Event-based intelligent controllers are good candidates for implementing more robust co-ordination schemes.

Figure 4 shows the three level control structure scheme of our autonomous controller.

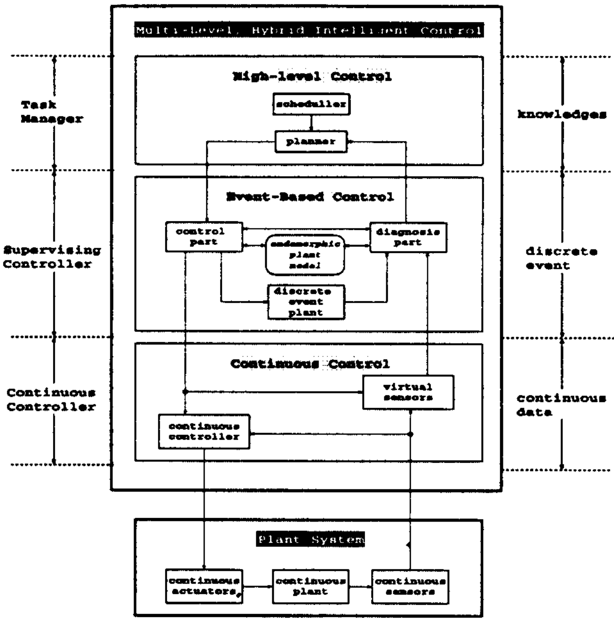


Fig. 4. Control Structure of Our Control System.

High-level control manages control objectives as an unit of task. The control tasks are scheduled by a scheduler with a task name and schedule time.. A planner decides the detailed state transitions for a scheduled task, and a planned target state is sent to the event-based control. An event-based controller obtains a real set point from the planned target state, and sends the set point to a continuous controller. The continuous controller tries to move the plant's state to the set point. The *virtual sensor* reads the output value of a continuous sensor and generates a threshold signal which indicates reaching the set point if the reading value equals to the set point. Using the threshold signal, the diagnosis part of the event-based control decides whether the plant operation is correct or not with the endomorphic plant model. If errors occur, then the module sends error informations to a diagnoser. Otherwise, the module sends a success signal to the planner. When the planner receives the success signal, it plans a new target state. A task can be generally represented as a circular state transition diagram. Thus the above sequences are repeated in corresponding to the current schedule task.

2. System Architecture

Based on the discussion of control structure in previous section, we designed our control system architecture as shown in Figure 5.

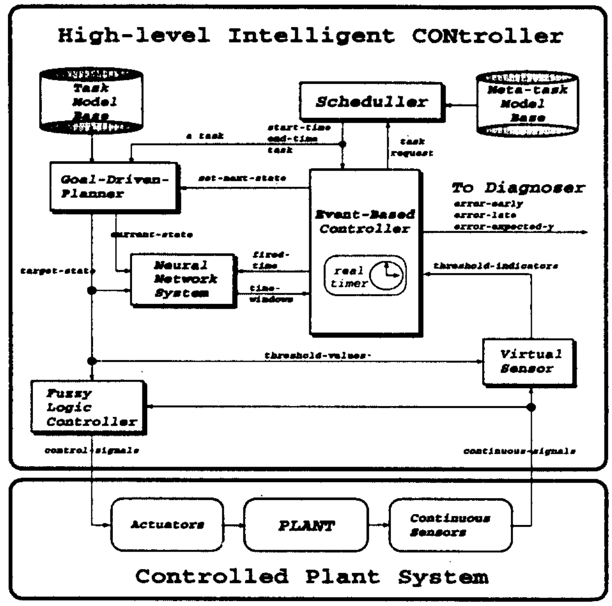


Fig. 5. verall Structure of HICON.

We used a fuzzy logic controller as a continuous controller because this control method is very powerful; and a neural network model as an endomorphic plant model because this modelling method offers adaptability and on-line learning.

A High-level Intelligent CONtroller (HICON) is composed of five modules--- a task management module, an event-based control module, an fuzzy logic control module, a neural network module, and a virtual sensor module. The task management module is composed of two active element--- *scheduler, goal-driven planner (GDP)*--- and two passive data bases---a meta-task model base and a task model base. Control objectives are classified as tasks and stored to task models; and the tasks are managed by a scheduler as a meta-task model. In this paper, we employed a FIFO queue as a simple scheduler with three items, *start-time, end-time,* and task name. Thus, tasks in a meta-task model are ordered with start and end time.. The time duration of a task should not be overlapped with any other tasks.

The control scenario of HICON is as follows. The event-based controller manages overall control sequences and keeps a real timer to manage diagnostic time informations. When the control starts, the event-based controller requests a task to a scheduler. Then the scheduler sends a task in a meta-task model to the GDP and the event-based controller. The GDP sets its current task to the scheduled task as soon as it receives the scheduled task. When the *start-time* is arrived, then the event-based controller sends a *set-next-state* signal to the GDP and starts the measure of *fired-time*. That is, the time for the continuous plant to transit to a target state is measured to make the neural network model learn the plant dynamics. Then the GDP sets its next state to a target state for the scheduled task and sends a real set point for the target state to the fuzzy logic controller. At that time, the value of target state are sent to the neural network system to learn the dynamics of the plant. The fuzzy logic controller controls the continuous plant using its rules which are given before the control starts. The plant output is sensed by a continuous sensors and the results are sent to the virtual sensor. The virtual sensor decides whether the target-state is reached or not by comparing the set point (a threshold-value) and the sensor output, and sends a *threshold-indicator* signal to the event-based controller. The function of the virtual sensor is to translate a continuous data to an event data to connect between a continuous layer and an event-based layer.

When the event-based controller receives the *threshold-indicator* signal, the controller informs the *fired-time* to the neural network system, and sends a *set-next-state* signal to the GDP to plan a next target state. The above control sequences for a task are repeatedly done until the *end-time* is arrived. When the *end-time* is arrived, the event-based controller requests a new task to the scheduler and repeats the above control sequences.

The neural network model in a neural network system will learn the dynamics of the controlled plant more and more as the control proceeds. This is because our control scheme employed on-line learning for constructing the endomorphic neural network model. This on-line learning helps the control system to be equipped with the dynamic and adaptable self-modelling capability. The event-based controller can diagnose only after the neural network model is sufficiently learned. Thus, the neural network system has two operational modes, i.e., a *learning-mode* and an *application-mode*. The neural network system changes the mode by consideration two factors: one is whether the current state transition from CS to TS has ever learned before, another is whether the mapping rate of the neural network model is sufficient. That is, the neural network system changes the mode to an application-mode only when the planned current and target state are already learned and the mapping rate of the model is sufficient. Otherwise, the neural network system remains the learning-mode. The mapping rate is determined by a Total Sum Square Error (TSSE) of the neural network.

If the neural network model is learned as an off-line manner, then the mode change is not necessary because the neural network has all dynamics of the plant prior to real control. In the case, the mode will be always an *application-mode*. The training data should be sufficiently large to cover all operating range of the plant. In method1, random set points are first applied to the plant and the result data are gathered. Then, the gathered data is mapped to the neural network model. Of course, the random set points are within operating range with uniform distribution. In method2, the data are gathered and mapped to the neural network model as an on-line manner. However, two problems are occurred in method 2: one is that the current planed data can not be exactly equal to the learned data because the values of the data are real numbers, another is that the initial gathered data has only one fired time, thus it cannot provides a time window. For the first problem, we employed a state window that offers the state tolerance in comparison of two data. Also we employed an initial width of a time window for the second problem. If the initial width is W, then a time window for a gathered fired time t becomes a time duration between *t* - *W/2* and *t* + *W/2*. This value W, however, must be diminished as the data is collected more and more. If the W remains a constant value, then the diagnose capability will be poor.

EBC : event-based control (diagnosis)
DG : data gathering
NNL : neural network learning
FLC : fuzzy logic control

CP : control set points
RP : random set points

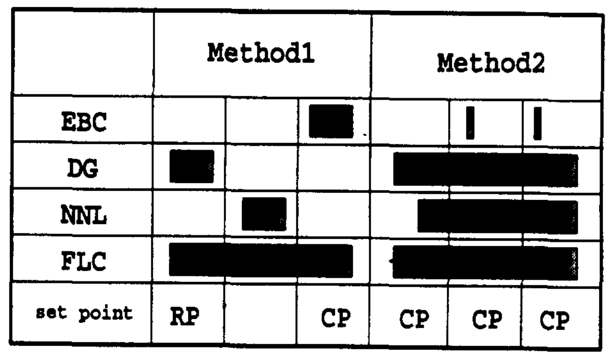


Fig. 6. two control scenarios.

The method 1 has one advantage that all state transitions for control set points can be diagnosed; and one disadvantage that the preprocessing for learning the neural network has much time. The method 2 also has one advantage that the preprocessing time is not necessary; and one disadvantage that only some state transitions can be diagnosed.

IV. Experimental Framework Construction

This section describes an overall experimental framework for temperature control and realization of a fuzzy logic controller.

1. Control environment

We employed a nonlinear laboratory-scale temperature plant to observe the control operation of our system. Dynamics of the temperature plant is hard to be modeled or may be impossible on an environment with large disturbance. This is because dynamics of the plant depends on nonlinear factors such as the temperature of environment air, humidity, ventilation, and so on. Moreover, the parameters of the model may be dynamically changed. Thus, this plant can be a good control example even simple. Figure 7 shows the overall experimental environment of temperature control.

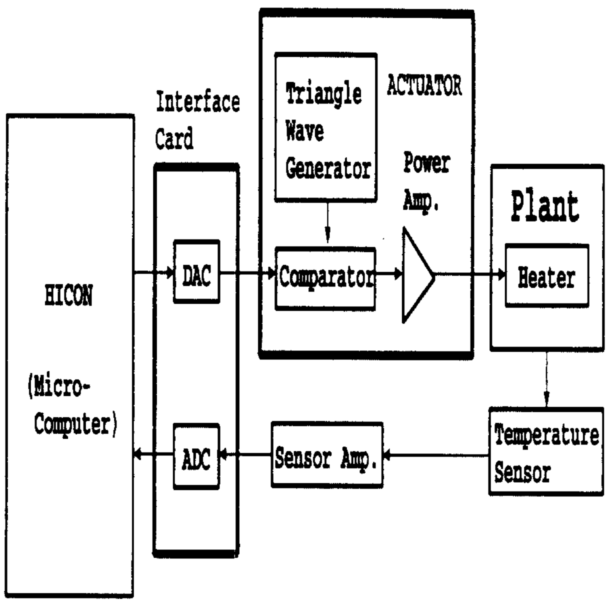


Fig. 7. Temperature Control Setup.

Our control system, HICON, is operated on a micro-computer and interfaced with external environment via an interface card, PCL-712. A control command (for example *turn-on, turn-off*) is translated to a value and this value is sent to DAC. The control value is amplified via actuator and applied to a heater to adjust room temperature. The room temperature is sensed by a temperature sensor. Then the result voltage is applied to an ADC via an amplifier because this sensed signal is very week. A digital value which represents the voltage is converted to a temperature value through a voltage-temperature lookup table. A virtual sensor uses this temperature value to make a threshold signal.

The HICON is implemented on an expert system shell *ART-IM/windows*. The ART-IM/windows is a complete toolkit for the development of rule-based, or knowledge-based system. This toolkit also supports the LISP-like ART-IM language and several development tools and graphic interface tools[18]. Also this can be easily expanded using dynamic interface facility.

2. *Fzzy Logic Controller*

The control structure of a FLC realized in our system is shown in Figure 8.

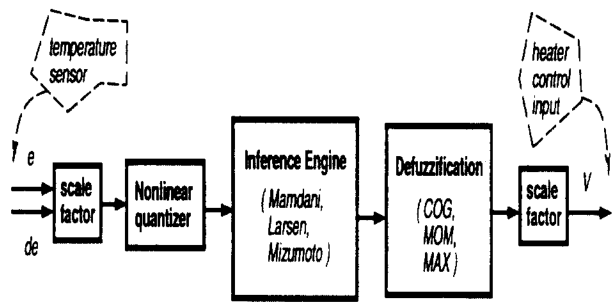


Fig. 8. FLC block diagram.

Two inputs---*e* and *de*---are directly applied to the inference engine through a scale factor and a nonlinear quantizer because we used singletons without complicated fuzzification as used in many practical applications. The scale factor makes all input variables have same universe of discourse one another. Scaled inputs are quantized such that a digital computer discretely manages continuous membership functions. In the quantization, we used a nonlinear quantization method because the FLC focuses on the vicinity of a set point for fine tuning. The inference engine calculates an output value using given fuzzy rules and input sigletones. We also used nonuniform membership functions in inferencing for fine tuning near a set point.

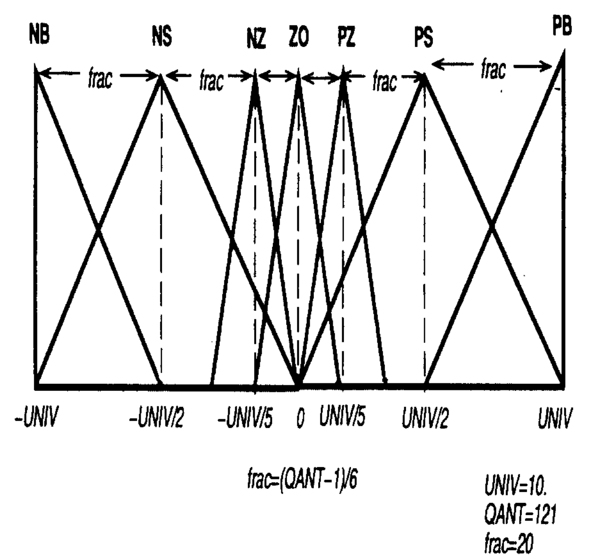


Fig. 9. Nonuniform support of membership and nonlinear quantization.

That is, ZO-label is divided into three parts —NZ, ZO, PZ.

Three direct reasoning methods---*Mamdani, Larsen,* *Mizumoto*---and three defuzzification methods---*center-of-area (COG)*, *middle-of-maxima (MOM)*, *first-of-maxima (MAX)* are implemented for flexible use.

Figure 10 shows fuzzy rules for the temperature plant which are obtained by a human expert.

e \ ce	nb	ns	nz	zo	pz	ps	pb
nb	nb	nb	nb	nb	nb	nb	ns
ns		nb	nb	ns	ns	nz	
nz		nb	ns	nz	pz	zo	
zo		ns	nz	zo	pz		
pz		pz	zo	pz	pz	pz	
ps		ps	pb	ps	ps	pb	
pb	ps	ps	pb	pb	pb	pb	pb

Fig. 10. Used Temperature Plant Rules.

For a more autonomous ability, automatic fuzzy rule generation methodologies have been studying.

V. Experimental Results and Discussion

This section deals with the detailed control operation of our control system, experimental results, and discussion about the results.

1. Control Scenario

We will describe control operations of our system with a simple control task. The aim of this control task is to regulate set points which are scheduled by a scheduler with time. All event time in our control system must be physical real time, not logical virtual time because we actually control a real plant. Thus, the event-based control module must manage a physical real timer. An event must be actually occurred only if physical time is equal to the logical time.

Figure 11 shows the flow chart of control operations of HICON.

When control starts, all modules first initialize their internal states and related parameters. The event-based controller sets its initial states and requests a task to a scheduler. A task which is the first entry of a meta-task model is sent to the GDP and the event-based controller. For example, a task message (10 30 5) implies "Do the task-5 in the task model during 10 to 30 seconds". The GDP finds the task-5 in the task model and sets the task to its current task. The event-based controller received the task message sends a *set-next-state* signal to the GDP if the start time is arrived. When GDP receives a *set-next-state*, then the GDP issues a *target-state* signal to a neural network system, a virtual sensor, and a fuzzy logic controller. If the task-5 is to circularly transit three states---*A*, *B*, and *C*, then the GDP continuously issues a *target-state* with the value of *A*, *B*, and *C* in order until the current task is changed. If an user wants to keep a state A during the scheduled time, then the state transition diagram must have only one state A with circular transition. According to the combination of tasks, control objectives of an user can be implemented.

The FLC uses the *target-state*, which is a numeric value to represent a target state, as its set point. The FLC continuously controls the plant with the set point, and the virtual sensor continuously monitors the plant output to make a threshold event which indicates that the plant output reaches the target state. If the event-based controller receives the threshold event signal from the virtual sensor, then the event-based controller informs the {\em fired-time} to the neural network system if the controller is in learning mode. Otherwise, the controller diagnoses the plant operation using the *fired-time* and time windows obtained from the neural network system.

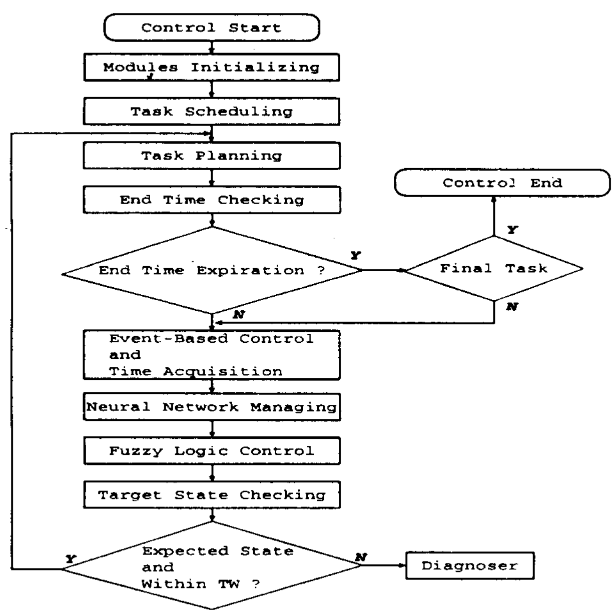


Fig. 11. Flow Chart of Control Operations of HICON.

The neural network system can learn the plant dynamics from this *fired-time* with a current and target state. If the end-time is not fired, then the event-based controller sends a *set-next-state* to the GDP again. Otherwise, the event-based controller requests a new task to a scheduler.

2. Experimental Results

Figure 12 shows an experimental result, illustrating all control modules with four windows, namely an event-based controller window, a fuzzy controller window, a temperature plant window, and an activity trace window. We describes all windows in detail.

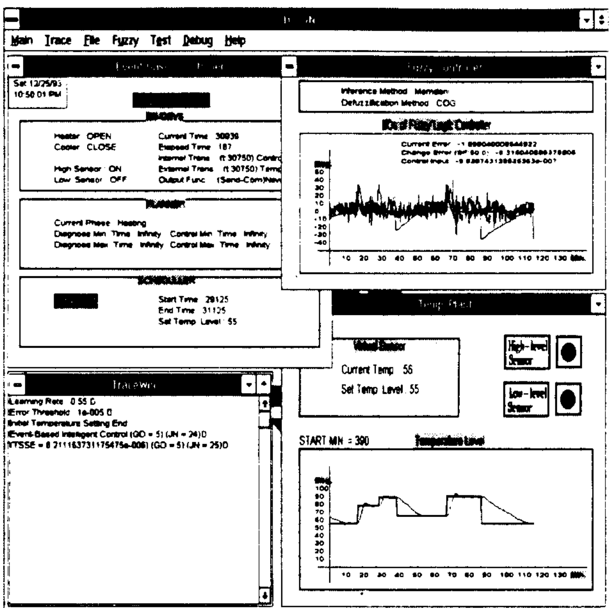


Fig. 12. Overall Control Environment.

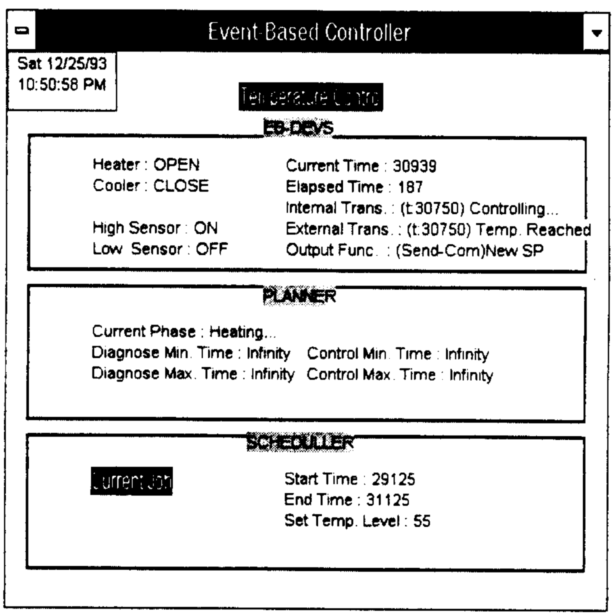


Fig. 13 Event-based Control Window.

Figure 13 shows the event-based control window. This window shows the related informations of high-level modules---a scheduler, a planner, and an event-based controller. The scheduler scheduled a task that the temperature of the plant is to regulate 55 during 29125 to 31125 seconds. The plant temperature was 90 degree on 29125 seconds and reaches 55 degree on 30750 seconds. At that time, an external event (a threshold-indicator signal) is entered to the event-based controller as shown in the EB-DEVS box. The external event immediately invokes an internal event to generate an output. Thus the internal transition function is done at the same time, and generate a new output, *New SP*, to the GDP. However, the target state is not changed because this task is to regulate a set point. The same target state with before cannot be diagnosed because no state transitions occurs. Thus the time windows have infinity as shown in the PLANNER box. The EB-DEVS box also shows the state of heater and cooler and sensors. A high sensor indicates that the plant temperature is higher than the set point and low sensor *vice versa*.

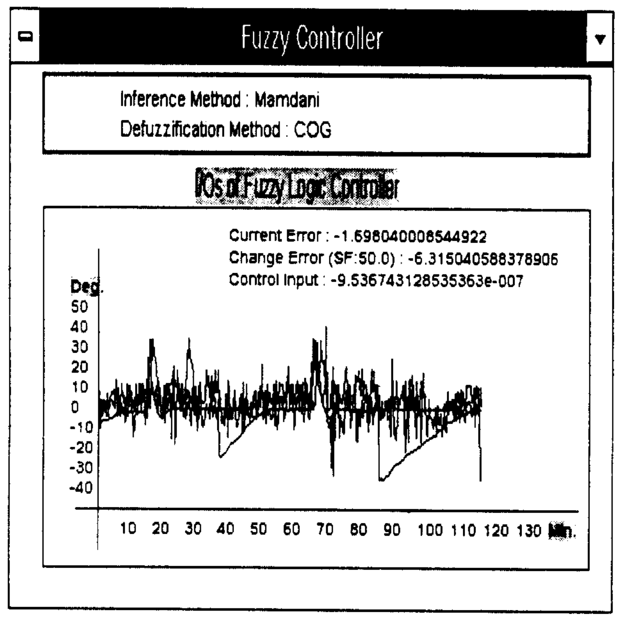


Fig. 14. Fuzzy Logic Control Window.

Figure 14 shows the fuzzy controller window which displays control operations of a fuzzy logic controller. The FLC receives a set point from the GDP, and it calculates a control input using the given fuzzy rules. In the calculation process, we employed a Mamdani inference method and a center-of-gravity defuzzification method because this combination shows best performance in our experiment. The FLC has two inputs and one control output because our control plant is a single input single output (SISO) system. The window also displayed the input and output crisp values.

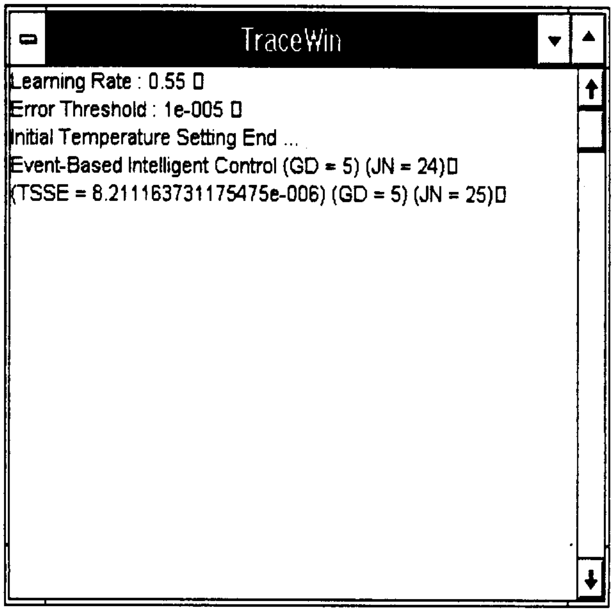


Fig. 15. Activity Trace Window.

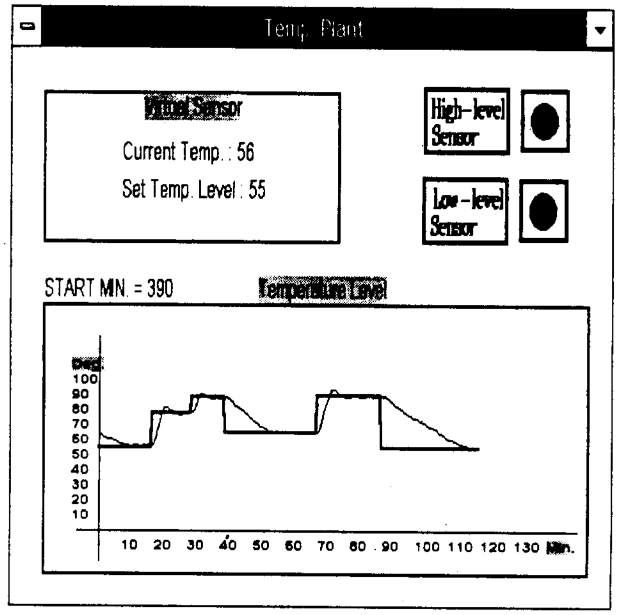


Fig. 16. Display of Status of Temperature Plant.

Figure 15 shows the activities in control operations. Trace levels in the menu-bar adjust the display level of activities. The trace has an event-level, a module level, and an operational level. In the figure, the level is operational level, thus only brief operational activities are displayed. This can be changed to other level by an user request in an on-line manner. In the figure, a neural network learning rate that is used in neural network modelling is shown. An error threshold value is a value to decide whether the neural network is well learned or not. That is, the event-based controller diagnoses the plant operation only if the error threshold is satisfied. In the figure, *GD* and *JN* are the number of gathered data and the number of controlled tasks after control starts. The current status is that only 5 data are obtained in control 25 tasks. A data is obtained at least a task is executed at least twice. This is because that a time window has minimum and maximum time window. Currently, the neural network is sufficiently learned because the TSSE ($8.21e^{-6}$) is less than the error threshold. Also currently scheduled task is already mapped to the neural network. Thus, the event-based controller can diagnose the plant operation. On the 30750 seconds, when an event is entered to the event-based controller, the controller diagnosed.

Figure 16 shows the output of the temperature plant with time. Current set point is 55 and current temperature is 56. Start time of the graph is 390 minutes. Thus, actual time is calculated by adding 390 minutes to the graph time. We does not use a cooler for cooling, so cooling needs much time than heating.

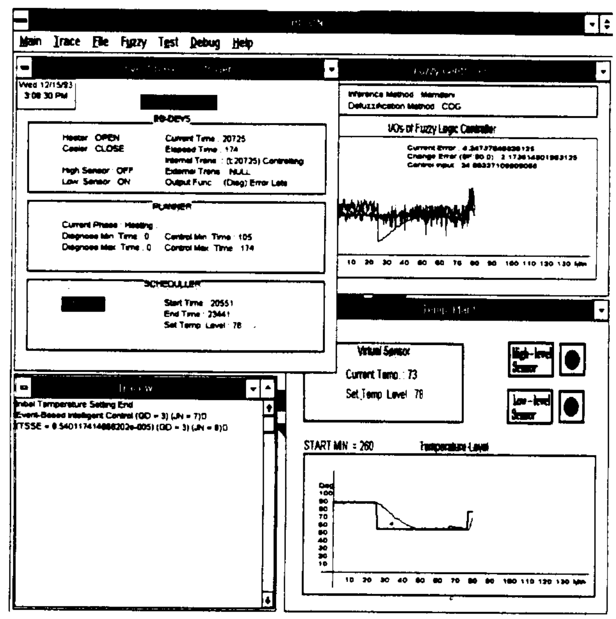


Fig. 17. Diagnosis Status of HICON.

The main advantage of the event-based controller is that this control has an ability to diagnose. If the neural network model is sufficiently learned and the current task is the same as the learned data. Figure 17 shows a diagnostic situation.

A time window for a temperature transition from 55 degree to 78 degree is given as (105, 174) as shown in the PLANNER box. The event-based controller issues a too-late error because the elapsed time exceeds the maximum time of the time window, (105, 174). For the fault generation, we forced the heater not to heat during some time to make the heating time large. This is very similar to human diagnostic ability because human can diagnose the plant operation using his brain mapped knowledges. However, human can not check whether the plant operation is correct or not if he does not know the plant dynamics in his brain neuron. Human who waits for a specific state of a plant will decide that it's too late in consideration past experiences if the state does not reach after a specific time.

VI. Conclusions

This paper proposed a framework for more autonomous control system with some intelligent control facilities such as an event-based control, neural network modelling, and a fuzzy logic control. We employed a hierarchical structure tô systematically manage the informations of plant with a modular manner. This control system has some autonomous capabilities: self-scheduling of tasks, self-planning of a task, self-modelling of a nonlinear plant, and self-diagnosis of the plant operation. A laboratory-scale temperature plant was employed to experiment our system. Experimental results showed that our control framework could offer more autonomous control capabilities than each intelligent systems. This indicates the functions of the unified system is improved by integration of each module. This control framework shall be extended to a more autonomous control system through incorporation of automatic rule generation capability for a fuzzy logic controller and autonomous task generation and scheduling from control objectives given by users [19, 20].

References

[1] Antsaklis P.J., K.M. Passino, and S.J. Wang, "Towards Intelligent Autonomous Control Systems: Architecture and Fundamental Issues:, *Journal of Intelligent and Robotic Systems,* vol. 1, no. 4, pp. 315-342, 1989.

[2] Cneong-Jye Luch and Bernard P. Zeigler, "Abstracting Event-Based Control Models for High Autonomy Systems", *IEEE Trans. on Systems, Man and Cybernetics*, vol., 23, no. 1, pp. 42-54, JANUARY/FEBRUARY 1993.

[3] Sungdo Chi, Modelling and Simulation for *High Autonomy Systems*, PhD thesis, University of Arizona, 1991.

[4] Bernard P. Zeigler, "High Autonomy Systems: Concepts and Models", *Proceedings of AI, Simulation, and Planning in HIgh Autonomy Systems*, pp. 2-7, Mar. 1990.

[5] Rahmat Shoureshi, "Intelligent Control Systems: Are They for Real?", *Journal of Dynamic Systems, Measurement and Control*, vol. 115, pp. 392-401, June 1993.

[6] Bernard P. Zeigler, "DEVS Representation of Dynamical Systems: Event-Based Intelligent Control", *Proceddings of the IEEE*, vol. 77, no. 1, pp. 72-80, Jan. 1989.

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

[8] Bernard P. Zeigler, *Theory of Modelling and Simulation*, John Wiley, 1976.

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

[10] Sung Hoon Jung, Tag Gon Kim, and Kyu Ho Park, "Event-Based Intellligent Control Using Endomorphic Neural Network Model", *Applied AI*, (accepted for publication).

[11] E.H. Mamdani, "Applications of fuzzy algorithms for control of simple dynamic plant", *Proceedings, of the IEEE*, vol. 121, no. 12, pp. 1585-1588, 1974.

[12] Chunen Chien Lee, "Fuzzy Logic in Control Systems: Fuzzy Logic Controller-Part I/II", *IEEE Trans. on Systems, Man and Cybernetics*, vol. 20, no. 2, pp. 404-435, MARCH/APRIL 1990.

[13] Wonweek Yu and Zeungnam Bien, "Design of Fuzzy Logic Controller with Inconsistent Rule Base", *5th IFSA World Congress*, vol. 2, pp. 1370-1373, 1993.

[14] Chuen-Chien Lee, "Intelligent Control Based on Fuzzy Logic and Neural Net Theory", *Proceedings of the INternational Conference on Fuzzy Logic*, pp. 759-764, July 1990.

[15] Chin-Teng Lin and C.S. George Lee, "Neural-Network-Based Fuzzy Logic Control and decision System", *IEEE Trans. on Computers,* vol. 40, no. 12, pp. 1320-1336, Dec. 1991.

[16] Jyh-Shing R. Jang, "Self-Learning Fuzzy Controllers Based on Temporal Back Propgation", *IEEE Trans. on Neural Networks*, vol. 3, no. 5, pp. 714-723, Sept. 1992.

[17] Panos Antsaklis, "Defining Intelligent Control", *IEEE Control Systems*, pp. 4-5, 58-66, June 1994.

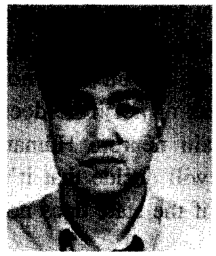
[18] "ART-IM/Windows Programming Language Reference", Inference Corporation, 1991.

[19] Tag Gon Kim and Bernard P. Zeigler, "AIDECS: An AI-Based, Distributed Environmental Control System for Self-Sustaining Habits", *Artificial Intelligence in Engineering*, vol., 5, no. 1, pp. 33-42, 1990.

[20] Tag Gon Kim, "Hierarchical Scheduling in an Intelligent Environmental Control Systems", *Journal of Intelligent and Robotic Systems*, vol. 3, pp. 183-193, 1990.

Sung Hoon Jung
He currently holds a postdoctoral position in the Department of Eelctrical Engineering at the Korea Advanced Institute of Science and Technology. He received his B.S.E.E. degree from Hanyang University, Korea, in 1988 and M.S. and Ph.D. degree from KAIST, in 1991 and 1995, respectively. Hisresearch interests are in the field of intellignet control, and in particular of neural networks, fuzzy control, genetic algorithms, and event-based control. Dr. Jung is a member of the Korea Fuzzy Mathematics and Systems Society (KFMS).

Tag Gon Kim
He received the B.S.E.E. and and M.S.E.E. degrees from Pusan National University, Korea, and Kyungpook National University, Korea, in 1975 and 1980, respectively. He received the PhD degree in Electrical Engineering from the University of Arizona, Tucson, AZ, in 1988. From 1987 to 1989, Dr. Kim worked as a Research Staff Engineer in the Environmental Research Lab of the University of Arizona. From 1989 to 1991, he was an Assistna tProvessor in the Department of Electrical and computer Engineering, the University of Kansas, Lawrence, Kansas. Since September 1991, he has been an Associate Professor in the Department of Electrical Engineering, KAIST, Taejon, Korea. His research interests include advanced modelling simulation methodology, computer systems analysis, and object-oriented software environment. Dr. Kim is an Associate Editor for international journals: International Journal in Computer Simulation, Simulation (SCS), Transactions for the Society of Computer Simulation, and Simulation Digest (IEEE/ACM). He is also on the editorial board of International Journal of Intelligent Cntrol and Systems and is a member of a KITE, the IEEE, the ACM, the AAAI, the SCS, and Eta Kappa Nu.



Kwang-Hyun Cho
He was born in Korea, in 1971. He received the B.S. and M.S. degrees in electrical engineering from Korea advanced Institute of Science and Technology in 1993 and 1995, respectively. His areas of research interest are supervisory control, fault tolerant system, robust control, and computer integrated manufacturing system. He is currently working towards the Ph.D. degree in electrical engineering, Korea Advanced Institute of Science and Technology. He is a student member of IEEE, IITE, KFMS, and ICASE

Kyu Ho park
He is a Professor in the Department of Electrical Engineering at Korea Advanced Institute of Science and Technology. He received his B.S. degree from Seoul National University, Korea, in 1973 and the M.S. degree from KAIST, in 1975. He received Dr. Ing. degree from the University de Paris, France, in 1983 in electrical engineering. His major interests include computer vision, computer architecture and parallel processing. Dr. Park is a member of KISS, KITE, and IEEE.