MODULAR ADAPTIVE NEURO-FUZZY PID CONTROLLER BASED ON ONLINE PROCESS RESPONSE ESTIMATION

Thesis submitted in partial fulfillment of the requirements for the award of Doctor of Philosophy

By

MYO THEIN KYAW

Department of Computer and Information Sciences
School of Mathematics and Computer/Information Sciences
University of Hyderabad
Hyderabad - 500 046

May, 2005

CERTIFICATE

Certified that, the work embodied in this thesis entitled **MODULAR ADAPTIVE NEURO-FUZZY PID CONTROLLER BASED ON ONLINE PROCESS RESPONSE ESTIMATION** has been carried out by Mr. Myo Thein Kyaw for the full period under Ph.D ordinances of the University, under my supervision. The same has not been submitted for the award of research degree of any other University.

Place : University of Hyderabad

Date :

[illegible] 24.5.05

Prof. Arun Agarwal
(Supervisor)

[illegible] 24.5.05

Head
Department of
Computer and Information Sciences,
University of Hyderabad.

HEAD
Department of CIS
UNIVERSITY OF HYDERABAD
HYDERABAD-500 046

[illegible] 27/5/05

Dean
School of Mathematics and
Computer/Information Sciences,
University of Hyderabad

DECLARATION

I hereby declare that the work embodied in this thesis has been carried out by me under the supervision of Prof. Arun Agarwal, in the Department of Computer and Information Sciences, University of Hyderabad, and the same has not been submitted at any other University.

[illegible]

Place : University of Hyderabad

Date : 23/05/2005

(Myo Thein Kyaw)

ACKNOWLEDGEMENTS

Firstly, I would like to thank my supervisor and Head of the Department, Prof. Arun Agarwal for providing indispensable feedback and guidance during my doctoral research.

I am thankful to Prof. A. K. Pujari, Dean, School of MCIS, for his support and encouragement. I also thank full to Prof. H. Mohanty, Dr. S. Bapi Raju who has introduced neural network technology to me and Dr. Atul Negi for their valuable comments and encouragement. I greatly appreciate their interest in my work. My due respects to all other faculty members of the department for their help and valuable discussions.

I am also much obliged to my colleagues Pammi Chandrasekhar, K. J. Pramod, Debasis Patnaik, Ashin Sihanalankara, Ashin Pannasihalankara, Ye Lwin Myint, Kyaw Naing Soe, Chit Htay Lwin, Dr. Tin Htwe, Dr. Phyu, Dr. Soe Win and T. Narasimhulu for their help.

I am grateful to my parents U Mg Mg Kyaw and Daw Khin Hlaing, my wife Htoo Htoo Aye and my children May Thu and Myat Thu for their constant support and encouragement. Their love and faith are my source of inspiration and strength.

Finally, I thank Government of the Union of Myanmar that allowed me to pursue my doctorial studies and ICCR (Indian Council for Cultural Relations) for providing financial assistance for my research work.

Contents

Abstract

Among a variety of control systems, the use of PID control has a long history in control engineering and is acceptable for many real applications. General applicability to most control systems and the simplicity of its structure are important features of conventional PID controller design. But, practical control problems are usually imprecise and can be changed by an unknown disturbance or other factors. Since the majority of conventional PID controller operates on error signal, unknown disturbance or noise effect is sensitive to degrade controller performance.

Though control theory highlights that sufficient flexible controller structure (modularity) is needed to implement high performance control system, generally, two degrees-of-freedom PID controller, non-modular various PID controller structures and non-modular artificial intelligence based PID controller structures are introduced up to present. In order to react appropriately for different manner of process responses, the diverse tuning strategies (gain scheduling) are required for different behavior of process response and modular structure can be regarded as a solution to it. Consequently, modular structure adaptive PID control system is needed to obtain an easily tunable and robust adaptive PID controller.

The adaptive control system can be represented by a nonlinear function, which maps the online information of process response into controller parameters. This map becomes the identity for direct algorithms. The main objective of proposed idea is to implement modular structure adaptive PID controller based on online qualitative information of process response. In our approach, the conventional PID controller is used as the standard controller. The gains of standard controller (*proportional gain*, *integral gain* and *derivative gain*) are tuned online based on information of process response.

We analyze both frequency response method and transient/steady-state analysis in time domain to define local tuning strategy for different behavior of process response.

These analyses reveal that it is necessary to decompose the whole process response into four categories (*rise time state, overshoot/oscillation state, disturbance state* and *steady-state*) for implementing individual local tuning strategies. Our proposed knowledge-based modular neural network classifier is responsible to provide online this information of process response. The design of our neural network classifier is developed based on well-known conventional control theory and multiple classifier strategy.

Fuzzy technology has some merits over traditional gain scheduling techniques such as providing smooth transaction between operations and incorporating existing knowledge. Accordingly, modular fuzzy adaptive mechanism is used to store the controller parameter transaction policy (local gain scheduling module) of different operating state. This mechanism consists of several fuzzy inference systems and they represent different tuning strategies. The appropriate fuzzy inference systems are selected online based on both qualitative information (provided by our proposed knowledge-based modular neural network classifier) and quantitative information (error signal) of process response.

Since the structure of proposed controller is modular, the overall representation of the controller design is more transparent and achieves greater flexibility. It is easy to modify, debug and possible to use a priori knowledge of existing control theory into the architecture by using a different module structure fitted for different operating regions.

The effectiveness of our proposed controller is observed with several experiments. First group of experiment examines the accuracy and validity of proposed knowledge-based modular neural network classifier. These results show that our proposed knowledge-based modular neural network classifier can provide sufficient process response estimation accuracy. The performance of proposed controller is observed with second group of experiments. It includes several types of experiments such as flexibility test, robustness test, disturbance rejection test and transient response characteristic test. These results are compared with other reported works. They show that proposed controller design is well-suited for both disturbance rejection and set point tracking and provides better performance than conventional PID controller along all the four important characteristics (stability, sensitivity, steady-state accuracy and transient response) of the control system design.

Notation and Abbreviations

α	Set-point weighting parameter for derivative gain
β	Set-point weighting parameter for proportional gain
δ_k	Peak value of process response in k^{th} sampling time
σ	Integration interval
ω_g	Gain crossover frequency
ω_p	Phase crossover frequency
2DOF	Two-Degrees-Of-Freedom
3-D	Three-dimensions
ANFIS	Adaptive-Network-based-Fuzzy-Inference System
b	Bias value of a neuron
c	The center of fuzzy inference system *f4*
C_a	Acceleration error constant
C_p	Position error constant
cr	Change in reference signal
cur_r	Current Region
C_v	Velocity error constant
dde	Error's second difference signal
de	Error's first difference signal
DE_NEG	Change in error is negative
DE_POS	Change in error is positive
d_i	i^{th} discriminant function
e(n)	Error signal in n^{th} sampling time (discrete form)
E(s)	Error of closed-loop system
e(t)	Error signal (continuous form)
E_NEG	Error is negative

E_POS	Error is positive
E_ZE	Error is zero
eq_b	Equilibrium
e_{ss}	Steady-state error
f_1	Fuzzy inference system that produces k_{pscale}
f_2	Fuzzy inference system that produces k_{iscale} in rise time state (near the set point)
f_3	Fuzzy inference system that produces k_{dscale}
f_4	Fuzzy inference system that produces k_{iscale} in overshoot/oscillation state
f_5	Fuzzy inference system that produces k_{iscale} in rise time state (far from the set point)
f_6	Fuzzy inference system that produces *offset1*
f_7	Fuzzy inference system that produces *offset2*
FInc	Fuzzy incremental controller
f_{kbmnn}	Knowledge-based modular neural network classifier
FLC	Fuzzy logic controller
FP	Fuzzy proportional controller
FPD	Fuzzy proportional derivative controller
FPD+I	Fuzzy proportional derivative controller + integral controller
G(s)	Transfer Function of open-loop system
g_1	A function that produces proportional gain scaling factor
g_2	A function that produces integral gain scaling factor
g_3	A function that produces derivative gain scaling factor
$G_c(s)$	Transfer Function of controller
GCE	Input scaling factor of change in error signal
GCU	Output scaling factor of *PI-FLC*
$G_d(s)$	Transfer Function of time delay
GE	Input scaling factor of error signal
GM	Gain Margin
$G_p(s)$	Transfer Function of process

GU	Output scaling factor of *PD-FLC*
H(s)	Transfer Function of feedback path
IMC	Internal model control
I-PD	Integral controller + Proportional Derivative controller
KBMNN	Knowledge-based modular neural network
k_d	Derivative gain of conventional PID controller
k_d'	Derivative gain of proposed PID controller
k_{dscale}	Derivative gain scaling factor
k_i	Integral gain of conventional PID controller
k_i'	Integral gain of proposed PID controller
k_{iscale}	Integral gain scaling factor
k_p	Proportional gain of conventional PID controller
k_p'	Proportional gain of proposed PID controller
k_{pscale}	Proportional gain scaling factor
MIMO	Multi Input Multi Output
MIT rule	The rule that is developed by Massachusetts Institute of Technology
MRAS	Model References Adaptive System
net	Activation value of a neuron
NN	Neural Network
o^1_i	i^{th} output of first combination unit of *KBMNN*
o^2_i	i^{th} output of second combination unit of *KBMNN*
offset1	The value to compute the center of fuzzy inference system *f4*
offset2	The value to compute increment of integral action in initial moment of disturbance state
P+ID	Proportional controller + Integral Derivative controller
PD	Proportional Derivative controller
PD+I	Proportional Derivative controller + Integral controller
PI	Proportional Integral controller
PID	Proportional Integral Derivative controller
PID-FLC	Proportional Integral Derivative type Fuzzy logic controller
PIDNN	Proportional Integral Derivative Neural Network

PM	Phase Margin
pre_c	Previous Class
pre_r	Previous Region
R(s)	Input of closed-loop system
S(s)	Sensitivity of closed-loop Transfer Function
SISO	Single Input Single Output
STR	Self-Tuning Regulator
T(s)	Transfer Function of closed-loop system
T1	Tuning strategy for rise time state (delay period)
T2	Tuning strategy for rise time state (far from the set point)
T3	Tuning strategy for rise time state (near the set point)
T4	Tuning strategy for overshoot/oscillation state
T5	Tuning strategy for steady-state
T6	Tuning strategy for disturbance state (diverging from the set point)
T7	Tuning strategy for disturbance state (approaching to the set point)
T_d	Derivative time constant
th1	Threshold value to define steady-state
th2	Threshold value to define classifier's activation
T_i	Integral time constant
TS	Sampling Time
$u_{pid}(t)$	Output of PID controller (continuous form)
$u_{pid}(n)$	Output of PID controller in n^{th} sampling time (discrete form)
w_i	Weight vector for each neuron
X	Input pattern vector
Y(s)	Output of closed-loop system

List of Figures

List of Tables

Chapter 1: Introduction

1.1. Control Engineering and Controller Design

The main concern in control theory is to study how signals are processed by dynamical systems and how this processing can be influenced to achieve a certain behavior. A control methodology can be defined as the set of techniques and procedures used to construct and/or implement a controller for a dynamical system. A schematic diagram of a general control system is shown in Figure 1-1.

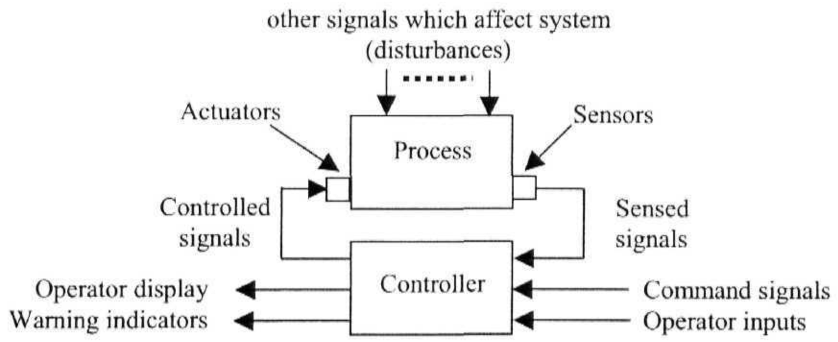


Fig. 1-1. A schematic diagram of a general control system.

Dynamical systems are found in all major engineering disciplines such as mechanical, electrical, fluid and thermal systems. They are merely a mapping from a certain input signal to an output signal. The future response of the system is determined by the present state of the system (the initial conditions) and the present input [Woods97].

Generally, dynamical systems can be described with three features. The first one is the state of a system, which is a representation of all the information about the system at some particular moment of time. The second feature is the set of all possible states to which a system can be assigned. The third is the state-transition function that is used to update and change the state from one moment to another.

The goal of control engineering is to improve, or in some cases enable, the performance of a system. The main objective of controller is to produce appropriate control action to drive actuators, which affect the behavior of the dynamical system. Sensors, which measure various signals in the system, and external command signals are also essential elements in automatic control applications.

Control engineering involves: Modeling or identification, Control configuration, Control law or controller design, Controller implementation, and Control system testing and validation. While each of these five tasks can be critical in control engineering, most research concentrates on controller design because it is difficult to precisely formulate the problems.

The controller or control law can be described as a signal processing to generate appropriate actuator signals based on sensor and other incoming signals. In order to agree with given specification (required characteristics), the designer must decide exactly how the actuators are to be driven by processing the incoming sensor signals. The important characteristics of control system are stability, sensitivity, disturbance rejection, steady-state accuracy and transient response of a system [Franklin00, Dorf04].

The controller design is actually being developed in the framework of three paradigms. The integral-differential paradigm, that includes all control methods using models starting from differential equations (transfer function and state-space), the data paradigm that includes all control methods based on experimental and empirical data (black-box models, neural models) and the linguistic paradigm that uses qualitative representations like fuzzy systems and controllers.

The first theoretical analysis of automatic control was a study of the stability of the fly-ball governor by James Clark Maxwell in 1868. His technique was to linearize the differential equations of motion to find the characteristic equation of the system. He studied the effect of the system parameters on stability and showed that the system is stable if the roots of the characteristic equation have negative real parts. E. J. Routh is credited with determining the first comprehensive mathematical stability criteria for higher order systems in 1877, an improvement upon Maxwell's earlier work which held good only for second and third order cases [RayChaudhuri95, Dorf04].

Control theory made significant strides in the past 120 years, with the use of frequency domain methods and Laplace transforms in the 1930s and 1940s. Optimal control methods and state space analysis was developed in the 1950s and 1960s. Then robust and adaptive control methods became famous from 1960s to today [Ogata02].

Over the past few decades the use of automatic controllers in industrial processes has accelerated and almost many aspects of our personal lives also depend on automatic controllers. But application to real time and safety critical systems is highly limited at present. One of today's major challenges is the design of high performance controllers for complex systems with high levels of uncertainty.

1.2. Conventional Control System

Modern control of the 1960's has now become a conventional (traditional) control. In conventional control system, the behavior of dynamical system is described by differential and difference equations. Powerful tools from linear algebra can be used for system analyses and control as long as the system behaves linearly. Thus assumption of system linearity has been made to develop a control theory on a solid basis.

1.2.1. The role of feedback concept in conventional control system

Although linear control theory has evolved a variety of powerful methods and has had a long history of successful industrial applications, it was found to be inadequate in many cases, for reasons such as requirement of increasingly greater in precision performance and for applications in large operating ranges. In practice, though linear systems possess the property of linearity only over a certain range of operations, all physical systems are nonlinear to some degree.

Thus it was necessary to acquire a facility for analyzing feedback control systems with varying degrees of nonlinearity. By using feedback properties, simpler plant models (i.e., linear models) can be used in the control design. The feedback mechanism provides some degree of immunity to discrepancies between the physical plant and the controller model. Consequently, the negative feedback concept has become the foundation for automatic control system design.

The earlier stage of the design of feedback control system was improved by trial-and-error together with a great deal of engineering intuition. Thus, it was more of an art than a science. Among them PID controller is the most common form of feedback control system because it needs no accurate model to implement a controller design [Ogata02].

A feedback loop measures the system response that includes both effects and disturbances and then compares the actual system response to the desired system response (reference signal) at each time instant and uses the error signal to drive the controller so as to generate real-time control corrections and keep errors small for all time. The feedback system supports to improve the stability, speed up the transient response, reduce the steady-state error and decreases the sensitivity to parameter variations [Franklin00].

1.2.2. Well-known mathematical techniques of conventional control system

A fundamental requirement in conventional control is mathematical modeling of the system to be controlled. Laplace transformation, the signal-flow diagram and the state-variable concept are well-known mathematical techniques of conventional control applications. In addition to these techniques, the design of digital controllers requires knowledge of the z transforms and some aspects of information theory [Franklin00, Dorf04]. These techniques are introduced briefly in this Sub-section.

Ordinary differential equations that described the behavior of systems can be solved by the Laplace transform. The resulting transformed equation is in purely algebraic terms, which can be easily manipulated to obtain a solution for the desired quantity as an explicit function of the complex variable. The Laplace transform of a function y(t) is defined in following Equation 1.1 and inverse Laplace transform, which returns to the time function is expressed by Equation 1.2. Utilizing these algebraic equations, a transfer function representation of the input-output relationship can be developed.

$$Y(s) = \mathcal{L}[y(t)] = \int_0^\infty y(t)e^{-st}\,dt \tag{1.1}$$

$$y(t) = \mathcal{L}^{-1}[Y(s)] = \frac{1}{2\pi}\int_{c-i\alpha}^{c+i\alpha} Y(s)e^{st}\,ds \tag{1.2}$$

Signal-flow graphs enable the control engineer to determine the response of a complicated linear and multi loop system. A signal-flow graph is a topological representation of a set of linear equations having the form of Equation 1.3.

$$y_i = \sum_{j=1}^{n} a_{ij}y_j, \quad i = 1,2,\ldots.,n. \tag{1.3}$$

A signal-flow graph that consists of branches and nodes represents a set of equation. Node i represents variable y_i and branch gains (a_{ij}) show the relation with other different variables. For example, Figure 1-2 represents the following set of linear equations.

$$y_2 = ay_1 + by_2 + cy_4, \tag{1.4}$$

$$y_3 = dy_2, \tag{1.5}$$

$$y_4 = ey_1 + fy_3, \tag{1.6}$$

$$y_5 = gy_3 + hy_4, \tag{1.7}$$

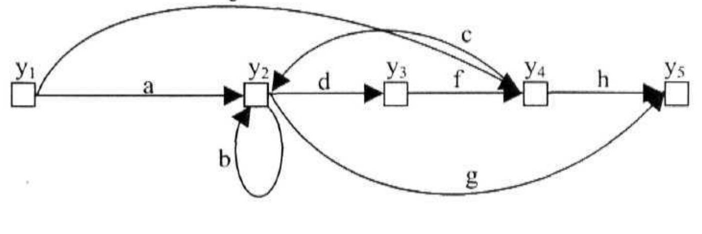


Fig. 1-2. Signal-flow graph.

In the analysis of state variable approach, the system is characterized by a set of first-order differential or difference equations that describe the state variable. For a dynamical system, the state of a system is described in terms of a set of state variables [x_1(t), x_2(t), x_3(t),....,x_n(t)]. A system's state refers to the initial, current and future behavior of a system. System analysis and design can be accomplished by solving a set of first-order equations rather than a single, high-order equation.

One of the mathematical tools devised for the analysis and design of discrete-data systems is the z-transform. The role of the z-transform to digital systems is similar to that of the Laplace transform to continuous-data system. The motivation for using the z-transform for the study of discrete-data systems can be explained by referring to the Laplace transform of a sampled signal. The Laplace transform of y*(t) (the output of an ideal sampler) is given by Equation 1.8.

$$Y^*(s) = \mathcal{L}[y^*(t)] = \sum_{k=0}^{\infty} y(kT)e^{-kTs} \qquad (1.8)$$

The relation between Laplace transform and z-transform can be expressed as Equation 1.9. Equation 1.10 expresses the z-transform. Notice that the z-transform describes the values of the time function at the sampling time only.

$$z = e^{Ts} \qquad (1.9)$$

$$Y^*[s = \frac{1}{T}\ln z] = Y(z) = Z[y(k)] = \sum_{k=0}^{\infty} y(kT)e^{-k} \qquad (1.10)$$

1.2.3. Brief explanation of conventional controller design

In many practical cases, it is necessary to redesign the system (by modifying the structure or by incorporating additional components) to alter the overall behavior so that the system will behave as desired and this additional device is called controller or compensator. In order to implement conventional controller design, there is a need for the system to be controlled as described with mathematical modeling. Then the controller can be designed to guarantee stability and to satisfy the given specifications based on following techniques:

a) The Nyquist stability criterion is a very valuable tool that determines the degree of stability or instability of a feedback control system. In addition, it is the basis for other methods that are used to improve both the steady-state and the transient response of a feedback control system. Application of the Nyquist stability criterion requires a polar plot of the open-loop transfer function, which is usually referred to as the Nyquist diagram [Ogata02, Dorf04]. This plot indicates clearly the manner in which the system should be modified.

b) The Bode-diagram approach is one of the most commonly used methods for the analysis and synthesis of linear feedback control systems. This method is basically an extension of the Nyquist stability criterion. It is useful to be able to plot the frequency response of a system in order to (a) design simple systems, (b) check computer based results and (c) understand the effect of compensation changes in design iterations [Ogata02, Dorf04].

c) The Root-locus is a technique, which investigates the movement of the characteristics roots on the s-plane as the system parameters are varied. The knowledge of the location of the closed-loop roots permits the accurate determination of a control system's relative stability and transient performance. The Root-locus technique helps to locate the closed-loop roots in order to get given specifications (relative stability and transient performance). It can be applied to obtain an approximate sketch for analyzing the initial design of a system and determine suitable alterations of the system structure and parameter values [Ogata02, Dorf04].

d) In time domain, state variable feedback system can be used to implement controller design. The main limitation of this technique is that controllability and observability are needed and it is simply not possible or practical to sense all the states and feed them back [Dorf04].

By observing the nature of the process with these techniques, the appropriate compensator or controller can be developed to achieve the desired system performance. Phase-lead, phase-lag, lag-lead, PI, PD and PID are practically useful controller design and the compensation is usually classified as: (a) series compensation or cascade compensation, (b) feedback compensation or parallel compensation, (c) state feedback compensation and (d) feedforward compensation [Bandyopadhyay03].

1.3. Adaptive Control System

Since the late 50's, control theorists have struggled to develop adaptive control laws that guarantee closed loop stability in the presence of unmodeled dynamics and external disturbances. Moreover, though physical laws help to build model of the system and find the shapes of the nonlinear functions, parameters of the system (e.g., the inertia parameters of a new object grasped by a robot) may depend on operational conditions and may not be precisely known in advance. Because of some factors such as aging effect, the parameters may also be slowly time-varying.

The adaptive control theory essentially deals with finding parameter adjustment algorithms that offer global stability and convergence guarantees. The adaptive controller can be defined as a controller that can modify its behavior in response to changes in the dynamics of the process and the nature of the disturbance. It can handle wider operating ranges than fixed controllers and can be considered to have higher degrees of autonomy than control systems employing fixed feedback controllers.

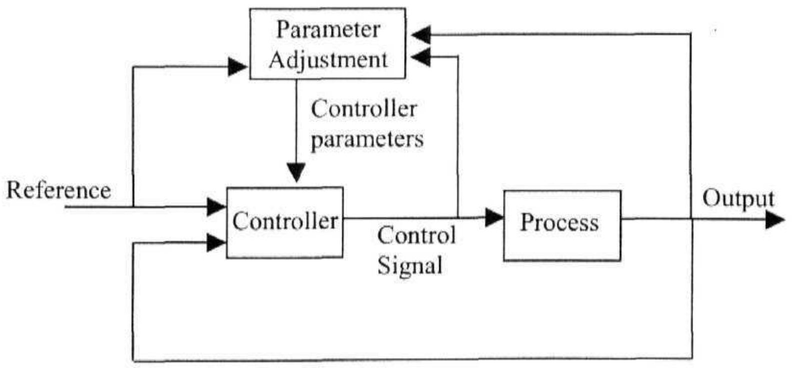


Fig. 1-3. Block diagram of an adaptive system.

Adaptive controllers generally contain two extra components on top of the standard controller itself. The first is a process monitor that detects changes in the process characteristics and the second component is the adaptation mechanism itself

[Driankov01]. It is initially designed as a function of a parameter set and state variables of a mathematical model. The parameters can be updated online during operation and the controller will adjust itself to the newly updated parameters. The Block diagram of an adaptive system is described in Figure 1-3 [Astrom03].

In most of the classical adaptive control literature it is common to assume the unknown dynamics to have a known structure with unknown parameters [Nam88]. The linear parameterization of unknown dynamics posses serious obstacles in adopting adaptive control algorithms in practical applications, because it is difficult to fix the structure of the unknown nonlinearities. This fact has been the motivating factor behind the interest in online function approximators to estimate and learn unknown function. The most common function approximators used in adaptive control are artificial neural network [Nardi00] and fuzzy logic structure [Driankov01].

The controller design can be represented by a nonlinear function, which maps the estimated parameters into controller parameters. This map becomes the identity for direct algorithms. Traditionally, there are four basic approaches for adaptive control: (i) gain scheduling, (ii) model reference adaptive system, (iii) self-tuning regulator and (iv) dual control [Astrom03].

1.3.1. Gain scheduling

The gain scheduling approach is perhaps one of the most popular nonlinear control design approaches which have been widely and successfully applied in fields ranging from aerospace to process control [Astrom03]. In many cases it is possible to find auxiliary variables that correlate well with changes in process dynamics. The appropriate controller parameters are tuned with respect to current process dynamics and this method is called gain scheduling.

A block diagram of a system with gain scheduling is shown in Figure 1-4. It has two loops: the inner loop composed of the process and controller and the outer loop that adjusts the controller parameters on the basis of the operating conditions [Astrom03]. The main problem in the design of systems with gain scheduling is to define suitable scheduling variables. When scheduling variables have been defined, the controller parameters are calculated at a number of operating conditions by using some suitable design method.

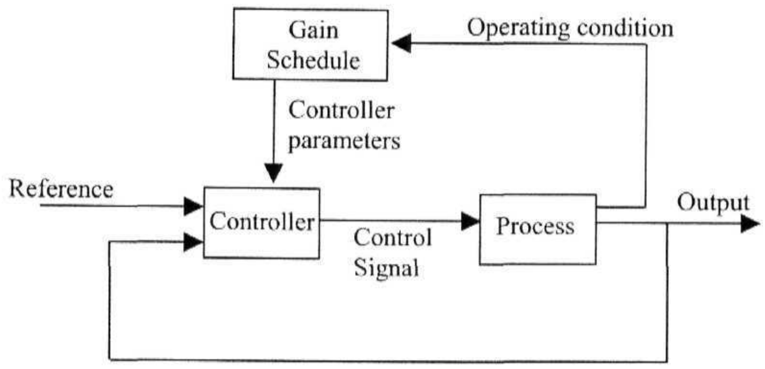


Fig. 1-4. Block diagram of a system with gain scheduling.

Gain scheduling design typically employs a divide and conquer approach whereby the nonlinear design task is decomposed into a number of linear sub-tasks. It involves three main tasks: partitioning of the operating region into several approximately linear regions, designing a local controller for each linear region and interpolation of controller parameters between the linear regions [Leith00]. The main purpose of gain scheduling can be viewed as a connection of a number of linear controllers, each of which functions around a different operating point, to form a global controller that is stable and functional not only at these operating points but also along the whole operating trajectory.

A gain-scheduled control system is the same as a feedback control system in which the feedback gains are adjusted using feedforward compensation. The feedback dynamics are scheduled between the operating points to cover the range of plant's dynamics. The measurement of operating conditions of the process based gain scheduling technique is often a good way to compensate for variations in process parameters or known nonlinearities of the process [Brdys02]. Moreover, controller parameters can be adjusted very quickly in response to changes in the plant dynamics and gain scheduling method is simpler to implement than automatic tuning or adaptation.

The classical gain scheduling design approach is based on the family of equilibrium linearization of the plant and may be applied directly to a broad range of nonlinear systems. The main drawback of classical gain scheduling is that the parameter change may be rather abrupt across the boundaries of the region, which may result in unacceptable or even unstable performance. Another problem is that accurate linear time invariant models at various operating points may be difficult to obtain.

In order to overcome these limitations, some researchers have applied fuzzy logic concept. Many earlier works reveal that fuzzy logic is used in supervisory role to provide how the gain variations are to be adjusted in the scheduling algorithm [Brdys02, Khiang96]. In these techniques, fuzzy linguistic rules and fuzzy inference mechanisms are used to establish the required control policy. The fuzzy gain scheduling is an effective way to adapt the parameters of the controller and permit smooth adaptation of the gain of the controller as a function of the state instead of step-wise manner as in conventional gain scheduling [Leith00].

1.3.2. Model-references adaptive systems (MRAS)

The model reference adaptive system is a method of comparing the performance of the actual system against an assumed mathematical model that describes the actual system. The controller produces appropriate control action to drive this comparison error to zero. The main requirement of model-based adaptive control is to select an appropriate reference model and adaptation algorithm, which modifies the feedback gains of the controller.

The objective of a model reference adaptive control system is to determine a feedback control law which forces the state of the plant to asymptotically track the state of a given reference model. At the same time, the unknown parameters in the plant model are estimated and used to update the control law [Kraaft89].

In MRAS, parameter adjustment mechanism must be developed in order to get a stable system, which brings the error to zero as described in Equation 1.11. The following parameter adjustment mechanism, called the MIT rule (Equation (1.12)), was used in the original MRAS:

$$e(t) = (G(p,\theta) - G_m(p)) \times u_c(t) \tag{1.11}$$

$$\frac{d\theta}{dt} = -\gamma e \frac{\partial e}{\partial \theta} \tag{1.12}$$

In these equations, $G_m(p)$ is transfer function of model, $G(p,\theta)$ is closed-loop transfer function of the plant and θ is a controller parameter. The quantity $\partial e/\partial\theta$ is the sensitivity derivative of the error with respect to parameter θ. The parameter γ determines the adaptation rate. It is noticed that there is no guarantee that an adaptive controller based on the MIT rule will give a stable closed-loop response and many researchers emphasize Lyapunov's theory to overcome this limitation [Astrom03].

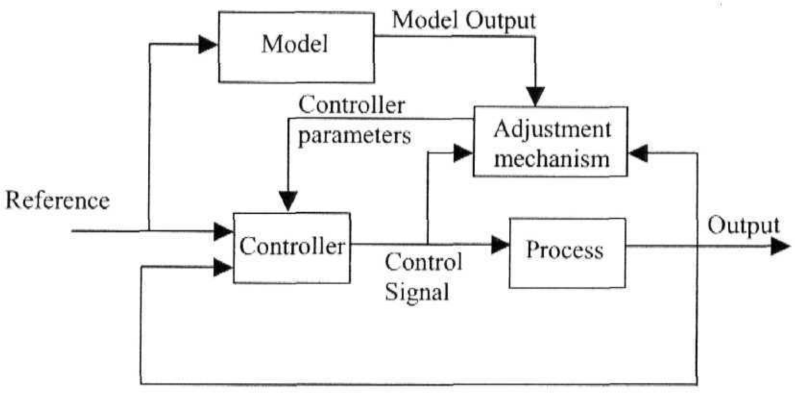


Fig. 1-5. Block diagram of a model-references adaptive system (MRAS).

On the other hand, fuzzy-rule-based modeling now plays a role as a so-called universal approximator, which can facilitate parameterization of vague system or uncertain systems. In Hasegawa et al. [Hasegawa95], fuzzy neural network is employed to describe the model of the unknown system and fuzzy logic controller represents both adjustment mechanism and controller.

1.3.3. Self-tuning regulators (STR)

Self-tuning regulator is originally proposed by Kalman and later developed by Astrom and Wittenmark in 1973. The basic of a self-tuning system is an algorithm that will automatically change its parameters to meet a particular requirement or situation. This is achieved by the addition of a mechanism, which monitors the system and adjusts the coefficients of the corresponding controller to maintain a required performance.

The self-tuning regulator is essentially a classical feedback system with online adjustable coefficients. The controller coefficients are adjusted during each control cycle according to the best estimate of the system parameters. The choice of model structure and its parameterization are important issues for self tuning regulator [Astrom03].

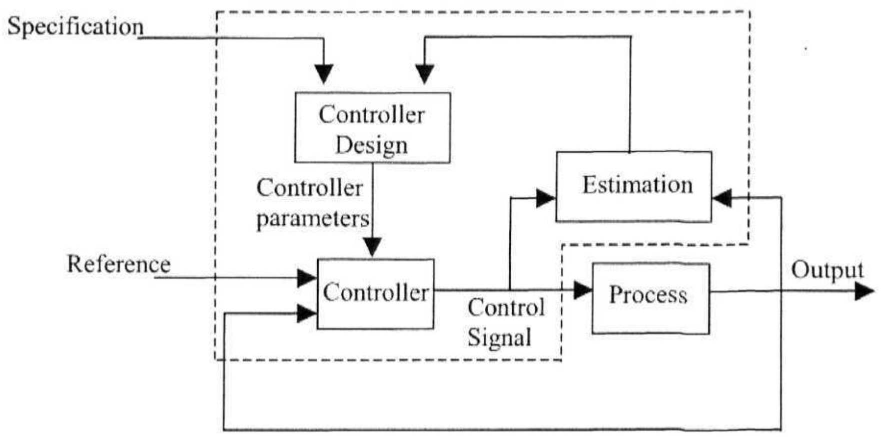


Fig. 1-6. Block diagram of a self-tuning regulator (STR).

The block diagram of a self-tuner is shown in Figure 1-6. Simple estimation methods (i.e., least square) and simple conventional control design techniques (i.e., pole placement) can be applied to implement self-tuning regulator design. The important condition for self-tuning regulator methodology to linear time-varying process is that the changes in process parameters are slow relative to the process dynamical response [Astrom03].

Alternately, fuzzy and neural concept can be used to implement self-tuning regulator adaptive controller since their parameters can be altered online. Adaptive fuzzy logic controller that modifies the fuzzy set definitions or the scaling factors is called self-tuning controller. Adaptive fuzzy logic controller is defined as self-organizing controller if it can modify the rules [Driankov01]. In Minkova et al. [Minkova98] proposed work, artificial neural network controller is utilized as an adaptive controller and its weights are updated online with well-known back propagation algorithm.

1.3.4. Dual control

The main concept of dual controller design comes from the process: its parameters and the environment can be described by using a stochastic model. It is noted that the uncertainties in the estimated parameters are taken into account in controller design.

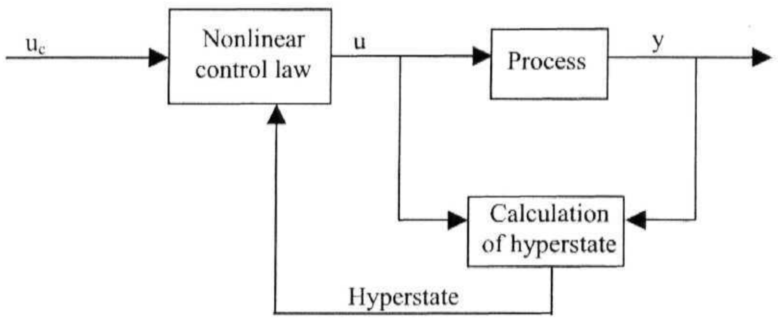


Fig. 1-7. Block diagram of a dual control.

The controller consists of two parts: a nonlinear estimator and a feedback controller. The estimator generates the conditional probability distribution of the state from the measurement, *p(z/y,u)*. This distribution is called the hyperstate of the problem. The feedback controller is a nonlinear function that maps the hyperstate into the space of control variables. The criterion is formulated so as to minimize the expected value of loss function.

To solve the dual control problem, it is necessary to evaluate the influence of the control signal on the future outputs and to estimate and predict the behavior of the stochastic parameters. The estimation problem is defined so as to compute the conditional probability distribution of the parameters, given the measured data. Since the approach is so complicated, so far it has not been possible to use it for practical problems [Astrom03].

1.4. Intelligent Control System

Today, our society needs high performance control system that has the ability to act appropriately in an uncertain environment or unexpected situations. It cannot be successfully addressed by the existing conventional control theory and adaptive control theory. The future control system needs high autonomy and flexibility. It is noted that intelligence is necessary to provide desirable functioning of systems under changing conditions. Consequently, the research area of intelligent system plays important role to fulfill this gap that are not characterized as conventional control. A general characterization of intelligent systems can be expressed as following [Antsaklis93]:

"In order for a man-made intelligent system to act appropriately, it may emulate functions of living creatures and ultimately human mental faculties. An intelligent system can be characterized along a number of dimensions. There are degrees or levels of intelligence that can be measured along the various dimensions of intelligence. At a minimum, intelligence requires the ability to sense the environment, to make decisions and to control action. Higher levels of intelligence may include the ability to recognize objects and events, to represent knowledge in a world model and to reason about and plan for the future. In advanced forms, intelligence provides the capacity to perceive and understand, to choose wisely, and to act successfully under a large variety of circumstances so as to survive and prosper in a complex and often hostile environment. Intelligence can be observed to grow and evolve, both through growth in computational power and through accumulation of knowledge of how to sense, decide and act in a complex and changing world".

Many researchers are approaching to emulate important characteristics of human intelligence to develop the intelligent control system [Boss00b, Chen02, Khiang96]. These characteristics include adaptation and learning, planning under large uncertainty and coping with large amounts of data. It either seeks to replace a human who performs a control task (e.g., a chemical process operator) or borrows ideas from how biological system solves problems and applies them to the solution of control problems (e.g., the use of neural network for control).

Artificial Intelligence (AI) is basically embedding human intelligence in a machine so that a machine can think intelligently like a human [Boss00a]. Recently, excitement over the field of intelligent control has risen due to progress in the area of artificial intelligence such as fuzzy control, neural networks, genetic algorithms and expert systems [Tyan96, Seng98]. These technologies permit building high autonomy controller designs supporting control systems that can perform well under significant uncertainties in the plant and the environment for extended periods of time [Antsaklis93].

On the other hand, in solving real-life engineering problems, domain knowledge or experience from domain experts are usually available. How one can take advantage of the domain expertise to either solve the problem or improve the performance is very important to the engineering world.

Formulating such information in words and use a suitable methodology to integrate it with traditional numerical designs are very desirable. Typically, the knowledge can be expressed in words or linguistic rules. Words can be integrated into controller design via fuzzy methodology. Some works show that control with words has much better performance as compared with pure numerical control [Tseng01].

Utilizing available knowledge to aid conventional numerical control design is helpful in both performance and stability requirements. The organization of knowledge is also one important attribute of intelligence. Many adaptive or learning control systems can be thought of as designing a control law to meet well-defined control objectives. This activity represents the system's attempt to organize or order knowledge of its own dynamical behavior to meet a control objective. If this organization is done autonomously by the system, then intelligence becomes a property of the system, rather than of the system's designer. The self-organization fuzzy controller design [Driankov01] can provide this feature and in the approach of Seng et al. [Seng98], their proposed neuro-fuzzy controller is able to self-organize the controller structure.

It is important to note that, intelligent control uses conventional control methods to solve "lower level" control problems and conventional control is included in the area of intelligent control [Antsaklis93] because conventional control technology is based on well-founded theory that has been successfully applied in many practical applications.

1.5. Modular Structure Intelligent Control System

Modularity has several advantages over non-modularity systems. Modularity is a natural way to ease the learning of complex behavior. It is well-suited for large-scale problems and provides a better speed in computation and better generalization [Jacobs90]. It is possible to introduce a priori knowledge into the architecture by using different module structure fitted for different plant operating regions. This is important when knowledge about the system is available.

In practical control applications, modularity brings in clarity in the overall representation of the controller design. The continuing goal of control systems is to provide extensive flexibility and a high level of autonomy [Dorf04]. Modularity in controller design is that the design of switching logic, the estimators and the candidate controllers are mutually independent and it achieves greater flexibility in applications [Hespanha03].

Most of the dynamical systems exhibit nonlinear behavior and linearization of these nonlinear systems is done by the assumption that the plant normally operates near the equilibrium. But linear approximations are no longer valid in many practical cases. Some works approach multiple models design to represent the system dynamics over the entire operating regime. By observing the physical properties of the problem under study, one can simplify the problem by modularizing the system and design the solution for each smaller and simpler module [Jacobs90].

On the other hand, multiple models, switching law and tuning law can be considered as a natural framework for intelligent control. Each model has different characteristic that is suitable for different modes of operation of the process. The switching law is used to select most suitable model to generate a new control input based on online prediction of the behavior of the system [Narendra97].

Modular structure controller consists of computing modules and a gating system. The computing modules are the operating part of the architecture. The gating system is used for the selection of the module valid for a current state of the plant [Ronco98]. Defining switching algorithm and task decomposition are key requirements to implement modular structure controller design.

Smith and Hunt [Smith95] presented the method for the automatic construction of local model networks. They showed that this architecture could represent nonlinear dynamic systems by smoothly integrating a number of simpler locally accurate models to apply in control and modeling applications.

Chen and Narendra introduced multiple models intelligent control system based on neural concept [Chen01, Chen02]. A linear robust adaptive controller and multiple nonlinear neural network based adaptive controllers are used in their design. A switching law is suitably defined to switch between these controllers based upon their performance in predicting the plant output.

1.6. Objectives of Thesis

In recent years, a number of authors have addressed a variety of control methods to handle unknown nonlinear systems with high level of uncertainty from different views. Some of the key concepts are [Antsaklis93, Ronco98, Tseng01, Dorf04]:

(1) Existing domain knowledge or experience of domain experts supports to improve the performance of control system. The controller design with both words and numerical technique is having relatively better performance as compared to pure numerical control.

(2) Artificial intelligence technology attempts to build upon and enhance the conventional control methodologies and enables to implement high autonomy controller to solve new challenging control problems.

(3) The modularity reduces the complexity of the controller design and provides for greater flexibility.

(4) The continuing goal of control systems is to provide extensive flexibility and a high level of autonomy.

Our thesis attempted to build a modular neuro-fuzzy gain scheduling PID controller based on these concepts. The key objective of this dissertation is to build an artificial intelligence based modular structure adaptive control system in order to get the high level of flexibility.

Figure 1-8 describes the fundamental idea of our proposed controller design. Conventional PID controller is used as a standard controller since the Proportional-Integral-Derivative (PID) controller is the most common algorithm and has clear relations between system response parameters and controller's gains. Moreover, sufficient flexible PID controller yields excellent results in many applications.

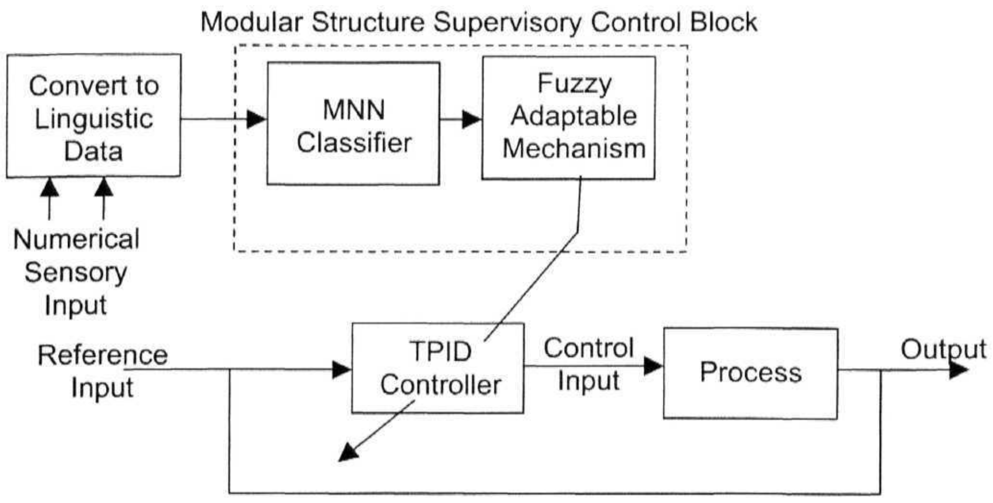

Modular Structure Supervisory Control Block
Convert to Linguistic Data
MNN Classifier
Fuzzy Adaptable Mechanism
Numerical Sensory Input
Reference Input
TPID Controller
Control Input
Process
Output

Fig. 1-8. Our proposed adaptive controller block diagram.

In adaptive control application, the observation of the system states is one of the important factors to implement adaptive law. The proposed adaptive law is mainly based on the current estimated process response. Modular neural network classifier is used to estimate the process response online since it has some merits over non-modular neural network classifiers and traditional classification systems.

The modular fuzzy gain scheduling mechanism consists of several local computing modules and they are responsible for different characteristics of process responses. It is responsible to produce gain scaling factors to update the parameters of the standard controller.

1.7. Organization of Thesis

This thesis consists of seven chapters including the Introduction. The remaining chapters are organized as follows:

Chapter 2: PID Controllers: A Research Review.

In recent years, many different ideas of PID controller design have been developed in order to improve performance of conventional PID controller such as a variety of conventional PID controller structures (P+ID, IPD, two degree-of-freedom PID controller, PID controller with resetting factor), conventional PID controller with compensation, PID-like artificial intelligence based controllers and gain scheduling PID controllers. Chapter 2 reviews the nature of conventional PID controller in frequency domain and a variety of PID controller designs.

Chapter3: A Proposed Modular Neuro-Fuzzy Control System.

Chapter 3 presents fundamental idea of our proposed modular fuzzy gain scheduling PID controller as shown in Figure 1-8. It reveals that the proposed adaptive law and the way of getting qualitative information of process response. The proposed knowledge-based modular neural network classifier and modular fuzzy gain scheduling mechanism are briefly discussed in this chapter.

Chapter 4: Knowledge-Based Modular Neural Network Classifier for Online Process Response Estimation.

Chapter 4 describes in detail how to implement knowledge-based modular neural network classifier to estimate the current process response online. In this work appropriate network topology, initial weights and biases are determined by the existing knowledge and engineering principles. This approach is the same as multiple classifiers strategy with simple task decomposition and recombining technique.

Chapter 5: Modular Fuzzy Gain Scheduling System.

Modular fuzzy gain scheduling mechanism is introduced in Chapter 5. It is combined with several independent modules and they are responsible for the different operating regions. It is developed based on well-known control theory and fuzzy concept. Since the structure of fuzzy gain scheduling is modular, it makes the system easier to understand, modify and incorporate expert's knowledge or existing theory.

Chapter 6: Experiments and Results.

Chapter 6 shows several experiments and results to demonstrate the performance of our proposed controller. These experiments explore the following features of our controller.

(1) The accuracy and reliability of our classifier,
(2) The transient and steady-state response analysis of our design,
(3) The robustness of our controller and
(4) The flexibility of our modular structure.

Chapter 7: Conclusions

Chapter 7 concludes this thesis and gives suggestions for the future directions of our research work.

Chapter 2: PID Controllers: A Research Review

2.1. Conventional PID Controller

General applicability to most control systems and the simplicity of its structure are important features of PID controller design. When the plant dynamics are not precisely known, however, analytical design methods cannot be applied in controller design. PID controller with experimental approaches can overcome this limitation. Several experimental methods for determining PID controller parameters had been developed over the past 50 years. Some techniques are based on open-loop step response (i.e., the Coon-Cohen reaction curve method) and other methods use knowledge of the Nyquist curve (i.e., the Ziegler-Nichols frequency-response method) [Ogata02]. The rules of thumb for tuning PID controller are described in Table 2-1 [Jantzen98].

Table 2-1. Rules of thumb for tuning PID controller.

Action	Rise time	Overshoot	Stability
Increase k_p	Faster	Increases	gets worse
Increase k_d	Slower	Decreases	Improves
Increase k_i	Faster	Increases	gets worse

The use of PID control has a long history in control engineering and is acceptable for many real applications. PID controllers play a dominant role in process control and industrial application that demands simple and transparent design procedure. The conventional PID controller structure is described in Figure 2-1. The controller has a proportional term (k_p), an integration term (k_i) and a derivative term (k_d) and the transfer function of this controller is:

$$G_c(s) = k_p + \frac{k_i}{s} + k_d s \tag{2.1}$$

and the equation for the control output in time domain is:

$$u_{pid}(t) = k_p e(t) + k_i \int e(t)dt + k_d \frac{de(t)}{dt} \tag{2.2}$$

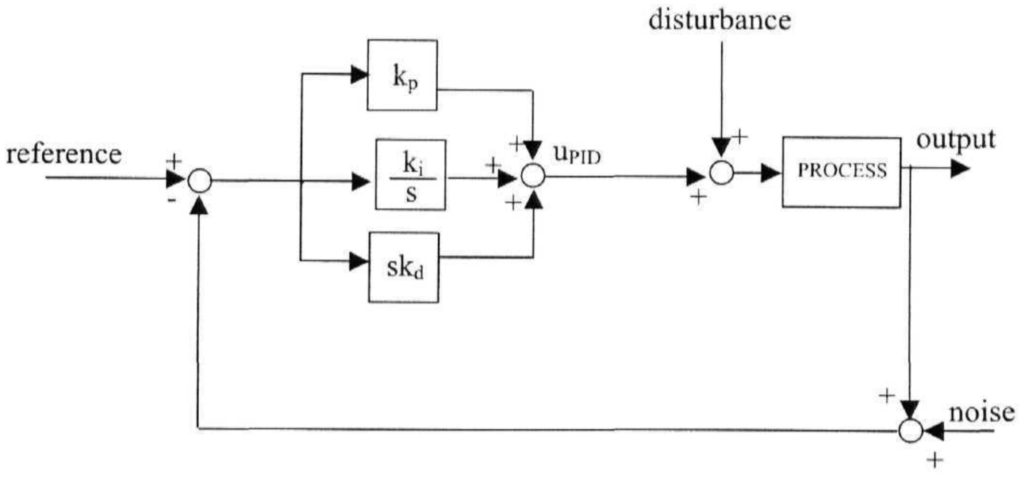


Fig 2-1. The conventional PID controller structure.

Despite their popularity, PID controllers are tuned mostly based on a small amount of information about the dynamic behavior of the system (i.e, critical gain and critical period) and often do not provide good tuning which results in poor control quality. However, among the industrial process controllers, PI control is the most common and practical since it can eliminate the steady-state error. The performance of this kind of controller for higher-order systems, systems with time delay and nonlinear systems may be very poor due to large overshoot and excessive oscillation.

Since the majority of the controller operates on error signal, disturbance and noise effect are sensitive to degrade controller performance. Due to their simple linear structure, they perform well only within the limited region of control that the process nonlinearities can be effectively ignored. Since nonlinearities are intrinsic part of the real world process, there is a need for PID adaptive algorithms that can handle systems with unknown or known nonlinearities without loosing their simple control structure.

Moreover, the appropriate setting of PID controller gains k_p (proportional gain), k_i (integral gain) and k_d (derivative gain) are different based on the nature of process responses. For example, the setting of loop gain depends on control objectives (stability, noise sensitivity and load regulation) [Jantzen98] and control objective depends on the nature of process responses (i.e,., stability and steady-state accuracy are more important in steady-state, performance is more important in transient state). As a result, non-adaptable PID controller is difficult to satisfy all the important requirements of controller design such as stability, sensitivity, steady-state error and transient response characteristic.

In recent years, many different aspects of PID controller design have been developed in order to improve performance of conventional PID controller and most works relied on artificial intelligence based technology. In the literature, various types of fuzzy PID controller have been proposed. In general, the application of fuzzy logic to PID controller design can be classified into three major categories: direct action type (PID-like fuzzy logic controller), gain scheduling type (fuzzy inference system is used for gain scheduling) and combined type (compensated fuzzy PID controller). This chapter explores the nature of PID controller in frequency domain and next reviews several techniques of PID controller design such as Modified PID controller schemes, PID controllers with compensation, auto-tuning PID controllers, gain scheduling PID controllers and PID-like artificial intelligence based controllers.

2.2. The Nature of PID Controller in Frequency Domain

The nature of conventional PID controller in frequency domain is discussed in this sub-section. The transfer function of PID controller that is described in Equation 2.1 can be expressed as:

$$G_c(s) = \frac{k_d s^2 + k_p s + k_i}{s} \quad (2.3)$$

A PID controller introduces a transfer function with one pole at the origin and two zeros that can be located anywhere in the left-hand plane (Equation 2.3). PID controller is a special case of lag-lead control with the lag pole at the origin and the lead pole migrating to infinity. Therefore we can analyze the nature of PID controller in frequency domain by observing lag-lead compensation. Figure 2-2 shows the relation of PID controller gains and the frequency region. The effectiveness of integral gain is in low frequency region, proportional gain affects to mid frequency range and derivative gain is for high frequency region.

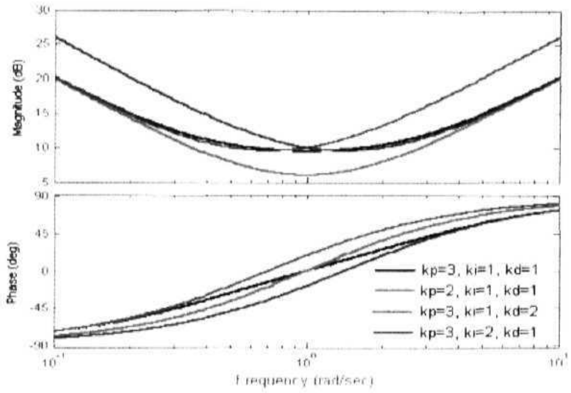


Fig. 2-2. Bode diagram of magnitude plot and phase plot for various PID controller's gains.

Lead compensation (PD controller) essentially yields an improvement in transient response and a small change in steady-state error since it mostly affects high frequency [Dorf04]. It may accentuate high frequency noise effects. From the frequency characteristic of derivative element it can be seen that it has phase lead of 90°. The primary function of the lead compensator (PD) is to reshape the frequency response curve to provide sufficient phase-lead angle to offset the excessive phase lag associated with the components of the fixed system. The addition of the lead compensator shifts the gain cross over frequency to the right and increases in bandwidth, but decrease the phase margin.

The primary function of a lag compensator (PI) is to provide attenuation in the high frequency range to give a system sufficient phase margin and suppresses the effects of high frequency noise signals [Dorf04]. PI controller can be considered as a low pass filter. Lag compensation permits a high gain at low frequencies (which improves the steady-state performance) and reduces gain in the higher critical range of frequencies so as to improve the phase margin. The attenuation due to the lag compensator will shift the gain crossover frequency to a low frequency point. Referring to Figure 2-2, a lower frequency point can be shifted by decreasing the loop gain. Thus, the lag compensator will reduce the bandwidth of the system and will result in slower transient response.

Lag-Lead (PID controller) compensation combines the characteristics of both lead compensation and lag compensation. The phase lead portion of the lag-lead compensator (PID) alters the frequency response curve by adding phase lead angle and increasing the phase margin at the gain crossover frequency (if assume gain crossover frequency point is fixed). The phase lag portion provides attenuation near and above the gain crossover frequency and thereby allows an increase of gain at the low frequency range to improve the steady-state performance.

In general, we note that PID controllers are particularly useful for reducing steady-state error (PI action) and improving the transient response (PD action). The PID controller is only a second order controller with integral action and therefore cannot provide more than 90° of phase lead. Because of these factors, we limit the closed loop bandwidth of the system to be no larger than the frequency at which the phase shift of the open loop plant is -180°.

2.3. Modified PID control schemes

The integral action has the potential to increase the system accuracy, but an important aspect of the use of integral term is that it can lead to integral windup. It is a condition that results when integral action saturates a controller without the controller driving the error signal toward zero. It can results in excessive oscillation or even instability.

A common modification in many PID controllers is the "anti-windup" feature of the integral term used in the algorithm. Although Lee [Lee93] tried to solve that problem with resetting factor that is driven by a fuzzy rule base, Rajani and Nikhil [Rajani01] showed that it is only suitable for some processes, but sometimes it can lead to steady-state error. This method destroys PI important ability to eliminate the steady-state error.

There is a limitation of standard backstepping method, which can also lead to a PD controller for such models. In order to overcome this limitation, Benaskeur and Desbiens [Benaskeur02] presented the adaptive backstepping-based PID that incorporates integral action within the definition of the first backstepping error. Ranger and Desbiens [Ranger03] removed some drawback of Benaskeur [Benaskeur02]'s proposed work (reducing one step in the design of the backstepping controller and eliminating the restriction of placing the regulation poles in the non-adaptive case).

The asymptotic stability domain in the space of feedback gains of the PID controller is usually less than that of the PD controller, but the accuracy of the PID controller is better than the accuracy of the PD controller. Changing the length of integration interval one can derive variety of different PID controller structure, which supports to improve performance [Radaideh02]. In PID controller, the integration is performed over an interval of length t, from the initial time 0 to the current time t. For $\sigma = 0$ and $\sigma = t$ (σ is an integration interval), the structure of PI_{σ} D is changed into the traditional PD and PID controllers, respectively.

If the reference input is a step function, the manipulated variable will involve an impulse function and such a phenomenon is called set-point kick. To avoid set-point kick phenomenon derivative action is placed in feedback path so that differentiation occurs only on the feedback signal and not on the reference signal. Furthermore, differentiating is always sensitive to noise. Therefore, a practical controller with derivative action is necessary to limit the high frequency gain of the derivative term. This problem can be avoided by filtering the reference value before feeding it to the controller [Dorf04].

Similarly, it may also be advantageous to move the proportional action to the feedback path so that it affects the feedback signal only. Thus, in I-PD control [Ogata02], it is imperative to have the integral control action for proper operation of the control system. Another possibility is to let proportional action act only on part of the reference signal and this is called set point weighting. The integral term is based on error feedback to ensure the desired steady-state and then Equation 2.2 has become following form (Equation 2.4).

$$u(t) = k_p(\beta r(t) - y(t)) + k_i \int e(t)d(t) + k_d(\alpha \frac{dr(t)}{dt} - \frac{dy(t)}{d(t)}) \qquad (2.4)$$

where β and α are additional weighting parameters. The overshoot is also reduced by the set-point weighting parameter β. Astrom and Hagglund [Astrom85] used only one weighting parameter β and assumed that $\alpha = 0$ and the calculation of parameter β was based on the approximate pole placement method. This type of modified PID controller structure is described in Figure 2-3. In this Figure, filter is usually used to filter out high frequency components from the controller output in order to spare actuator from unwarranted action.

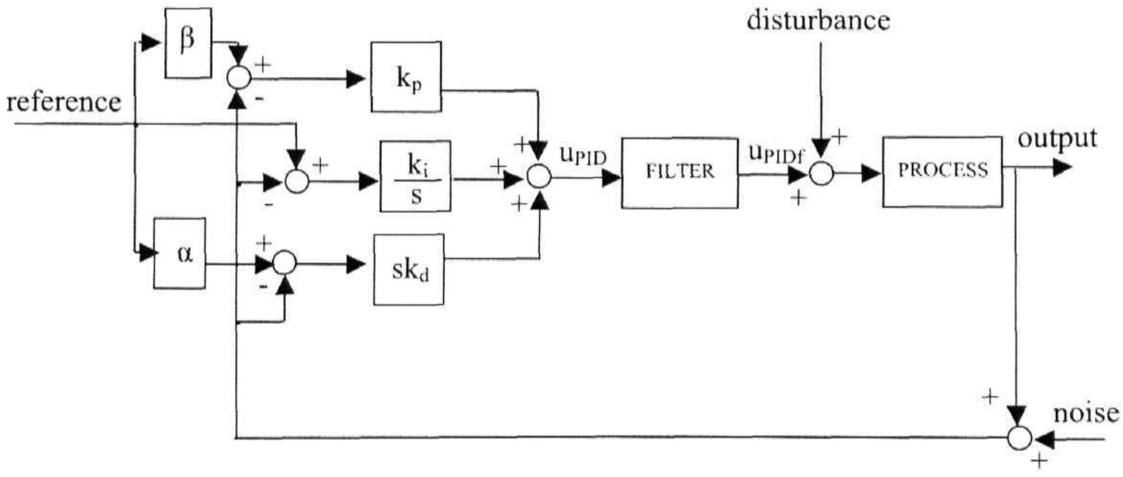


Fig 2-3. A type of modified form of PID controller structure.

In [Henriques99], hierarchical control strategy was used to combine multiple linear PID controllers to apply in the distributed collector field of a solar power plant. To guarantee good performances in all operating points, a local PID controller was tuned to each operating point with the help of the combination of a dynamic recurrent neural network model and a pole placement control design. A fuzzy logic supervisory strategy was applied to switch among these controllers accordingly to the actual measured conditions. The c-mean clustering technique was used to reduce the number of local controllers to be selected by the supervisor.

2.3.1. Two-Degrees-of-Freedom PID controllers

In practical cases, there may be one requirement on the response to disturbance input and another requirement on the response to reference input. Often these two requirements conflict each other and it is very difficult to satisfy with the single-degree-of-freedom controller design. In a two-degrees-of-freedom (*2DOF*) PID control system, both the closed-loop characteristics and the feedback characteristics can be adjusted independently to improve the system response performance.

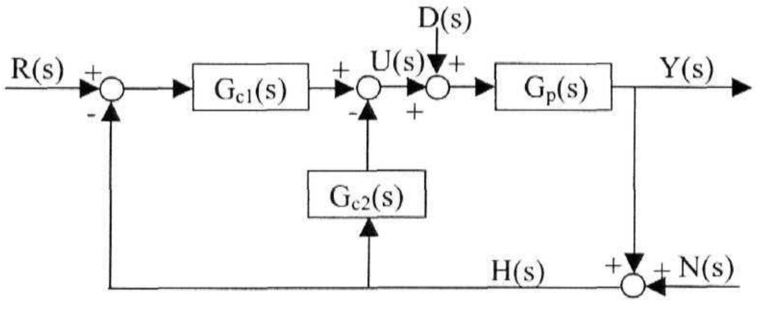


Fig. 2-4. Two-degrees-of-freedom control system (*H(s)* = 1).

The perfect command tracking and disturbance rejection are indicated by *T(s)* = 1 (*T(s)* is the transfer function of closed-loop system) and *S(s)* = 0 (*S(s)* is the system sensitivity), respectively. It is called the *2DOF* controller, if one can adjust *T(s)* and *S(s)* separately. The example of two-degrees-of-freedom controller is illustrated in Figure 2-4 and the transfer functions of output to reference, output to disturbance and output to noise are described in Equation 2.5 [Ogata02].

$$G_{yr}(s) = \frac{Y(s)}{R(s)} = \frac{G_{c1}(s)G_p(s)}{1+(G_{c1}(s)+G_{c2}(s))G_p(s)}$$

$$G_{yd}(s) = \frac{Y(s)}{D(s)} = \frac{G_p(s)}{1+(G_{c1}(s)+G_{c2}(s))G_p(s)} \tag{2.5}$$

$$G_{yn}(s) = \frac{Y(s)}{N(s)} = \frac{(G_{c1}(s)+G_{c2}(s))G_p(s)}{1+(G_{c1}(s)+G_{c2}(s))G_p(s)}$$

Hence, we have

$$G_{yr}(s) = G_{c1}(s)G_{yd}(s)$$

$$G_{yn}(s) = \frac{G_{yd}(s)-G_p(s)}{G_p(s)} \tag{2.6}$$

In Equation 2.6, if G_{yd} is given, then G_{yn} is fixed, but G_{yr} is not fixed, because G_{c1} is independent of G_{yd}. Thus, two closed-loop transfer functions among three closed-loop transfer functions G_{yr}, G_{yd} and G_{yn} are independent.

The I-PD structure with set-point weighting can also be considered as the two-degrees-of-freedom controller because the transfer function of output to input is different from the transfer function of reference to input. The variation of parameter β only affects the input signal of proportional gain and it is primarily responsible for set-point tracking performance. The parameter α is normally zero to avoid large transients in the control signal due to sudden changes in the set point [Astrom85].

A robust two-degree-of-freedom control design was presented by Robert and Zhiquiang [Robert04] based on the active disturbance rejection control. The active disturbance rejection control was originally proposed as a nonlinear solution to industrial control problems. The effectiveness of the active disturbance rejection control is to estimate and cancel disturbance effect in real time. In their work, the active disturbance rejection control was in feedback path and it produced some correctness action based on estimation. This method is suitable for parameter variation with known structure and the structure of process to be controlled must be known in advance.

A proportional-integral (PI) controller has been used to compensate the effects of deadtime and disturbance without degrading tracking performance in two-degrees-of-freedom controller [Hur00]. Increasing the gains of the PI controller that is in feedback path enhances the disturbance-rejection property, but the adjustment of PI controller gain does not affect the transfer function between the current command and its output.

2.3.2. PID controllers with compensation

Li et al. [Li99] introduced a fuzzy P+ID controller based on the concept of the proportional term, which plays an important role in improving overshoot and rise time response. It was constructed by using an incremental fuzzy logic controller in place of the proportional term in a conventional PID controller as shown in Figure 2-5(b). It has only one additional parameter to be adjusted based on the original P+ID controller. This technique is similar to set-point weighting scheme by modifying the set point only in proportional term. The difference between the set-point weighting scheme and the fuzzy P+ID structure lies in their linear and nonlinear property.

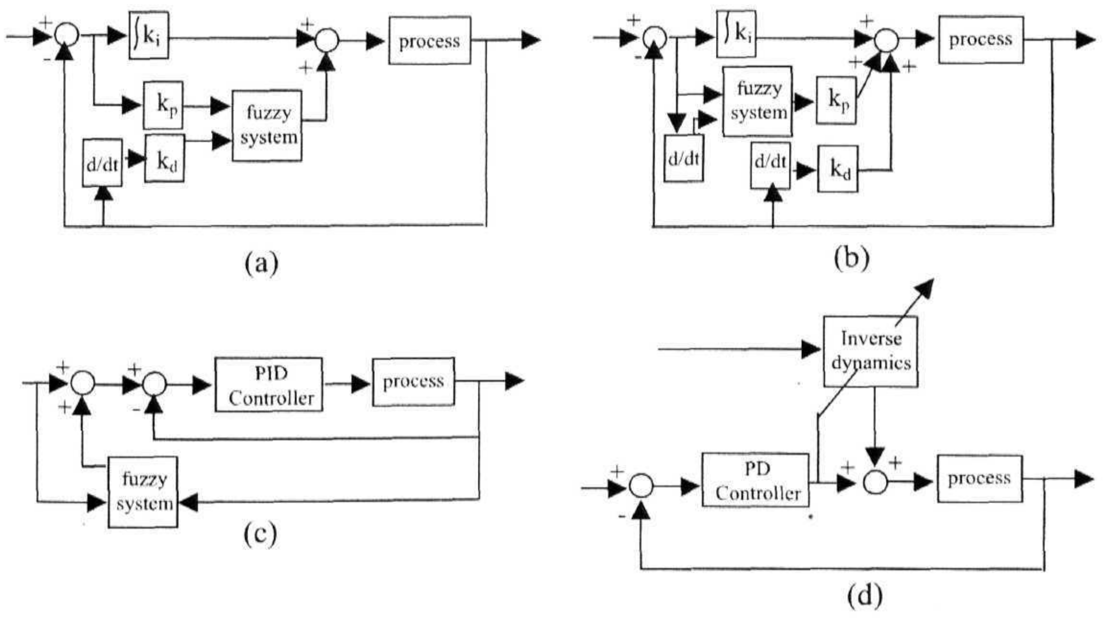


Fig. 2-5. The various structures of PID controller with compensation.

The transient response is affected mostly by the proportional signal and the derivative signal is the key idea behind Kim and Chung [Kim02] proposed fuzzy PD+I controller design. It is illustrated in Figure 2-5(a). The proportional and derivative gains vary with the output of the system under control. The proportional and derivative part of the conventional linear PID controller is modified by the two-input fuzzy system. The weights of the proportional and derivative actions vary with the tracking error. The nonlinearity emphasizes the proportional gain when the tracking error was relatively large and accelerates decreasing speed of tracking error. The nonlinearity in the derivative gain suppresses overshoot and increased damping from the beginning of the settling. Their simulation results showed that this structure enhanced the tracking performance over a wide range of input and parameter variations.

To eliminate heavy and/or unbalanced load effects is the important requirement in motor applications. In [Shieh98], the proposed strategy was intended to improve the performance of the original control system by using fuzzy logic controller. In this approach fuzzy logic controller was used in a supplementary role to enhance the existing control system when the control conditions change. The fuzzy logic controller produces appropriate different control action based on load effect that is observed by error and change in error value. Figure 2-5(c) represents this structure.

An adaptive neural network compensator can be used to compensate for the control error of the preinstalled conventional control systems. In Choi et al. [Choi01] work, the reference input feeds into the adaptive neural network compensator and the output of the compensator generates a new command input to the conventional control system. The control error between the reference input and actual output of the system is used to adjust the weights of the adaptive neural network compensator to eliminate the nonlinearities and other unwanted effects. The structure is almost the same as Figure 2-5(c), but apart from this, it employs neural network instead of fuzzy logic controller. The result showed that their proposed *NN*-compensated control system is adaptable to the environment change and is less sensitive to the disturbance than the conventional PID controller.

Most of today's robots and machine tools are controlled with simple linear-axis controllers, such as PD feedback controllers. The tracking performance that can be achieved with such linear controllers is however limited, especially when the system has highly nonlinear dynamics and/or when the velocities are high. Honegger et al. [Honegger00] presented the application of a nonlinear adaptive controller to a *6* degrees-of-freedom parallel manipulator. It consisted mainly of a feedforward part with the inverse dynamics and nonlinear feedback loop. In their design, the feedback PD controller is used only to correct unmodelled effects. It is illustrated in Figure 2-5(d).

A neural network based simple and effective method is used for trajectory tracking of robotic manipulator [Wu96]. It is the same structure as shown in Figure 2-5(d). The first step was to employ a neural network to learn the characteristics of the inverse dynamics of the robotic manipulator in an offline manner. Then the network is placed in series with the robotic manipulator for online operation and the desired trajectory is fed into the neural network to obtain the corresponding torque. Since it is difficult to get sufficient training patterns of the neural network to excite all modes of the system, a classical PID controller is added for compensation.

2.3.3. Auto-tuning techniques for PID controller

PID controllers need to be regularly retuned because the plant to be controlled is time varying or there could be component aging or changes in process operation conditions. Consequently, auto-tuning becomes one essential feature in PID control system. After the widespread application of microprocessors, a variety of auto-tuning techniques have been developed in PID control applications and fuzzy logic system dominates most of the works.

Bandyopadhyay et al. [Bandyopadhyay01] presented a fuzzy-genetic approach auto-tuning technique for PID controller. They developed the algorithm with the objective of achieving dead-beat control and the controller parameters are tuned in such a way that the error at the next sampling instant becomes zero. In their algorithm, only the future value of controller output is unknown and Takagi-Sugeno model fuzzy inference system has been used to predict it. The rule base of fuzzy inference system is derived with the help of a genetic algorithm. In their approach, rule base of fuzzy inference system is initialized with experts' knowledge. Then their proposed genetic algorithm provides to refine/check the consistency and correctness of the rule base.

The auto-tuning neurons are responsible for finding gains of multivariable PID controller in Chang et al. [Chang03]. There are two adjustable parameters in the activation function of each auto-tuning neuron. A modified hyperbolic tangent function is used as the activation of an auto-tuning neuron. The magnitude and shape of activation function is changed by the variation of these two adjustable parameters. The gains of multivariable PID controller are tuned online by using corresponding auto-tuning neurons and the required parameters of the auto-tuning neurons are updated based on the steepest decent method.

A method for auto-tuning *SISO* and *MIMO* PID fuzzy logic controller is explored in [Almeida02]. The fuzzy auto-tune procedure adjusts online the parameters of a conventional PID controller located in the forward loop of the process. Fuzzy rules are developed with the human expertise's operating knowledge of the process. The method is based on gain margin and phase margin specification and needs system identification under relay experiments. In both *SISO* and *MIMO* cases, gain and phase margins are determined by a set of Mandani rules. The required membership functions of the fuzzy sets are based on the system error and its difference.

Lee and Teng [Lee02] presented a PID tuning method for unstable processes using an adaptive-network-based-fuzzy-inference system (*ANFIS*) for given gain and phase margin specification. In this paper, the *ANFIS* is adopted to identify the relationship between the gain-phase margin specifications and the PID controller parameters. Then, it is used to automatically tune the PID controller parameters for different gain and phase margin specifications. It can estimate the stabilizing region of PID controller parameters and valid region for gain-phase margin. There are two advantages to use the *ANFIS* for formulating gain and phase margin problems. First, the trained network automatically tunes the PID controller parameters for different gain and phase margin specification so that neither numerical methods nor graphical methods need to be used. Second, the *ANFIS* can also find the relationship between PID controllers' parameters and specifications (gain-phase margin) in the weight parameters in the networks.

2.3.4. PID controllers with gain scheduling

Zhao et al. [Zhao93] introduced a fuzzy gain scheduling scheme of PID controllers for process control. Fuzzy rules and reasoning are utilized online to determine the controller parameters based on the error signals and first difference. In this approach, a rule-based scheme for gain scheduling of PID controllers are developed based on human knowledge and experience in control system design. Their simulation results are based on set point tracking and it is noticed that some rules mismatch with disturbance rejection criteria.

Blanchett et al. [Blanchett00] also developed PID gains scheduling using fuzzy logic. Their proposed fuzzy gain scheduling has only one fuzzy input variable and is proportional to the derivative of the conventional PID manipulated variable. They present an explicit control formula with two variables. Online control improvement is achieved by the independent tuning of these two parameters, while the previously tuned conventional PID parameters are retained.

In Viljamaa and Koivo proposed fuzzy logic gain scheduling strategy [Viljamaa95], the tunings and the operating points are collected into a fuzzy rule base. The plant output and the reference signal are used as scheduling variables and the fuzzy reasoning is based on the plant output, the reference signal and a table of the offline tuned PID parameters. The controller changed its parameters online based on the current operating point and table. This technique needs to tune offline at different operating points with the help of *IMC* tuning rules in advance.

2.4. PID-like Artificial Intelligence Based Controllers

In the literature, various types of PID-like artificial intelligence based controllers (most of them are based on fuzzy logic) have been proposed. A typical fuzzy logic control is constructed as a set of heuristic control rules and the control signal is directly deduced from the knowledge base. From the input-output relationship point of view, their structures are almost equivalent to the conventional PID controller.

The major advantage of this technology over the conventional PID control technology is its capability of capturing and utilizing qualitative human experience and knowledge in a quantitative manner through the use of fuzzy logic. Moreover, this is the simple way of developing nonlinear controller because fuzzy controllers are inherently nonlinear in nature.

Initially, the PID-FLC structures were designed considering three terms as inputs. Obviously, the rule base of the fuzzy control is three-dimensional, which makes it difficult to obtain since *3-D* information is usually beyond the sensing capability of a human expert. By combining the use of an incremental PI algorithm and a positional PD algorithm, a PID fuzzy control strategy can be implemented simply from two input variables. This idea has been developed basically in two ways, a parallel combination and a hybrid combination [Mann99, Ambrosio02].

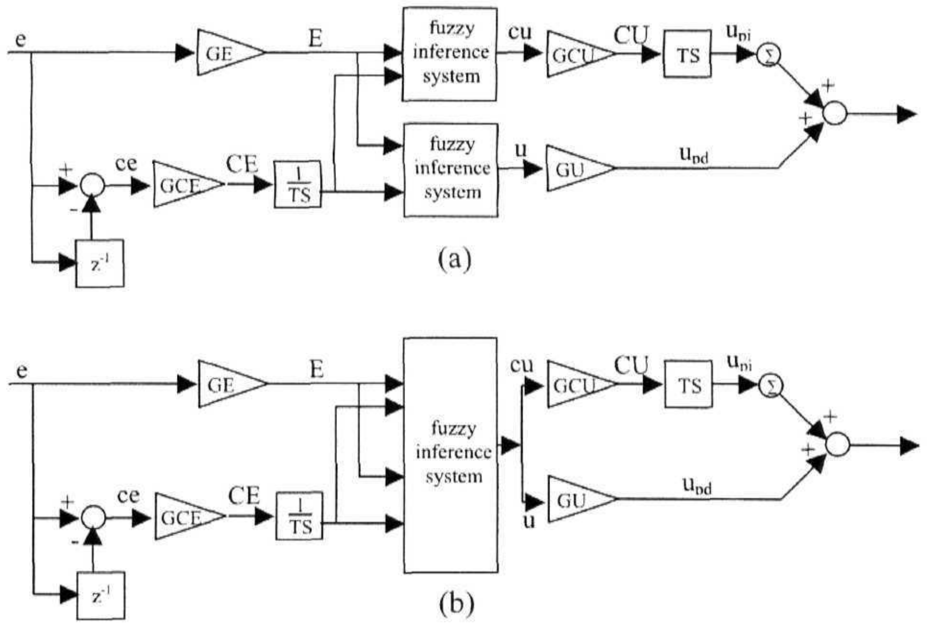


Fig. 2-6. PID-FLC controller structures (a) parallel combination ($u \neq cu$), (b) hybrid combination ($u = cu$).

Figure 2-6 expresses the basic structures of these controllers. The output of the PID-FLC controller can be expressed as the following Equation 2.7 [Ambrosio02].

$$u_{pid} = u_{pi} + u_{pd}$$

$$u_{pid} = GCU \times Ts \times \sum_{i=0}^{k} cu_i + u_k \times GU\ , \qquad (2.7)$$

and the continuous form (for hybrid structure $u = cu$) is

$$u_{pid} = GCU \times \int cudt + GU \times u\ , \qquad (2.8)$$

Although PID-FLC controller has several advantages over conventional PID controller, there are no systematic way to analytically analyze and design of fuzzy control system. It has no clear relation within fuzzy scaling factors and process responses. On the other hand, the linear PID controllers are easy to implement, and sufficient tuning rules are available to cover wider range of process specifications. Moreover, the available PID tuning heuristics are easy to understand and implement for practical control problems. Therefore, some works focus on the construction of an explicit link between the scaling factors of PID type FLC and the three actions of conventional PID control.

$$\begin{aligned} k_p &= GCU * GCE + GU * GE \\ k_i &= GCU * GE \qquad (2.9) \\ k_d &= GU * GCE \end{aligned}$$

Table 2-2. Relationship between fuzzy scaling factors and PID gains (*GIE* is linear integral gain, $T_i = k_p/k_i$, $T_d = k_d/k_p$).

Controller type	k_p	$1/T_i$	T_d
FP	*GE*GU*	-	-
FInc	*GCE*GCU*	*GE/GCE*	-
FPD	*GE*GU*	-	*GCE/GE*
FPD+I	*GE*GU*	*GIE/GE*	*GCE/GE*

Equation 2.9 expresses the relationship between the gains of conventional PID controller and the scaling factors of a modified hybrid PID-FLC that was developed by Ambrosio and Mort [Ambrosio02] and Table 2-2 reveals that the Jantzen's development [Jantzen98]. In some other approach [Guzelkaya03], the relation of conventional PID parameters and fuzzy scaling factors were expressed as the following Equation 2.10.

$$k_p = GU \times GE \times P + GCU \times GCE \times D,$$
$$k_i = GCU \times GE \times P, \tag{2.10}$$
$$k_d = GU \times GCE \times D,$$

where P and D are the variables of fuzzy inference system. In case the variables P and D approach to one, it is the same as Equation 2.9. The Equation 2.9 is derived from Tagaki-Sugeno fuzzy inference system and Equation 2.10 is based on the fuzzy inference system with product-sum inference method, center of gravity defuzzification method and triangular uniformly distributed membership function for the inputs and crisp output.

Some self tuning mechanisms have been proposed in literature in order to improve the performance of PID-like fuzzy logic controller. Parameter adaptive method for PID-type fuzzy logic controller based on a peak observer has been proposed in [Qiao96]. The peak observer keeps watch on the process response and sends a signal at each peak time and measures the absolute peak value. The parameter regulator tunes the controller parameter *GCE* and *GCU* simultaneously at each peak time according to the peak value. The algorithm for tuning these parameters is as follows:

$$GCE = \frac{GCE_s}{\delta_k},$$
$$GCU = \delta_k \times GCU_s, \tag{2.11}$$

where GCE_s and GCU_s are initial values of *GCE* and *GCU*, δ_k is the peak value of process response in k^{th} sampling time. In Woo et al. [Woo00], the parameter adaptation is based on the following functions.

$$f(e(t)) = a_1 \times abs(e(t)) + a_2, \tag{2.12a}$$
$$g(e(t)) = b_1 \times (1 - abs(e(t))) + b_2, \tag{2.12b}$$

and the adaptation law becomes

$$GCE = GCE_s \times f(e(t)), \tag{2.13a}$$

$$GCU = GCU_s \times g(e(t)), \tag{2.13b}$$

In the Equations 2.12 and 2.13, a_1, a_2, b_1 and b_2 are parameters to be tuned. The objective of function $f(e(t))$ (Equations 2.12a and 2.13a) is to decrease *GCU* with the change of error. In steady-state, the value of function $f(e(t))$ is in its minimum value (i.e., $f(e(t)) = a_2$). The function $g(e(t))$ (Equations 2.12b and 2.13b) has inverse objective and it will be equal to $b_1 + b_2$ in steady-state.

In Guzelkaya and Yesil proposed idea [Guzelkaya03], the tuning strategy depends not only error signal but also on normalized acceleration. The normalized acceleration gives relative rate information about the system response. Therefore their self-tuning mechanism is based on relative rate observed and tuning algorithm can be described as the Equation 2.14.

$$GCE = GCE_s \times k_{fd} \times k_f \times \gamma,$$
$$GCU = \frac{GCU_s}{k_f \gamma} \tag{2.14}$$

where k_f is the output scaling factor for the fuzzy parameter regulator block, k_{fd} is the additional parameter that affects only the derivative factor of the PID-like fuzzy logic controller. In their approach, the required proportional control strength can be set based on the parameter k_{fd}. The fuzzy parameter regulator produces the variable γ based on these following concepts:

- When the system response is slow, the derivative effect of the PID-like fuzzy logic controller must decrease.
- When the error is small and system response is fast, the derivative effect of the PID-like fuzzy logic controller must increase.

Rajani and Nikhil [Rajani00] introduced a self-tuning fuzzy PI controller and PI-like fuzzy logic controller (i.e., it includes only fuzzy PI part in our Figure 2-6). The output scaling factor (*GCU*) of the controller was modified online by updating factor. The proposed self-tuning mechanism fuzzy inference system produced the appropriate value of updating factor based on error and change in error. The required fuzzy rule base is developed based on the task of a skill operator who can manipulate the process to be controlled properly. The background idea behind their rule based can be summarized as following.

- To achieve a lower overshoot and reduce settling time but not at the cost of increased rise time the controller gain should be set at a small value when the error is very big.
- Depending on the situation, gain around the set point should be allowed to vary widely to prevent large overshoot and undershoot.
- To improve the control performance under load disturbance, gain around the steady-state condition is made sufficiently large. But at steady-state gain should be very small.

Chen and Linkens [Chen98] presented a hybrid neuro-fuzzy control strategy and its corresponding rule generating technique. There are two features in their approach. First one is the required fuzzy control rules are generated in accordance with linear conventional PID control theory and the second is single neuron based online adaptation characteristic. The relationship between the input and output of the neuron is described as

$$u_{pid} = w_1 u_{pi} + w_2 u_{pd} \tag{2.15}$$

where w_1 and w_2 are adjustable parameters for PI-like fuzzy controller output and PD-like fuzzy controller output respectively and these parameters are tuned based on well-known back propagation algorithm.

In the work [Li00], a fuzzy PID controller is investigated on the basis of genetic algorithms to eliminate the undershoot of a non-minimum phase system and to improve the transient response at the same time. The proposed controller employs a two-stage control structure (i.e., it includes two independent controller) and the block diagram is described in Figure 2-7. The fuzzy PI controller is used to cancel out the effect of unstable zeros and the fuzzy PD controller is introduced for the reason of reducing overshoot. The required fuzzy control rules for both controllers are evolved and determined by the genetic algorithm and the nature of non-minimum phase processes. The final controller output is described as

$$u_{pid} = k_{pi} \times u_{pi} + k_{pd} \times u_{pd} \tag{2.16}$$

where u_{pi} represents fuzzy PI controller, u_{pd} is the output of fuzzy PD controller and the combination of coefficients k_{pi} and k_{pd} are determined by the following Equation 2.17.

$$k_{pi} = \begin{cases} 1, & |e| \geq 0.9 \\ 2 - 20y, & 0.8 \leq |e| < 0.9 \\ 0, & |e| < 0.8 \end{cases} \qquad (2.17)$$

where $k_{pd} = 1 - k_{pi}$ and y is the output of process. Cho et al. [Cho97] also developed the automatic rule generating method based on genetic algorithm.

An important feature of decomposed PID-like fuzzy logic controller [Marjan01] is their simple structure. Moreover, it has a direct relationship between fuzzy control and conventional control. It consists of three one-input/one-output inferences with three separate rule bases. The basic idea is to decompose the multivariable control rule base into three sets of one-dimensional rule bases for each input (error, change-of-error and integral-of-error). Referring to their results, the performance of hybrid PID-FLC (Fig. 2-6(b)) is better than their proposed decomposed PID-like fuzzy logic controller.

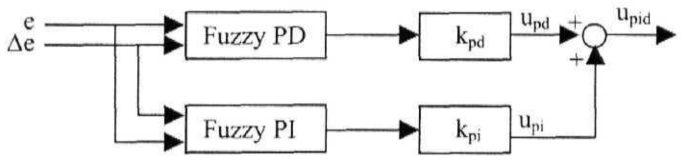


Fig. 2-7. The block diagram of two stages fuzzy PID controller.

Shu and Pi developed PID neural network (*PIDNN*) [Shu00] that consists of three layers. The input layer has two neurons (reference value and output value), there are three neurons (proportional action, integral action and derivative action) in hidden layer and one neuron (controller output) in output layer. *PIDNN*'s weights are adjusted by back propagation algorithm and their results show it has abilities to control different time delay systems.

2.5. Discussions

Over the past 50 years, several methods of PID control system have been developed based on different concept. However, PID controller has some advantages such as robustness, design simplicity and a clear relation between PID controller's gains and system response parameters. It has some limitation to implement high performance controller design for the processes with highly nonlinear or uncertainty parameters or unmodelled delay.

It is important to note that, depending on the behavior of process operating condition or the different objective, the appropriate setting of PID controller's gains are different or sometimes different structures are needed (PID or PI or PD). In order to fulfill this requirement, online auto-tuning PID controllers and gain scheduling PID controllers have been developed with the help of artificial intelligence based technology especially in fuzzy logic.

One noticeable point here is that fuzzy controller gradually play an important role in PID controller design and it has been found that the combination of PID and fuzzy produces better quality control. One main reason behind this is that in conventional PID control theory, rules of thumb for tuning rules are already developed in linguistic nature, which is well-suited for fuzzy systems. Most of the fuzzy systems are utilized in supervisory role by observing the process dynamics. Therefore the precise prediction of dynamical behavior online is needed for supervisory fuzzy system.

Furthermore, control theory highlights that the sufficient flexible controller design (modular structure controller or more degrees-of-freedom controller) is needed to implement high performance control system. Generally, two degrees-of-freedom PID controller, non-modular various PID controller structures and non-modular artificial intelligence based PID controller structures are introduced up to present. Although there have been several approaches in the past to evaluate design procedures, until now there was no success in finding universally valid controller design. Consequently, several improvements would still be needed to obtain an easily tunable and robust adaptive PID controller.

Chapter 3: A Proposed Modular Neuro-Fuzzy Control System

3.1. A Practical Control Problem

Practical control problems are usually imprecise. It is merely mapping from a certain input signal to an output signal (desired dynamical behavior of process to be controlled) and can be described as a transfer function of input and output as shown in Figure 3-1. The transfer function that represents the relation of input and output (Box-A in Figure 3-1) may be uncertain and it can be changed by an unknown disturbance or other factors. The flexible variation of the controller (both parameters and structure) is needed in order to establish the appropriate relation between inputs and output based on function of process response. In Figure 3-1, Box-B (adaptive mechanism) is responsible to alter the controller parameters and structure. The required information can be extracted from sensory inputs.

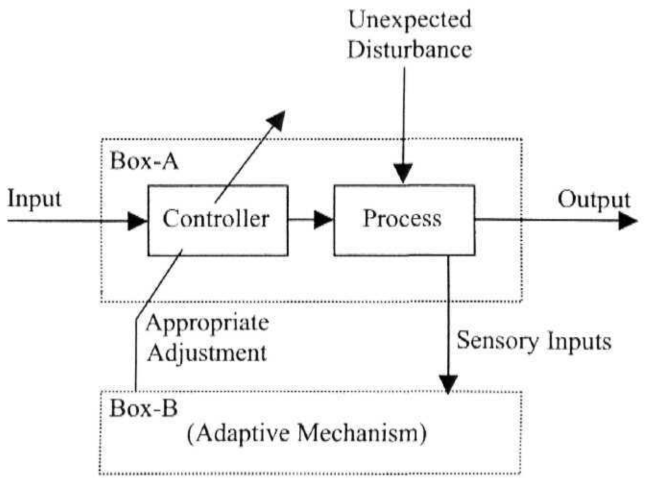


Fig. 3-1. A practical control problem.

The appropriate setting of controller parameters and control resolution (the range of gain variation) are different based on the characteristics of process response and control objectives. For example, set point tracking and disturbance rejection are fundamental requirements in practical control applications. Often these two requirements conflict with each other and a flexible adaptive algorithm is needed in order to satisfy both of them. As a result, a flexible controller design (i.e., a flexible adjustment algorithm) and providing sufficient information of process response are needed to implement high performance controller design. The next section digs into an in-depth analysis of required characteristics of an adaptive controller that form a basis of our proposed model.

3.2. The Analysis of Required Characteristics of Controller in Different Operating Region

The prime capability of a flexible controller design is to react appropriately to different manner of process responses. The diverse tuning strategies (parameter transaction policy) are required for different behavior of process response and modular structure can be regarded as a solution to it. The modular structure supports development of multiple modules that are responsible for different tuning strategies. In order to define each local tuning strategy for different behavior of process response, we analyze both frequency response method and transient/steady-state analysis in time domain that are practical and important approach for the analysis and design of a control system.

Phase margin, gain margin, the gain crossover frequency, the phase crossover frequency and bandwidth provide important insight in the design of high performance controller design under high level of uncertainty [Dorf04]. The block diagram of closed-loop control system as shown in Figure 3-2 is used to analyze these important characteristics.

The transfer function *T(s)* of closed-loop system as shown in Figure 3-2 is:

$$T(s)=\frac{Y(s)}{R(s)}=\frac{G_c(s)G_p(s)}{1+G_c(s)G_p(s)} \qquad (3.1)$$

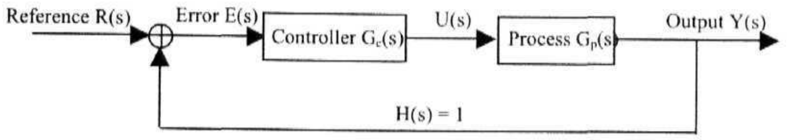


Fig 3-2. Block diagram of closed-loop control system.

3.2.1. Stability analysis in frequency domain

In a controller design, we require that not only the system be stable but it is also necessary that the system has an adequate relative stability. The relative stability can be defined in terms of the gain and phase margin. In frequency domain mid frequency region represents the relative stability.

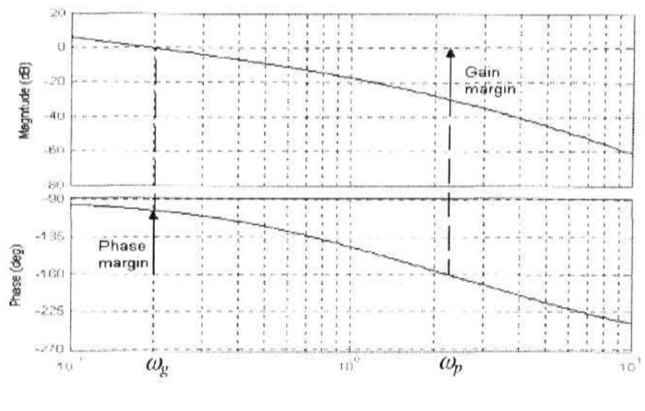


Fig. 3-3. Bode diagram of magnitude plot and phase plot for *T(s)*.

The gain margin (*GM*) is the factor by which the system gain can be increased to drive it to the verge of stability. The phase margin (*PM*) is the amount of additional phase lag at the gain crossover frequency needed to take the system towards instability [Bandyopadhyay03]. The gain and phase margins are easily evaluated from the Bode diagram (Figurer 3-3) and the formulas of gain and phase margins are as given in Equations 3.2 and 3.3.

$$GM = -20\log\frac{1}{\left|G_c(j\omega_p)G_p(j\omega_p)\right|}dB \tag{3.2}$$

$$PM = [\angle G_c(j\omega_g)G_p(j\omega_g)] + \pi \tag{3.3}$$

where ω_p is the phase crossover frequency and ω_g is gain crossover frequency. It is noticed that low loop gain supports to reduce gain crossover frequency and it helps to improve relative stability.

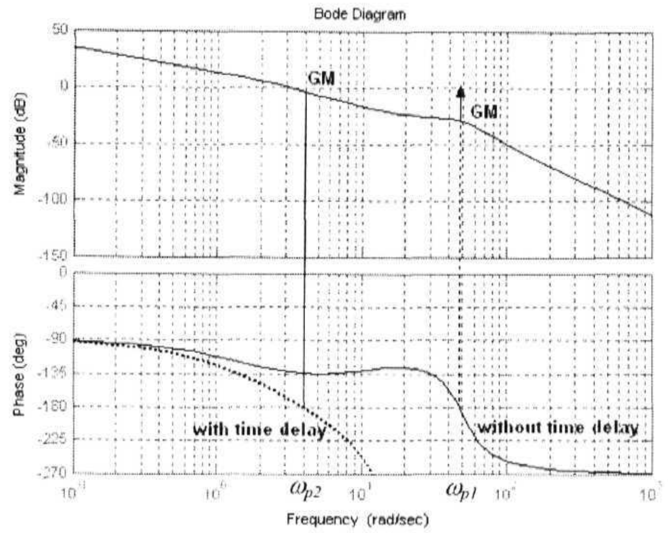


Fig. 3-4. Analyzing the time delay effect.

Many control systems have a time delay within the closed loop of the system and it affects the stability of the system. A time delay in a feedback system introduces an additional phase lag and results in a less stable system. It can be expressed as Equation 3.4 [Dorf04].

$$G_d(s) = e^{-sT} \tag{3.4}$$

Then loop transfer function of Figure 3-2 with a time delay becomes:

$$G(s) = G_c(s)G_p(s)e^{-sT} \tag{3.5}$$

The frequency response of this system is:

$$G(j\omega) = G_c(j\omega)G_p(j\omega)e^{-jwT} \tag{3.6}$$

The delay factor $e^{-j\omega T}$ results in a phase shift *($-\omega T$)* to the frequency response (from ω_{p1} to ω_{p2}) without altering the magnitude curve as shown in Figure 3-4. Therefore low loop gain is required to minimize the effect of delay factor and to obtain reasonably large relative stability.

3.2.2. Sensitivity analysis in frequency domain

The plant model can be an inaccurate representation of the actual physical system because of parameter changes, unmodeled dynamics, unmodeled time delays, change in equilibrium point (operating point), sensor noise and unpredicted disturbance inputs. The goal of robust system design is to keep system response within limits in spite of model inaccuracies and changes.

The system sensitivity is defined as the ratio of the percentage change in the system transfer function to the percentage change of the process transfer function [Ogata02, Dorf04]. The sensitivity *S(s)* of the transfer function that is described in Equation 3.1 is:

$$S(s) = \frac{\Delta T(s)/T(s)}{\Delta G_p(s)/G_p(s)} \tag{3.7}$$

$$S(s) = [1 + G_c(s)G_p(s)]^{-1} \tag{3.8}$$

then,

$$S(s) + T(s) = 1 \tag{3.9}$$

Low sensitivity is an important requirement in robust controller design and it is important to make *S(s)* small in above Equation 3.9. Accordingly, high loop gain is required to minimize the sensitivity. But, low loop gain is necessary to improve stability.

3.2.3. Steady-state error analysis in frequency domain

Any physical control system inherently suffers from steady-state error in response to certain types of inputs. The steady-state error of our example transfer function (Figure 3-2) for a unit step input is:

$$\frac{Y(s) - R(s)}{R(s)} = \frac{G(s) - 1 - G(s)}{1 + G(s)} \tag{3.10}$$

$$\frac{-E(s)}{R(s)} = \frac{-1}{1 + G(s)} \tag{3.11}$$

$$E(s) = \frac{R(s)}{1 + G(s)} \tag{3.12}$$

The steady-state error is evaluated from the final value theorem,

$$e_{ss} = \lim_{s\to 0}\frac{sR(s)}{1+G(s)} \tag{3.13}$$

$$e_{ss} = \lim_{s\to 0}\frac{s}{1+G(s)}\frac{1}{s} \tag{3.14}$$

$$e_{ss} = \frac{1}{1+G(0)} \tag{3.15}$$

The summary of steady-state errors for various kinds of processes is described in Table 3-1 [Bandyopadhyay03].

Table 3-1. The summary of steady-state error analysis.

Number of integrations in process, Type number	Input		
	Step, r(t) = A, R(s)=A/s	Ramp, r(t) = At, R(s) = A/s^2	Parabola, r(t) = $At^2/2$, R(s) = A/s^3
0	$e_{ss} = A/(1+C_p)$	Infinite	Infinite
1	$e_{ss} = 0$	A/C_v	Infinite
2	$e_{ss} = 0$	0	A/C_a

where C_p is a position error constant, C_v is a velocity error constant and C_a is designated the acceleration error constant. The error constants *(C_p, C_v and C_a)* of a control system describe the ability of a system to reduce or eliminate the steady-state error. Consequently, the gain in the low frequency region (k_i) should be large enough to minimize the steady-state error since low frequency indicates the steady-state behavior. But high integral gain (*low frequency gain*) can cause large overshoot and excessive oscillation.

3.2.4. Transient response analysis in frequency domain

Another objective in the design of control system is that the controlled system's output should exactly and instantaneously reproduce its input. That is the system's transfer function should be unity.

$$T(s) = \frac{Y(s)}{R(s)} = 1 \tag{3.16}$$

In other words, the system should be representable on a Bode gain versus frequency diagram with a 0 dB gain of infinite bandwidth and zero phase shift. The range of frequencies over which the *T(jω)* is equal to or greater than 1/√2 (3 dB) is defined as the bandwidth. The speed of response to a step input will be roughly proportional to bandwidth. For the system to follow arbitrary inputs accurately, it must have a large bandwidth. From the viewpoint of noise, the bandwidth should not be too large. Thus, there are conflicting requirements on the bandwidth and a compromise is usually necessary for a good design.

3.2.5. Transient and steady-state analysis in time domain

In many practical cases, the desired performance characteristics of control systems are specified in terms of time-domain quantities. The time response of a control system consists of two parts: transient response and steady-state response as shown in Figure 3-5. Delay time, rise time, maximum overshoot and settling time are important factors for transient response and the accuracy and stability are important in steady state [Ogata02].

The transient response of a practical control system can be decomposed into three parts based on different state: (a) rise time state, (b) overshoot/oscillation state and (c) disturbance state as shown in Figure 3-6. Rise time state is defined as the time required for the response to rise from 0% to 100% of its final value and the performance of the system is more important in this state (high bandwidth). But stability is more important in delay period or near the set point.

The transient response of a practical control system often exhibits a damped oscillation before reaching the steady-state. The intermediate period between rise time state or disturbance state and steady-state is defined as overshoot/oscillation state. It is necessary to reach steady-state as soon as possible and stability is more important (low loop gain and fine control resolution). At the same operating point (same reference value), when the process response diverges from set point (reference point), it can be defined as disturbance state and two different parameters settings are required as shown in Figure 3-6. Based on these observations, some important facts are summarized to develop individual local tuning strategy modules and they are tabulated in Table 3-2.

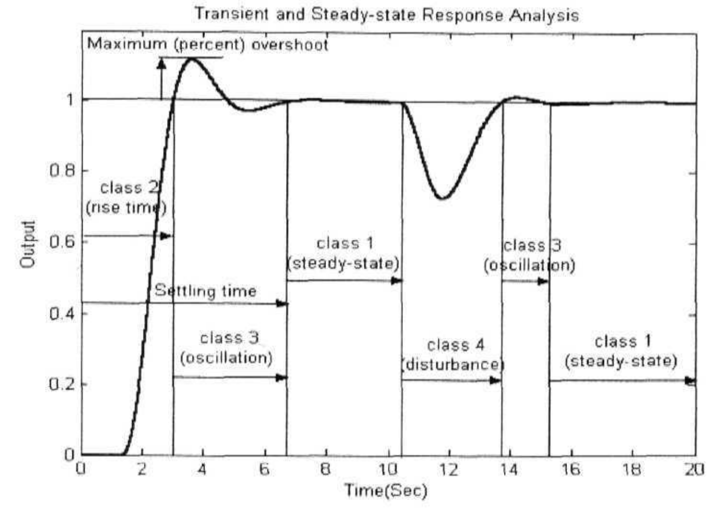


Fig. 3-5. Transient and steady-state response analysis in time domain.

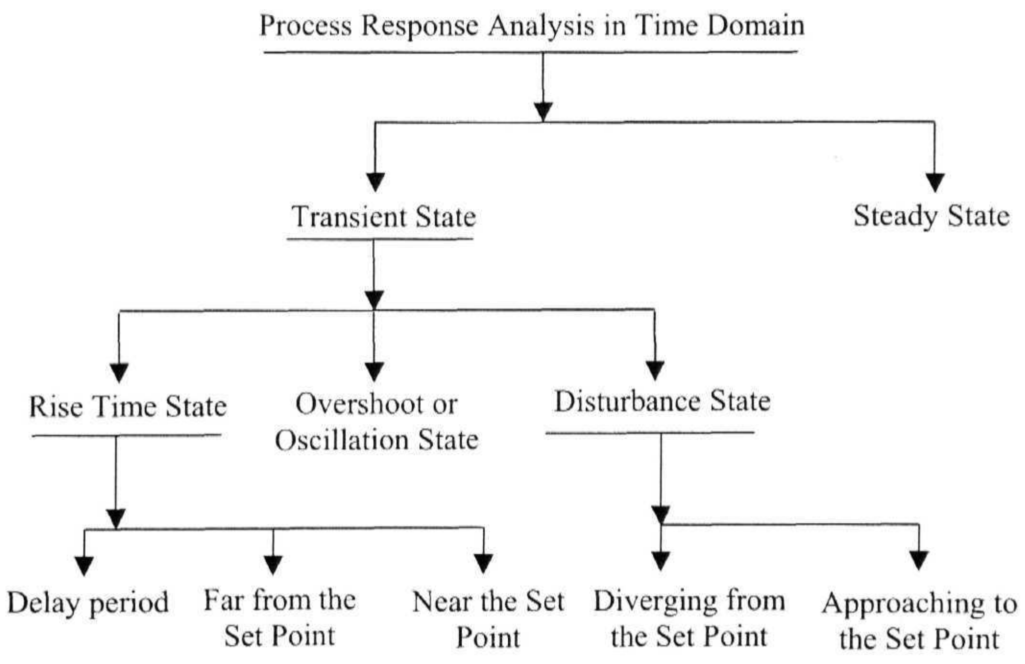

Process Response Analysis in Time Domain
Transient State
Steady State
Rise Time State
Overshoot or Oscillation State
Disturbance State
Delay period
Far from the Set Point
Near the Set Point
Diverging from the Set Point
Approaching to the Set Point

Fig. 3-6. Process response analysis in time domain.

Table 3-2. Summary of required parameters setting and control resolution for different states of process response.

No.	State of Process Response	Appropriate Parameter Setting	Control Resolution	Referred Subsection
1.	Rise time state (Delay period)	Low loop gain	-	3.2.1
2.	Rise time state (Far from the set point)	High loop gain (Especially in mid and high frequency gain (k_p and k_d))	Coarse	3.2.4 & 3.2.5
3.	Rise time state (Near the set point)	Low loop gain (But sufficiently large low frequency gain k_i)	Fine	3.2.3 & 3.2.5
4.	Steady-state	Low loop gain	-	3.2.3 & 3.2.5
5.	Overshoot/Oscillation state	Low loop gain	Fine	3.2.5
6.	Disturbance state (diverging from the set point)	High loop gain (Especially in low frequency gain k_i)	Coarse	3.2.2 & 3.2.5
7.	Disturbance state (approaching to the set point)	High loop gain (Especially in low frequency gain k_i)	Fine	3.2.2 & 3.2.5

Accordingly, it is necessary to decompose the whole process response into four categories (rise time state, overshoot/oscillation state, disturbance state and steady-state) for implementing individual local tuning strategies.

3.3. Brief Review of Proposed Modular fuzzy Gain Scheduling Mechanism

As we discussed in earlier section, it is necessary to define multiple local gain tuning strategies with respect to process response in order to implement flexible controller design. Our proposed modular adaptable mechanism is based on multiple fuzzy inference systems and they are responsible for controlling different regions of process response. Next section discusses briefly the advantages of fuzzy logic.

3.3.1. Fuzzy logic in control applications

L. A. Zadeh introduced the theory of fuzzy sets in 1965 [Zadeh65]. Fuzzy logic posses the ability to mimic the human reasoning and decision-making strategies that are approximate rather than exact. In 1974, E. H. Mamdani [Mamdani74] has developed the first fuzzy controller for industrial applications and since then, fuzzy logic control has evolved as an alternative or complementary to the conventional control strategies in various engineering areas. Mamdani's pioneering work introduced the most common and robust fuzzy reasoning method, called Zadeh-Mamdani min-max gravity reasoning [Mann99].

Fuzzy control departs significantly from conventional control theory. Instead of deriving a controller via modeling the controlled process numerically or mathematically, the fuzzy control methodology tries to build the controller directly from domain experts or operators who are controlling the process manually and successfully. Fuzzy control theory usually provides nonlinear controllers that are capable of performing different complex nonlinear control actions.

Unlike conventional controls, designing a fuzzy logic controller does not require precise knowledge of the system model. The control knowledge can be expressed by a set of linguistic rules with the form of *IF situation THEN action*. Fuzzy rules have been advocated as a key tool for expressing pieces of knowledge in "fuzzy logic" and the performance of the system rests largely on the quality of the rule-base. The form of fuzzy logic controller can be expressed as:

IF a_i is A_l AND b_i is B_l THEN c_i is C_l

$$c_i = FLC(a_i, b_i) \tag{3.17}$$

where a_i and b_i are two fuzzy input variables and c_i is output variable. The advantages of using fuzzy control usually fall into one of the four categories: (i) implementing expert knowledge for a higher degree of automation, (ii) robust nonlinear control, (iii) reduction of development and maintenance time and (iv) marketing and patents [Driankov01].

Fuzzy logic control technique has found many successful industrial applications and demonstrated significant performance improvements. However, fuzzy controller design is insufficient to apply analytical design technique in contrast with the well-developed linear control theories. An often-remarked drawback of the fuzzy logic based methods is the lack of appropriate tools for analyzing the controller performance, such as stability, optimality, robustness, etc.

An attractive alternative way to build fuzzy logic controller design is to utilize fuzzy computing as a supervisory block for a conventional controller. This approach takes full advantage of the combination because fuzzy computing can thus be more easily applied in different control applications and controller performance can be analyzed based on conventional control theory.

Fuzzy gain scheduling is considered to be the most promising alternative combination of fuzzy logic and conventional controllers. The approach means that the values of the controller parameters are gathered to build fuzzy rule base with the operating conditions where they are valid. During the control, the fuzzy gain scheduler provides appropriate values for the controller parameters and fuzzy logic offers smooth transaction between operations.

There are two general sources of information for building fuzzy systems such as prior knowledge and data (measurements). Prior knowledge can be a rather approximate nature that usually originates from experts (i.e., control engineers or process operators). Alternately, data about the process operation can be recorded in many applications. It is also possible to arrange special experiments in order to obtain the relevant data. Building fuzzy systems from data involves two major types of learning: (i) structure learning (to find appropriate fuzzy logic rules) and (ii) parameter learning (to fine-tune the membership function and other parameters). The crucial problem with data-driven techniques is what kind of data must be prepared, how much data is required and how to collect it.

According to conventional control theory based analyses (described in section 3.2) and advantages of fuzzy logic, we implement multiple fuzzy gain scheduling modules that represent individual local tuning strategies.

3.3.2. Defining local tuning strategies

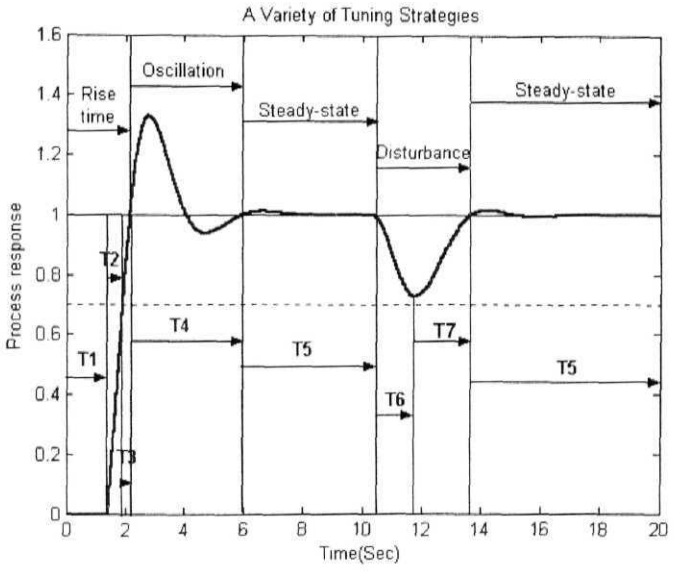


Fig. 3-7. The relation of proposed tuning strategies and state of process response.

As we described in Figure 3-6, we define seven different tuning strategies in our gain scheduling mechanism. Figure 3-7 illustrates the relation of tuning strategies and states of process response. *T1* tuning strategy is responsible for the delay period in rise time state, *T2* is for rise time state (far from set point), *T3* is for rise time state (near set point), *T4* is for overshoot/oscillation state, *T5* is for steady-state and *T6, T7* are for disturbance state respectively. The required gain transaction policies are stored in multiple fuzzy inference systems and detail explanations of implementation of each fuzzy inference system are discussed in Chapter 5.

3.4. The Proposed Switching Algorithm

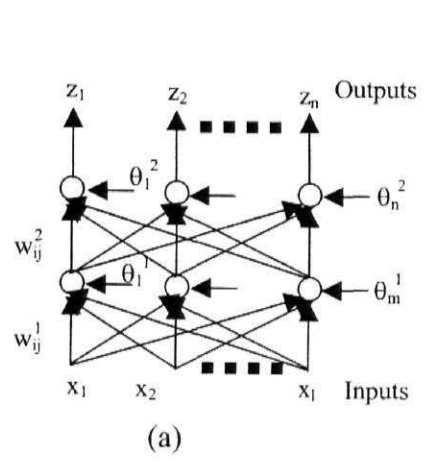
In order to select online appropriate fuzzy inference systems, it is necessary to define switching algorithm. Switching algorithm is developed based on characteristics of process response. Most of the previous process response estimation works [Li95, Li99, Blanchett00] are based on quantitative information (error and its first difference), whereas our proposed switching algorithm is based on both qualitative and quantitative (error and its first difference) information of process response. It is necessary to define the current state of the process response (rise time state or oscillation state or disturbance state or steady-state) and online process response classification becomes an essential task in our controller design. This is modeled by a knowledge-based modular neural network classifier.

3.4.1. Neural network in classification process

Neural networks have been very successfully applied in the identification and control of dynamic systems. The universal approximation capabilities of the multilayer perceptron have made it a popular choice for modeling nonlinear systems and for implementing general-purpose nonlinear controller [Norgaard00]. The neural network has capability to learn with appropriate learning rule and adaptation can be accomplished by this means.

A variety of training rules have been developed for setting the weights. The most popular rule is back propagation algorithm that was originally developed by Paul Werbos. This technique is based on gradient descent in the weight space over an error surface created by the sum-squared error at each output node over the entire training set. A trained neural network, like a human brain, can associate with a large number of output patterns corresponding to each input pattern [Bose00b].

In classifier design, neural network technology has some merits over traditional classification systems with respect to capability of adaptive learning, generalization ability with noisy or sparse learning data [Auda96]. The topology of a neural network can be defined by a directed graph where the vertices correspond to the neurons and the edges indicate the flow of information. The performance of a neural network depends on both its topology and weights that are associated with its edges.



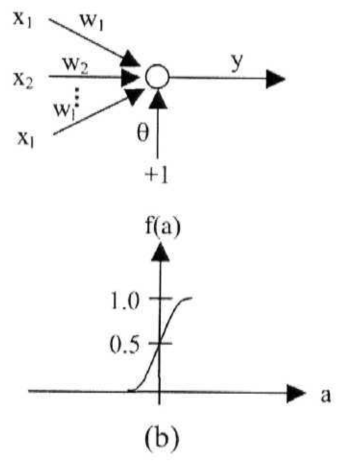


Fig. 3-8. Neural network classifier.

$$y = f(\textstyle\sum_{i=1}^{L} w_i x_i + \theta) \tag{3.18a}$$

$$z_j = f_2(\sum_{i=1}^{m}\sum_{j=1}^{n} f_1(\sum_{i=1}^{l}\sum_{j=1}^{m} w_{ij}^1 x_i + \theta_j^1) w_{ij}^2 + \theta_j^2) \tag{3.18b}$$

Figure 3-8 illustrates the simple neural network classifier system. The input to this network (Figure 3-8(a)) is the feature vector extracted from the object to be classified, and the output is typically a block code where one output is high that indicates the class of objective and all others outputs are low. Each node in the network performs a simple function as shown in Figure 3-8(b). The output of each node is expressed in Equation 3.18a and Equation 3.18b represents overall output. The weights connecting the nodes are determined using some training rule with the training set.

Theoretically, a feedforward multilayer network with as few as one hidden layer is capable of approximating any function with any desired degree of accuracy. But determining the sufficient number of hidden units is an inherently unsolved problem. There are no general criteria for choosing the network's size (number of hidden nodes) since the parameters of each application demands different network capabilities [Higgins93]. If the network has fewer neurons than necessary, the learning algorithm will fail to find the desired classification function. If the network has more neurons than necessary, it can result in overfitting of the training data leading to a poor generalization.

When prior knowledge is rich enough to form an approximately complete and correct domain theory, knowledge compilation can be used to build knowledge-based neural networks. Knowledge-based neural networks are those whose topology is determined by mapping the domain-specific rule base into a neural network [Fu95]. The rules-to-network algorithm conceived by Towell [Towell94] was an important step in the development of knowledge-based artificial neural network.

On the other hand, current generation of non-modular neural network classifiers is unable to cope with classification problems that have a wide range of overlap among classes. This is due to the high coupling among the networks' hidden nodes. Usually, there are regions in the class feature-space, which show overlap due to the resemblance of two or more classes. Meanwhile, there are other regions, which show little or even no overlap, according to the uniqueness of the classes therein.

The modular neural network classifier's approach is to decompose the classification task into simpler sub-tasks. Then, sub-solutions are integrated via a multi-module decision-making strategy [Auda96]. It can reduce the effect of conflicting training information (crosstalk) or high coupling among the networks' hidden nodes. The important strength of the modular architecture is that its structure is well suited for incorporating domain knowledge.

3.4.2. Brief review of proposed modular neural network classifier

Our proposed knowledge-based modular neural network classifier produces necessary qualitative information of process response online. The design of our neural network classifier is developed based on well-known conventional control theory and multiple classifier strategy. This is accomplished by a feedforward (three layers) perceptron neural network and two knowledge-based neural networks. It has four inputs (error (*e*), error's first difference (*de*), error's second difference (*dde*) and change in reference (*cr*)) and these values are defined as:

$$e(n) = r(n) - y(n), \tag{3.19a}$$

$$de(n) = e(n) - e(n-1), \tag{3.19b}$$

$$dde(n) = de(n) - de(n-1), \tag{3.19c}$$

$$cr(n) = r(n) - r(n-1), \tag{3.19d}$$

where *r(n)* is reference input and *y(n)* is output for n^{th} sampling time. Since there are only four inputs (*e, de, dde, cr*), it allows better understanding to observe the relation between inputs and outputs. The detailed description of proposed model is discussed in Chapter 4.

In Figure 3-9, function f_{kbmnn} represents knowledge-based modular neural network classifier and it produces online qualitative current estimated process response among four pre-defined classes. It is a nonlinear mapping of a given inputs vector (*e, de, dde, cr*) to one-of-four pre-defined classes (steady-state, rise time, oscillation and disturbance). It can be expressed with the following Equation 3.20:

$$f_{kbmnn}(e, de, dde, cr) = \begin{cases} [1,0,0,0] \\ [0,1,0,0] \\ [0,0,1,0] \\ [0,0,0,1] \end{cases} \tag{3.20}$$

3.5. Brief Overview of Proposed Adaptive Law

The combination of these two approaches (modular fuzzy gain scheduling mechanism and switching algorithm) represents proposed adaptive law. In our approach, the conventional PID controller is used as the standard controller. The digital form *(n^{th} sampling time)* of conventional PID controller structure (Equation 2.2) can be expressed as given in Equation 3.21.

$$u_{pid}(n) = k_p'(n) \times e(n) + k_i'(n) \times \sum_{i=0}^{n} e(i) \times Ts + k_d'(n) \times \frac{e(n)}{Ts} \tag{3.21}$$

where $k_p'(n)$ is the proportional gain of PID controller, $k_i'(n)$ is the integral gain of PID controller and $k_d'(n)$ is derivative gain of PID controller, *e(n)* is error value and *Ts* is sampling time, respectively.

The appropriate setting of PID controller parameters k_p (proportional gain), k_i (integral gain) and k_d (derivative gain) are different based on the characteristics of process response. For example, the setting of loop gain depends on control objectives (stability, noise sensitivity and load regulation) [Jantzen98].

A suitable structure of PID controller also relate to the current process response. PI controller has the potential to increase system accuracy and it is most suitable in steady-state. PD controller structure is well-suited to improve performance [Kim02] and PID structure can improve both performance and system accuracy. A flexible PID controller design is needed in order to get excellent results. Both structure and parameters adjustment is occasionally necessary in high performance PID controller and online process response estimation is an important requirement for tuning of PID controller's parameters.

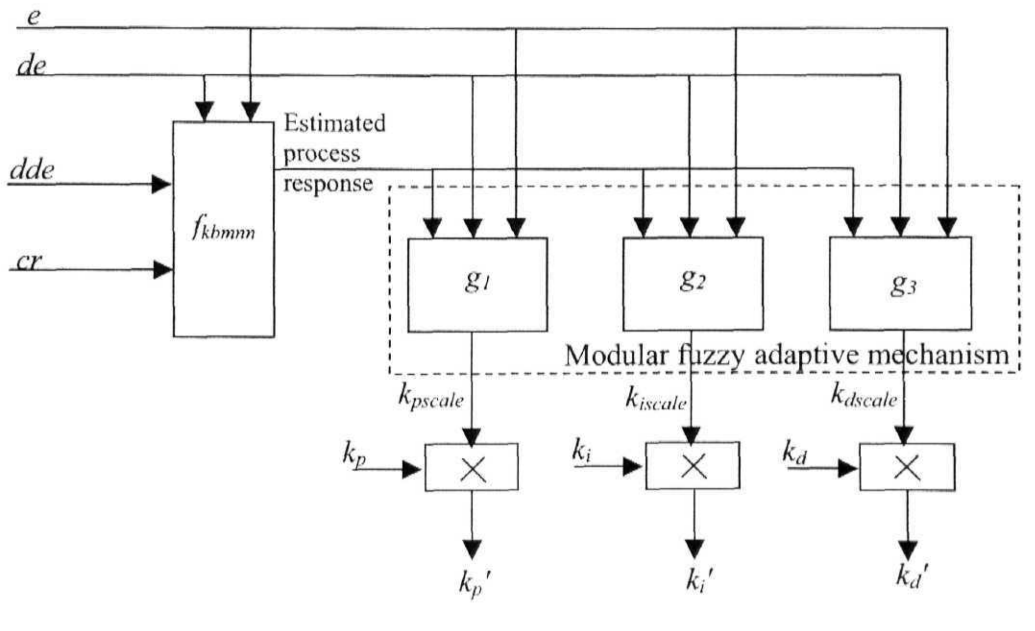


Fig. 3-9. Proposed adaptive law.

In our adaptive law, there are three independent functions (g_1, g_2 *and* g_3) to produce parameter scaling factors based on three input variables (*e, de, f_{kbmnn}(e, de, dde, cr)*) and function f_{kbmnn} classifies the current process state. Figure 3-9 presents a block diagram of our adaptive law. It includes four independent functions (f_{kbmnn}, g_1, g_2 and g_3). Modular fuzzy adaptable mechanism (function g_1, g_2 and g_3) produces parameter scaling factors with the following equations:

$$k_{pscale} = g_1(e, de, f_{kbmnn}(e, de, dde, cr)), \quad (3.22a)$$

$$k_{iscale} = g_2(e, de, f_{kbmnn}(e, de, dde, cr)), \quad (3.22b)$$

$$k_{dscale} = g_3(e, de, f_{kbmnn}(e, de, dde, cr)), \quad (3.22c)$$

The following Equation 3.23 updates the parameters of the standard PID controller.

$$k_p' = k_p \times k_{pscale} \quad (3.23a)$$

$$k_i' = k_i \times k_{iscale} \quad (3.23b)$$

$$k_d' = k_d \times k_{dscale} \quad (3.23c)$$

In the Equations 3.23, k_p, k_i and k_d are upper bound constant and they are derived from well-known Ziegler-Nichols recommendations. The variations of PID controller parameters only depend on parameter scaling factors. In our design, the range of proportional parameter is $0 < k_p' \leq k_p$ and integral and derivative parameters' ranges are $0 \leq k_i' \leq k_i$ and $0 \leq k_d' \leq k_d$ since all scaling factors lie within the bounded region [0, 1] (detailed discussion are described in Chapter 5).

3.6. Proposed Modular Neuro-Fuzzy Controller Structure

Our proposed modular fuzzy gain scheduling PID controller is illustrated in Figure 3-10. The conventional PID controller is used as a standard controller since it has some advantages such as general applicability to most control applications, the simplicity of its structure and clear relation between controller's parameters and process response. The parameters of the standard controller are tuned online with proposed adaptive law that is combined with Block-1 (knowledge-based modular neural network classifier) and Block-2 (modular fuzzy gain scheduling mechanism).

As shown in Figure 3-9, our controller design needs four sensory inputs (error, error's first difference, error's second difference and change in reference) and the way of getting these sensory inputs are described in Equations 3.19. The proposed knowledge-based modular neural network classifier (Block-1 in Figure 3-10) is responsible for providing qualitative information of process response (described in section 3.4).

This qualitative information is well-suited not only to get more precise nature of process response but also to implement modular structure. In Figure 3-10, Block-2 is composed of several fuzzy inference systems and they are responsible for different characteristic of process response (described in section 3.3). An appropriate fuzzy inference system is selected based on proposed switching algorithm that is based on qualitative information of process response and numerical error value (described in section 3.5).

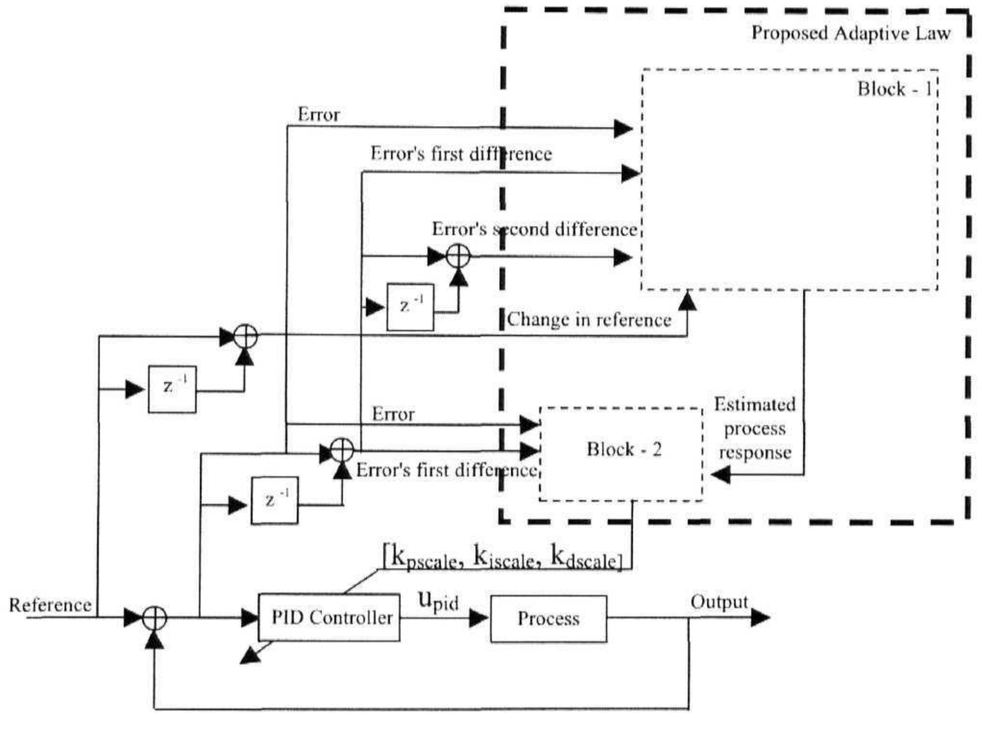


Fig. 3-10. Proposed modular fuzzy gain scheduling PID control system.

3.7. Discussions

The conventional control theory (both frequency domain and time domain) reveals that the high performance controller design needs sufficient information of dynamical behavior and flexible controller structure. Conventional control theory alone cannot fulfill this requirement. Many works present that the artificial intelligence technology helps to enhance the conventional control methodologies. The combination of these two technologies enables to implement high autonomy controller to solve new challenging control problems.

Table 3-3. The Comparison of proposed approach and other AI based PID controllers.

No.	Controller	Structure of controller	Process response/dynamical behavior estimation	Set-point tracking	Disturbance rejection	Delay effect
[1]	Zhao93	Non-modular (fuzzy gain scheduling)	Numerically (error& its first difference)	Yes	-	-
[2]	Viljamaa95	Non-modular (fuzzy gain scheduling)	Numerically (output & reference)	Yes	-	-
[3]	Li99	Non-modular	Numerically (error& its first difference)	Yes	-	-
[4]	Hur00	Two-degrees-of-freedom	Numerically (error& reference)	Yes	Yes	-
[5]	Rajani00	Non-modular (PI-type FLC)	Numerically (error& its first difference)	Yes	Yes	-
[6]	Guzelkaya 03	Non-modular (PID-type FLC)	Numerically (error, its first difference & its second difference)	Yes	-	Yes
[7]	Chang03	Non-modular (NN based gain scheduling)	Numerically (error)	Yes	-	-
[8]	**Proposed approach**	**Modular (fuzzy gain scheduling)**	**Both Numerically and Qualitatively**	**Yes**	**Yes**	**Yes**

This work introduces a new adaptive modular neuro-fuzzy PID controller design based on the combination of these two technologies (artificial intelligence technology and conventional control theory). The parameters of conventional PID controller are adjusted online based on qualitative information of process response and numerical value of error and its first difference (quantitative information). The proposed modular neural network classifier gives the qualitative information of the current process response. This work reveals that our process monitor provides more precise information about the dynamical behavior of the process.

Since the structure of proposed controller is modular, the overall representation of the controller design is more transparent and achieves greater flexibility. It is easy to modify, debug and possible to use a priori knowledge of existing control theory into the architecture by using a different module structure fitted for different operating regions. Table 3-3 summarizes the comparison of other Artificial Intelligence based PID controllers and our approach.

In proposed adaptive law, each of local computing modules is responsible for the different task and they are developed based on conventional control theory. It is an important requirement to implement high performance and robust controller design because suitable controller structure, parameters and resolution of parameter variation are different with respect to various characteristics of process response and objectives.

Consequently, our proposed controller design provides very flexible relation between input and output based on online information of process response. Also it is well-suited for both disturbance rejection and set-point tracking, and it minimizes the delay effect that is detrimental to the stability of the system.

Chapter 4: Knowledge-Based Modular Neural Network Classifier for Online Process Response Estimation

4.1. Introduction

As discussed in Chapter 3, our proposed switching algorithm needs qualitative information of current process response. In our approach, knowledge-based modular neural network classifier is developed in order to accomplish this nonlinear mapping process. This Chapter introduces the transformation of four numerical sensory (visual) inputs (*e, de, dde, cr*) that was defined in Equations 3.19, into qualitative information of current process response (one of four pre-defined classes: *(i)* steady-state, *(ii)* rise time state, *(iii)* overshoot/oscillation state and *(iv)* disturbance state). In next section, we analyze the requirement of classifier design.

4.2. Preliminary Classifier Design Considerations

In real-life applications, classification is often not a stand-alone problem but rather a part of a larger system involving optimization, explanation and evaluation of decisions, and interactions with the environment. The goal of classification is to assign a physical object, event, or phenomenon to one of pre-specified classes with a minimal rate of misclassification.

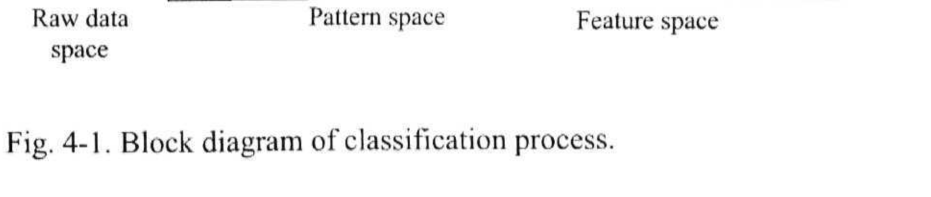


Fig. 4-1. Block diagram of classification process.

In general, the transformation of input data into class membership is highly complex and non-invertible. Figure 4-1 represents the block diagram of classification process [Zurada01]. The transducer or sensor provides the input pattern data to the feature extractor. Typically, inputs to the feature extractor are sets of data vectors that belong to a certain category. The feature extractor performs the reduction of dimensionality. Then a set of discriminant functions (Classifier Block in Figure 4-1) determines the appropriate category among pre-specified classes. Practical classification problems require selection of a subset of attributes or features to represent the patterns to be classified and a set of discriminant functions.

4.2.1. Features selection

One of the most difficult and important requirements in the design of automatic pattern classifier is the selection of appropriate features for the classification problem at hand. In most cases, each data object (pattern) is represented numerically by a vector composed of the values of some measurable features. Although, all of these features constitute the inputs of a classifier, they have different impacts on the classification performance. Some features may not increase the discriminative power of the classifier among pattern classes. The main task of feature extractor is to retain the minimum number of data dimensions while maintaining the probability of correct classification. A number of approaches to feature subset selection have been proposed in the literature [Ruck90, Rezaee99, Zhu04].

Feature subset selection algorithms can be classified into two categories based on whether or not feature selection is done independently of the learning algorithm used to construct the classifier. If feature selection is done independent of the learning algorithm, the technique is said to follow a filter approach. Otherwise, it is said to follow a wrapper approach [john94].

While the filter approach is generally computationally more efficient than the wrapper approach, its major drawback is that an optimal selection of features may not be independent of the inductive and representational biases of the learning algorithm to be used to construct the classifier. The wrapper approach, on the other hand involves the computational overhead of evaluating candidate feature subsets by executing a selected learning algorithm on the training dataset.

4.2.2. Discriminant functions selection

In the final classification step, the membership in a category is determined by the classifier based on the comparison of the results of R discriminant functions ($d_1(X)$, $d_2(X)$, ..., $d_R(X)$). The pattern X belongs to the i^{th} category if and only if

$$d_i(X) > d_j(X), \qquad \text{for } i, j = 1,2,....,R, i \neq j. \tag{4.1a}$$

$$X = [x_1, x_2, ..., x_n]' \tag{4.1b}$$

A special case of the classifier into R classes for $R = 2$, such a classifier is called the dichotomizer. The discriminant functions of dichotomizer for both linear and nonlinear cases are illustrated in Figure 4-2. If a priori knowledge is rich enough to implement classifier design, the discriminant functions can be derived from this knowledge. It can be defined as a knowledge-based classifier.

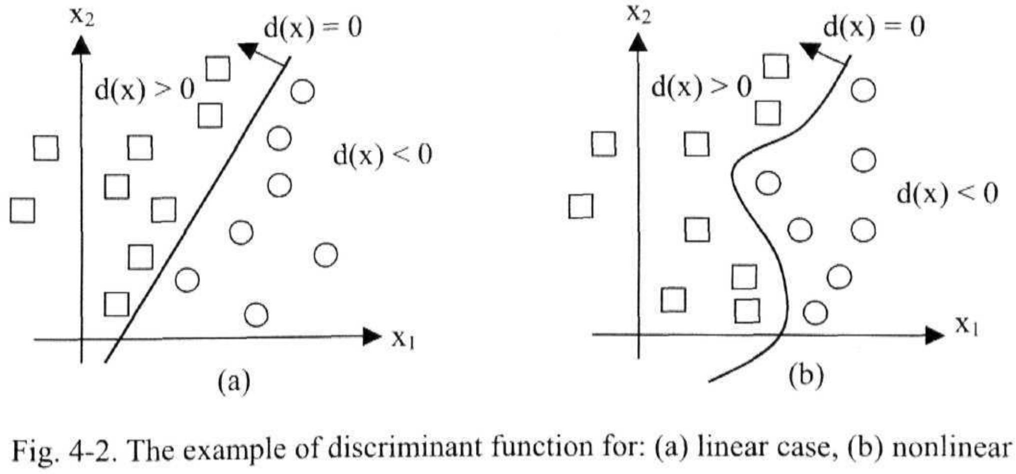


Fig. 4-2. The example of discriminant function for: (a) linear case, (b) nonlinear case.

4.2.3. Knowledge-based classifier system

In a knowledge-based classifier system, the classification process can be explicitly described by a number of *If-Then* rules. The rules may look like:

IF *A* is *SMALL* and *B* is *LARGE* THEN *Class1*,

IF *A* is *LARGE* and *B* is *MEDIUM* THEN *Class2*,

where *A* and *B* are features, *SMALL*, *MEDIUM* and *LARGE* are fuzzy sets/crisp sets and rules represent discriminant functions.

Knowledge acquisition is one of the key elements in the development of knowledge-based classifier system. It means transferring expertise from existing theory, principles and/or human experts who use natural language marked by vagueness. Although knowledge base can be represented with simple *IF-THEN* rules, avoiding contradiction to develop a sufficiently complete rule set is difficult to handle in a larger problem. This difficulty is known as the knowledge acquisition bottleneck.

One of the most powerful techniques for solving problems is to break them down into small and more easily solvable pieces. Smaller problems are less overwhelming, and they permit us to focus on details that are lost when we are studying the entire problem. It is a necessary to decompose a large-scale and complex problem into multiple smaller and simpler sub-problems. Therefore, many researchers have tried to implement multiple classifier system [Giacinto01, Toygar04].

4.2.4. Multiple classifier system

There are limitations on using a single classification (i.e., a large scale and complex problem) technique. Though commonly the main objective of the classifier design process is constructing a good performance classifier, there are some additional aspects, which we need to consider like, for instance, it may be a part of the requirements that the classifier model is transparent and has the ability to provide an easily interpretable explanation of the suggested classification decision. The rationale is that it may be more difficult to optimize the design of a single complex classifier than to optimize the design of a combination of relatively simpler classifiers. Moreover, the computational speed of the model is of primary concern in the real time control application.

Multiple classifier system has emerged over recent years to address the practical problem of designing classification systems with improved accuracy and efficiency [Giacinto01, Toygar04]. A popular method for creating an accurate classifier from a set of training data is to build several classifiers and combine their predictions. The aim is to design a composite system that outperforms any individual classifier by pooling together the decisions of all classifiers.

It is intuitively accepted that classifiers to be combined should be diverse. If they were identical, it is not possible to get any improvement by combining them. Combination and measurement of diversity become key issues to build multiple classifiers. Diversity measured can be categorized into two types (pair-wise and non-pair-wise) and the various pair-wise diversity measures are investigated in [Windeatt05]. Although it is known that diversity among base classifiers is a necessary condition for improvement in ensemble performance, there is no general agreement about how to qualify the notion of diversity among a set of classifiers.

There are two families of combining multiple classifiers: serial combination and parallel combination. In serial combination, the order of arrangement is crucial for the classification performance of the system since it arranges classifiers sequentially. In parallel combination, system performance depends on the combination algorithm.

It is noticed that modular neural network classifier divides the classification task into separate partially or fully independent smaller or simpler neural modules. Then, sub-solutions are integrated via a multi-module decision-making strategy. It can be seen as the multiple classifier system.

4.3. Proposed Features Selection for Process Response Estimation

The choice of features affects the performance of classifier. It is generally desirable to reduce the number of features that are used while still maintaining adequate classification accuracy and it strongly depends on the particular problem. Our requirement is to decompose a whole process response into four states of process response.

In conventional control theory, phase plane method describes the relation between region and error signals that tabulated in Table 4-1. Li and Gatland [Li95] show that the current existing region provides some important information of process response. Accordingly, we observe the relation between region and process response in order to select appropriate features for our classification process.

Table 4-1. The relation of error signals and region.

No.	Error Signals		Region
	Error	Change in Error	
1.	Positive	Negative	*A1*
2.	Negative	Negative	*A2*
3.	Negative	Positive	*A3*
4.	Positive	Positive	*A4*

4.3.1. Extracting useful knowledge from process response analysis

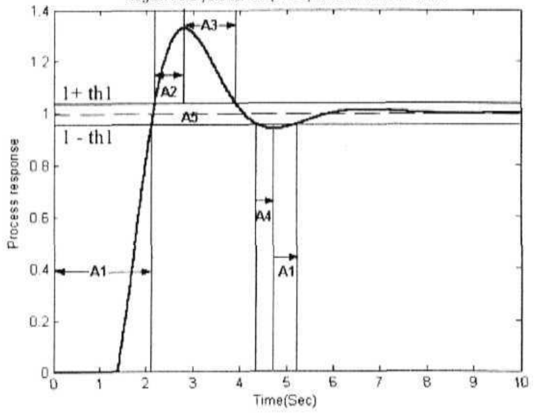


Fig. 4-3. Proposed region partition of process response in time domain.

Generally, although four regions (*A1, A2, A3, A4*) are defined process response analysis in time domain and phase plane [Li95], one more region (*A5*) in addition to the above four in our proposed approach that is needed to represent steady-state condition. The threshold value *th1* is used to define this *A5* region as shown in Figure 4-3.

In practical application, output process response is difficult to exactly agree with reference input (i.e., error is zero). Generally, steady-state condition is defined based on the settling time. It is the time required for process response to reach and stay within the range (mostly defined as *2%*) of final reference value [Ogata02]. In our approach, threshold value *th1* is responsible to define settling time. Accordingly, it can be defined within the range between *0* and *0.02* (*0%* of final reference value $\leq$ *th1* $\leq$ *2%* of final reference value). In case of *th1* = *0*, the region demarcated in process response is the same as phase plane (The value of *th1* is set to *0.02* in our classification process).

We observe a variety of process responses with different aspects in order to define useful features in our classification process. Figures 4-4 shows that current process response state can be defined based on function of previous estimated state, previous existing region and current existing region. For instance, **IF** previous estimated state is *Disturbance* and previous region is *A2* and current region is *A2* **THEN** estimated current process response state is *Disturbance* (Figure 4-4(a)). **IF** previous estimated state is *Disturbance* and previous region is *A5* and current region is *A2* **THEN** estimated current process response state is *Overshoot/oscillation* (Figure 4-4(b)).

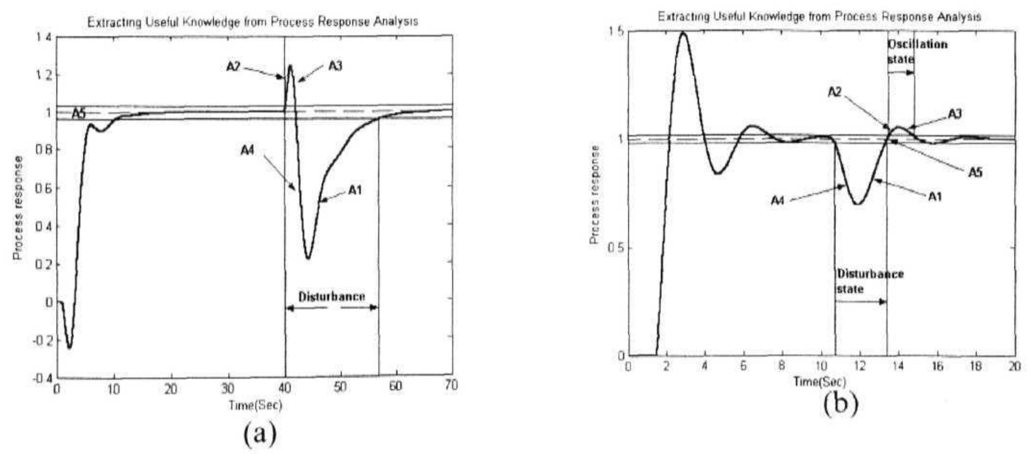


Fig. 4-4. Extracting useful knowledge from process response analysis (a) non-minimum phase process response, (b) minimum phase process response.

As we stated earlier, *A5* region represents steady-state condition. But, Figure 4-5 expresses that other states can occur in this region. Therefore, there is a necessity to use one more feature that can separate steady-state and other states in this region. The settling time is important factor to define steady-state condition. Generally, it depends on the error and its velocity (i.e., error is within the bounded region (*±2%* of its final value) and its velocity is zero) [Ogata02].

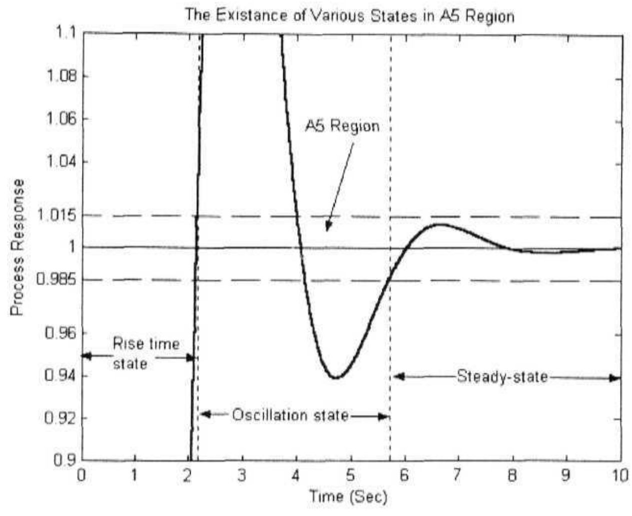


Fig. 4-5. The existence of various states in *A5* region.

Figure 4-6 reveals that only these two parameters are not sufficient to define settling time correctly. In this example, although error is less than *± 2%* of its final value (i.e., error is in *A5* region) and the velocity of error is zero, it is in oscillation state. Therefore, we define one more feature (equilibrium state) to overcome this limitation.

Viljiamaa [Viljiamaa02] suggests that reference signal is important to estimate process response online in order to get better response for the big change in the reference signal. In our assumption, rise time state depends on the reference value as it can occur due to change in reference only.

As a result, (i) **current region**, (ii) **previous region**, (iii) **previous estimated state**, (iv) **change in reference** and (v) **equilibrium state** are defined as appropriate features for our classification process and the next section presents the way of extracting these features from available sensory (visual) inputs.

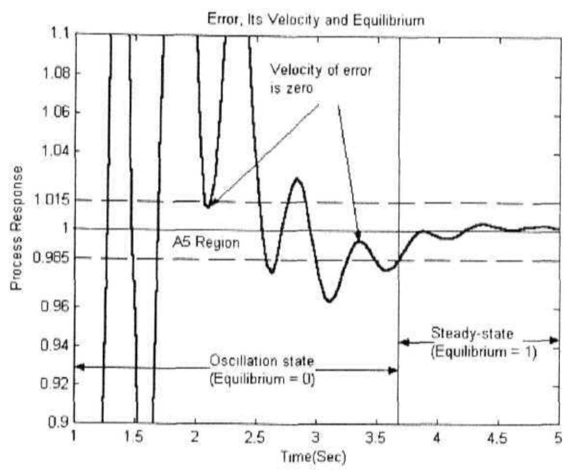


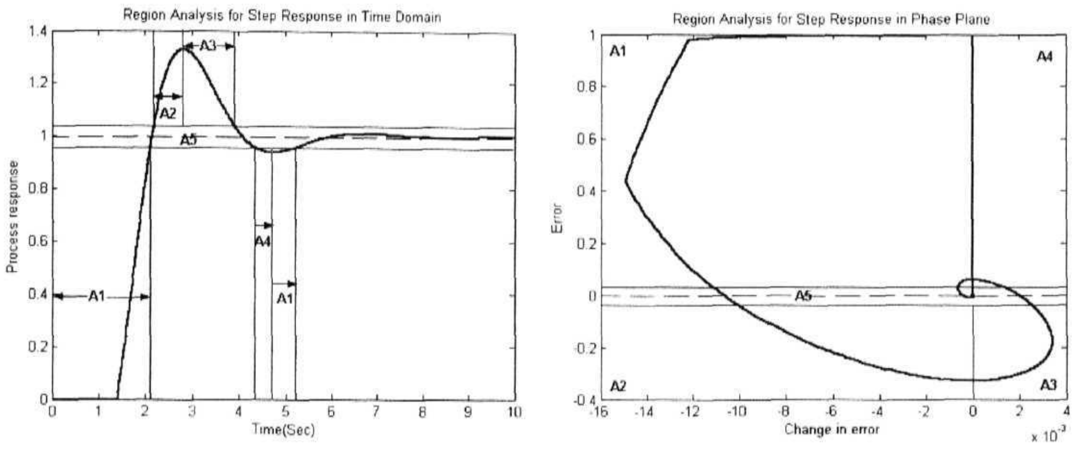
Fig. 4-6. The relation between error, its first difference and equilibrium.

4.3.2. Transformation of sensory data into useful features

The processing of sensory data into useful features is highly depended on the problem at hand. There are five features in our classification process. Among them, change in reference is computed from reference input (described in Equation 3.19d) and previous region and previous estimated state (for n^{th} sampling time) is achieved in the following way: (i) Previous region (n^{th} sampling time) = Current region ($(n-1)^{th}$ sampling time) and (ii) Previous estimated state (n^{th} sampling time) = Current estimated state ($(n-1)^{th}$ sampling time). The remaining two features (current region and equilibrium state) are discussed in the subsequent sections.

4.3.2.1. Defining current region

As we tabulated in Table 4-1, mapping from time domain to phase plane [Li95] reveals the relation between regions and error values. Figure 4-7 illustrated the proposed region partition both time domain and phase plane. Figure 4-7(a) represents process response in time domain. It can be illustrated by plotting *e(t)* (error value at time *t*) versus *de(t)* (change in error value at time *t*). As *t* varies, the process response describes a curve in the *e-de* plane as shown in Figure 4-7(b). Accordingly, current value of error signal and its first difference can be used to define current region in our classification process. For instance, **IF** error signal is *Negative* and change in error signal is *Negative* **THEN** current region is *A2* (Figure 4-7(b)).



(a) (b)

Fig. 4-7. Mapping from (a) time domain to (b) phase plane.

Therefore, it is a necessity to qualitatively define current error signal (*Negative (E_NEG)* or *Positive (E_POS)* or *Zero (E_ZE)*) and change in error signal (*Negative (DE_NEG)* or *Positive (DE_POS)*). In our approach, they are crisp set {0, 1} and Equations 4-2 and 4-3 describe how to convert numerical sensory data (error signal and change in error signal) [-1, 1] into crisp sets {0, 1} with threshold value *th1*.

$$E\_ZE = \begin{cases} 0, & if\,|e| \geq th1 \\ 1, & if\,|e| < th1 \end{cases} \tag{4.2a}$$

$$E\_POS = \begin{cases} 0, & if\,e < th1 \\ 1, & if\,e \geq th1 \end{cases} \tag{4.2b}$$

$$E\_NEG = \begin{cases} 0, & if\,e > -th1 \\ 1, & if\,e \leq -th1 \end{cases} \tag{4.2c}$$

$$DE\_POS = \begin{cases} 0, & if\,de < 0 \\ 1, & if\,de \geq 0 \end{cases} \tag{4.3a}$$

$$DE\_NEG = \begin{cases} 0, & if\ de \geq 0 \\ 1, & if\ de < 0 \end{cases} \tag{4.3b}$$

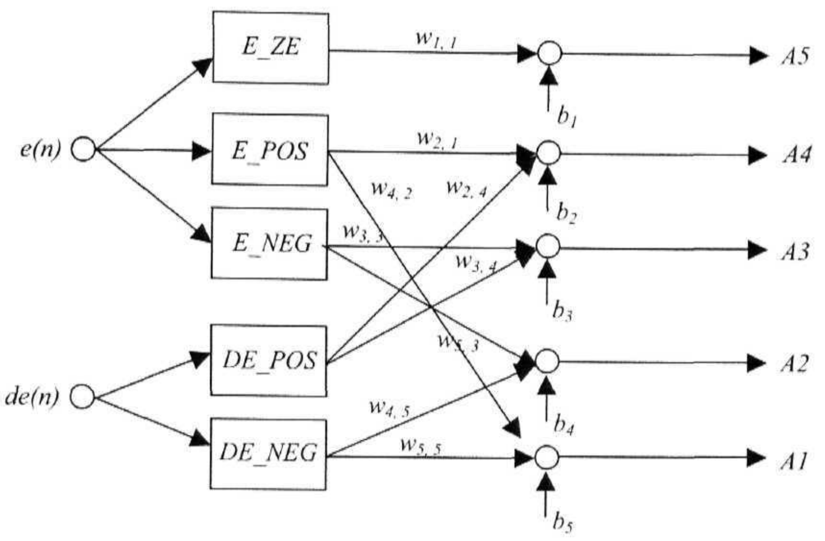


Fig. 4-8. Detail neural network diagram to define current region.

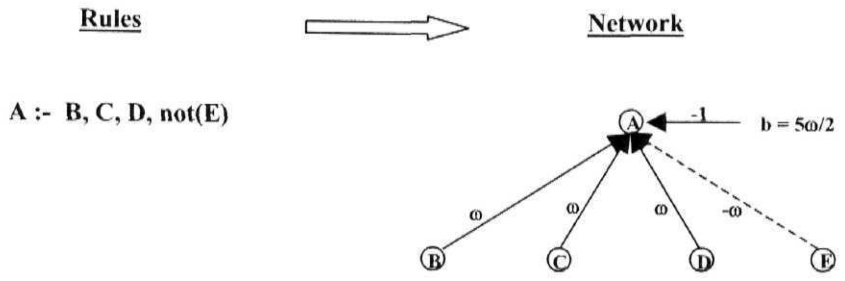
In our approach, a neural network produces current existing region based on status of these inputs. Detail neural network diagram to define current region is described in Figure 4-8. The appropriate network topology relies on the existing knowledge (the relation of region and error signals that was described in Figure 4-7(b)): for instance, **IF** error signal is *zero* (*E_ZE =1*) **THEN** current region is *A5*.

The weights and biases of our proposed neural network are initialized by the rules to network translation algorithm that is introduced by Towell [Towell94], and shown in Figures 4-9. Equation 4.4(a) represents how to compute a bias value *b* for a conjunctive rule and Equation 4.4(b) represents for a disjunctive rule.

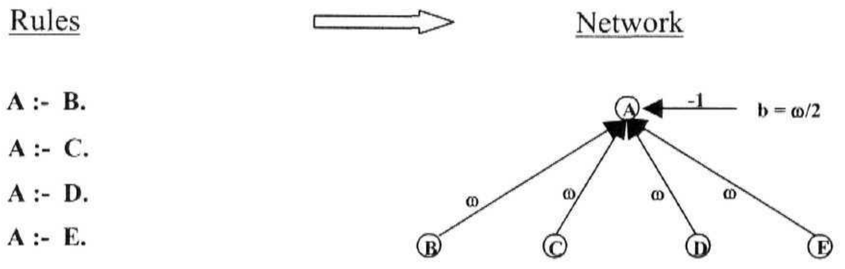
$$b = \frac{(2n-1)\omega}{2} \tag{4.4a}$$

$$b = \frac{\omega}{2} \tag{4.4b}$$

where, ω is a random positive weight value, b is a bias and n is number of positive inputs. The following explanation shows how to initialize weights ($w_{2,1}$ and $w_{2,4}$) and bias b_2 of proposed neural network (Figure 4-8): $w_{2,1} \approx w_{2,4} \approx w$ (w is a random positive value and $w_{2,1}$, $w_{2,4}$ and w are almost the same value) and $b_2 = 3w/2$.



(a)



(b)

Fig. 4-9. Towell proposed translation of rules into network algorithm for (ω represents weights and θ represents bias) (a) conjunctive rules and (b) disjunctive rules.

4.3.2.2. Defining equilibrium state

The equilibrium state is required in order to define steady-state properly. In this work, available knowledge is not sufficient to implement the equilibrium state observer. Therefore, our proposed equilibrium state observer neural network is developed based on training dataset. We extract required training dataset based on extensive experiments and Figure 4-10 expresses the way of getting required training dataset to implement appropriate equilibrium state observer neural network.

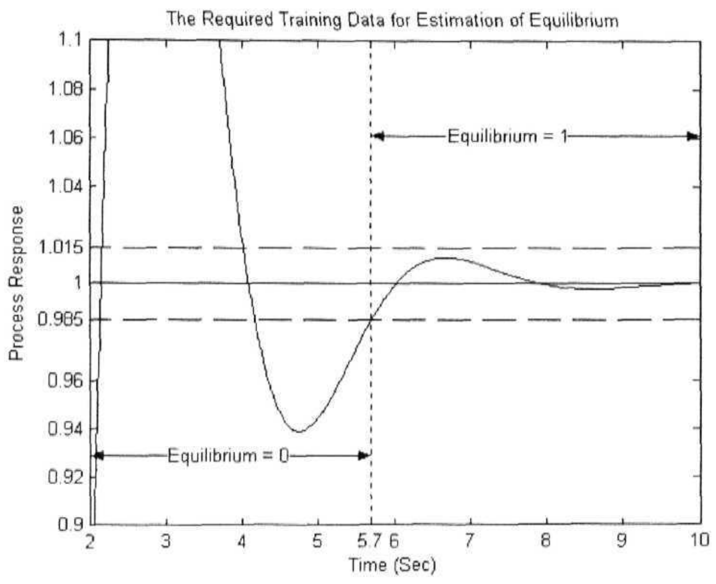


Fig. 4-10. The way of getting training data set to define equilibrium state.

The required inputs for this process are error's first difference (velocity) and second difference (acceleration) because equilibrium state is function of these two inputs (i.e., the process response is in equilibrium when velocity and acceleration of error are approximately zero). The appropriate structure, weights and biases of our neural network are chosen based on results of extensive experiments and is tabulated in Table 4-2.

Accordingly, it needs four inputs and the proposed structure consists of six neurons in first hidden layer, nine neurons in second hidden layer and one neuron in output layer as shown in Figure 4-11. The activation function of each neuron is log sigmoid that is described in Equations 4.5 (x_i represent input vector [*de(n), de(n-1), dde(n), dde(n-1)*] and w_i represent weight vector for each neuron that are to be trained).

Table 4-2. A variety of equilibrium state observer neural network structures and their accuracy (890 training datasets and 140 test datasets).

No.	NN Structure	MSE (Mean Square Error)	Epoch	Accuracy (Training dataset)	Accuracy (Test dataset)
1.	4-13-1	0.013971	12000	91.461%	75%
2.	4-15-1	0.013375	12000	92.022%	79.286%
3.	4-4-9-1	0.0125341	12000	97.303%	92.143%
4.	**4-6-9-1**	**2.079×10^{-7}**	**12000**	**100%**	**100%**
5.	4-4-11-1	0.0114073	12000	97.416%	92.143%
6.	4-6-11-1	1.964×10^{-4}	12000	99.663%	97.857%

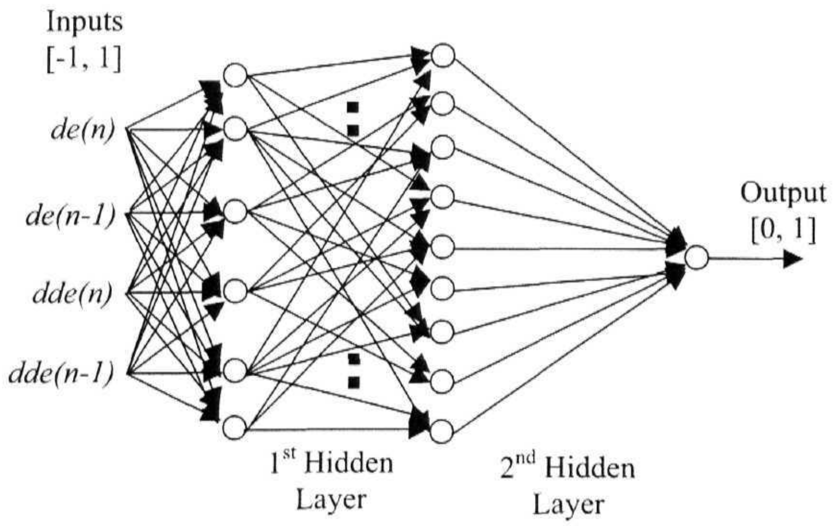


Fig. 4-11. The proposed equilibrium state observer neural network.

$$f(net) = \frac{2}{1+\exp(-net)} - 1 \tag{4.5a}$$

$$net = \sum_{i=1}^{m} w_i x_i \qquad (4.5b)$$

4.4. Proposed Descriminant Functions Selection for Process Response Estimation

The appropriate category among pre-specified states is determined by a set of discriminant functions. Our proposed scheme is based on multiple classifier strategy and existing well-known control theory based task decomposition technique.

Task decomposition method can be roughly classified into the following categories: functional modularity, domain modularity, class decomposition and state decomposition according to different partition strategies. Our classification process is a four-classes (four-states) problem and it is divided into four two-classes problems based on class decomposition technique. Then second decomposition step relies on input space partition.

4.4.1. Detail structure of proposed discriminant functions

Table 4-3. The required discriminant functions (rules) for Class 1 (steady-state).

Rules	Input Attributes					Class Attributes
	Current Region	Previous Region	Previous Class	Change in Reference	Equilibrium	
1	A5	Do not care	Do not care	No	Yes	Class 1
2	A5	Do not care	Class 1	No	Do not care	Class 1

Tables 4-3 to 4-6 describe the set of discriminant functions for steady-state (Class 1), rise time state (Class 2), overshoot/oscillation state (Class 3) and disturbance state (Class 4), respectively.

Table 4-4. The required discriminant functions (rules) for Class 2 (rise time state).

Rules	Input Attributes					Class Attributes
	Current Region	Previous Region	Previous Class	Change in Reference	Equilibrium	
3	Do not care	Do not care	Do not care	Yes	Do not care	Class 2
4	A1	Do not care	Class 2	Do not care	Do not care	Class 2
5	A4	Do not care	Class 2	Do not care	Do not care	Class 2
6	A5	Do not care	Class 2	Do not care	No	Class 2

Table 4-5. The required discriminant functions (rules) for Class 3 (overshoot / oscillation state).

Rules	Input Attributes					Class Attributes
	Current Region	Previous Region	Previous Class	Change in Reference	Equilibrium	
7	A2	Do not care	Class 2	No	Do not care	Class 3
8	A2	Not (A2)	Class 4	No	Do not care	Class 3
9	A1	Do not care	Class 3	No	Do not care	Class 3
10	A2	Do not care	Class 3	No	Do not care	Class 3
11	A3	Do not care	Class 3	No	Do not care	Class 3
12	A4	Do not care	Class 3	No	Do not care	Class 3
13	A5	Do not care	Class 3	No	No	Class 3

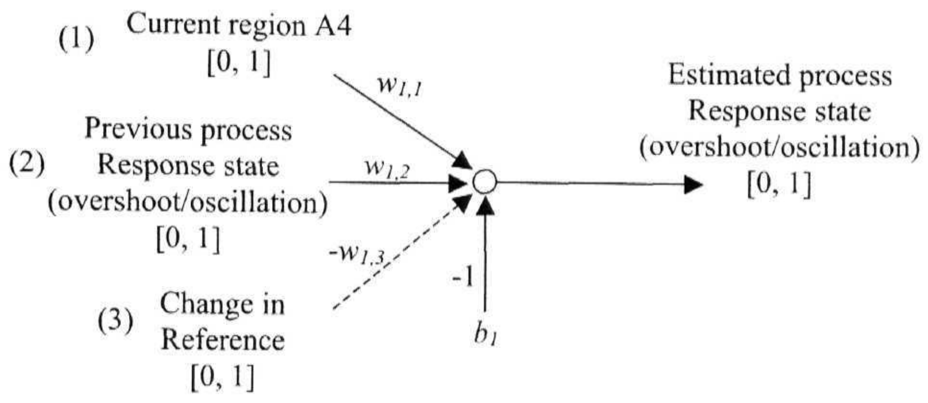

(1) Current region A4
[0, 1]
(2) Previous process Response state (overshoot/oscillation)
[0, 1]
(3) Change in Reference
[0, 1]
$w_{1,1}$
$w_{1,2}$
$-w_{1,3}$
-1
b_1
Estimated process Response state (overshoot/oscillation)
[0, 1]

Fig. 4-12. Example of an independent simple classifier (Rule 12).

Table 4-6. The required discriminant functions (rules) for Class 4(disturbance state).

Rules	Input Attributes					Class Attributes
	Current Region (*cur_r*)	Previous Region (*pre_r*)	Previous Class (*pre_c*)	Change in Reference (*cr*)	Equilibrium (*eq_b*)	
14	A2	Do not care	Class 1	No	Do not care	Class 4
15	A4	Do not care	Class 1	No	Do not care	Class 4
16	A2	A2	Class 4	No	Do not care	Class 4
17	A3	Do not care	Class 4	No	Do not care	Class 4
18	A4	Do not care	Class 4	No	Do not care	Class 4
19	A1	Do not care	Class 4	No	Do not care	Class 4
20	A5	Do not care	Class 4	No	No	Class 4

Our task decomposition technique divides a complex problem into twenty very simple problems (a set of discriminant functions) and each simple problem represents the independent classifier (Rule). Using Towell's [Towell94] rules-to-network algorithm, Figure 4-12 shows such a mapping for Rule 12. In this Figure, $w_{1,1} \approx w_{1,2} \approx w_{1,3} \approx w$ (w is a random positive value) and $b_1 = 3w/2$.

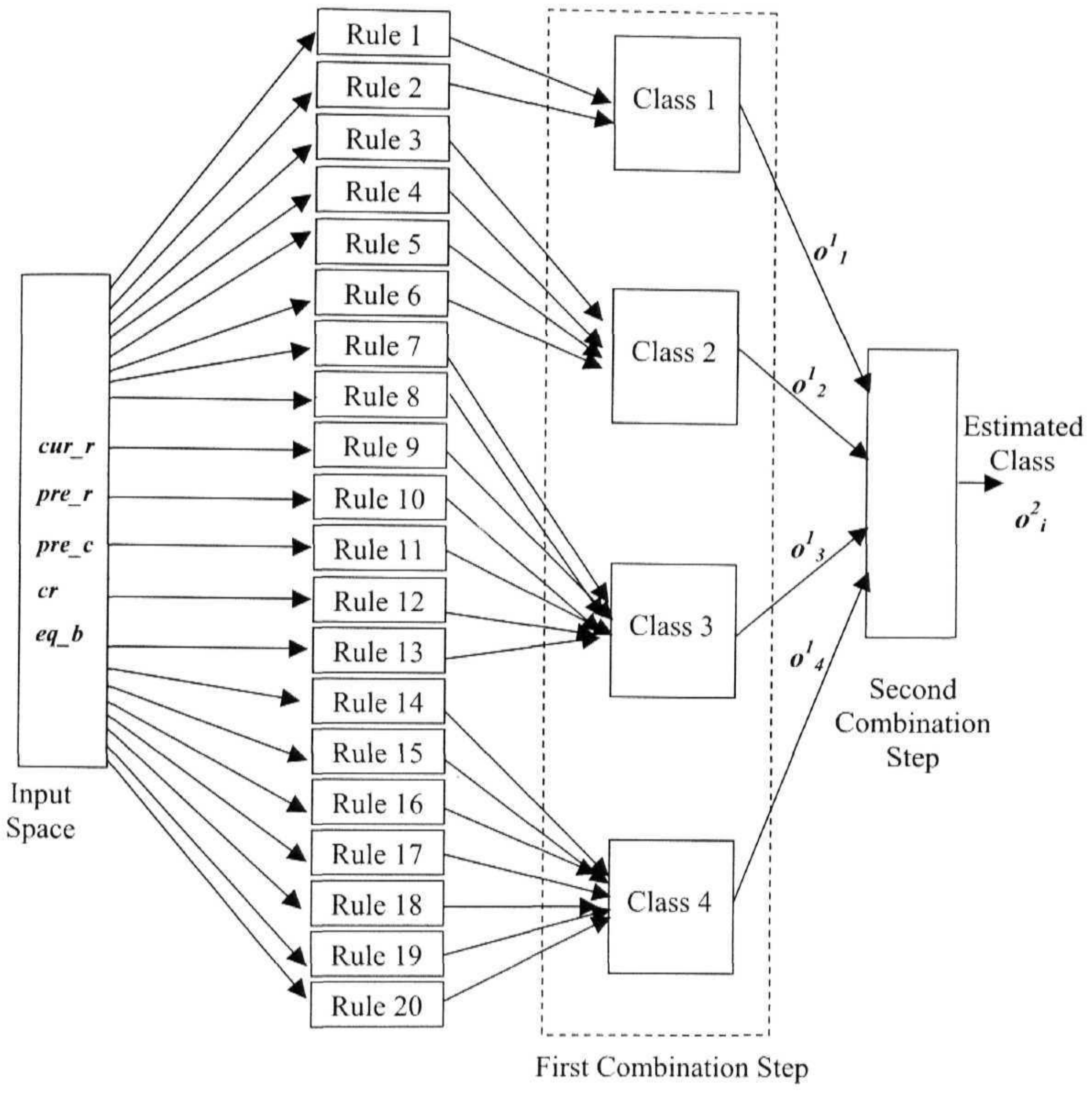


Fig 4-13. The whole structure of proposed discriminant functions.

Then, sub-solutions are integrated via a multi-module decision-making strategy. It is noticed that decision making rules are transparent and easy to debug. There are two parallel combination steps in this classification process. First parallel combination step is the same as translation of disjunctive rules by Towell [Towell94] (described in Equation 4.4(b) and Figure 4-9(b)) and a maximum selector is used for second combination step.

The membership in a category is based on the comparison of outputs from first parallel combiner and it is illustrated in Equation 4.6.

$$o^2_i = \begin{cases} 1, & if \quad o^1_i > o^1_j, \quad for\, i, j = 1,2,3,4, \quad i \neq j \\ 0, & otherwise \end{cases} \tag{4.6}$$

where, o^2_i is the output of second combination unit and o^1_i [0, 1] is the output of first combination unit.

The whole structure of proposed discriminant functions is described in Figure 4-13. It is a nonlinear mapping from input space that includes five features, into output space (estimated process response state). It consists of twenty individual simple classifiers (discriminant functions) and two parallel combination steps. The important facts of state transactions are described in Figure 4-14.

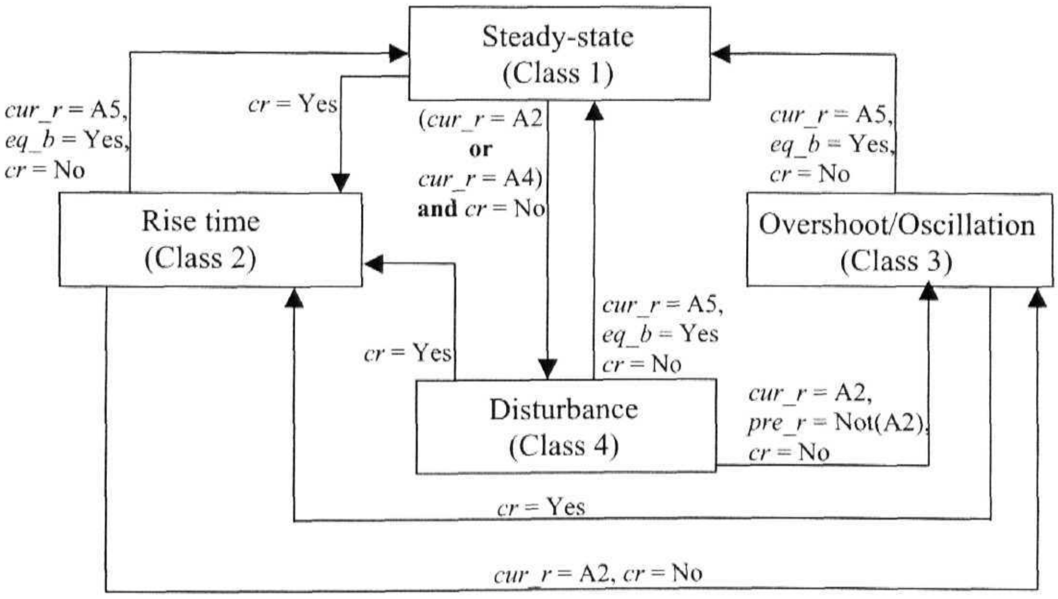

Steady-state (Class 1)
Rise time (Class 2)
Overshoot/Oscillation (Class 3)
Disturbance (Class 4)
cur_r = A5, eq_b = Yes, cr = No
cr = Yes
(cur_r = A2 or cur_r = A4) and cr = No
cur_r = A5, eq_b = Yes, cr = No
cur_r = A5, eq_b = Yes cr = No
cr = Yes
cur_r = A2, pre_r = Not(A2), cr = No
cr = Yes
cur_r = A2, cr = No

Fig. 4-14. The state transactions diagram of our classification process.

4.4.2. Refinement algorithm for proposed discriminant functions

The design of our classifier (Classifier-2 Block) is based on well-known control theory's domain knowledge (i.e., a task-specific collection of inference rules). But, in practical applications it is very difficult to get perfect knowledge base and refinement algorithm is necessary to improve imperfect domain knowledge, if it exists.

The main concept of knowledge base refinement algorithm is that at least or at most one output of first combination unit (described in Figure 4-12) must be totally matched with current input subspace (features) at that moment of time (i.e., only one output of first combination unit can be activated at the moment of time).

When input subspace partially matches each classifier in Classifier-2 Block (i.e., incomplete knowledge base), all outputs of first combination units are less than threshold value (threshold value $th2 = 0.9$) at that moment of time. In that case, our knowledge base is incomplete and knowledge refinement task is necessary. When more than one classifier (rule) are totally matched with current input subspace at that moment of time, these classifiers are enabled (i.e., their output is greater than or equal to threshold value $th2$) and they contradict each other. Based on these facts, we propose a refinement algorithm for our proposed Classifier-2 Block and it is summarized as follows.

Knowledge Base Refinement Algorithm

```
Stage 1.    Determine incomplete knowledge base and mismatch classifiers:
    Step 1: Initialize sampling time [n:=1].
    Step 2: Repeat Step 3 to Step 4 until the reach end of training data.
    Step 3: Read each output of individual classifiers from Classifier-2 Block (n^th
            sampling time).
    Step 4: Examine how many classifiers produce totally true (i.e., classifier output is
            greater than threshold value th2).
            Step 4a:    If each classifier does not produce totally true, go to Step 1
                        in Stage 2.
            Step 4b:    Else If more than one classifier that represent different
                        states (for instance, Rule 11 and Rule 20), produce totally
                        true, go to Step 2 in Stage 2.
            Step 4c:    Else n := n + 1 and go to Step 3 in Stage 1.
            [End of If Structure]
```

Step 5: Exit.

Stage 2. Refine the knowledge base.

Step 1: Refine the incomplete knowledge based on current input subspace and target response (i.e., it builds a new rule based on training dataset) and go to Step 3 in Stage 1.

Step 2: Refine the mismatch classifiers based on current input subspace and target response (i.e., it keeps the most suitable rule and deletes others based on training dataset as shown in following Example 4.1) and go to Step 3 in Stage 1.

Example 4.1:

Training dataset: current region is ***A2*** and previous region is ***A2*** and previous class is ***Class 4*** (input data), Estimated class is ***Class 4*** (output data).

Mismatch Rule1: current region is ***A2*** and previous class is ***Class 4*** (input data), Estimated class is ***Class 3*** (output data).

Mismatch Rule2: current region is ***A2*** and previous region is ***A2*** and previous class is ***Class 4*** (input data), Estimated class is ***Class 4*** (output data).

In this Example, outputs of both rules are greater than threshold value *th2*. Our refinement algorithm selects Mismatch Rule2 based on training dataset and deletes Mismatch Rule1. The above algorithm can be used not only to refine knowledge-based but also to examine the validity of classification accuracy. For instance, the current estimated process response is not certain when more than one classifier that represent different states, are energetic or no classifier is energetic.

4.5. Our Proposed Modular Neural Network Classifier

In order to provide qualitative information of process response, we attempted to build knowledge-based modular neural network classifier based on priori knowledge of process response and conventional control theory. The appropriate selections of features and discriminant functions for this classification process have been described in previous sections. This approach gleans existing knowledge (conventional control theory) to determine an appropriate network topology, initial weights and biases of our neural network.

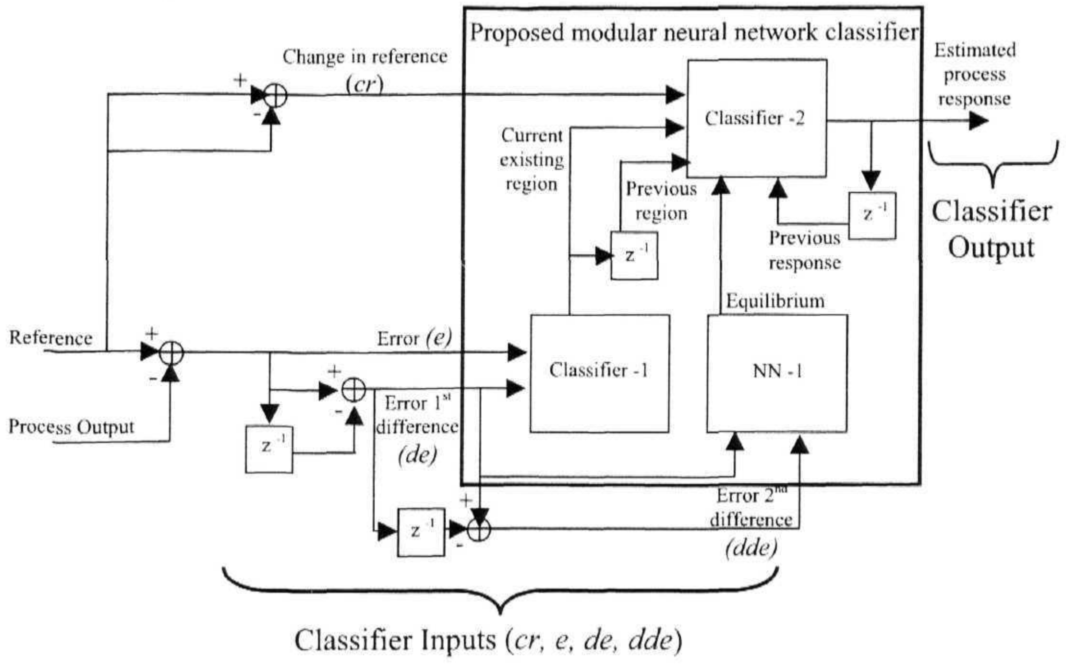


Fig. 4-15. Proposed modular neural network classifier.

Figure 4-15 depicts the complete structure of proposed knowledge-based modular neural network classifier. It is a transformation of four numerical sensory (visual) inputs (*e, de, dde, cr*) into qualitative information of current process response (one of four pre-defined classes: (i) steady-state, (ii) rise time state, (iii) overshoot/oscillation state and (iv) disturbance state). There are three main independent classifiers (Classifier-1 Block, Classifier-2 Block and NN-1 Block) in this classification process. Classifier-1 Block that is described in section 4.3.2.1, is responsible to define current region. NN-1 Block observes the equilibrium state that is described in section 4.3.2.2. Classifier-2 Block represents the set of discriminant functions that is described in section 4.4. Classifier-1 Block and Classifier-2 Block consist of several independent simple classifiers and it is seen that the whole structure of our proposed classifier is the same as multiple classifiers strategy.

4.6. Discussions

The process monitor that detects change in the process characteristics, is an essential component of adaptive control applications. In order to implement high performance controller, it is necessary to provide precise information of dynamical behavior of process to be controlled. Although many adaptive controller designs are based on numerical information of process response (i.e., error, change in error, etc.), in our approach, qualitative information is also required. Our proposed knowledge-based modular neural network classifier is responsible to provide qualitative information of current process response.

This Chapter described in detail how to build knowledge-based modular neural network classifier design to estimate qualitatively current process response. The selection of a subset of features to represent the patterns to be classified and a set of discriminant functions are key requirements in practical classification process. In our approach, the selection of features and a set of discriminant functions for this specific classification process are based on transient and steady-state response analysis and phase plane method.

The proposed approach is the same as multiple classifier strategy since the whole structure of our classifier is combination of several simple classifiers. It provides adequate classification accuracy and the decision making strategy is easier to understand and modify. Knowledge base refinement algorithm is introduced in order to improve imperfect domain knowledge because it is difficult to get perfect domain knowledge in practical applications. It provides useful information to examine the validity of current classification accuracy.

Moreover, the future control technology is embracing modular structure controller design. Modular structure controller consists of computing modules and a gating system. The gating system is used for the selection of the module valid for a current state of the process and proposed current response estimation can be used as a switching logic. For instance, the controller parameters adjustment strategies are different based on the state of the process response (steady-state, rise time state, overshoot/oscillation state and disturbance state) and this information supports to develop modular parameter adjustment structure. As a result, proposed modular classifier provides some qualitative information of process response and it is well-suited to implement modular structure control system.

Chapter 5: Modular Fuzzy Gain Scheduling System

5.1. Introduction

Brief review of proposed modular adaptive mechanism was described in section 3.3. We now present detail structure of this mechanism in this Chapter. Our proposed modular structure adaptive mechanism is strongly based on well-known conventional control theory and in next section, we discuss in detail how to define a variety of tuning strategies for different state of process response based on fuzzy system concept.

5.2. The Proposed Tuning Strategies for Our Modular Fuzzy Gain Scheduling Mechanism

Gain scheduling means a technique where controller parameters are changed during control in a predefined way. It is not merely a way to enlarge the operation area of a linear controller, but a way to improve performance of the control system. The system schedules the controller parameters according to predefined parameter values with respect to changing operation conditions. Scheduling variables can be the measurement signal, the controller output signal or an external signal that has influence on the current state of operation conditions.

Our proposed scheduling variables are qualitative information of current process response (described in Chapter 4) and measurement signals (error and its first difference). Detail explanations of different tuning strategies with respect to these scheduling variables are taken up in subsequent sections based on the relation of tuning strategies and states of process response that was described in section 3.3.

5.2.1. Tuning strategy for steady-state

PI controller (lag compensation) has the potential to increase the system accuracy and it is most suitable in steady-state (described in section 2.2). Referring to characteristic analysis of controller design (described in section 3.2), low loop gain is needed to improve the stability of the system in steady-state.

In our tuning strategy for this state, proportional gain and integral gain keep previous setting (i.e., $k_{pscale}(n) = k_{pscale}(n\text{-}1)$ and $k_{iscale}(n) = k_{iscale}(n\text{-}1)$) and derivative gain is set to zero ($k_{dscale}(n) = 0$). Our PID control system's behavior is like PI controller (only lag compensation) and digital form of our modular fuzzy gain scheduling PID controller becomes as the below Equation 5.1 (tuning strategy *T5* in Figure 3-7).

$$u_{pid} = k_p^{'}(n-1) \times e(n) + k_i^{'}(n-1) \times \sum_{i=0}^{n} e(i)Ts \tag{5.1}$$

5.2.2. Tuning strategy for rise time state

As discussed in section 3.3.2, there are three different tuning strategies (*T1*, *T2* and *T3*) in this state. At the beginning of rise time period, when error is very large and change in error is zero (approximately zero), it can be assumed as the delay period. Low proportional action and very low integral action (approximately zero) are set to minimize the delay effect (described in Section 3.2.1). Controller structure is like PD controller (lead compensation with low loop gain) and is captured in Equation 5.2 (tuning strategy *T1* in Figure 3-7).

When tracking error is relatively large, our tuning strategy emphasizes on performance improvement. Higher proportional action and derivative action are set because larger bandwidth (described in Section 3.2.4) supports to improve transient response (tuning strategy *T2* in Figure 3-7). In this state the setting of integral action depends on the process response speed (change in error).

Near the set point, our controller structure is like PID control system (lag-lead compensation) with small proportional action (to improve stability), large derivative action (to improve stability and to reduce settling time) and sufficiently large integral action (to improve system accuracy). The variation of integral parameter depends on the change in error value so as to reduce settling time (tuning strategy *T3* in Figure 3-7).

$$u_{pid}(n) = k_p'(n) \times e(n) + k_d'(n) \times \frac{e(n)}{Ts} \tag{5.2}$$

5.2.3. Tuning strategy for overshoot/oscillation state

The setting of integral action and derivative action are important in this state. In our proposed strategy, gain scaling factor for proportional parameter keeps previous setting (i.e., $k_{pscale}(n) = k_{pscale}(n-1)$). The derivative action is set to maximum value (high derivative gain supports to improve stability and reduce overshoot). The setting of integral action is dependent on both region (*A1* or *A2* or *A3* or *A4* or *A5*) and value of change in error. For instance, large integral action is needed to protect overshoot in *A2* region (error is negative and change in error is negative) and undershoot in *A4* (error is positive and change in error is positive) region. Large integral action in *A3* region (error is negative and change in error is positive) leads to sluggish response (very small change in error value). In *A1* region (error is positive and change in error is negative), the adjustment of integral action is necessary to balance between undershoot and overshoot.

It is noticed that fine control resolution (i.e., small universe of discourse of output variable in fuzzy inference system) is needed since stability is important in this state. On the other hand, in order to reduce settling time and overshoot, the required integral action is different based on dynamical behavior of process response. It means that in oscillation state, the variation of integral action should be within a limited range (i.e., within a range of [0.5, 0.6]) and appropriate range (i.e., range of integral action is [0.5, 0.6] or [0.7, 0.8]) are different based on dynamical behavior of process response.

In our approach, an appropriate range for integral gain scaling factor is computed by defining *offset1* as a function based on change in error. Then the appropriate range for integral scaling factor in oscillation state is defined as:

$$[c - 0.05 \leq c \leq c + 0.05]$$

where

$$c = k_{iscale}(n-1) + offset1 \tag{5.3}$$

and $k_{iscale}(n-1)$ is the previous integral gain scaling factor. The digital form of our controller design in this state can be expressed as Equation 5.4 (tuning strategy *T4* in Figure 3-7).

$$u_{pid} = k_p'(n-1) \times e(n) + k_i'(n) \times \sum_{i=0}^{n} e(i) \times Ts + k_d \times \frac{e(n)}{Ts} \tag{5.4}$$

5.2.4. Tuning strategy for disturbance state

To improve the control performance under load disturbance, gain should be sufficiently large because large loop gain is necessary to reduce sensitivity (Equation (3.9) in Section 3.2.2). There are two distinct objectives in this state. First one is to remove the disturbance effect as soon as possible (i.e., diverging from the set point) and another one is to maintain the stability near the set point (i.e., approaching to the set point).

In first tuning strategy, in order to get large loop gain, proportional gain and derivative gain is suddenly set to largest value (i.e., the same as original value of k_p and k_d) and the increment of integral action is function of change in error. This tuning strategy (tuning strategy *T6* in Figure 3-7) continues until it reaches the maximum perturbation (i.e., change in error value is approximately zero). In second tuning strategy (tuning strategy *T7* in Figure 3-7), the adjustment of proportional action and derivative action are the same as rise time state tuning strategy, but integral action scaling factor keeps previous value because large variation of integral action can affect stability of the system in this state.

5.3. The Required Fuzzy Inference Systems for Proposed Tuning Strategies

In the previous section, it is seen that multiple modules are needed in order to implement above mentioned tuning strategies. In our approach, multiple fuzzy inference systems are used to represent these gain tuning strategies because fuzzy inference system produces smooth transaction and is a way to incorporate the knowledge.

Table 5.1 illustrates the corresponding fuzzy inference system in different process state. Fuzzy inference system f_1 is responsible to produce proportional gain scaling factor, Fuzzy inference systems f_2, f_4 and f_5 produce integral gain scaling factors and fuzzy inference system f_3 is responsible to produce derivative gain scaling factor respectively. Fuzzy inference system f_6 produces *offset1* value that supports to set center of output variable of fuzzy inference system f_4 (Equation 5.3). Fuzzy inference system f_7 is responsible to produce *offset2* value in order to compute increment of integral action in initial moment of disturbance state.

Table 5-1. The appropriate fuzzy inference systems for different process state.

Tuning Strategy	Fuzzy Inference Systems	Process State
T1	f_1, f_3	Rise time (Delay period)
T2	f_1, f_3, f_5	Rise time (Far from the set point)
T3	f_1, f_2, f_3	Rise time (Near the set point)
T4	f_4, f_6	Overshoot/Oscillation
T5	-	Steady-state
T6	f_7	First tuning strategy in Disturbance
T7	f_1, f_3	Second tuning strategy in Disturbance

The main difference between tuning strategies *T1* and *T7* is very low integral action in *T1* and sufficiently large integral action in *T7*. Accordingly, seven fuzzy inference systems are needed in our modular adaptive mechanism. Fuzzy inference systems f_1, f_2, f_3 and f_4 are function of error and change in error (i.e., two-inputs/one-output), and Fuzzy inference systems f_5, f_6 and f_7 are function of change in error (i.e., one-input/one-output). In order to develop each of these fuzzy inference systems, firstly we analyze the structure of fuzzy inference system.

5.4. The Structure of Fuzzy Inference System

Fuzzy system consists of a number of rules that specify linguistic relation between the linguistic labels of input and output variables of the system. The working environment of the fuzzy system is of a linguistic nature. However, the majority of engineering problems involve only numerical variables, such as in case of control and modeling. To apply the fuzzy system to those numerical environments, two procedures, namely fuzzification and defuzzification are normally employed. Every fuzzy system is composed of four principal blocks (Figure 5-1): (*i*) Fuzzification module, (*ii*) knowledge base, (*iii*) inference engine and (*iv*) defuzzification module.

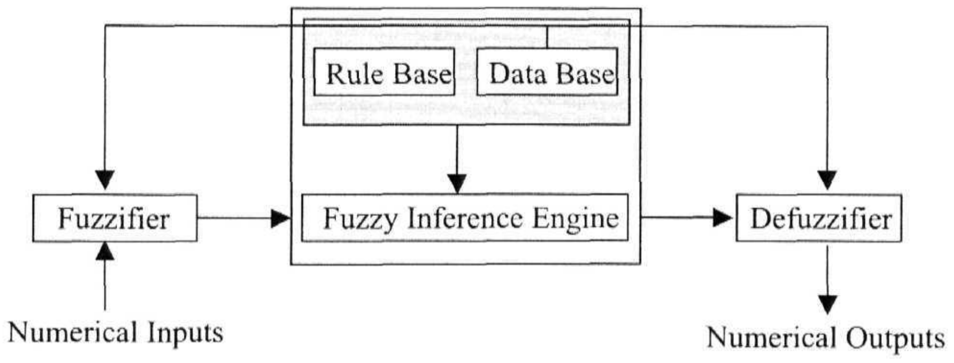


Fig 5-1. Block diagram of fuzzy inference system.

Fuzzifier block converts a set of crisp variables into a set of fuzzy variables to enable the application of logical rules. It performs two functions: (*i*) a scale transformation (i.e., an input normalization) which maps the physical values of the current process state variables into a normalized universe of discourse (normalized domain) and (*ii*) fuzzification which converts a crisp current value of a process state variable into a fuzzy set.

Defuzzifier block is responsible to convert fuzzy state value back into numerical environments. It includes two processes: (*i*) defuzzification and (*ii*) output denormalization. The most often used defuzzification methods are Center-of-Area/Gravity, Center-of-Sums, Center-of-Largest-Area, First-of-Maxima, Middle-of-Maxima and Height defuzzification [Driankov01].

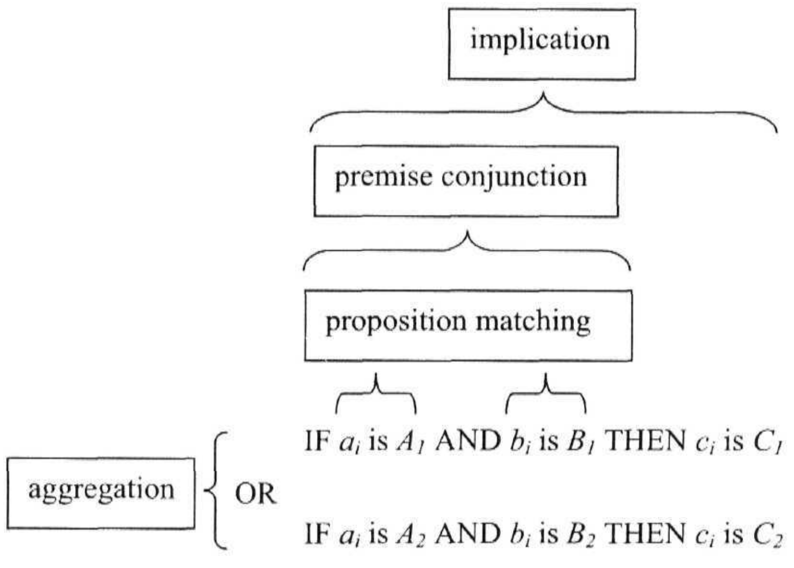

implication
premise conjunction
proposition matching
IF a_i is A_1 AND b_i is B_1 THEN c_i is C_1
aggregation
OR
IF a_i is A_2 AND b_i is B_2 THEN c_i is C_2

Fig. 5-2. Steps of fuzzy inference algorithm and corresponding linguistic operators.

Rule base stores a set of logical *IF-THEN* rules and data base stores parameters of membership functions. The basic function of the data base is to provide the necessary information for the fuzzification module and the defuzzification module. The information includes the place and the shape of membership functions. The designer can be attracted to try many different shapes of membership function. However, usually very simple membership functions can produce the control performance wanted. The most important point in the selection of the membership functions is the transparency of the obtained controller. The most popular choice for the shape of the membership function includes triangular, trapezoidal and bell-shape functions.

The basic function of the rule base is to represent in a structured way the control policy of an experienced process operator and/or control engineer in the form of a set of production rules. These two bases (rule base and data base) can be regarded as the linguistic layer (or knowledge base) of the system. The inference engine defines mapping from input fuzzy sets to output fuzzy sets. It consists of six steps [Riid02] and the Figure 5-2 illustrates the steps of fuzzy inference algorithm and corresponding linguistic operators.

Design of fuzzy systems may be seen as a general algorithm consisting of six steps [Yager94] (there are five steps in Li proposed idea [Li95]) and steps *(i)-(v)* can be regarded as the structure determination of the fuzzy system.

(i) Selection of the input and output variables;

(ii) Selection of the appropriate reasoning mechanism for the formalization of the fuzzy model;

(iii) Determination of the universe of discourse;

(iv) Determination of the linguistic labels into which the variables are partitioned;

(v) Formation of the set of linguistic rules that represent the relationships between the system variables;

(vi) Evaluation of system adequacy.

Briefly, design of the fuzzy inference system means selection of fuzzy rule base structure that includes the number of fuzzy sets for each input and output. In next section we discuss about the structure of each of required fuzzy inference systems for our modular adaptive mechanism.

5.5. The Proposed Structure of Fuzzy Inference System

We divided the whole development task of fuzzy inference system into three main categories: (i) selection of inputs and outputs variables (required inputs/outputs and universe of discourses), (ii) selection of fuzzy parameters (membership functions type and linguistic labels with appropriate partitions) and (iii) selection of rule base.

5.5.1. Selection of input output variables

In practice, there are several signals, which should be taken into account to compute control signals. In the selection of input variables, it is necessary to restrict the number of variables because the more inputs can cause the larger rule base. The number of output variables does not affect the size of the rule base and they are dependent on problem at hand.

Usually the design problem is well-defined with respect to the output variables of the fuzzy system (i.e., the input signals which affect the process output are known). Most commonly, the error and the change in error are used to determine a better-suited control input for the open-loop process. Li introduces [Li95] that a phase plane method can bridge the gap between the process dynamics and rule base of fuzzy logic controller. The fundamental goal of this architecture is to generate a control input based upon the error information resulting from the difference between the input signal and feedback signal.

In our approach, phase plane method provides some information to proposed modular gain scheduling mechanism. Consequently, error and its first difference are used as inputs for our fuzzy inference systems. Our modular gain scheduling mechanism produces three parameter scaling factors (k_{pscale}, k_{iscale} and k_{dscale}) and they are defined as output variables of fuzzy inference systems.

Input normalization processes help to define universe of discourses for input variables. The appropriate scaling factors for these normalization processes are set based on several experiments. Figure 5-3 describes the variation of change in error value for different types of process responses (i.e., sluggish response and aggressive response) in different states. Based on these results, the normalized scaling factors for each input of fuzzy inference systems are selected and is tabulated in Table 5.2.

Apart from fuzzy inference system f_4, the universe of discourse of each input variable is defined within the range $[0, 1]$ and input range of fuzzy inference system f_4 is $[-1, 1]$. Based on extensive simulation study on various processes, the universe of discourse of output variable of fuzzy inference system (f_1) is defined within the range $[0.75, 1.0]$ because very small proportional gain can cause sluggish response in rise time state and in worst case it can become unstable system (i.e., sufficient energy is needed to keep stability of the system). Detail explanation of the selection of the universe of discourse of output variable of fuzzy inference system (f_1) is described in Appendix I.

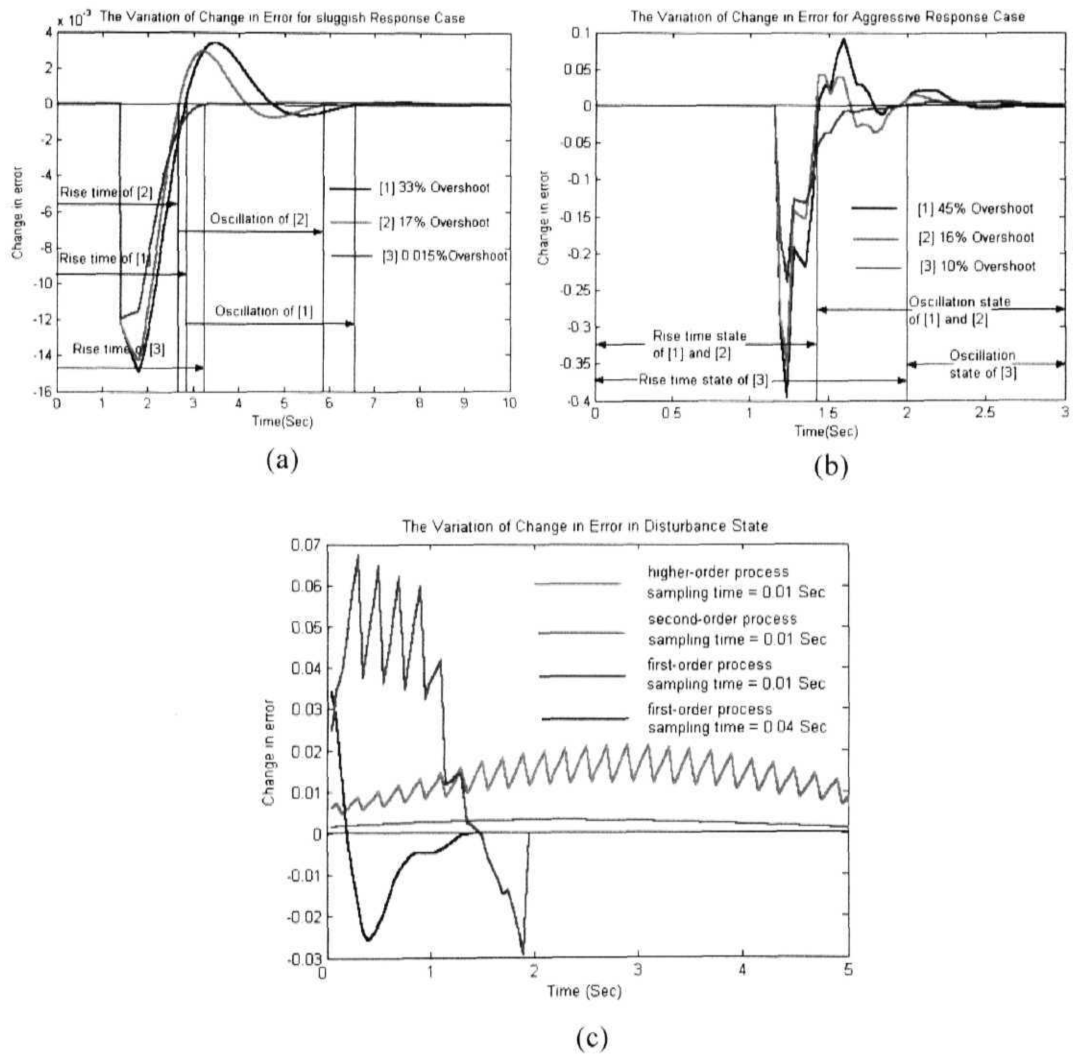


Fig. 5-3. The variation of change in error for (a) aggressive response case, (b) sluggish response case, (c) disturbance state.

The output range of fuzzy inference system f_2 is $[0.2,\ 1.0]$ because appropriate setting of integral action is a trade-off between undershoot and overshoot near the set point in rise time state. Fuzzy inference system f_3 is defined within the range $[0.8, 1.0]$ due to large derivative gain is necessary to improve performance in rise time state. Fuzzy inference system (f_4) produces the output within a limited range (i.e., $c - 0.05 \leq c \leq c + 0.05$) as described in Equations 5.4. The output range of fuzzy inference system f_5 is $[0.0, 1.0]$, fuzzy inference system f_6 that produces *offset1* value, is $[-0.2, 0.1]$ and fuzzy inference system f_7 that produce *offset2* value, is $[0.0, 0.4]$ respectively.

Table 5-2. Normalized scaling factors of input variables.

Inputs	Fuzzy Inference Systems						
	f_1	f_2	f_3	f_4	f_5	f_6	f_7
e	1.0	0.25	1.0	1.0	-	-	-
de	0.75	0.3	0.75	0.1	0.75	0.75	0.1

5.5.2. Selection of fuzzy parameters

Membership functions play a central role in converting qualitative linguistic labels into quantitative graded values. They form the basis for fuzzy computing. The shape, the overlapping, peak values and their continuity properties determine how the fuzzy system should be designed and how it behaves. For computation efficiency, efficient use of memory and performance analysis needs a uniform representation of the membership functions is required. This uniform representation can be achieved by employing the same type of membership function with uniform shape.

The overlap of input membership functions is also one of the most important factors influencing interpolation in fuzzy systems. Some works suggested using within a minimum of 25% and a maximum of 75% has been established experimentally [Riid02].

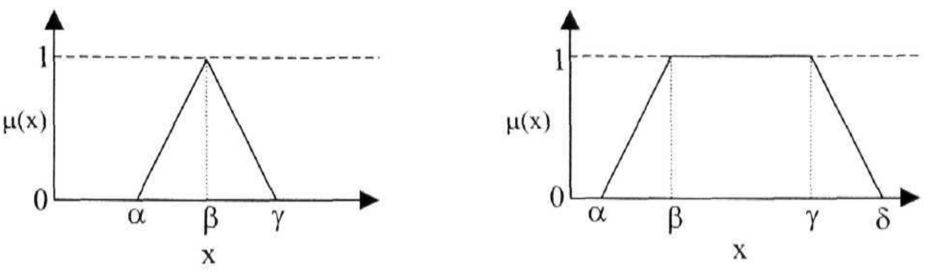
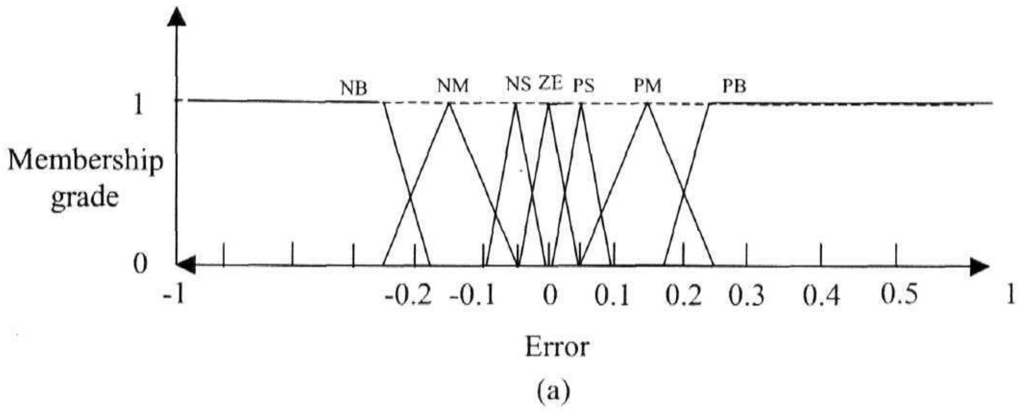


Fig. 5-4. The standard membership function types (a) Triangular and (b) Trapezoidal (*x* is crisp input and *μ(x)* is membership grade).

In our approach, most of the membership functions are uniform triangular shape with 50% overlap (in some cases less than 50%), but trapezoidal membership functions are selected for some cases (i.e., to define a partition properly in oscillation state as shown in Figure 5-5) based on the important characteristics of input variables (i.e., more than *±2%* of the final value can be defined as very big oscillation). The function of triangular membership function type is illustrated in Figure 5-4(a) and Equation 5.5. Figure 5-4(b) and Equation 5.6 describe the function of trapezoidal function.

$$f(x;\alpha,\beta,\gamma)=\begin{cases} 0, & for\, x<\alpha \\ \dfrac{x-\alpha}{\beta-\alpha}, & for\, \alpha\le x\le\beta \\ \dfrac{\gamma-x}{\gamma-\beta}, & for\, \beta\le x\le\gamma \\ 0, & for\, x>\gamma \end{cases} \tag{5.5}$$

$$f(x;\alpha,\beta,\gamma,\delta)=\begin{cases} 0, & for \quad x<\alpha \\ \dfrac{x-\alpha}{\beta-\alpha}, & for \quad \alpha\le x\le\beta \\ 1, & for \quad \beta\le x\le\gamma \\ \dfrac{\delta-x}{\delta-\gamma}, & for \quad \gamma\le x\le\delta \\ 0, & for \quad x>\delta \end{cases} \tag{5.6}$$



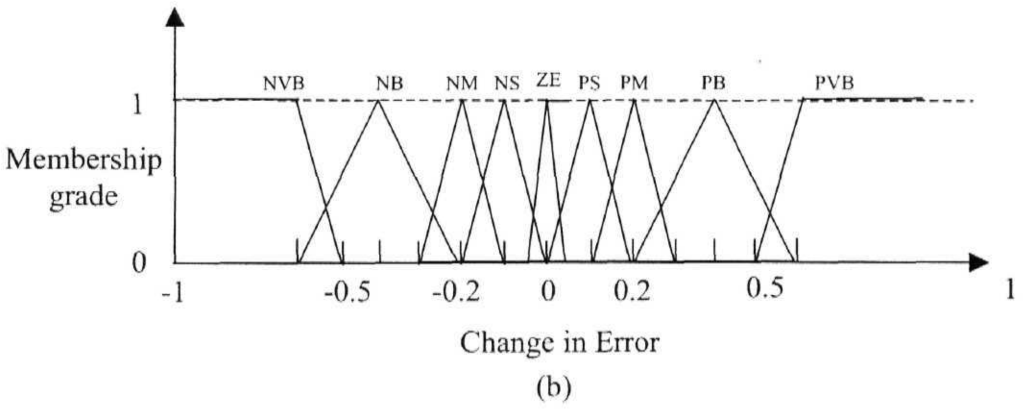
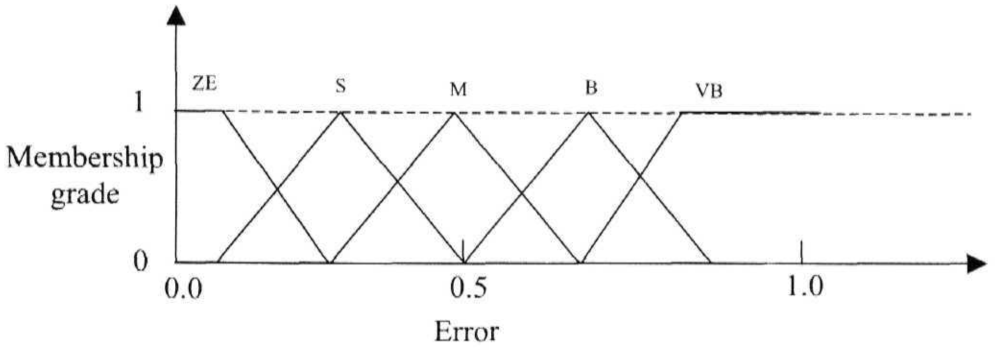


Fig. 5-5. Membership functions of fuzzy inference system f_4 (oscillation state) for (a) input variable error (normalized scaling factor = 1) and (b) input variable change in error (normalized scaling factor = 0.1) (NVB = Negative Very Big, NB = Negative Big, NM = Negative Medium, NS = Negative Small, ZE = Zero, PS =Positive Small, PM = Positive Medium, PB = Positive Big, PVB = Positive Very Big).

The change in error is more sensitive in oscillation state. As shown in Figures 5-3(a) and 5-3(b), most of the change in error values are less than *0.05* (normalized value is 0.5). Referring to these results, we define the partitions of change in error input for overshoot/oscillation state as shown in Figure 5-5 (b).



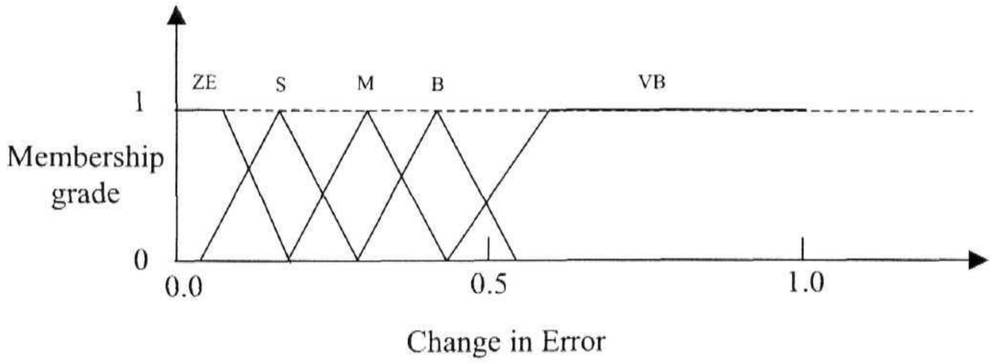


Fig. 5-6. Membership functions of each of fuzzy inference systems (apart from f_4) for (a) input variable error and (b) input variable change in error (ZE = Zero, S = Small, M = Medium, B = Big, VB = Very Big).

The membership functions of the remaining input (error variable) in oscillation state is set based on priori knowledge (i.e., more than 20% overshoot/oscillation is defined as excessive oscillation) as shown in Figure 5-5(a). The membership functions of input variables for remaining fuzzy inference systems are described in Figure 5-6. The appropriate membership functions of output variables for each of fuzzy inference systems are described in Figure 5-7.

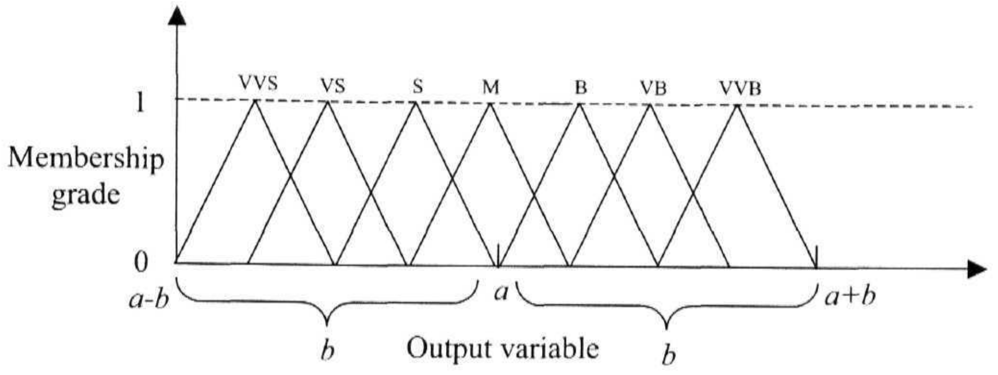


Fig. 5-7. Membership functions of output variable for each of fuzzy inference systems (value of variable a and b are dependent on fuzzy inference system, for example, $a=0.6$ and $b=0.4$ for fuzzy inference system f_2) (VVS = Very Very Small, VS = Very Small, S = Small, M = Medium, B = Big, VB = Very Big, VVB = Very Very Big).

5.5.3. Selection of rule base

The inference engine of our scheme is mainly based on Mamdani type inference because this is the most common and robust fuzzy reasoning method in fuzzy control applications [Mann99, Drainkov01]. The success and performance of rule base control systems depend largely on the availability and the performance of the rule base. An obvious and fundamental question arising from the real implementation of rule based control systems is how a set of control rules can be derived.

The development of rule base in this approach is based on domain knowledge (proposed tuning strategy) that was described in section 5.3. Fuzzy inference system f_1 is responsible to produce scaling factor for proportional gain (Equation 3.19a, Section 3.4.1). The appropriate rules for fuzzy inference system f_1 are described in Table 5-3 (detail explanation is described in Appendix II):

Table 5-3. Proposed rules for fuzzy inference system f_1.

		Error				
		ZE	S	M	B	VB
Change in Error	ZE	M	M	VVB	VVB	VVS
	S	S	M	VVB	VVB	VVB
	M	VS	M	VB	VVB	VVB
	B	VS	S	B	VB	VB
	VB	VVS	S	B	B	VB

Table 5-4. Proposed rules for fuzzy inference system f_2.

		Error				
		ZE	S	M	B	VB
Change in Error	ZE	M	M	B	VB	VVB
	S	S	M	B	B	VB
	M	S	M	M	B	B
	B	VS	S	M	M	B
	VB	VS	S	S	M	M

Table 5-4 illustrates proposed rules for fuzzy inference system f_2 that produces scaling factor for integral gain (Equation 3.19b, Section 3.4.1). It's responsibility is to balance between overshoot and undershoot at near the set point.

Table 5-5. Proposed rules for fuzzy inference system f_3.

		Error				
		ZE	S	M	B	VB
Change in Error	ZE	VB	B	M	S	VS
	S	VB	B	M	S	VS
	M	VB	VB	B	M	S
	B	VVB	VB	B	M	M
	VB	VVB	VB	VB	B	M

Table 5-6. Proposed rules for fuzzy inference system f_4.

		Error						
		NB	NM	NS	ZE	PS	PM	PB
Change in Error	NVB	VVB	VVB	VB	VB	VS	S	S
	NB	VVB	VB	VB	B	S	S	M
	NM	VB	VB	B	B	M	M	M
	NS	VB	VB	B	M	M	M	B
	ZE	VB	B	M	M	M	B	B
	PS	B	M	M	S	M	B	B
	PM	M	M	S	S	M	B	VB
	PB	M	M	S	VS	B	VB	VB
	PVB	S	S	VS	VS	B	VB	VVB

Near the set point, high derivative gain reduces overshoot and improves stability. It reshapes the frequency response curve to provide sufficient phase-lead angle to offset the phase lag that is caused by the process. The proposed rules for fuzzy inference system f_3 are shown in Table 5-5 and it produces scaling factor for derivative gain (Equation 3.19c, Section 3.4.1)

Fuzzy inference system f_4 produces a scaling factor to adjust integral action in oscillation state. The control policy of this fuzzy inference system is dependent on sign of error and its first difference. When process response departs from set point (error is positive and change in error is positive (*A4* region) or error is negative and change in error is negative (*A2* region), high integral action is required to eliminate this effect as soon as possible and when process response approaches to set point the required integral action is dependent on error's first difference.

In rise time state, when error is relatively large, fuzzy inference system f_5 balances between performance and stability (speed of the response and overshoot) with respect to the function of change in error because change in error value is related with integration interval (i.e., small change in error is the same as large integration interval and it can cause large overshoot). The appropriate rules are tabulated in Table 5-7.

Table 5-7. Proposed rules for fuzzy inference system f_5.

	Change in Error				
	ZE	S	M	B	VB
Output	VVS	S	M	M	M

The adjustment of fuzzy inference system f_6 helps to reach an appropriate interval of integral action and can activate only at the initial moment of oscillation state. The fuzzy inference system f_7 is responsible to adjust integral gain (low frequency gain) in disturbance state. It is a trade-off between performance and stability because large integral gain (low frequency gain) is required to eliminate disturbance effect, but it leads to become unstable. Table 5-8 presents required rules for fuzzy inference system f_6 and proposed rules for fuzzy inference system f_7 are described in Table 5-9.

Table 5-8. Proposed rules for fuzzy inference system f_6.

	Change in Error				
	ZE	S	M	B	VB
Output	PM	PM	PS	ZE	NS

Table 5-9. Proposed rules for fuzzy inference system f_7.

	Change in Error				
	ZE	S	M	B	VB
Output	S	M	M	B	VB

5.6. The Proposed Online Gain Scheduling Algorithm

As we discussed earlier, there are seven fuzzy inference systems in our proposed modular structure adaptive mechanism. It produces three gain scaling factors to adjust gains of conventional PID controller. The appropriate fuzzy inference systems are selected online based on both qualitative information (estimated current process response) and quantitative information (value of error signal) of process response. Brief explanation of overall structure of our proposed online gain-scheduling algorithm is as follows (Modular neural network classifier process monitor estimates the current process response and only one state can be activated at any moment of time).

IF estimated current process response is **steady-state**,

$k_{pscale}(n) = k_{pscale}(n-1)$,

$k_{iscale}(n) = k_{iscale}(n-1)$,

$k_{dscale}(n) = 0.0$.

ELSE IF estimated current process response is **rise time state**,

```
        $k_{pscale}(n) = f_1$ (error, change in error),
        $k_{dscale}(n) = f_3$ (error, change in error),
        IF error is very large and change in error is zero (i.e., delay period)
            $k_{iscale}(n) = 0.0$.
        ELSE IF error is relatively large (i.e., far from the set point)
            $k_{isclae}(n) = f_5$ (change in error).
        ELSE (i.e., near the set point),
            $k_{iscale}(n) = f_2$ (error, change in error).
        [End Inner IF Loop]
ELSE IF current estimated process response is **overshoot/oscillation state**,
        IF previous state is not overshoot/oscillation state (i.e., initial moment of
        overshoot/oscillation state), compute parameter $c$ (parameter $c$ is the center of
        output membership function of $f_4$).
            offset1 = $f_6$ (change in error),
            $c = k_{iscale}(n-1) + offset1$.
        [End of Inner IF Loop]
            $k_{iscale}(n) = f_4$ (error, change in error),
            $k_{pscale}(n) = k_{pscale}(n-1)$,
            $k_{dscale}(n) = 1.0$.
ELSE IF current estimated process response is **disturbance state**,
        IF it is not reaching the maximum perturbation (first tuning strategy),
            IF previous state is not disturbance state (i.e., initial moment of
                disturbance state, compute offset2 to increase integral gain.
                Offset2 = $f_7$ (change in error),
                $k_{iscale}(n) = k_{iscale}(n-1) + offset2$,
                IF $k_{iscale}(n) > 1.0$,
                        $k_{iscale}(n) = 1.0$.
                [End of Inner IF Loop]
            ELSE
                $k_{iscale}(n) = k_{iscale}(n-1)$.
            [End of Inner IF Loop]
            $k_{pscale}(n) = 1.0,$
            $k_{dscale}(n) = 1.0.$
    ELSE (i.e., second tuning strategy),
            $k_{pscale}(n) = f_1(error, change\ in\ error),$
            $k_{dscale}(n) = f_3(error, change\ in\ error),$
            $k_{iscale}(n) = k_{iscale}(n-1).$
    [End Inner IF Loop]
[End Outer IF Loop]
```

5.7. Discussions

Usually the controller must be retuned at different operation conditions. Thus it is necessary to define the controller parameter transition policy with respect to dynamical behavior of process response. This work presents a novel approach to modular fuzzy gain scheduling technique for conventional PID controller (that is widely used in process control).

In our approach, modular fuzzy adaptive mechanism represents the controller parameter transition policy of different operating state and provides smooth transaction between operations. The proposed modular fuzzy gain scheduling mechanism consists of seven fuzzy inference systems and they are responsible for different characteristics of process responses. Both qualitative information (current process response) and quantitative information (value of error signal) are utilized as a switching system among these fuzzy inference systems.

Since performance of this kind of controller largely depends on rule base, the proposed rule base is developed based on both frequency response method and time domain analysis that are practical and important approach to control system. Both priori knowledge (existing control theory) and data (measurement) provide essential information to select appropriate inputs/outputs and fuzzy parameters. The necessary input data (i.e., change in error) is investigated with several experiments in order to implement relevant membership functions type, partition and normalization factors.

Since the structure of fuzzy gain scheduling is modular, it makes the system easier to understand, modify and incorporate the expert's knowledge or existing theory. It allows tuning of controller's parameters independently for each of the different characteristics of process responses (i.e., a delay period, rise time period and disturbance period) and achieves greater flexible relation between inputs and outputs. The main benefit of our approach is that it provides a flexible and transparent mechanism for gain scheduling. It not merely improves the performance of the process, but very useful to analyze the effect of individual parameters of conventional PID controller in various different process states.

Chapter 6: Experiments and Results

6.1. Introduction

This chapter presents extensive experiments and simulations in order to show the effectiveness of our proposed controller design. These experiments are divided into two main groups. First group of experiments shows the accuracy and validity of qualitative information of process responses and second group of experiments describes the performance and flexibility of our proposed controller. To demonstrate the effectiveness of our proposed controller design, four types of process are used for illustration. They are:

(1) First-order plus dead time process

$$G_1(s) = \frac{e^{-\beta s}}{(s+a)} \tag{6.1}$$

(2) Second-order plus dead time process

$$G_2(s) = \frac{e^{-\beta s}}{(s+a)^2} \tag{6.2}$$

(3) Third-order process

$$G_3(s) = \frac{a}{(s+0.5)(s^2+1.64s+8.456)} \tag{6.3}$$

(4) Higher-order process

$$G_4(s) = \frac{a}{(s+1)(s+3)^3} \tag{6.4}$$

The required specifications for the experiments are given in Table 6-1 [Zhao93, Ambrosio02].

Table 6-1. Required data for our experiments.

	$G_1(s)$	$G_2(s)$	$G_3(s)$	$G_4(s)$
Proportional gain k_p	4.8021	3.2448	2.19	3.1806
Integral gain k_i	13.34	2.207	2.1262	2.356
Derivative gain k_d	0.432	1.19268	0.565	1.0718
Sampling time TS	0.04 Sec	0.01 Sec	0.01 Sec	0.01 Sec
Delay parameter β	0.2	0.4	-	-
Process parameter a	1	1	4.228	27

6.2. Knowledge-based Modular Neural Network Classifier Test

As we described in earlier chapters, the appropriate setting of controller parameters/structure are different based on the nature of process responses or control objectives. In Chapter 3, we decomposed the whole process response into four classes (rise time state, oscillation state, disturbance state and steady-state) in order to provide qualitative information to adaptive mechanism. Our proposed knowledge-based modular neural network classifier provides online this qualitative information of process response. It is used to switch the appropriate tuning strategy among a variety of tuning strategies.

Our aim of online process response estimation (qualitative information) is to apply in adaptive application. The stability of that kind of control system relies on estimation accuracy because it decides the appropriate tuning strategy and setting of parameters at that moment of time. We observe the accuracy and validity of proposed knowledge-based modular neural network classifier based on two types of experiments: (i) accuracy test and (ii) refinement algorithm test.

6.2.1. Accuracy test

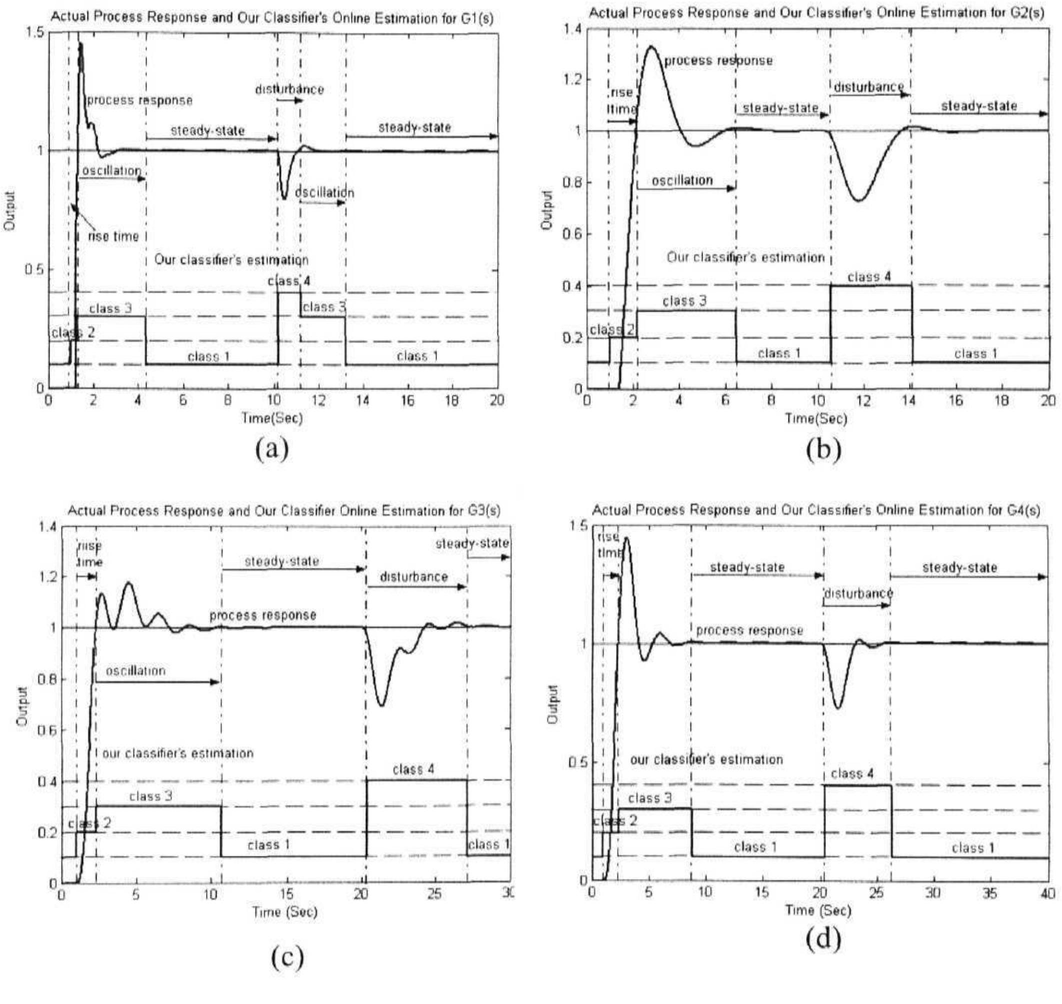


Fig. 6-1. The comparison of process response and our classifier's estimation (a) $G_1(s)$; (b) $G_2(s)$; (c) $G_3(s)$ and (d) $G_4(s)$.

In PID control system, online qualitative process response estimation would improve performance because there is a connection between parameters of PID controller and process responses (described in section 2.1.). Our goal of online qualitative estimation of process response is to switch to appropriate tuning strategy. There are seven different tuning strategies in our proposed modular adaptive mechanism (described in section 5.2) and they are developed based on the frequency response method and the nature of PID controller parameters in frequency domain.

In our work, a modular neural network classifier estimates online process response qualitatively. Firstly, we analyze the estimation accuracy of our proposed knowledge-based modular neural network classifier (described in section 3.2 and section 5.2).

Figures 6-1(a), (b), (c), and (d) show the comparison of actual process responses for four different types of process responses and our classifier's estimations. In Figures 6-1, *Class 1* represents steady-state, *Class 2* represents rise time state, *Class 3* represents oscillation state and *Class 4* represents disturbance state respectively. By observing these results, our classifier's estimation is almost same as the actual response. There are slight delays when process response changes from transient state to steady-state.

6.2.2. Refinement algorithm test

Although, our classifier design is based on well-known existing theory and principles, refinement of rules developed are necessary to guarantee the estimation accuracy. Our proposed knowledge-based refinement algorithm emphasizes on the set of discriminant function (Classifier-2 Block) that includes twenty individual classifiers (rules). There are two tests for incomplete domain knowledge and mismatch rule base, respectively. In this experiment and threshold value *th2* is defined as *0.9*.

In order to examine incomplete domain knowledge, one rule (9^{th}) is removed from Classifier-2 Block. During this test, our refinement algorithm examines the outputs of all remaining nineteen rules from Classifier-2 Block. Figure 6-2 shows activation of each rule at the specific time (the classifier is activated when its output is greater than threshold value *th2*).

Referring to this result, input subspace region is only partially matched with each rule at time *t* = *2.36* second (i.e., each rule produces less than threshold value *th2*). After which the knowledge refinement algorithm add on an independent rule for this input subspace based on target response (described in section 4.4).

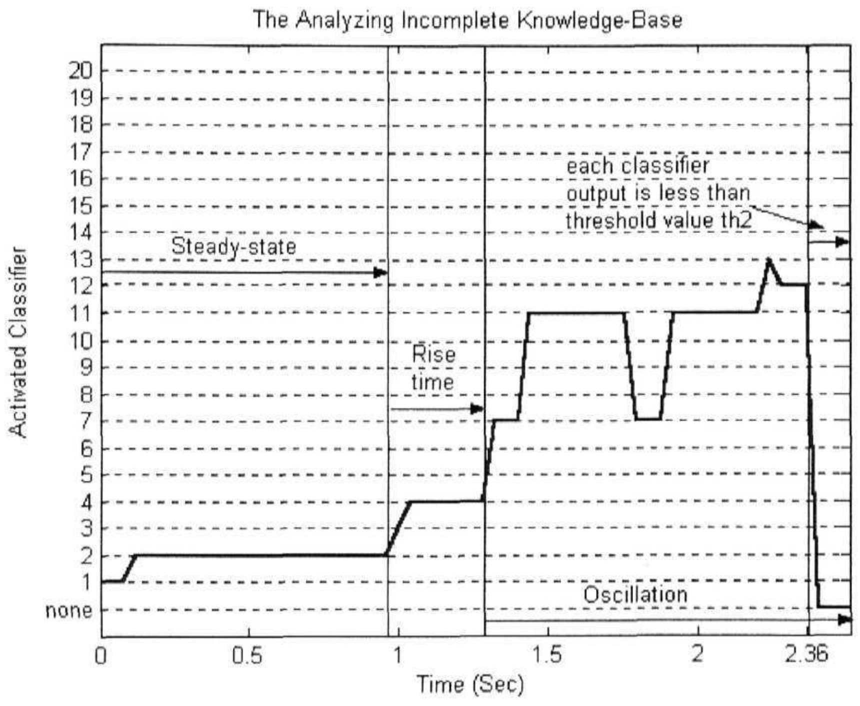


Fig. 6-2. The analyzing the incomplete knowledge-base.

In test 2, one additional rule (*21st*) is added into Classifier-2 Block. During this test, our algorithm notices that more than one classifier are totally matched with the current input subspace at time *t* = *2.28* second (outputs of more than one rule are greater than threshold value *th2*). Our refinement algorithm debugs which classifiers are responsible for that and it selects a classifier that is most suitable in the current situation (described in Example 4.1). The main restriction of our proposed refinement algorithm is that it is not possible to use this in online applications because training data sets (input/output data pairs) are required to detect these mismatch rules. The above two experiments reveal that the proposed refinement algorithm supports to improve the performance of classifier.

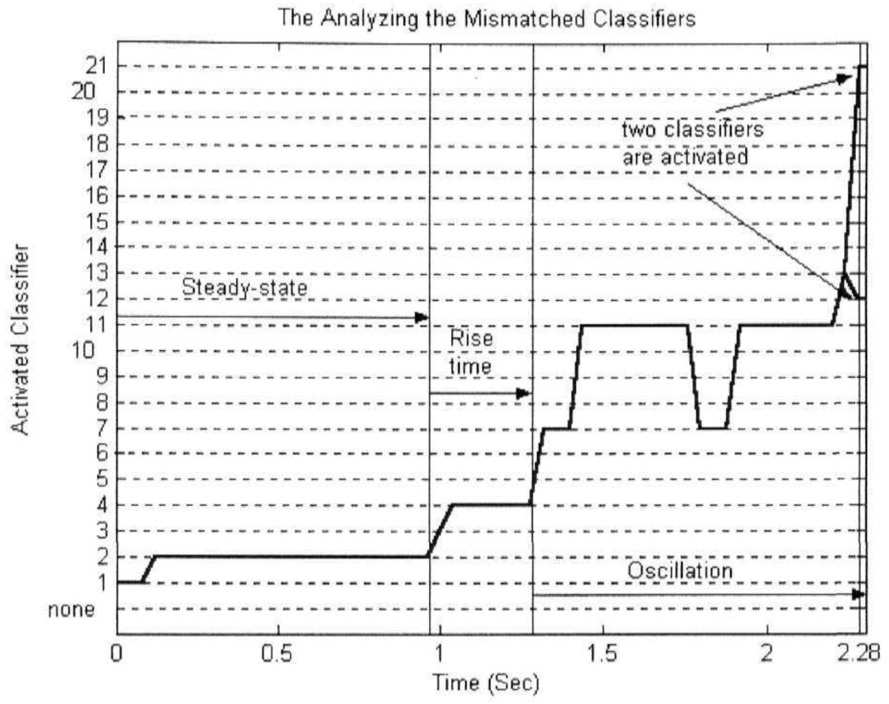


Fig. 6-3. The analyzing the mismatched rules (classifiers).

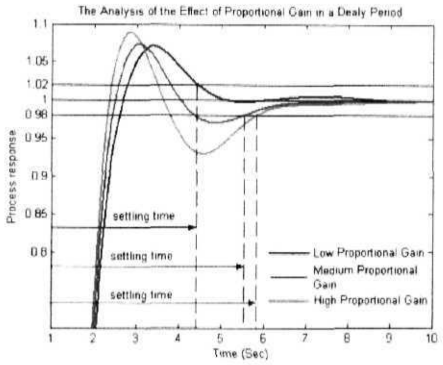
6.3. Performance Tests of Our Proposed Controller

In order to show effectiveness of our controller structure, the following three features are observed through several experiments.

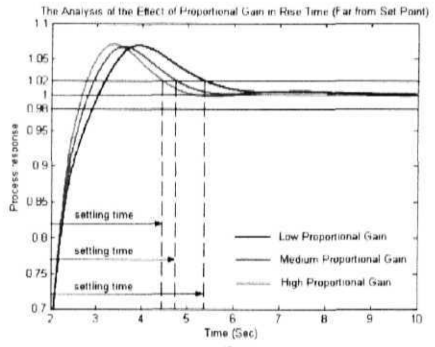
1. First, the flexibility and transparency of our controller structure,
2. Second, the transient and steady-state response of our design and compare it with the other reported work (it includes set point tracking, disturbance rejection, stability and steady-state accuracy) and
3. Third, the robustness (sensitivity) of our controller structure.

6.3.1. Flexibility and transparency characteristics test

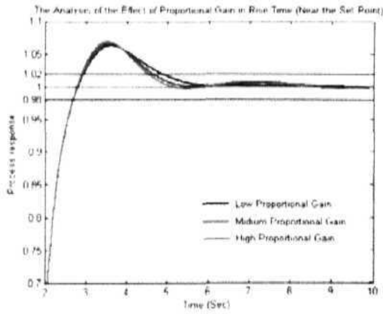
Flexibility and transparency are important characteristics for future high performance control system. Therefore, the observations of these characteristics are important in the controller design. In our proposed approach, there are seven fuzzy inference systems in modular structure gain scheduling mechanism and each fuzzy inference system has own tuning strategy (described in section 5.5). In this subsection, we analyze flexibility and transparency characteristics of proposed controller design based on processes $G_1(s)$ (Equation 6.1) and $G_2(s)$ (Equation 6.2) viz on our seven fuzzy inference system.



(a)



(b)



(c)

Fig. 6-4. The effect of fuzzy inference system f_1 (proportional gain) (a) in a delay period, (b) far from the set-point and (c) near the set-point.

Firstly, we analyze the effect of fuzzy inference system f_1 that is responsible to update proportional gain. The results (shown in Figures 6-4) show that high proportional gain in a delay period (tuning strategy *T1*) can cause oscillatory response. When process response is far from the set-point (tuning strategy *T2*), high proportional gain is needed to improve performance. Near the set-point (tuning strategy *T3*), low or medium proportional gain can improve stability without affecting the speed of the process response. It is noticed that the speed of process response can vary based on the different setting of proportional gain especially when process response is far from the set-point (Figure 6-4(b)).

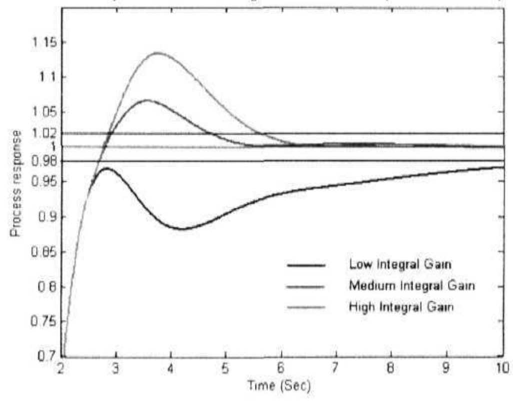
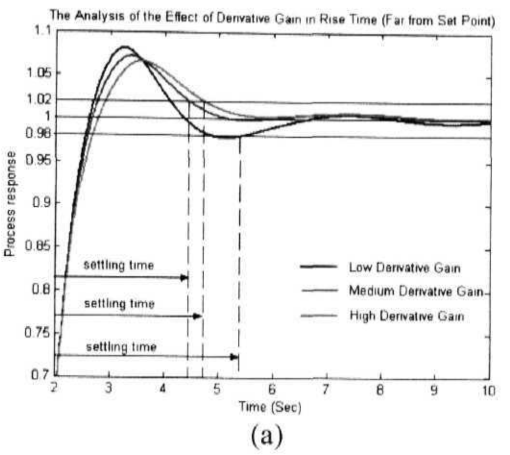


Fig. 6-5. The effect of fuzzy inference system f_2 (integral gain) near the set-point.

Near the set-point in rise time interval, sufficiently large integral action is necessary to remove steady-state error. Figure 6-5 shows that low integral action causes steady-state error and an excessive overshoot occurs by large integral action. In this state, integral action is very sensitive and our proposed fuzzy inference system f_2 is adjusted based on two numerical sensory inputs (error and its first difference). The effectiveness of our proposed fuzzy inference system f_3 (derivative gain) is described in Figures 6-6. These results show that near the set-point, derivative action is more sensitive to reduce overshoot and settling time.



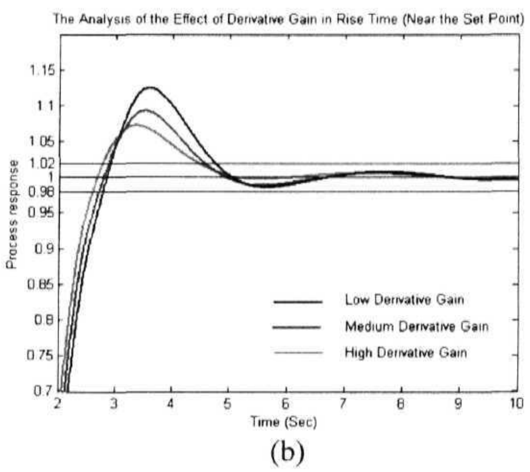


Fig. 6-6. The effect of fuzzy inference system f_3 (derivative gain) (a) far from the set-point and (b) near the set-point.

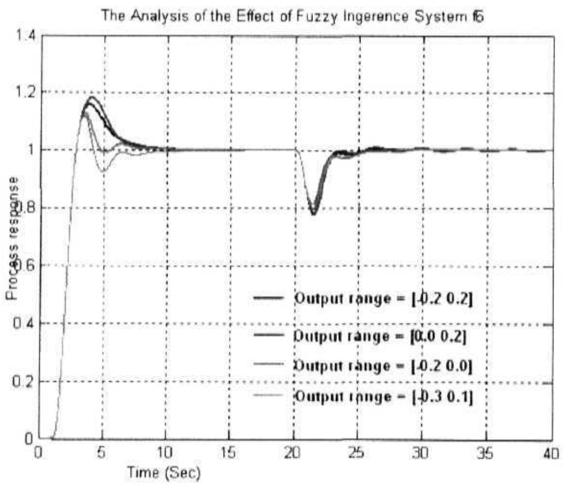


Fig 6-7. The effectiveness of proposed tuning strategy in oscillation state (f_4 and f_6)

The effectiveness of proposed tuning strategy in oscillation state (fuzzy inference systems f_4 and f_6) is described in Figure 6-7. The main objective of these fuzzy inference systems is to reach the steady-state as soon as possible. Therefore, fine control resolution (the small amount of gain variation) is necessary in order to keep stability. The proposed fuzzy inference system f_4 is responsible to produce fine control resolution with the help of fuzzy inference system f_6 (Described in section 5.2). In this experiment we observed the effectiveness of various output ranges of fuzzy inference system f_6. The result shows that appropriate setting of the output range of fuzzy inference system f_6 helps to reduce overshoot and settling time without affecting the other characteristics.

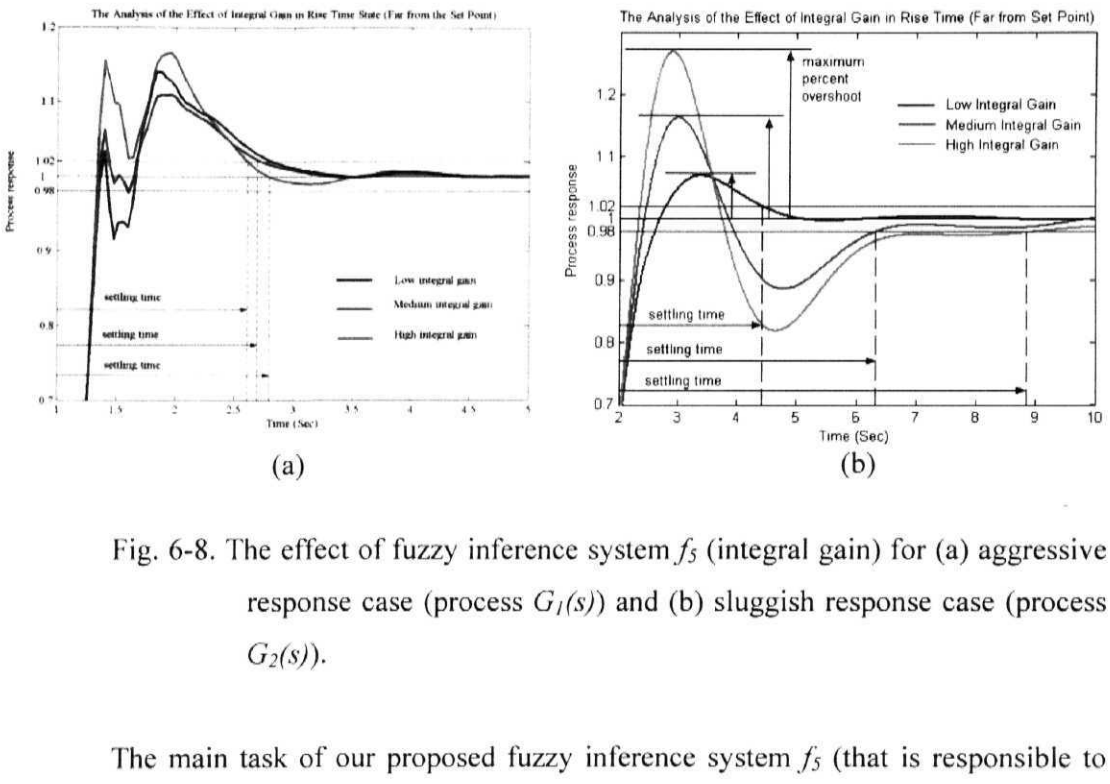


Fig. 6-8. The effect of fuzzy inference system f_5 (integral gain) for (a) aggressive response case (process $G_1(s)$) and (b) sluggish response case (process $G_2(s)$).

The main task of our proposed fuzzy inference system f_5 (that is responsible to adjust integral gain when far from set-point in rise time state) is to eliminate excessive overshoot and oscillation. The effect of this fuzzy inference system is described in Figures 6-8. These results show when far from the set-point, low integral gain is necessary to reduce excessive overshoot and oscillation. But, in aggressive response case (Figure 6-8(a)), low integral gain can cause more oscillatory response. In our approach, appropriate integral action is selected online based on the current value of change in error signal.

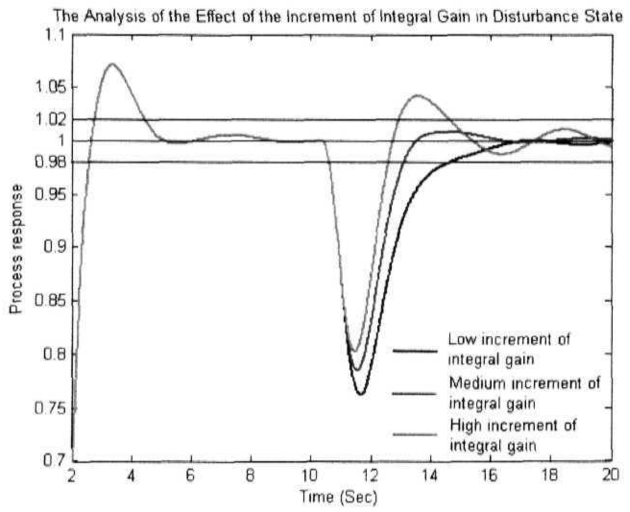


Fig 6-9. The effect of the variation of integral action in disturbance state.

Effect of disturbance is eliminated by low frequency gain (integral gain). Figure 6-9 shows the effect of the variation of integral action in disturbance state. High integral action can eliminate fast the disturbance effect, but it leads to unstable system. The proposed fuzzy inference system f_7 is responsible for producing appropriate integral action in order to remove disturbance effect.

These experiments reveal that the flexibility of our proposed controller structure. Based on these experiment results, each fuzzy inference system has individual task and they are less sensitive to other performance. The corresponding task of each fuzzy inference system can be described as following:

(1)	f_1	- to speed up the process response
(2)	f_2	- to eliminate steady-state error
(3)	f_3	- to eliminate overshoot and improve settling time
(4)	f_4	- to improve settling time
(5)	f_5	- to eliminate excessive overshoot and oscillation
(6)	f_6	- to eliminate overshoot and improve settling time
(7)	f_7	- to remove disturbance effect

In our approach, the specific requirement can be achieved by adjusting output range of appropriate fuzzy inference system. It is also easy to understand and analyze the relationship between the variation of conventional PID controller parameters and a variety of process responses.

6.3.2. Transient and steady-state response characteristics test

In many practical cases, the desired performance characteristics of control systems are specified in terms of time-domain quantities. We analyze the performance characteristics (transient and steady-state response of control system) based on transient response to a unit-step input and unit-step load since it is easy to generate and is sufficiently drastic. Steady-state accuracy, rise time, maximum percent overshoot and settling time are analyzed in these experiments. We extracted and reproduced some results from other reported works in order to compare with our work.

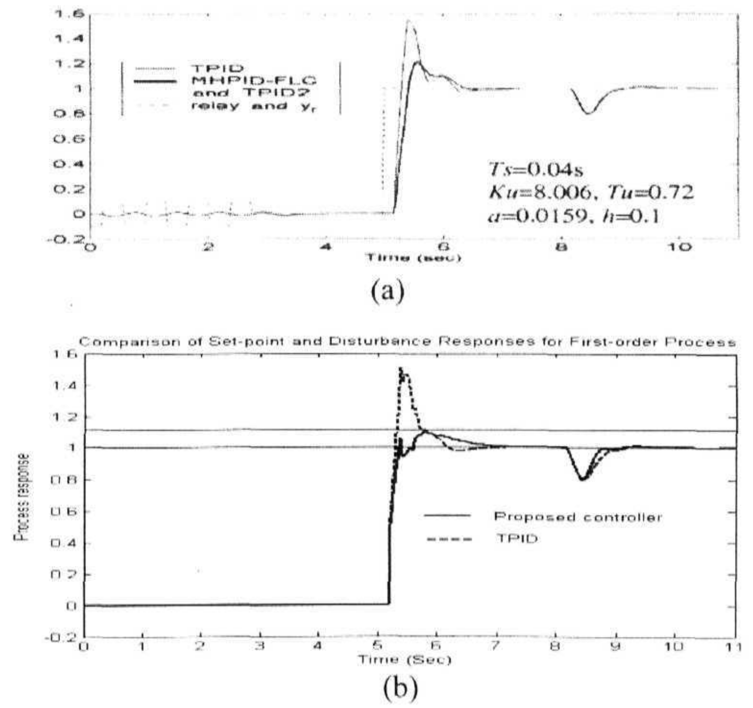
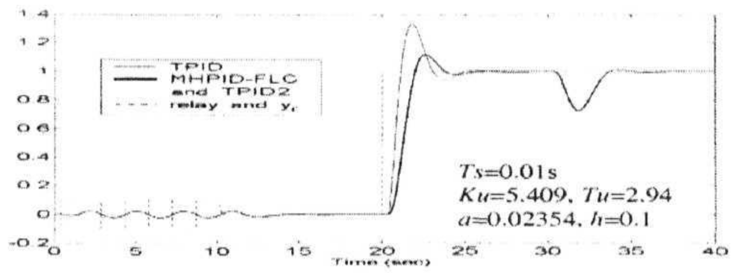
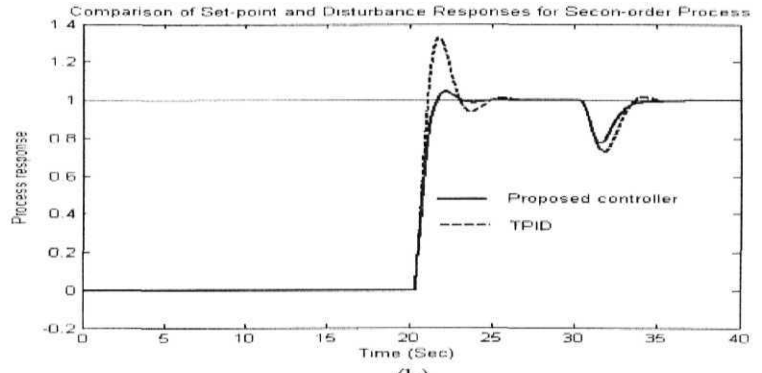


Fig. 6-10. The comparison of set point tracking and disturbance rejection for first-order process plus time delay (a) Ambrosio's work [Ambrosio02], (b) proposed work and TPID.

The performance of rise time and maximum percent overshoot normally conflict with each other. Ambrosio's [Ambrosio02] work is the same as conventional PID controller with set point weighting (set point factor is *0.5*). Though it can reduce the overshoot, rise time is increased. According to the results from Figures 6-10(b), 6-11(b), 6-13(a) and 6-14(b), obtained from our proposed idea, overshoot is obviously reduced without too much effect in rise time for three processes (apart from third-order process). The settling time (± 2% of final value) is also improved. Disturbance rejection behavior of our controller is somewhat better than traditional PID controller in second-order process, third-order process and higher-order process. However, previous reported works only emphasize on the set point tracking as shown in Figures 6-12(a), 6-12(b), 6-12(c), 6-13(b) and 6-14(c) [Zhao93, Chen98, Guzelkaya03]. These results show that our proposed controller design can handle both disturbance rejection and set point tracking.

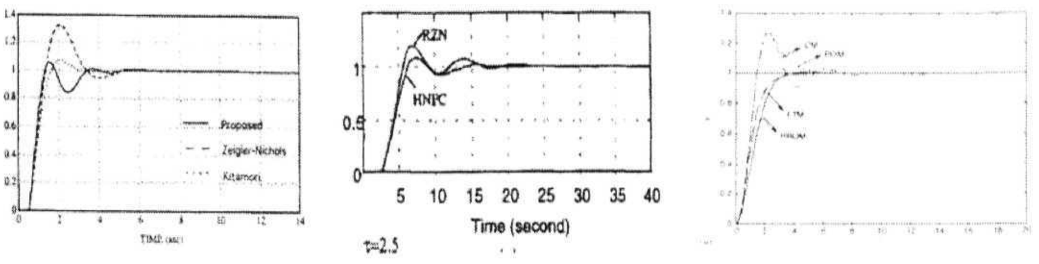


(a)



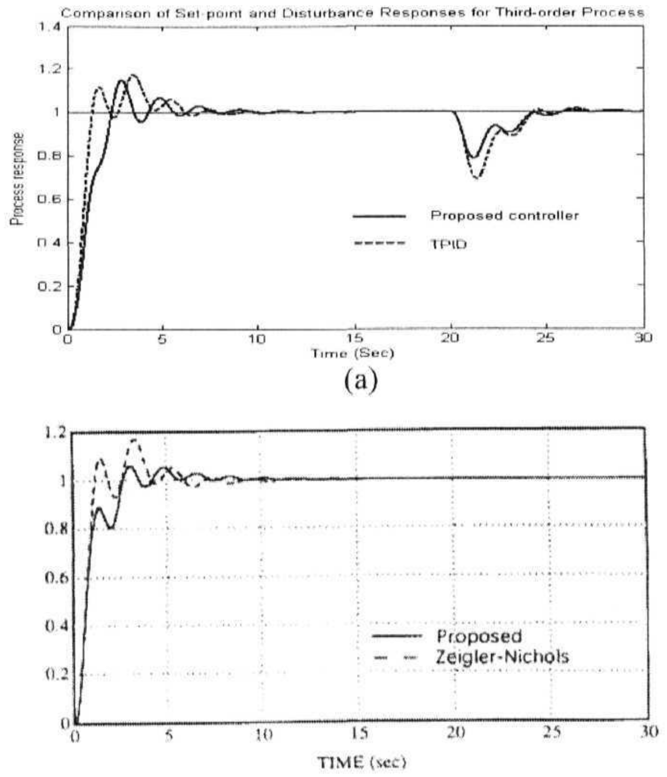
(b)

Fig. 6-11. The comparison of set point tracking and disturbance rejection for second-order process plus time delay (a) Ambrosio's work [Ambrosio02] and (b) proposed work and TPID.



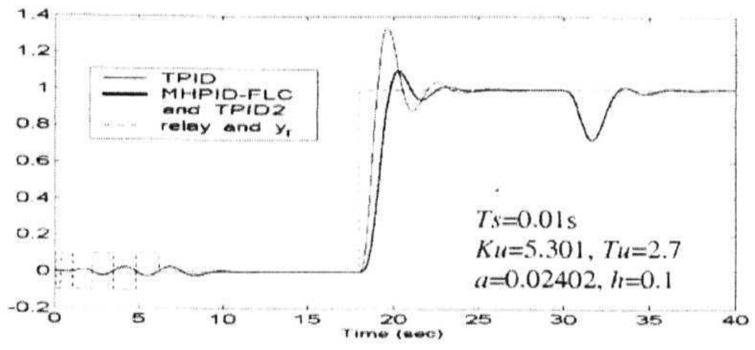
(a) (b) (c)

Fig. 6-12. The comparison of set point tracking for second-order process (a) Zhao's work (with time delay) [Zhao93], (b) Chen's work (with time delay) [Chen98], and (c) Guzelkaya's work [Guzelkaya03] (RROM = Relative Rate Observer Method).

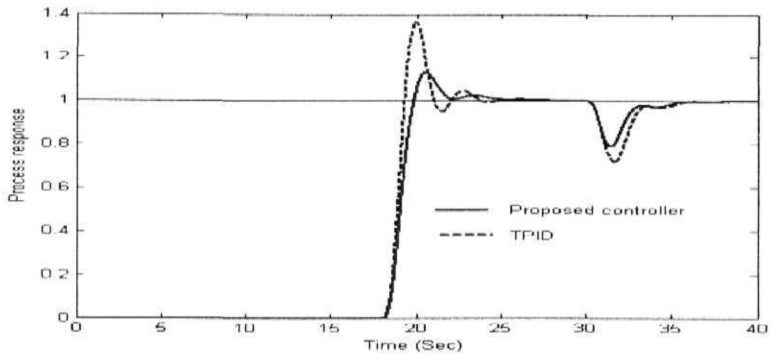


(b)

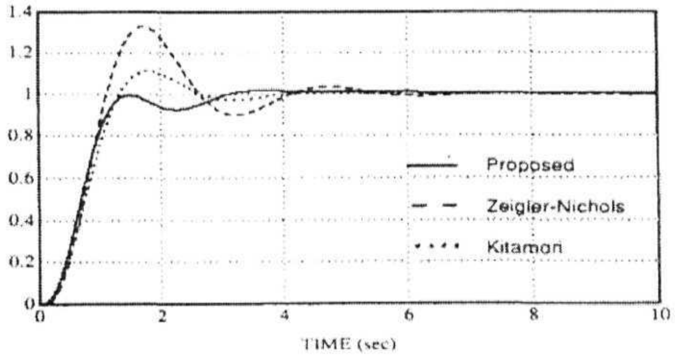
Fig. 6-13. The comparison of set point tracking for third-order process (a) proposed work and TPID (with disturbance rejection) and (b) Zhao's work [Zhao93].



(a)



(b)

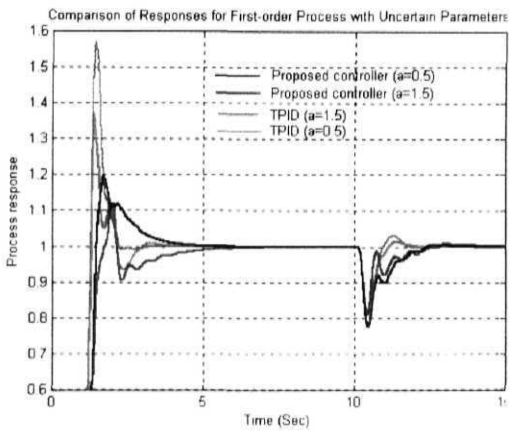


(c)

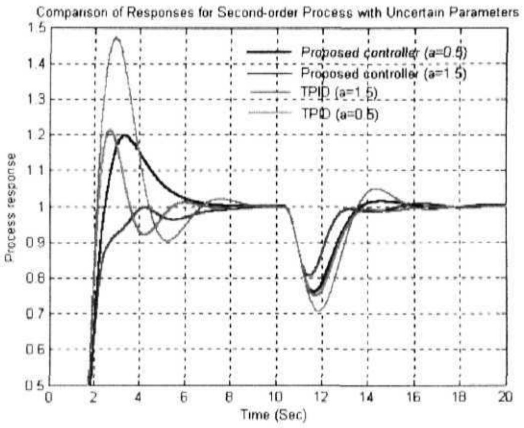
Fig. 6-14. The comparison of set point tracking and disturbance rejection for higher-order process (a) Ambrosio's work [Ambrosio02], (b) proposed work and TPID and (c) Zhao's work [Zhao93] (without disturbance rejection).

6.3.3. Robustness test

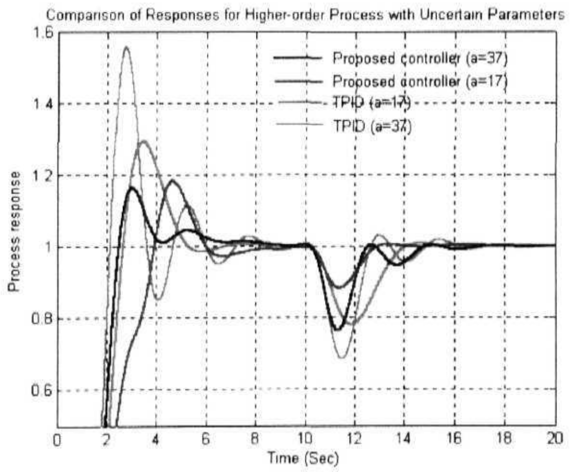
Many systems have several parameters that are constants but uncertain within a range. To ascertain stability of the system, investigation is necessary for all possible combinations of parameters. The sensitivity of a control system to a parameter variation is of prime importance that is known as robustness of controller design.



(a)



(b)



(c)

Fig. 6-15. The comparison of responses with different value of parameter a for (a) first-order process, (b) second-order process, (c) higher-order process.

A control system is robust when it has low sensitivities or it is stable over the range of parameter variations or the performance continues to meet the specifications in the presence of a set of change in the system parameters. The system should be able to withstand unexpected effects such as disturbance, unmodeled delay and unmodeled dynamics.

To analyze the robustness of our controller design we assume some parameters of system (parameter a and delay period β) can vary within bounded range. For characteristic equation of first-order process with known coefficient within bound is

$$s + a = 0\,, \tag{6.5}$$

where $0.5 \le a \le 1.5$.

For characteristic equation of second-order process with known coefficients within bounds is

$$s^2 + 2s + a = 0\,, \tag{6.6}$$

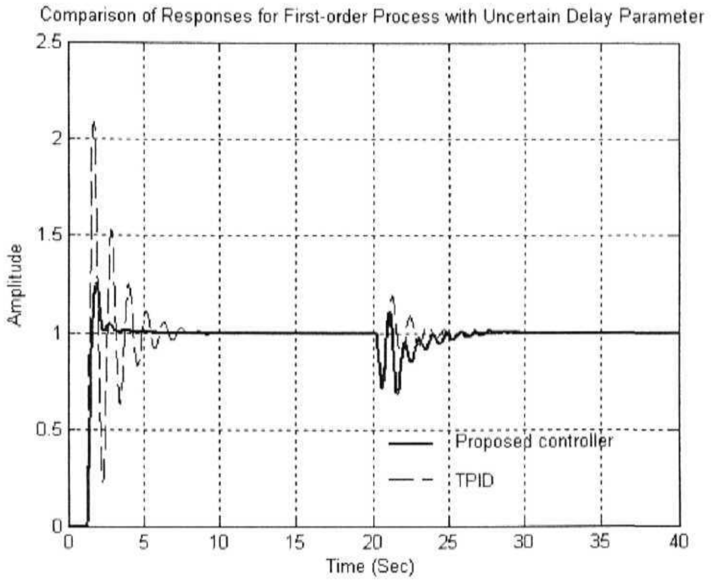
where $0.5 \le a \le 1.5$.

Known coefficients within bounds of higher-order process that is described in Equation 6.4 is ($17 \le a \le 37$).

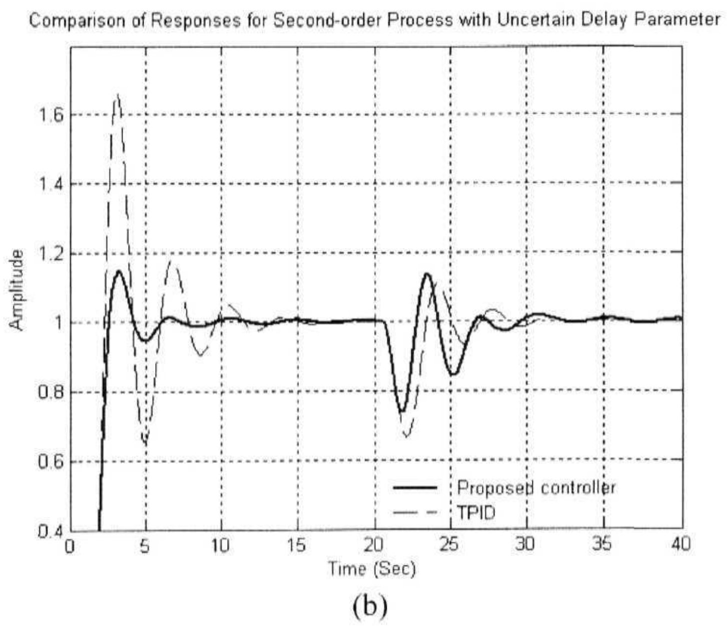
Figure 6-15(a), (b) and (c) describe the robustness of our controller design and conventional PID control system for various parameters values. The simulation results show that our controller design is well-suited for aggressive response case (i.e., a = 0.5 for Figure 6-15(a), 6-15(b) and a = 37 for Figure 6-15(c)) and in sluggish response case rise time performance of TPID is better than our model.

In practical control applications, the time delay introduces an additional phase lag and results in a less stable system. The robustness of unmodeled time delay is observed with the various time delay factors and results are given in Figure 6-16 (a) and (b). The proposed modular gain scheduling mechanism reduces the loop gain especially in low frequency gain (integral gain) in the time delay period of rise time and it becomes less sensitive to the effect of delay.

These results show proposed controller design can minimize the time delay effect and traditional PID controller cannot be considered adequate robust in unmodeled time delay parameter. It is also noticed that our proposed controller is less sensitive than traditional PID controller for parameters variation of the process.



(a)



(b)

Fig. 6-16. Comparison of responses with uncertain delay parameter for (a) first-order process (delay time is 0.3 sec), (b) second-order process (delay time is 0.6 sec).

6.4. Discussions

The accuracy and validity of our proposed knowledge-based modular neural network classifier and the performance of our proposed controller are analyzed with several experiments. Our fuzzy gain scheduling mechanism is a modular structure and it includes several different tuning strategies. It allows variation to both structure and parameters of standard controller (Conventional PID controller).

In adaptive control application, online observation of the process response is one of the important factors to implement adaptive law. In our approach, knowledge-based modular neural network classifier provides online estimation of process response. This information is used as a switching law to select appropriate module of modular adaptable mechanism. Consequently, the performance and stability of proposed technique is largely dependent on the accuracy of proposed modular structure classifier.

First group of experiments describe the accuracy and validity of our proposed knowledge-based modular neural network classifier. These experiment results (Figure 6-1, 6-2 and 6-3) show that proposed classifier has sufficient estimation accuracy in order for it to be applied in adaptive control applications and our proposed knowledge-based refinement algorithm can debug offline incomplete knowledge base or incompatible rules.

Second group of experiments show that the performance of our proposed controller. It has been empirically found that sufficient flexible PID controller provides excellent results in many practical applications. Several experiments (Figures 6-4, 6-5, 6-6, 6-7, 6-8 and 6-9) reveal the flexibility of proposed structures. They also show the effectiveness of each of tuning strategies (each of fuzzy inference systems) in different operating regions.

We then analyze the transient/steady-state response and the robustness of proposed controller design. First test observes the steady-state accuracy, disturbance rejection and set point tracking performance of our controller. Second test describes the robustness of our controller. It analyzes the unmodeled delay effect and parameter of uncertainty. These results show that it can handle wider operating ranges than non-modular adaptive controllers.

Though most of the adaptive PID control methods available so far concentrate on transient and steady-state response characteristics, this approach considers all of the key characteristics of control system such as stability, sensitivity, steady-state accuracy and transient response characteristic. We show that our controller design provides better performance than traditional PID controller along all the four important characteristics of the control system design.

Chapter 7: Conclusions

7.1. Review of our Work

Improving performance of conventional PID controller, that plays a dominant role in process control and industrial application, is the main objective of this approach. Control theory reveals that high performance controller design needs high autonomy and flexibility. High autonomy can be accomplished by Artificial Intelligence technology and modular structure helps to achieve flexibility. This work introduces a way to implement flexible modular structure PID controller design based on Artificial Intelligence technology (neural network and fuzzy logic).

In our controller design, conventional PID controller is used as the standard controller. The proposed adaptive law adjusts online parameters (k_p, k_i and k_d) of conventional PID controller based on both qualitative and quantitative information of process response. This adaptive law consists of two main parts: modular structure fuzzy gain scheduling mechanism and knowledge-based modular neural network classifier.

The required multiple tuning strategies are defined in modular fuzzy gain scheduling mechanism. This mechanism is developed based on well-known conventional control theory (frequency response method and transient /steady-state response analysis in time domain). It produces three gain scaling factors (k_{pscale}, k_{iscale} and k_{dscale}) that are used to adjust parameters of conventional PID controller.

There are seven fuzzy inference systems in this mechanism and they represent different individual tasks. Since the structure of our proposed fuzzy gain scheduling mechanism is modular, it provides extensive flexibility and transparency. It is a way to incorporate with many different tuning strategies (controller's parameters transaction policies) and allows variation to both structure and parameters of the conventional PID controller. The information of process response (both qualitative and quantitative) is used as a switching law to select appropriate modules (fuzzy inference systems) of modular adaptive mechanism.

The quantitative information is extracted from error signal. The proposed knowledge-based modular neural network classifier process monitor provides online qualitative information of process response. This knowledge-based modular neural network classifier is implemented based on conventional control theory such as transient and steady-state response analysis and phase plane method. It decomposes the whole state of process response into four categories (rise time state, steady-state, overshoot/oscillation state and disturbance state) that is very useful information in the conventional PID control system.

The proposed knowledge-based modular neural network classifier is the same as multiple classifier strategy since the whole structure of our classifier is combined with several simple classifiers. It makes the process monitor to become more transparent and easy to debug. We also introduce the idea of theory refinement algorithm to debug and refine incomplete and incompatible knowledge-based. Though proposed theory of refinement algorithm still needs to be improve, it is useful to examine online validity of classification accuracy.

The modularity of controller structure and capability of giving qualitative information of process response can be regarded as important quality of our proposed controller design. Understanding in-depth about knowledge of process to be controlled, is not necessary in order to develop our proposed knowledge-based modular neural network classifier and modular fuzzy gain scheduling mechanism. The following four features are observed with several experiments in order to demonstrate the effectiveness of our proposed controller:

1. First, the accuracy and reliability of our classifier,
2. Second, the flexibility and transparency of our controller structure,
3. Third, the transient and steady-state response of our design and compare it with the other reported work (it includes set point tracking, disturbance rejection, stability and steady-state accuracy) and

4. Fourth, the robustness (sensitivity) of our controller structure.

These experiments results show that the proposed classifier has sufficient estimation accuracy in order to apply in adaptive control applications and our proposed knowledge-based refinement algorithm can debug offline incomplete knowledge base or incompatible rules. Though most of the adaptive PID control methods available so far concentrate on transient and steady-state response characteristics, this approach considers all of the key characteristics of control system such as stability, sensitivity, steady-state accuracy and transient response characteristic.

The experiments results reveal that each of our proposed local computing modules (fuzzy inference systems) can be tuned independently in order to get specific requirement and this controller design provides better performance than traditional PID controller for all of these four important characteristics in controller design.

7.2. Suggestions for Further Research

This work presents general adaptive law for conventional PID control system. The effectiveness of proposed adaptive law is examined with first-order plus time delay process, second-order plus time delay process, third-order process and fourth-order process. In order to get better performance, it is necessary to develop specific adaptive law for individual control problems.

In this approach, the value of qualitative information of process response is also introduced. In order to provide this information, we decompose a whole process response into four pre-defined categories. It can be regarded as an important step to replace well-developed mathematical model based conventional control technique.

For instance, analytical design techniques find the locations of the characteristic equation roots that indicate the performance of the control system. The correlation between the locations of the characteristic equation roots in the s-plane and the transient response is well-known in control system. In order to get required performance of a control system, controller or compensator relocates the characteristic equation roots based on this knowledge. The main reason finding the location of the characteristic equation roots is to observe the transient response and stability of the system.

In our assumption, if we can estimate process response with more complete information (i.e., current process response is *rise time state* and it is *sluggish* or current process response is *oscillation state* and it is *stable* or current process response is *steady-state without error*), it can be regarded as an alternative way of finding the location of the characteristic equation roots. Consequently, it is necessary to implement high performance process monitor that can provide more precise and complete information of process response.

Appendix I: Selection of the Universe of Discourse for Output Variable of Fuzzy Inference System f_1

Fuzzy inference system f_1 produces proportional gain scaling factor k_{pscale} in rise time state (*T1*, *T2* and *T3*) and second tuning strategy in disturbance state (*T7*). Very small proportional gain can cause slow response in both cases. But, large proportional gain can cause oscillatory output. Zhao [Zhao93] proposed the range of proportional gain [$0.32k_u$, $0.6k_u$] based on an extensive simulation study on various processes. It is the same as our proposed proportional gain scaling factor range [0.53, 1.0] ($0.6k_u = k_p$).

We also examine several experiments with various processes in order to find out appropriate the universe of discourse of output variable of fuzzy inference system f_1. The required data of process $G_1(s)$ (Figure AI-1(d)), $G_2(s)$ (Figure AI-1(d)) and $G_3(s)$ (Figure AI-1(d)) are described in section 6.2. The remaining third-order process $G_5(s)$ [Ogata02] are shown in following Equations AI.1.

$$G_5(s) = \frac{1}{s(s+1)(s+5)} \qquad \text{(AI.1)}$$

The required data for these processes are tabulated in Table AI-1 [Ogata02].

Table AI-1. Required data for Process $G_5(s)$.

Process	Proportional gain	Integral gain	Derivative gain	Sampling time
$G_5(s)$	18	12.811	6.32232	0.01 Sec

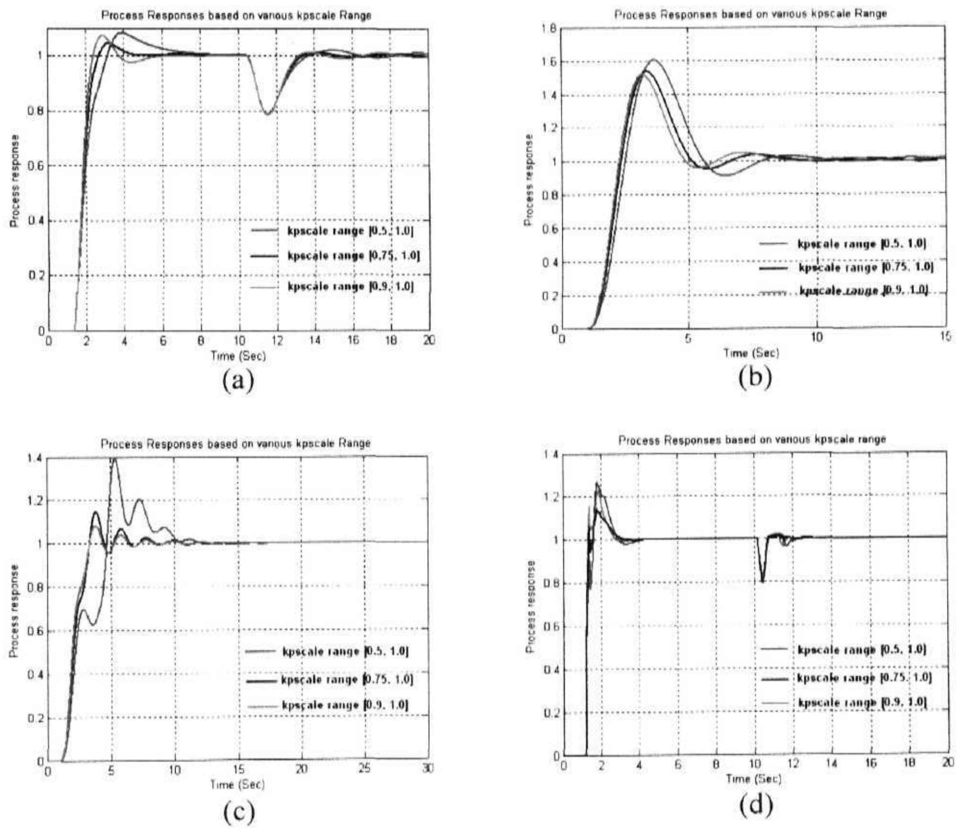


Fig. AI-1. A variety of process responses based on various k_{pscale} range for (a) process $G_2(s)$ (Equation 6.2), (b) process $G_4(s)$ (Equation AI.1), (c) process $G_3(s)$ (Equation 6.3) and (d) process $G_1(s)$ (Equation 6.1).

These results are shown in Figures AI-1. Accordingly, we select the range [*0.75, 1.0*] for the universe of discourse of output variable of fuzzy inference system f_1.

Appendix II: Description of Selection of Rules for Fuzzy Inference System f_1

The task of fuzzy inference system f_1 can be described as follows and corresponding surface plane for these rules as shown in Figure AII-1 (surface planes of remaining fuzzy inference systems are shown in Figures AII-2):

(i) It is necessary to produce small proportional gain scaling factor (to keep stability) in a delay period (tuning strategy T_1) and the required rule can be described as,

If error is *VB* and change in error is *ZE* **Then** Proportional gain scaling factor is *VVS*.

(ii) To improve performance (large proportional gain) in the beginning and middle of rise time state (Tuning strategy T_2), the required rules can be described as,

If error is *VB* and change in error is *M* **Then** Proportional gain scaling factor is *VVB*.

If error is *VB* and change in error is *VB* **Then** Proportional gain scaling factor is *VB*.

If error is *M* and change in error is *VB* **Then** Proportional gain scaling factor is *B*.

(iii) It is necessary to reduce proportional gain scaling factor (based on value of change in error) to improve stability and reduce overshoot (small proportional gain) near the set-point (tuning strategy T_3 and T_7), the required rules can be described as,

If error is *S* and change in error is *VB* **Then** Proportional gain scaling factor is *S*.

If error is *S* and change in error is *M* **Then** Proportional gain scaling factor is *M*.

If error is *ZE* and change in error is *VB* **Then** Proportional gain scaling factor is *VVS*.

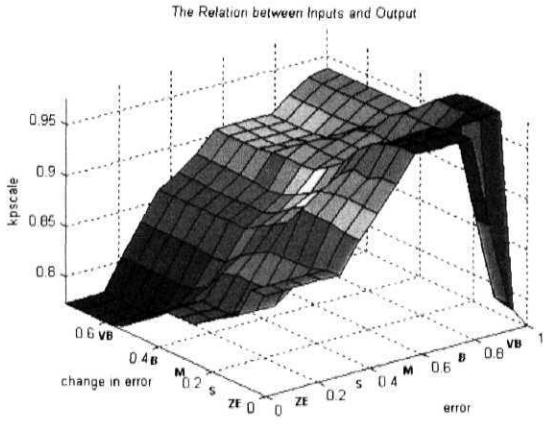


Fig. AII-1. The relation between inputs and output of fuzzy inference system f_1.

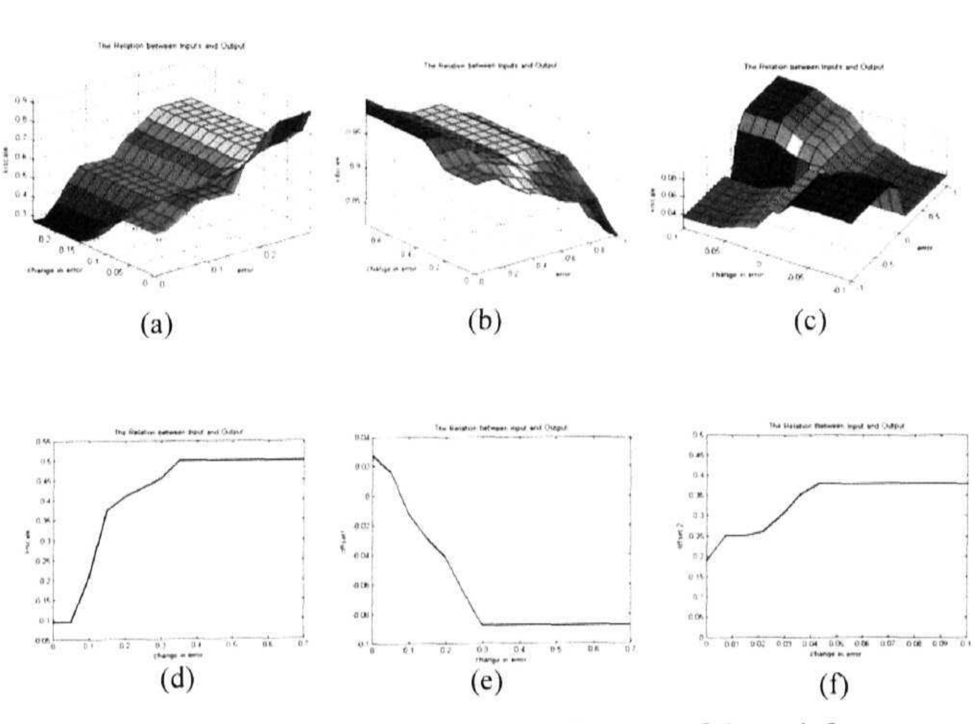


Fig. AII-2. The relation between inputs and output of fuzzy inference system; (a) f_2, (b) f_3, (c) f_4, (d) f_5, (e) f_6, (f) f_7.

References

[1] Almeida, O. D. M., Coelho, A. A. R., (2002). A Fuzzy Logic Method for Auto-Tuning a PID Controller: SISO and MIMO Systems. 15th Triennial World Congress, Barcelona, Spain, IFAC.

[2] Ambrosio, P. J. E., Mort, N., (2002). Auto-Tuning of Fuzzy PID Controllers. 15th Triennial World Congress, Barcelona, Spain, IFAC.

[3] Antsaklis, P., (1993). Defining Intelligent Control. Report of the Task Force of Intelligent Control. IEEE Control Systems Society.

[4] Astrom, K. J., B. Wittenmark, B., (2003). Adaptive Control. Person Education, India.

[5] Astrom, K. J., Hagglund, T., (1985). Automatic Tuning of PID Controllers Based on Dominant Poles Design. IFAC Adaptive Control of Chemical Processes. Pergamon Press, New York, 205-210.

[6] Auda, G., (1996). Cooperative Modular Neural Network Classifiers. Ph.D. Thesis, University of Waterloo, Waterloo, Ontario, Canada.

[7] Bandyopadhyay, M. N., (2003). Control Engineering Theory and Practice. Prentice-Hall, India.

[8] Bandyopadhyay, R., Chakraborty, U. K., Patranabis, D., (2001). Autotuning A PID Controller: A Fuzzy-Genetic Approach. Journal of Systems Architecture 47, 663-673.

[9] Benaskeur, A. R., Desbiens, A., (2002). Backstepping-based Adaptive PID Control. IEE Proceedings, Control Theory and Applications. 149(1), 54-59.

[10] Blanchett, T. P., Kember, G. C., Dubay, R., (2000). PID Gain Scheduling Using Fuzzy Logic. ISA Transaction 39, 317-325.

[11] Bose, B. K., (2000). Expert System, Fuzzy Logic, and Neural Networks in Power Electronics and Drives. Power Electronics and Variable Frequency Drives. IEEE Press.

[12] Bose, B. K., May/June (2000). Fuzzy Logic and Neural Networks. IEEE Industry Applications Magazine.

[13] Brdys, M. A., Littler, J. J., (2002). Fuzzy Logic Gain Scheduling for Nonlinear Servo tracking. International of Appl. Math. Comut. Sci., 12(2), 209-219.

[14] Chang, W. –D., Hwang, R. –C., Hsieh, J. -G., (2003). A Multivariable On-line Adaptive PID Controller Using Auto-tuning Neurons. Engineering Applications of Artificial Intelligence 16, 57-63.

[15] Chen, L., Narendra, K. S., (2001). Nonlinear Adaptive Control Using Neural Networks and Multiple Models. Automatica, Special Issue on Neural Network Feedback Control, 37(8), 1245-1255.

[16] Chen, L., Narendra, K. S., December (2002). Intelligent Control Using Neural Networks and Multiple Models. Proceedings of the 41st IEEE Conference on Decision and Control, Vol. 2, 1357-1362.

[17] Chen, M., Linkens, D. A., (1998). A Hybrid Neuro-Fuzzy PID Controller. Fuzzy Sets and Systems 99, 27-36.

[18] Cho, H. –J., Cho, K. –B., Wang, B. -H., (1997). Fuzzy-PID Hybrid Control: Automatic Rule Generation Using Genetic Algorithms. Fuzzy Sets and Systems 92, 305-316.

[19] Choi, Y. –K., Lee, M. –J., April (2001). Sungshin Kim, Young-Chul Kay. Design and Implementation of a Adaptive Neural-Network Compensator for Control Systems. IEEE Transactions on Industrial Electronics. 48(2).

[20] Dorf, R. C., Bishop, R. H., (2004). Modern Control Systems. Pearson Education, India.

[21] Driankov, D., Hellendoom, H., Reinfrank, M., (2001). An Introduction to Fuzzy Control. Narosa Publishing House, India.

[22] Franklin, G. F., Powell, J. D., Workman, M., (2000). Digital Control of Dynamic Systems. Third Edition, Pearson Education, India.

[23] Fu, L., December (1995). Introduction to Knowledge-Based Neural Networks. Knowledge-Based Systems, Vol. 8 (6).

[24] Giacinto, G., Roli, F., (2001). An Approach to the Automatic esign of Multiple Classifier Systems. Pattern Recognition Letters 22, 25-33.

[25] Guzelkaya, M., Eksin, I., Yesil, E., (2003). Self-Tuning of PID-Type Fuzzy Logic Controller Coefficients via Relative Rate Observer. Engineering Applications of Artificial Intelligence 16, 227-236.

[26] Hasegawa, T., Horikawa, S., Furuhashi, T., Uchikawa, Y., (1995). On Design of Adaptive Fuzzy Controller Using Fuzzy Neural Networks and A Description of Its Dynamical Behavior. Fuzzy Sets and Systems71,3-23.

[27] Henriques, J., Cardoso, A., Dourado, A., (1999). Supervision and c-Means Clustering of PID Controllers for A Solar Power Plant. International Journal of Approximate Reasoning 22, 73-91.

[28] Hespanha, J. P., Liberzon, D., Morse, A. S., May 15(2003). Overcoming the limitation of adaptive control by means of logic-based switching. Systems and Control Letters. Vol. 49(1), 49-65.

[29] Higgins, C. M. Jr., (1993). Classification and Approximation with Rule-Based Networks. Ph.D. Thesis, California Institute of Technology, Pasadena, California.

[30] Honegger, M., Brega, R., Schweitzer, G., April (2000). Application of a Nonlinear Adaptive Controller to a 6 DOF Parallel Manipulator. Proceedings of the IEEE International Conference on Robotics & Automation. San Francisco, CA.

[31] Hur, N., Nam, K., Won, S., June (2000). A Two-Degrees-of_Freedom Current Control Scheme for Deadtime Compensation. IEEE Transactions on Industrial Electronics 47(3).

[32] Jacobs, R. A., Jordan, M. I., Barto, A. G., March (1990). Task decomposition through in a modular connectionist architecture: The what and where vision tasks. COINS Technical Report 90-27.

[33] Jantzen, J., 30 September (1998). Tuning of Fuzzy PID Controllers. Tech. Report no. 98-H 871, Technical University of Demark, Department of Automation, Bldg 326, DK-2800 Lyngby, Denmark.

[34] John, G., Kohavi, R., Pfleger, K., (1994). Irrelevant Features and the Subset Selection Problem. Machine Learning: Proceedings of the Eleventh International Conference.

[35] Khiang, T. K., Khalid, M., Yusof, R., December (1996). Intelligne Elevator Control by Ordinal Structure Fuzzy Logic Algorithm. Proc. of ICARCV 97, Singapore.

[36] Kim, B. J., Chung, C. C., May 8-10 (2002). Design of Fuzzy PD+I Controller for Tracking Control. Proceedings of the American Control Conference. Anchorage, AK.

[37] Kraaft, L. G., Campagna, D. P., June 21-23 (1989). A Comparison between CMAC Neural Network Control and Two Traditional Adaptive Control Systems. American Control Conference, Pittsburgh, Pennsylvania.

[38] Lee, C. –H., Teng, C. -C., (2002). Tuning of PID Controllers for Unstable Processes Based on Gain and Phase Margin Specifications: A Fuzzy Neural Approach. Fuzzy Sets and Systems 128, 95-106.

[39] Lee, J., (1993). On Methods for Improving Performance of PI-type Fuzzy Logic Controllers. IEEE Transactions on Fuzzy Systems 1(4), 298-301.

[40] Leith, D., Leithead, W., (2000). Survey of Gain Scheduling Analysis and Design. International Journal of Control (73) 1001-1025.

[41] Li, H. -X., Gatland, H. B., March (1995). A New Methodology for Designing a Fuzzy Logic Controller. IEEE Transactions on Systems, Man, and Cybernetic, Vol. 25(3).

[42] Li, T. –H. S., Shieh, M. -Y., (2000). Design of a GA-based Fuzzy PID Controller for Non-minimum Phase Systems. Fuzzy Sets and Systems 111, 183-197.

[43] Li, W., Chang, X. G., Wahl, F. M., Tso, S. K., (1999). Hybrid Fuzzy P+ID Control of Manipulators under Uncertainty. Mechatronics 9, 301-315.

[44] Mamdani, E. H., (1974). Application of Fuzzy Algorithms for the Control of a Dynamic Plant. Proc. IEE, 121, 1585-1588.

[45] Mann, G. K. I., Hu, B. G., Gosin, R. G., (1999). Analysis of Direct Action Fuzzy PID Controller Structures. IEEE Transaction on Systems, Man, and Cybernetic, Part B 29, 371-388.

[46] Marjan, G., (2001). Decomposed Fuzzy Proportional-Integral-Derivative Controllers. Applied Soft Computing 1, 201-214.

[47] Minkova, M. D., Minkov, D., Rodgerson, J. L., Harley, R. G., (1998). Adaptive Neural Speed Controller of a DC Motor. Electric Power Systems Research 47, 123-132.

[48] Nam, K., Arapostathis, A., September (1998). A Model Reference adaptive Control Scheme for Pure-Feedback Nonlinear Systems. IEEE Transactions of Automatic Control 33(9).

[49] Nardi, F., (2000). Neural Network based Adaptive Algorithms for Nonlinear Control. Ph.D Thesis. Georgia Institute of Technology.

[50] Narendra, K. S., Balakrishnan, J., (1997). Adaptive control using multiple models. IEEE Transaction on Automatic Control. 42(2), 171-187.

[51] Norgaad, M., Ravn, O., Poulsen, N. K., Hansen, L. K., (2000). Neural Networks for Modelling and Control of Dynamic Systems. Springer-Verlag London Limited.

[52] Ogata, K., (2002). Modern Control Engineering. Fourth ed. Pearson Education, India.

[53] Qiao, W. Z., Mizumoto, M., (1996). PID Type Fuzzy Controller and Parameters Adaptive Method. Fuzzy Sets and Systems 78, 23-35.

[54] Radaideh, S. M., Hayajneh, M. T., (2002). A Modified PID Controller. Journal of the Franklin Institute 339, 543-553.

[55] Rajani, K. M., Nikhil, R. P., (2000). A Self-Tuning Fuzzy PI Controller. Fuzzy Sets and Systems 115, 327-338.

[56] Rajani, K. M., Nikhil, R. P., (2001). A Note on Fuzzy PI-Type Controllers with Resetting Action. Fuzzy Sets and Systems 121, 149-159.

[57] Ranger, P., Desbiens, A., (2003). Improved Backstepping-based Adaptive PID Control. The International Conference on Control and Automation, Montreal, Canada, 123-127.

[58] RayChaudhuri, T., Hamey, L. G. C., (1995). From Conventional Control to Intelligent Neurocontrol Methods: A Survey of the Literature. Macquarie University Techanical Report 95/170c. Australia.

[59] Rezaee, M. R., Goedhart, B., Lelieveldt, B. P. F., Reiber, J. H.C., (1999). Fuzzy Feature Selection. Pattern Recognition 32, 2011-2019.

[60] Riid02, A., (2002). Transparent Fuzzy Systems: Modeling and Control. Ph.D. Thesis, Talinn Technical University.

[61] Robert, M., Zhiquiang, G., October 3-7 (2004). A Robust Two-Degree-of-Freedom Control Design Techniques and its Practical Application. IEEE Industrial Application Society 2004 Annual Meeting and World Conference.

[62] Ronco, E., Gawthrop, P. J., Hill, D. J., (1998). Gated Modular Neural Networks for Control Oriented Modelling. Technical Report: EE-98009. Center for systems and Control, University of Glasgow UK.

[63] Ruck, D. R., Rogers, S. K., Kabrisky, M., (1990). Feature Selection Using a Multilayer Perceptron. Neural Network Computing 2 (2), 40-48.

[64] Seng, T. L., Khalid, M., Yusof, R., Omatu, S., (1998). Adaptive Neuro-Fuzzy System by RBF and GRNN Neural Networks. Journal of Intelligent and Robotic systems. 23, 267-289.

[65] Shieh, M. –Y., Li, T. –H. S., (1998). Design and Implementation of Integrated Fuzzy Logic Controller for a Servomotor System. Mechatronics 8, 217-240.

[66] Shu, H., Pi, Y., (2000). PID Neural Networks for Time-Delay Systems. Computers and Chemical Engineering 24, 859-862.

[67] Smith, R. M., Hunt, K., (1995). Local Model Architectures for Nonlinear Modeling and Control. Neural Network Engineering in Dynamic Control Systems. Springer-verlag.

[68] Towell, G. G., Shavlik, J. W., (1994). Knowledge-Based Artificial Neural Networks, Artificial Intelligence 70, 119-165.

[69] Toygar, O., Acan, A., (2004). Multiple Classifier Implementation of a Divide-and-Conquer Approach Using Appearance-Based Statistical Methods for Face Recognition. Pattern Recognition Letters 25, 1421-1430.

[70] Tseng, C., (2001). Control with Word: the modular approach. Information Sciences 134, 111-133.

[71] Tyan, C. –Y., Wang, P. P., D. R. Bahler, D. R., (1996). An application on intelligent control using neural network and fuzzy logic. Neurocomputing 12, 345-363.

[72] Viljamaa, P., (2002). Fuzzy Gain Scheduling and Tuning of Multivariable Fuzzy Control- Methods of Fuzzy Computing in Control Systems. Ph.D. Thesis, Tampere University of Technology, Finland.

[73] Viljamaa, P., Koivi, H. N., August 28-31 (1995). Fuzzy Logic in PID Gain Scheduling. Third European Congress on Fuzzy and Intelligent Technologies EUFIT'95, Aachen, Germany.

[74] Windeatt, T., (2005). Diversity measures for Multiple Classifier System Analysis and Design. Information Fusion 6, 21-36.

[75] Woo, Z. W., Chaung, H. –Y., Jyelin, J., (2000). A PID Type Fuzzy Controller with Self-Tuning Scaling Factor. Fuzzy Sets and Systems 115, 321-326.

[76] Woods, R. L., Lawrence, K. L., (1997). Modeling and Simulation of Dynamic Systems. Prentice-Hall, New Jersey 07458.

[77] Wu, C. –J., Huang, C. -H., (1996). A Neural Network Controller with PID Compensation for Trajectory Tracking of Robotic Manipulators. Journal of the Franklin Institute 333, 523-537.

[78] Yager, R., Filev, D., (1994). Generation of Fuzzy Rules by Mountain Clustering. Journal of Intelligent and Fuzzy Systems, No. (2), 209-219.

[79] Zadeh, L. A., (1965). Fuzzy Sets. Information and Control, Vol. (8), 338-353.

[80] Zhao, Z. –Y., Tomizuka, M., Isaka, S., (1993). Fuzzy Gain Scheduling of PID Controllers. IEEE Transaction on Systems, Man, and Cybernetic 23 (5).

[81] Zhu, F., Guan, S., (2004). Feature Selection for Modular GA-Based Classification. Applied Soft Computing 4, 381-393.

[82] Zurada, J. M., (2001). Introduction to Artificial Neural Systems. Jaico Publishing House, India.