THE GLOBAL ANALYSIS OF FUZZY DYNAMICAL SYSTEMS

by

Yung-Yaw Chen

Memorandum No. UCB/ERL M89/107

29 August 1989

THE GLOBAL ANALYSIS OF FUZZY DYNAMICAL SYSTEMS

by

Yung-Yaw Chen

Memorandum No. UCB/ERL M89/107

29 August 1989

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

THE GLOBAL ANALYSIS OF FUZZY DYNAMICAL SYSTEMS

by

Yung-Yaw Chen

Memorandum No. UCB/ERL M89/107

29 August 1989

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

The Global Analysis of Fuzzy Dynamical Systems

by

Yung-Yaw Chen

Abstract

In this report we focus on the behavioral analysis of fuzzy dynamical systems, which has become an increasingly important issue due to the prominent success of fuzzy logic control. Fuzzy logic control was first proposed (Zadeh (1968)) as a possible application of fuzzy set theory. Many industrial applications of it have been demonstrated in the past two decades. A fuzzy logic controller utilizes the knowledge of a human expert to acquire better control strategies for a process. It starts from some simple process controls which can be easily handled by human operators, such as steam engine control. However, as the constructed fuzzy control systems become more complex and larger in scale, there is an increasing need for a systematic analysis and design methodology.

Among all the problems concerning the analysis of a fuzzy dynamical system, stability is one of the most interesting topics. In this thesis, a notion of the "expert's Lyapunov function" is first proposed to give an insight into the stable nature of a fuzzy control system. A cell-to-cell mapping method, which is successfully used in the nonlinear system analysis, is then adopted to attack the problem

*Research supported by NASA Grant NCC-2-275, AFOSR Grant 89-0084 and California State MICRO 88-094.

of the global analysis of a fuzzy dynamical system. An analytical method is developed to describe the behavior of a fuzzy dynamical system. Both the real and fuzzy initial state responses are also studied. A definition of *cellular stability* is proposed to give a complete description of the analyzed fuzzy system. A follow-up stability theorem is also formulated to provide a sufficient condition for a stable fuzzy dynamical system. To illustrate the advantages as well as the disadvantages of the fuzzy logic control, an experiment involving an inverted pendulum was conducted to compare the differences on the performance and controller design procedure between the fuzzy logic control and the more conventional state feedback control. The results of this experiment are described subsequently.

Chapter 1

Introduction

1.1 Motivation

Fuzzy Dynamical Systems (FDS) are a class of nonlinear systems which are represented in a rule-base form with the rules described in terms of linguistic variables. The representation of an FDS is entirely different from the traditional descriptions of a dynamical system using differential or difference equations. Conventional system theorists have constructed a very good, if not perfect, universe for linear time-invariant (LTI) systems. There exist many powerful methods, both in the time domain and the frequency domain, for the analysis and design of an LTI system. The task is straight forward since there is a simple relationship between their input-output behavior and their internal dynamics. It is especially easy to characterize their input-output behavior in the frequency domain as the Fourier transform of the impulse response. On the other hand, there is not yet a universal method for the analysis and design of a general nonlinear system. This is even more difficult for an FDS because there is no precise mathematical formulation.

Fuzzy Control Systems (FCS) are parts of the fuzzy dynamical systems. Their basic structure consists of a plant and a Fuzzy Logic Controller (FLC) in the feedback loop. An FLC is designed based on the knowledge of the process operators and, like an FDS, is in a rule-base form and described by linguistic variables. Human operators are usually able to summarize their control strategies and incorporate them into the design of an FLC to achieve the control goals. The advantage of utilizing operators' knowledge accounts for part of the excellent performance achieved by fuzzy logic control. However, the power as well as the weakness of an FCS is its total dependency on the experts' knowledge. The controller design is impossible in cases when there is no experienced operator available. In order to automate the design process of an FLC, it is firstly necessary to have a systematic way of analyzing the behavior of an FDS. Many

researchers have tried to solve the problem, but with few satisfactory results. As a matter of fact, there are not even commonly accepted definitions of stability and controllability, the two most important properties required to describe an FDS.

A powerful technique, namely *cell-to-cell mapping* (Hsu (1980)), is used in nonlinear system analysis with great success. The basic idea underlying the usage of the cell-to-cell mapping is to transform the original nonlinear infinite numbered point-to-point mapping into a finite numbered cell-to-cell mapping. Then the original nonlinear system is approximated by this new cell-to-cell mapping and can be analyzed accordingly. As the cost of the computer memory and the cpu time decreases, this method has proven to be very useful in nonlinear system analysis and controller design.

Motivated by the success of the cell-to-cell mapping in nonlinear system analysis, we introduce this method into the analysis of fuzzy dynamical systems. A detailed analysis of an FDS using the cell-to-cell mapping technique is developed. Both the real and the fuzzy initial state responses are discussed to give a complete picture of the behavior of the analyzed FDS. Definitions of stability based on this method are also proposed. Under a NASA project, a hardware setup of an inverted pendulum was constructed to provide us with a way to practically compare the performance of the fuzzy logic control and the more conventional state feedback control.

1.2 Review of Previous Work

The original idea of fuzzy control was first proposed by Zadeh (1968, 1972,1973) as a possible application of fuzzy set theory. The idea of fuzzy logic control is to construct a controller which utilizes the linguistic, imprecise knowledge of the human operators. The first stage of development was mostly contributed to Mamdani (1974) and his colleagues in Queen Mary College, London. Most of the research conducted in this period were merely experiments in laboratories, such as the work by Kickert (1976) and Rutherford (1976). One exception is the work by

Ostergaard (1977), which led to the development of a commercially available fuzzy controller for a cement kiln and was the first industrial application of fuzzy logic control.

Many applications of fuzzy logic control have appeared in a wide range of fields since Mamdani's first steam engine controller. They can be listed as follows: warm water process (Kickert (1976)), sinter plant (Rutherford (1976)), robot (Uragami (1976)), stirred tank reactor (King (1977)), heat exchanger (Ostergaard (1977), Sinha (1977)), traffic junction (Pappis and Mamdani (1977)), strongly perturbed motor (Willaeys (1977)), activated sludge processes (Flanagan (1980), Tong (1980)), cement kilns (Umbers (1980), King (1982)), box annealing furnace (Yonekura (1981)), casting plant (Bartolini (1982)), pump operation (Kokawa (1982)), model cars (Takagi and Sugeno (1983), Sugeno and Nishida (1984)), automobile (Murakami (1983)), diesel engine (Murayama (1984)), aircraft flight (Larkin (1984)), steam generating unit (Kumar and Majumder (1985)), vehicle navigation (Hogle and Bonissone (1985)), robot arm (Scharf (1985)), blast furnace (Hong (1985)), turning process (Sakai and Ohkusa (1985)), multilayer incinerator (Sugeno and Kang (1986)), crane (Yasunobu (1987)), nuclear plant (Bernard (1986), Fukuzaki (1988)), automobile transmission (Kasai and Morimoto (1988)), Sendai automatic train (1988).

With the increasing number of successful industrial applications, researchers became more interested in formalizing the structure of fuzzy logic control. The format of a fuzzy control system, which will be explained in detail in Chapter 2, has not changed much since it was first introduced. The main problem considered by most researchers regarding the controller design is the method to assemble the control statements that constitute the rule-base. Braae and Rutherford (1979) and later Czogala and Pedrycz (1981) discussed the completeness of a fuzzy logic controller in their work. A lot of work, such as Baldwin and Pilsworth (1980), Bandler and Kohout (1980), Mizumoto (1981), and Sugeno and Takagi (1983), focused on the appropriateness and validity of the implication rules. A self-organizing fuzzy controller was proposed by Procyk and Mamdani (1979) who tried to construct a *learning* fuzzy logic controller by using a "performance measure decision table". Czogala and Pedrycz (1982) adopted a similar approach to generate

control rules through a performance index and the subsequent solution of a relational equation.

Many researchers involved in the analysis of fuzzy control systems have theoretical control backgrounds. The notions of state, stability and controllability from conventional system theories are studied and extended to fuzzy control systems. Kickert and Mamdani (1978) used the describing function method to evaluate the stability of fuzzy control systems. Braae and Rutherford (1979) proposed a linguistic phase plane trajectory for analyzing the stability. An energetistic stability criterion was formed by Kiszka, Gupta, and Nikiforuk (1985). There was also the work of deGlas (1984) that included a fuzzy stability theory. Chen (1987) proposed a notion of "expert's Lyapunov function" to explain the intrinsic stability of fuzzy control systems.

While people were looking for a suitable tool to analyze a fuzzy control system, Hsu and his students successfully developed the *cell-to-cell mapping* method for analyzing a wide range of nonlinear dynamical systems. The details of the method can be found in a series of publications: Hsu (1980), Hsu and Guttalu (1980), Hsu (1981), Hsu, Guttalu and Zhu (1982), Hsu (1982), Hsu (1985), Hsu and Chiu (1986) and Chiu and Hsu (1986). The prospect of combining the cell-to-cell mapping with the analysis of fuzzy dynamical systems further leads to the work by Chen (1988), Chen and Tsao (1988) and Chen and Tsao (1989).

1.3 Contributions of the Thesis

In this dissertation, we analyze the behavior of the fuzzy dynamical systems with more emphasis on the fuzzy control systems and also introduce the cell-to-cell mapping technique to discuss:

(1) the global behavior of a fuzzy dynamical system,

(2) the real and fuzzy initial state response of a fuzzy dynamical system,

(3) the cellular stability of a fuzzy dynamical system.

The outline of the thesis is as follows:

In Chapter 2, we give an introduction to a fuzzy control system including both the plant modeling and the fuzzy logic controller. The definitions of a fuzzy set and its basic operations are explained. A functional block diagram is shown to fully describe the difference between a fuzzy control system and a conventional control system. Detailed descriptions of each block, such as the plant modeling, the fuzzification, the defuzzification and the rule inferences of an FLC, are also included in the chapter. Lastly, two industrial applications are described to exemplify the practicality of fuzzy logic control.

In Chapter 3, we give an explanation for the seemingly intrinsic stability of a fuzzy control system. A new concept, namely "expert's Lyapunov function", is proposed to represent the stability notion of a process operator. We believe a fuzzy control system is stablized because of the incorporation of experts' stability notion into the controller design. A stability criterion is developed and can be used to help the design of an FLC.

The most important part of this dissertation is the introduction of the cell-to-cell mapping into the analysis of a fuzzy dynamical system. In Chapter 4, the idea behind cell-to-cell mapping is explained. This is then adapted to the analysis of an FDS. The analytical method gives an approximate description of the behavior of the analyzed system. The description cannot be precise because the system is represented in fuzzy linguistic terminology in the first place. From the analysis, both the real and fuzzy initial state responses are shown to further illustrate the behavior of the system. A definition of the cellular stability and of an FDS, which is based on the results of this method, is also proposed in this chapter.

In Chapter 5, an experimental comparison between the conventional state feedback control and the fuzzy logic control is described. A hardware setup of an inverted pendulum under a NASA project was built for this purpose. Both the advantages and the disadvantages of a fuzzy control system are investigated and verified through the experiment.

Chapter 2

General Description of Fuzzy Control Systems

2.1 Introduction

The construction of a fuzzy control system (FCS) is based on the idea of incorporating the "experience" or "expert knowledge" of a human process operator to derive a better strategy for the control of a process and thereby achieve better system performance. An FCS is similar to an expert system in the sense that they both model human experience and human decision making behavior. The main advantage of this approach is its ability to implement "rule of the thumb" experience and heuristics. It can also work without a model of a process. However, an FCS is different from an expert system in that it is a real-time feedback control mechanism and not a consulting tool like most expert systems.

The basic configuration of an FCS is shown in Figure 2.1.1. A plant and a fuzzy logic controller (FLC) constitute a feedback loop with the FLC supplying the input to the plant. The configuration is basically the same as a conventional control system except for the replacement of the controller block with an FLC. The plant in Figure 2.1.1 is generally a process which is difficult to describe and/or control by conventional methods but which can be controlled by human experts with quite satisfactory results. Figure 2.1.2 shows the detailed structure of an FLC. The major part of a fuzzy logic controller is a rule-base which consists of a number of *if-then* rules expressed in linguistic terms, such as:

$$\text{if } x_1 \text{ is } \textit{small}\text{, and } x_2 \text{ is } \textit{zero}\text{, then } u \text{ is } \textit{medium}\text{,} \tag{2.1.1}$$

where x_1 and x_2 are state variables and u is the plant input. The linguistic terms, *small*, *zero*, and *medium*, are fuzzy sets whose membership functions are defined in the database. The premise and the consequent parts of the control rules are constructed based on the expert's knowledge

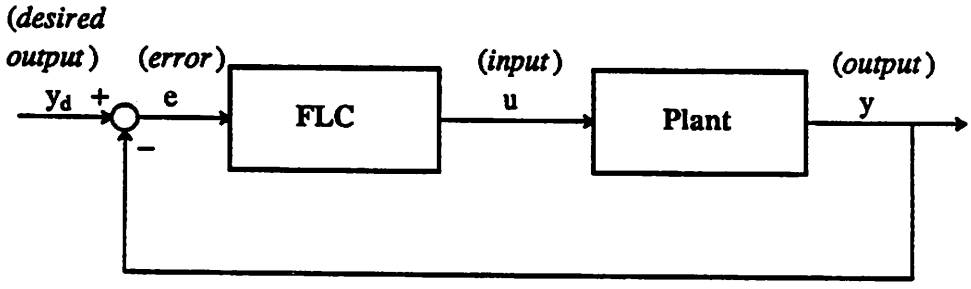

Figure 2.1.1: Basic structure of a fuzzy control system

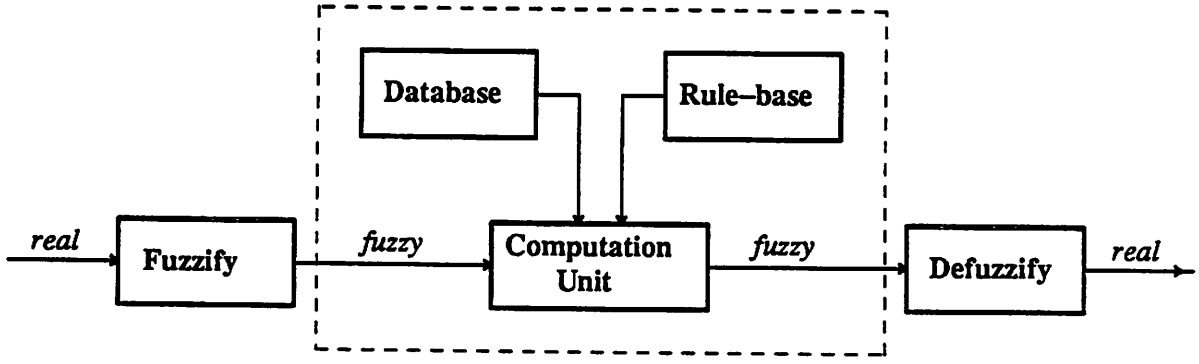

Figure 2.1.2: The configuration of a fuzzy logic controller

of the plant. The compositional inference is performed by the computation unit to combine the effects of different rules. Practically, the input and output of a fuzzy logic controller are real vectors and not fuzzy sets. A fuzzification stage is needed to convert the real-valued inputs into fuzzy set expressions and a defuzzification block is also needed to transform the "fuzzy" results of the inference operations into real vectors again. Each functional block of an FCS will be discussed in detail in the rest of the chapter.

Section 2 gives some basic definitions and operations of fuzzy sets and commonly used terminology. Section 3 shows the plant modeling of an FCS. Section 4 introduces the formulation of a fuzzy logic controller and explains its inference operations. These include the fuzzification and the defuzzification processes. Section 5 describes two successful industrial applications of fuzzy logic control on cement kiln and automatic train control.

2.2 Definitions and Terminology

We shall start with some basic definitions and operations of fuzzy sets in this section. Only definitions which are relevant to the representation and the operation of an FCS are introduced. More detailed descriptions can be found in Zimmermann (1985) and Kandel (1986).

2.2.1 Basic Definitions of Fuzzy Set Theory

Let X be a space of objects, namely *object space* and x be a generic element of X. A classical set A is defined as a collection of elements or objects $x \in X$, which can be finite, countable, or uncountable. Each element in X can either belong to or not belong to a set A, $A \subset X$. By defining a characteristic function (or membership function) for the elements in a set where 1 indicates membership and 0 non-membership, a classical set can be represented by a set of ordered pairs $(x, 0)$ or $(x, 1)$. For example, assume that X contains 5 elements x_1, x_2, x_3, x_4 and x_5, then a set $A \subset X$, containing x_1, x_2, and x_5, can be represented by

$$A = \{ x_1, x_2, x_5 \}, \tag{2.2.1}$$

$$= \{ (x_1, 1), (x_2, 1), (x_3, 0), (x_4, 0), (x_5, 1) \}. \tag{2.2.2}$$

Unlike the conventional "crisp" set, a fuzzy set expresses the different "degrees" to which an element belongs to a set. In the real world, most of the human concepts and descriptions are full of terms of "degrees". People, without knowing the true measure, tends to give an approximate statement of an event, such as John is *tall*. Zadeh (1965) proposed the fuzzy set theory as a means of representing this concept of "matters of degree". For a fuzzy set, the characteristic function is extended to have values between 0 and 1 to denote different degrees of membership for the elements of a given set.

Definition 2.2.1: Fuzzy Set

If X is a collection of objects denoted generically by x, then a fuzzy set $\tilde{A}$ in X is defined as a set of ordered pairs:

$$\tilde{A} = \{ (x, \mu_{\tilde{A}}(x)) \mid x \in X \} \tag{2.2.3}$$

$\mu_{\tilde{A}}(x)$ is called the membership function or grade of membership of x in $\tilde{A}$, which maps X to the membership space M, $M = [0, 1]$.

□

When M contains only two points 0 and 1, $\tilde{A}$ is non-fuzzy (crisp) and $\mu_{\tilde{A}}(x)$ is identical to the characteristic function of a non-fuzzy set. Elements with a zero degree of membership are normally not listed. In the literature, there is another commonly used notation for fuzzy sets:

$$\tilde{A} = \mu_{\tilde{A}}(x_1)/_{x_1} + \mu_{\tilde{A}}(x_2)/_{x_2} + \cdots = \sum_{i=1}^{n} \mu_{\tilde{A}}(x_i)/_{x_i} \tag{2.2.4}$$

$$or \int_x \mu_{\tilde{A}}(x)/_x \tag{2.2.5}$$

Example 2.2.1:

Assume that $\tilde{A}$ is a fuzzy set representing the temperature of a DC motor, then $\tilde{A}$ can be expressed as

$\tilde{A}$ = *Positive Medium* , (2.2.6)

The membership values of "*Positive Medium*" are given by

$\tilde{A}$ = { (110, 0.1), (120, 0.5), (130, 0.8), (140, 1), (150, 0.8), (160, 0.5), (170, 0.1) } . (2.2.7)

By (2.2.1.4), $\tilde{A}$ can also be represented by

$\tilde{A} = 0.1/_{110} + 0.5/_{120} + 0.8/_{130} + 1.0/_{140} + 0.8/_{150} + 0.5/_{160} + 0.1/_{170}$. (2.2.8)

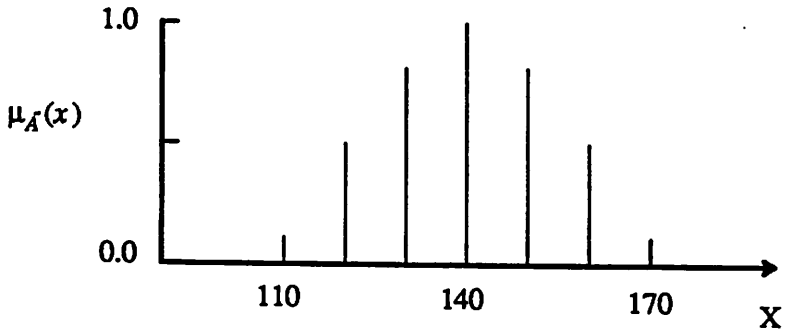

Figure 2.2.1: The membership function of $\tilde{A}$

Definition 2.2.2: Support of a Fuzzy Set

The *support* of a fuzzy set $\tilde{A}$, $S(\tilde{A})$, is the crisp set of all $x \in X$, such that $\mu_{\tilde{A}}(x) > 0$, i.e.

$$S(\tilde{A}) = \{ x \in X \mid \mu_{\tilde{A}}(x) > 0 \} . \quad (2.2.9)$$

□

Definition 2.2.3: Crossover Point

For a fuzzy set $\tilde{A}$ defined in X, x is called the *crossover point* if

$$\mu_{\tilde{A}}(x) = 0.5 . \quad (2.2.10)$$

□

Definition 2.2.4: Fuzzy Singleton

A fuzzy set $\tilde{A}$ defined in X is called a *fuzzy singleton* if the support of $\tilde{A}$ contains only one point and its membership value is 1.

□

Definition 2.2.5: α-Cut

The (crisp) set of elements which belong to the fuzzy set $\tilde{A}$ at least to the degree α, is called the α-level-set or α-cut of $\tilde{A}$,

$$A_\alpha = \{ x \in X \mid \mu_{\tilde{A}}(x) \geq \alpha \} \tag{2.2.11}$$

and $A'_\alpha = \{ x \in X \mid \mu_{\tilde{A}}(x) > \alpha \}$ is called "strong α-level-set" or "strong α-cut".

□

Note that both α-cut and strong α-cut of a fuzzy set $\tilde{A}$ are crisp sets.

Example 2.2.2:

For the $\tilde{A}$ in example 2.2.1, the α-cut of $\tilde{A}$ for $\alpha = 0.3, 0.7, 1.0$ are shown in the followings:

$$A_{0.3} = \{ 120, 130, 140, 150, 160 \}, \tag{2.2.12}$$

$$A_{0.7} = \{ 130, 140, 150 \}, \tag{2.2.13}$$

$$A_{1.0} = \{ 140 \}. \tag{2.2.14}$$

2.2.2 Basic Operations on Fuzzy Sets

A fuzzy set is defined in terms of its membership function. Hence it is obvious that the set operations of fuzzy sets will be also defined via their membership functions. Though there is actually more than one possible way to give a consistent definition of the fuzzy sets operations, the min-max operation proposed by Zadeh (1965) is the most widely accepted one.

Definition 2.2.6: Intersection

The membership function $\mu_{\vec{C}}(x)$ of the intersection $\vec{C} = \vec{A} \cap \vec{B}$ is pointwise defined by

$$\mu_{\vec{C}}(x) = \min \{ \mu_{\vec{A}}(x), \mu_{\vec{B}}(x) \}, x \in X. \tag{2.2.15}$$

□

Definition 2.2.7: Union

The membership function $\mu_{\vec{D}}(x)$ of the union $\vec{D} = \vec{A} \cup \vec{B}$ is pointwise defined by

$$\mu_{\vec{D}}(x) = \max \{ \mu_{\vec{A}}(x), \mu_{\vec{B}}(x) \}, x \in X. \tag{2.2.16}$$

□

Definition 2.2.8: Complement

The membership function of the complement of a fuzzy set $\vec{A}$, $\mu_{not\ \vec{A}}(x)$ is defined by

$$\mu_{not\ \vec{A}}(x) = 1 - \mu_{\vec{A}}(x), x \in X \tag{2.2.17}$$

□

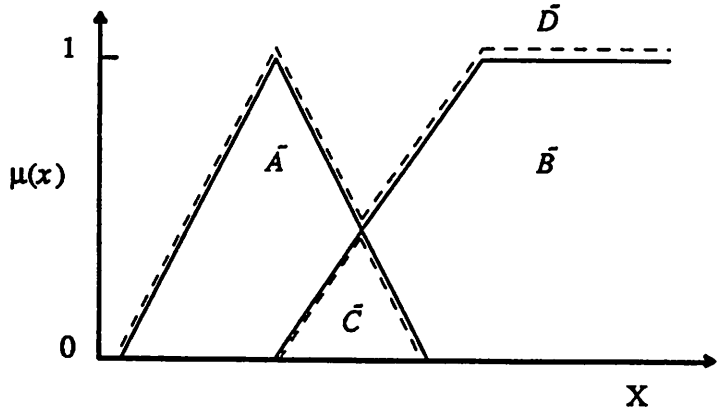

Figure 2.2.2: An example of the fuzzy set operation

Definition 2.2.9: Cartesian Product

Assume that $\tilde{A}_1, \ldots, \tilde{A}_n$ are fuzzy sets in $X_1, \ldots, X_n$ respectively. The Cartesian product of $\tilde{A}_1, \ldots, \tilde{A}_n$ is a fuzzy set in the product space $X_1 \times \cdots \times X_n$ with the membership function

$$\mu_{\tilde{A}_1 \times \cdots \times \tilde{A}_n}(x_1, \ldots, x_n) = \min \{ \mu_{\tilde{A}_1}(x_1), \ldots, \mu_{\tilde{A}_n}(x_n) \} \tag{2.2.18}$$

where $(x_1, \ldots, x_n) \in X_1 \times \cdots \times X_n$.

□

A fuzzy relation is a fuzzy subset of the product space $X_1 \times \cdots \times X_n$. In the two dimensional case, a fuzzy relation can be considered as a mapping from $X_1 \rightarrow X_2$.

Definition 2.2.10: Fuzzy Relation

Let $X_1, \ldots, X_n \subset R$ be universal sets. Then

$$\tilde{R} = \{ ((x_1, \ldots, x_n), \mu_{\tilde{R}}(x_1, \ldots, x_n)) \mid (x_1, \ldots, x_n) \in X_1 \times \cdots \times X_n \} \tag{2.2.19}$$

is called a fuzzy relation on $X_1 \times \cdots \times X_2$.

□

2.3 Plant Modeling

Most of the existing fuzzy logic controllers are designed without using any mathematical model of the underlying process. The construction procedures are generally based on the experts' understanding of the process without involving any detailed mathematical descriptions. This approach has its benefit but is not without its shortcomings. The main disadvantage is the extreme difficulty in automating the controller design of a fuzzy control system. As a result, the design of an FLC has not progressed much since it was first proposed. Therefore, some researchers have tried to work on the fuzzy identification and/or fuzzy modeling in order to develop some model-based fuzzy logic controller design theories.

One of the most notable contributions to fuzzy modeling was by Takagi and Sugeno (1985). The proposed format of a fuzzy model consists of a number of implication rules which are written as:

R:

$$\text{If} \quad f\,(x_1 \textit{ is } \tilde{A}_1, \ldots, x_k \textit{ is } \tilde{A}_k) \tag{2.3.1}$$

$$\text{then} \quad y = g\,(x_1, \cdots, x_k) \tag{2.3.2}$$

where

y Variable of the consequent whose value is inferred.

x_i Variables of the premise that also appear in the consequent.

$\tilde{A}_i$ Fuzzy sets with linear membership functions representing a fuzzy subspace in which the implication R can be applied for reasoning.

f Logical functions connecting the propositions in the premise.

g Function that implies the value of y when $x_1 \cdots x_k$ satisfies the premise.

The function f and g can be some "appropriate" functions. However, the simpliest one used is an identity function for f and linear equations for g, i.e. an implication rule is expressed as:

R:

$$\text{If} \quad x_1 \textit{ is } \tilde{A}_1 \text{ and } \cdots \text{ and } x_k \textit{ is } \tilde{A}_k \tag{2.3.3}$$

$$\text{then} \quad y = p_0 + p_1 x_1 + \cdots + p_k x_k \tag{2.3.4}$$

2.3.1 Reasoning Operations of the Fuzzy Model

Suppose that we have n implication rules R^i, $i = 1, \ldots, n$ in the format of (2.3.3)-(2.3.4) to form a rule base. Given the states

$$x_1 = x_1^0 , \ldots , x_k = x_k^0 \tag{2.3.5}$$

where $x_1^0 , \ldots , x_k^0$ are real numbers, the reasoning operations of the model are as follows:

(1) For each implication rule R^i, the consequent variable y^i is calculated by

$$y^i = p_0^i + p_1^i x_1^0 + \cdots + p_k^i x_k^0 \tag{2.3.6}$$

(2) The truth value of the i-th implication rule is given by

$$| R^i | = \bar{A}_1^i(x_1^0) \wedge \cdots \wedge \bar{A}_k^i(x_k^0) \tag{2.3.7}$$

where $|*|$ denotes the truth value of the implication and $\wedge$ is the "min" operator.

(3) The combined output of the n implications is calculated by the weighted average method, i.e.

$$y = \frac{\sum_i^n |R^i| \times y^i}{\sum_i^n |R^i|} \tag{2.3.8}$$

2.3.2 Identification

Based on the structure of (2.3.3)-(2.3.4) the identification algorithm is basically divided into three steps, i.e.

(1) *Consequent parameter identification*,

(2) *Premise parameter identification*,

(3) *Choice of premise variables*.

The identification is done by minimizing a performance index which is defined to be the root mean square of the output errors. The outline of the algorithm is shown in Figure 2.3.1. Details of the algorithm will be discussed in the following sections.

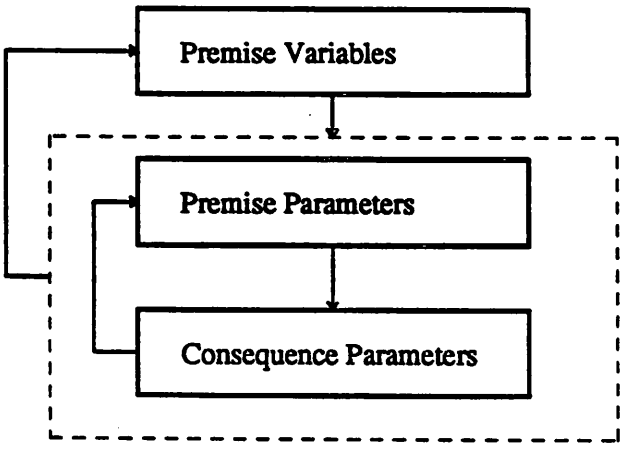

Figure 2.3.1: Outline of the algorithm

2.3.2.1 Consequent Parameters Identification

Suppose that the rule base consists of n implication rules in the form of (2.3.3)-(2.3.4). The output y with the input $(x_1, \ldots, x_k)$, by (2.3.8), is obtained as

$$y = \frac{\sum_i^n (\bar{A}_1^i(x_1) \wedge \cdots \wedge \bar{A}_n^i(x_n)) \times (p_0^i + \cdots + p_k^i x_k)}{\sum_i^n (\bar{A}_1^i(x_1) \wedge \cdots \wedge \bar{A}_n^i(x_n))} \tag{2.3.9}$$

Let

$$\beta_i = \frac{(\bar{A}_1^i(x_1) \wedge \cdots \wedge \bar{A}_n^i(x_n))}{\sum_i^n (\bar{A}_1^i(x_1) \wedge \cdots \wedge \bar{A}_n^i(x_n))} \tag{2.3.10}$$

then

$$y = \sum_i^n \beta_i (p_0^i + \cdots + p_k^i x_k) \tag{2.3.11}$$

$$= \sum_i^n (p_0^i \beta_i + \cdots + p_k^i x_k \beta_i) \tag{2.3.12}$$

With (2.3.11)-(2.3.12), we can obtain the consequent parameters by using the least square estimation method whenever the input-output pairs are given.

2.3.2.2 Premise Parameters Identification

The task of deciding the premise parameters is equivalent to determining how to divide the state space into some fuzzy subspaces so that the performance index can be minimized. The task is simplified by assuming the linear shape of the membership functions. The problem of premise identification is then reduced to a nonlinear programming problem which is solved by a complex method for the minimization.

2.3.2.3 Choice of Premise Variables

The basic concept of fuzzy modeling is to represent a dynamical process with a certain number of implication rules. The implication rules are supposed to be few and representative. Alternatively, the major task of fuzzy modeling is to divide the input-output space into regions such that each region can be described by one implication rule.

The choice of premise variables is done by successfully dividing the state space into subspaces, then choosing the one associated with the minimal performance index. For example:

Suppose that we wish to build a fuzzy model for a k-input ($x_1, \ldots, x_k$) and single-output y system.

(1) Divide the range of x_1 into two fuzzy subspaces "small" and "big", and do not divide the other variables $x_2, \ldots, x_k$. The model will thus contain only two implication rules as:

if x_1 is $small_1$ then ...

if x_1 is big_1 then ...

Call it model 1-1. Similarly, a model with x_2 divided into two subspaces and other states undivided is called model 1-2 and so on. In this way, we shall have k models, each with 2

implication rules.

(2) Find the optimum parameters for both premise and consequent by the methods described in the previous sections. Then choose the model with the least performance index value.

(3) Proceed with the second level partition which is basically the same as the first level partition and create the 2-i model.

(4) Repeat the procedure of (2) and (3) until the performance index becomes less than the predetermined value or the number of implications exceeds a predetermined number.

2.3.3 Remarks

Fuzzy modeling is a very important issue in developing a model-based fuzzy logic controller design theory. Sugeno's work is a foundation for future research. However, it is not clear at this moment as to how to systematically design an FLC based on his model. We also believe that the extension from a linear consequent equation to a polynomial one should improve the performance of the modeling. The tradeoff is, of course, the increase of the computational complexity. We feel that the goal of automating FLC design is an important and worthwhile one and that the progress towards this is very promising.

2.4 Fuzzy Logic Controller

2.4.1 Basic Structure

As shown in Figure 2.1.2, a fuzzy logic controller consists of five major parts: (1) a rule base which contains a number of control rules, (2) a database which defines the membership functions of the linguistic terms used in the rule base, (3) a computation unit which performs the necessary inference operations upon the control rules, (4) a fuzzification interface which relates the real sensor inputs with the fuzzy terms so that they can be processed by the inference

mechanism, and (5) a defuzzification interface which transforms the "fuzzy" results of the inferences in the rule base to real numbers so that they can be used to control the process.

A rule base usually consists of a number of *if –then* control rules which are of the form:

$$\text{if } x_1 \text{ is } PS \text{ , and } x_2 \text{ is } PL \text{ , then } u \text{ is } PS \qquad (2.4.1)$$

where x_1 and x_2 are the state variables, u is the output of the fuzzy logic controller, *PS* (*Positive Small*) and *PL* (*Positive Large*) are fuzzy sets with membership functions defined by the experts.

In fuzzy logic control, the control rules are given by the experts. Therefore, it is necessary for the experts to specify the universe of discourse to give the semantics of a fuzzy variable, i.e. the membership function. An example of the membership functions of *PS* (*Positive Small*) and *PL* (*Positive Large*) is shown in Figure 2.4.1.

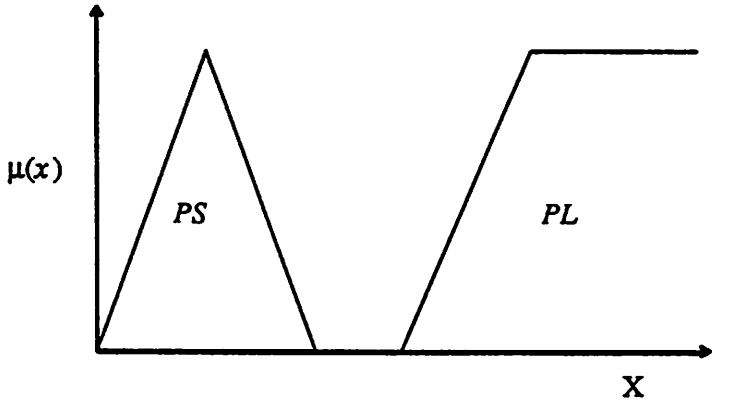

Figure 2.4.1: The membership functions

Generally, there is no restriction on the shape of a membership function. However, there are some functions, such as the bell-shaped function, the triangular function, and the monotonic linear function, which are usually adopted in the formulation of the membership function of a fuzzy variable. Since the meaning of a fuzzy variable is normally defined by the expert according to his own terminology, a fuzzy term may represent different meanings for different variables.

For example, the *PS* (*Positive Small*) in (2.4.1) has different interpretations for x_1 and x_3 respectively.

To summarize, a rule base with n control rules of a fuzzy logic controller is of the following form:

$$\text{if } x_1 \text{ is } \tilde{A}_{11}, \ldots, x_p \text{ is } \tilde{A}_{1p}, \text{ then } u_1 \text{ is } \tilde{B}_{11}, \ldots, u_q \text{ is } \tilde{B}_{1q}, \tag{2.4.2}$$

$$\text{if } x_1 \text{ is } \tilde{A}_{21}, \ldots, x_p \text{ is } \tilde{A}_{2p}, \text{ then } u_1 \text{ is } \tilde{B}_{21}, \ldots, u_q \text{ is } \tilde{B}_{2q}, \tag{2.4.3}$$

$$\ldots$$

$$\ldots$$

$$\text{if } x_1 \text{ is } \tilde{A}_{n1}, \ldots, x_p \text{ is } \tilde{A}_{np}, \text{ then } u_1 \text{ is } \tilde{B}_{n1}, \ldots, u_q \text{ is } \tilde{B}_{nq}, \tag{2.4.4}$$

where $x_i, i = 1, \ldots, p$ are the state variables, $u_j, j = 1, \ldots, q$ are the action variables or the controller outputs, $\tilde{A}_{ij}$, $\tilde{B}_{ij}$, $i = 1, \ldots, p$, $j = 1, \ldots, q$ are fuzzy variables. In most cases, the number of the action variables is one, i.e. we normally consider only one controller output.

2.4.2 Operations of a Fuzzy Logic Controller

The operation of a fuzzy logic controller depends on the fuzzy reasoning method adopted. There are generally two kind of fuzzy reasonings:

(1) based on compositional rules of inference,

(2) based on fuzzy logic, e.g. fuzzified Lukasiewicz logic.

We shall introduce both methods in the following sections. However, in fuzzy logic control, the second method based on fuzzy logic is usually used because of the simplicity in computation.

2.4.2.1 The First Type

To simplify the notation, we shall use only two rules to represent the rule base and each rule has only two state variables and one action variable. This can be easily extended to systems with more state and action variables. Let the rule base be:

$$\text{if } x_1 \text{ is } \tilde{A}_{11}, x_2 \text{ is } \tilde{A}_{12}, \text{ then } u \text{ is } \tilde{B}_1 \tag{2.4.5}$$

$$\text{if } x_1 \text{ is } \tilde{A}_{21}, x_2 \text{ is } \tilde{A}_{22}, \text{ then } u \text{ is } \tilde{B}_2 \tag{2.4.6}$$

Fuzzification

Suppose that the premise part of the control rules is given from the sensor readings which may be fuzzy terms or real numbers. For fuzzy terms, each reading $\tilde{R}_{ij}$ is matched to the premise $\tilde{A}_{ij}$ by finding the matching coefficient m_{ij} to be:

$$m_{ij} = \min_{x_i} (\tilde{A}_{ij}(x_i), \tilde{R}_{ij}(x_i)) \tag{2.4.7}$$

However, the sensor readings are generally real numbers in most practical situations. These real-valued sensor measurements, e.g. x_1^0 and x_2^0, are matched to their corresponding fuzzy variables by finding their corresponding membership values, such as:

$$\tilde{A}_{11}(x_1^0), \tilde{A}_{12}(x_2^0), \tilde{A}_{21}(x_1^0), \tilde{A}_{22}(x_2^0) \tag{2.4.8}$$

Graphically, the operations of a fuzzy controller is shown in Figure 2.4.2.

Fuzzy Reasoning

For all the control rules, (2.4.5)-(2.4.6), in the rule base, we derive the truth values of each rule in the premise by forming the conjunction of all the fuzzy variables either by:

$$w_1 = \tilde{A}_{11}(x_1^0) \wedge \tilde{A}_{12}(x_2^0) \tag{2.4.9}$$

$$w_2 = \tilde{A}_{21}(x_1^0) \wedge \tilde{A}_{22}(x_2^0) \tag{2.4.10}$$

or by

$$w_1 = \tilde{A}_{11}(x_1^0) \times \tilde{A}_{12}(x_2^0) \tag{2.4.11}$$

$$w_2 = \tilde{A}_{21}(x_1^0) \times \tilde{A}_{22}(x_2^0) \tag{2.4.12}$$

The latter one (the multiplicative weights) has been used more often than the previous "min" operator because of better smoothness properties.

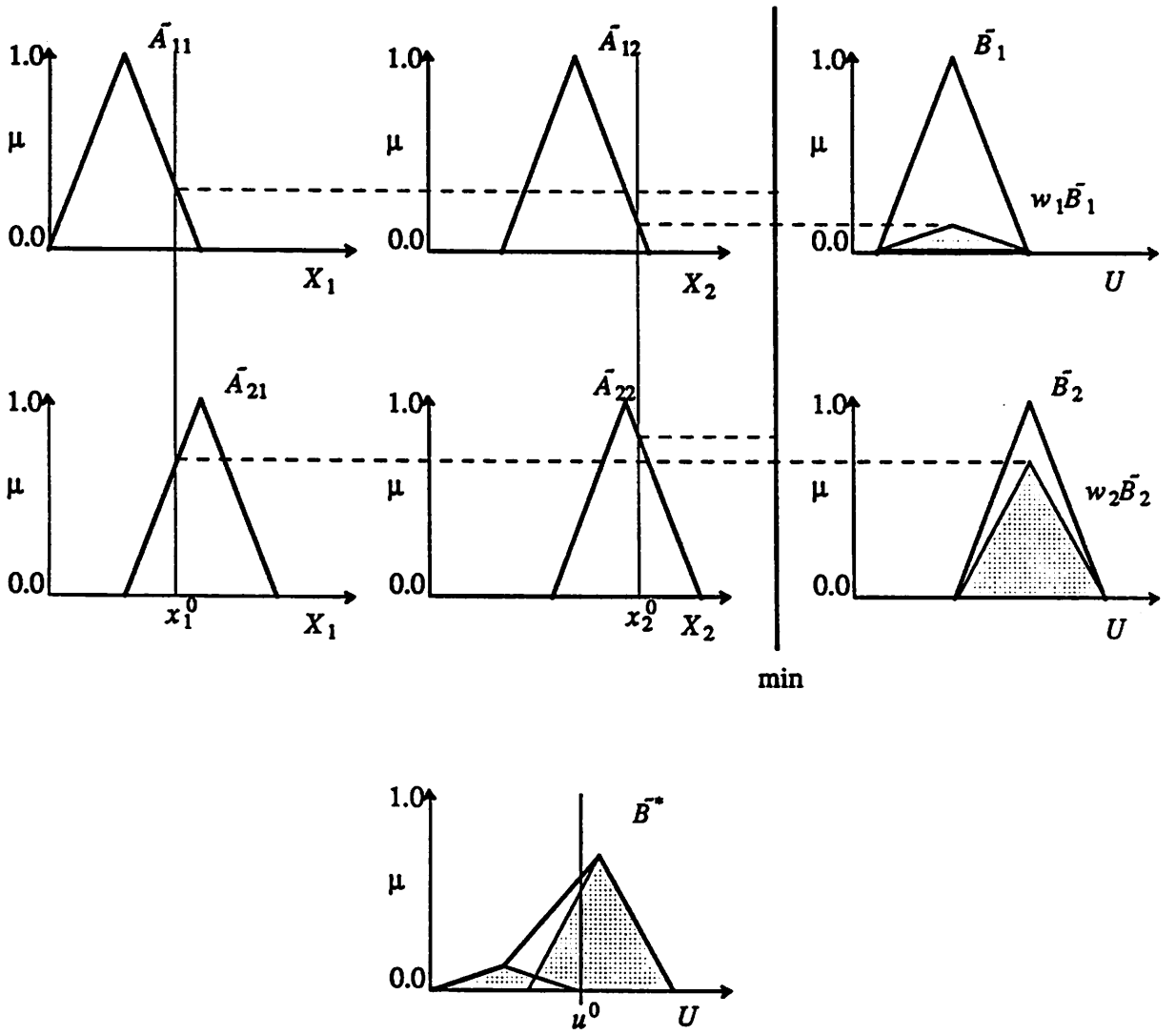

Figure 2.4.2: The first type of fuzzy reasoning

The effect of the control rules or their action part are represented as the fuzzy sets $w_1\tilde{B}_1$ and $w_2\tilde{B}_2$ and are calculated by:

$$w_1\tilde{B}_1(u) = w_1 \times \tilde{B}_1(u) \tag{2.4.13}$$

$$w_2\tilde{B}_2(u) = w_2 \times \tilde{B}_2(u) \tag{2.4.14}$$

Then the combined result of the inferences forms a fuzzy set $\tilde{B}^*$ as:

$$\tilde{B}^* = w_1\tilde{B}_1 \cup w_2\tilde{B}_2 \tag{2.4.15}$$

where the $\cup$ operation is generally the "max" function.

Defuzzification

The purpose of the defuzzification process is to transform the output of the inferences, which is a fuzzy set, to a real number so that it could be used to control a process, i.e. we need to find a single number to represent a fuzzy set. The most commonly adopted method is to find the value corresponding to the center of mass of the membership function as shown in (2.4.16).

$$u^0 = \frac{\int \tilde{B}^*(u)\, u\, du}{\int \tilde{B}^*(u)\, du} \tag{2.4.16}$$

There may be many rules in the rule base of a fuzzy controller. Generally, if the weights of some rules are smaller than a certain threshold value, then the rules are omitted in reasoning. A fuzzy logic controller using this type of reasoning method has been proved to function like a multirelay (Braae and Rutherford (1979)).

2.4.2.2 The Second Type

The second type of reasoning is based on fuzzy logic. One major difference between the first type and the second type of reasoning is the shape of the membership functions used. The first type, as shown in Figure 2.4.2, uses bell-shape or triangular functions for the membership functions. The second type is applicable only when monotonic membership functions are used

for the consequent in the rule base. The reasoning operation of the second type is shown in Figure 2.4.3.

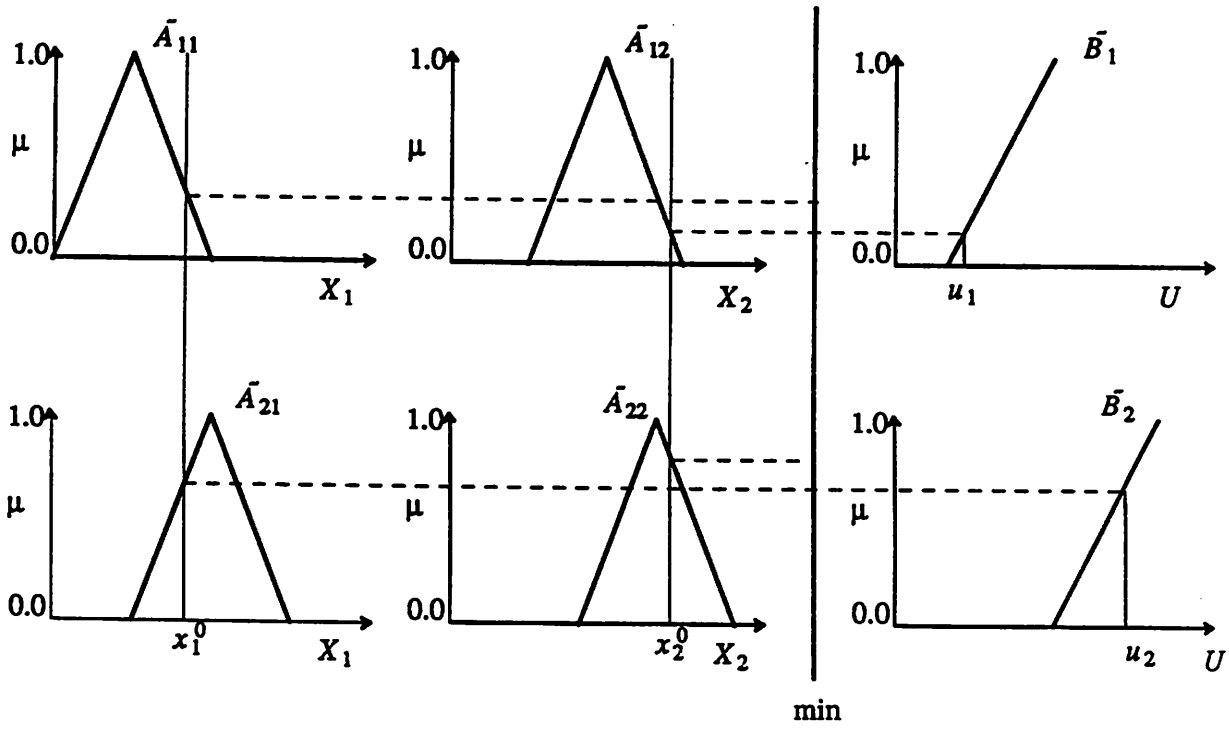

Figure 2.4.3: The second type fuzzy reasoning

The fuzzification procedures for the second type are the same as the first. The truth values or weights of the different rules are also derived by (2.4.9)-(2.4.12). However, since the membership functions of $\bar{B}_1$ and $\bar{B}_2$ are monotonic functions, the action part can be calculated by

$$u_1 = \bar{B}_1^{-1}(w_1) \tag{2.4.17}$$

$$u_2 = \bar{B}_2^{-1}(w_2) \tag{2.4.18}$$

Then the output of the inferences is given by the weighted average method to be

$$u^0 = \frac{w_1 u_1 + w_2 u_2}{w_1 + w_2} \tag{2.4.19}$$

The advantage of the second type of reasoning is its computational simplicity. The first type operates with fuzzy sets throughout the whole procedure. It takes to a lot of memory space and computation time to derive the final control output. The second type works with real numbers as much as possible to reduce the usage of memory and computation time. Since fuzzy logic controllers are more complicated in structure than other kinds of controllers, computational efficiency is very important to real-time fuzzy logic control.

2.5 Industrial Applications

Two most well-known industrial applications of fuzzy logic control are described in this section. The cement kiln control (Holmblad and Ostergaard (1982)) was the first successful example that practically implemented a fuzzy logic controller on an industrial process. The automatic train control by Hitachi, Japan was the most notable application which considered not only the quantitative factors, such as fuel efficiency and braking frequency, but also human factors, such as the comfortness of its passengers.

2.5.1 Cement Kiln Control

2.5.1.1 Plant and Process Description

Cement is manufactured by heating a slurry consisting of clay, limestone, sand and iron ore to a high temperature to form the complex compounds of cement, dicalcium silicate (Ca_2Si), tricalcium silicate (Ca_3Si), tricalcium aluminate (Ca_3Al), and tetracalcium aluminoferrite ($Ca_4Al\,Fe$), Three different heating stages are involved to complete the process. The kiln consists of a long steel shell about 130 m in length and 5 m in diameter and is mounted at a slight inclination to the horizontal. The shell rotates slowly at approximately 1 rev/min and the slurry is fed in at the upper back end of the kiln. The complete process of transporting the material through the kiln takes about 3 hours and 15 minutes with a further 45 minutes spent in the clinker (referred to the end product) cooler. The heat in the kiln is provided by pulverized coal mixed with air. The hot combustion gas is sucked through the kiln by an induction fan at the back end of the kiln.

There are five measurable state variables and three control variables used in the process. The state variables are:

(1) exhaust gas temperature: back end temperature (BT),

(2) intermediate gas temperature: ring temperature (RT),

(3) burning zone temperature (BZ),

(4) oxygen percentage in exhaust gas (O_2),

(5) litre weight (LW): indicates clinker quality.

The three control variables are:

(1) kiln speed (KS),

(2) coal feed (CS),

(3) induced drought fan speed (BF).

The cement manufacturing process is very complicated and involves many factors, time delays, heat conductions, etc. The dynamic responses were modeled by using the "reaction" curve method with a pure time delay. The representation is very much simplified because the kiln is subject to random disturbances from several sources which can not be quantified and the kiln's dynamics is extremely nonlinear and variable depending on the prevailing kiln conditions. The whole process was controlled manually by operators. An experienced operator monitored the control panel and made proper control actions based on the sensor information, operational procedures, and most importantly, experiences.

2.5.1.2 Fuzzy Logic Controller Design and Its Results

The operational procedures and operators' experiences were formulated in the form of an if-then statement.

if <condition> then <control actions>

The condition part consists of linguistic statements of the five state variables, such as "the litre weight is slightly high" or "the back end temperature is somewhat low", and the action part are statements like "make a medium reduction in coal feed rate" or "open slightly the exhaust gas fan damper". The key items in the control rules are terms like "medium reduction", "slightly high", and "somewhat low". These linguistic terms were represented by fuzzy sets with membership values between 0 and 1. The fuzzy control system was then coded and incorporated into the F. L. Smidth's computerized process monitoring system, FLS Supervision, Dialogue and Reporting (SDR) system.

The system had then put into continuous operations for two years when Holmblad and Ostergaard reported that it was capable of maintaining a good and stable operation of the kiln consistently. The success of this example demonstrated the plausibility of fuzzy logic control with a real industrial application for the first time and was considered an important milestone of its development.

2.5.2 Automatic Train Control

Another important industrial application of fuzzy logic control was applied on the automatic train operation (Yasunobu and Miyamoto (1985)).

2.5.2.1 Automatic Train Operations

The automatic train control system (ATC) has two sub-systems, namely, the automatic train operation system (ATO) and the automatic train protection system (ATP). The input data of ATO are distance pulses of a tacho-generator, cab signals of the ATP on-board system, position marker detected signals which indicate the train position, and supervisory commands from the automatic train supervision system (ATS). The output data of ATO are powering and braking commands to a traction controller and a brake controller.

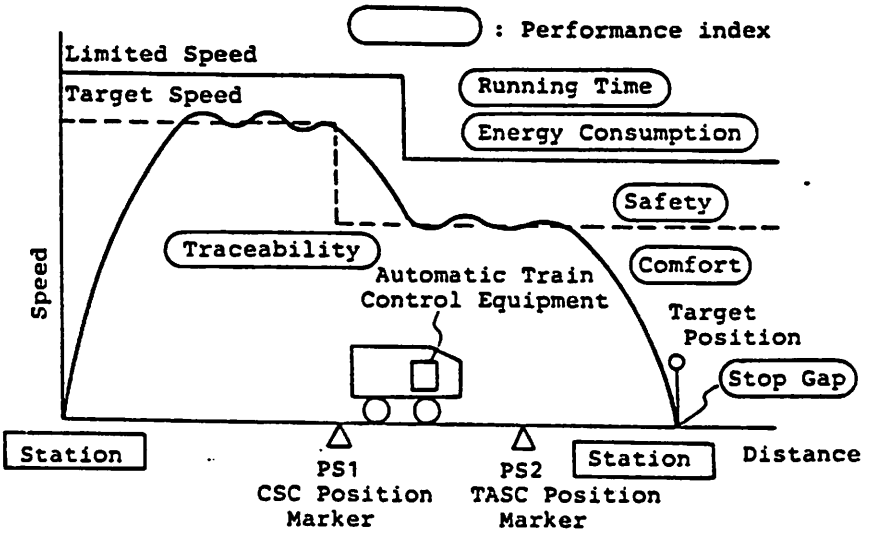

Figure 2.5.1: The automatic train operation

(Source: (Yasunobu, 1985))

The ATO was operated manually by human experts whose control actions were decided by evaluating the sensor inputs and various performance indices such as safety, riding comfort, and stop accuracy. The human operator's strategies were basically divided into two parts. One was the Constant Speed Control (CSC) and the other was the Train Automatic Stop Control (TASC) as shown in Figure 2.5.1. With the departure command, the operator first brought the train to a target speed. The CSC was then performed according to rules such as:

(C-1) For safety: if the train speed exceeds the speed limit, the maximum brake notch is selected.

(C-4) For riding comfort: if the train speed is in the predetermined allowance range, the control notch will not be changed.

In the second stage of the train operation, the human operator started the TASC by detecting the position marker which indicated the distance to the target station. Then the control command was selected with rules like:

(T-1) For riding comfort: when the train is in the TASC zone, the control notch will not be changed if the train will stop at the predetermined allowance zone.

(T-3) For the stop gap accuracy: when the train in the TASC zone, change the notch a little if the train will not stop within the predetermined allowance zone.

2.5.2.2 Fuzzy Logic Controller Design and Its Results

The meaning of the performance indices, such as danger, safety, and comfort, were first characterized by fuzzy sets with different membership functions. The linguistic rules of the CSC and TASC operations were transformed into fuzzy control rules like:

(C-4) IF (DN is 0 -> S is SS and T is TG) Then DN is 0.

where DN is the difference of control notch, S is the safety index, SS is the fuzzy term representing "safe", T is the traceability index, and TG is the fuzzy term representing "good trace".

The fuzzy ATO control system was built into an on-board controller by a micro-computer (M-6800) with a sampling time of 100 ms. A conventional PID controller was also built for the purpose of comparison. In field tests, the fuzzy ATO system gave significant improvements over a conventional PID controller with respect to the safety, riding comfort, accuracy of stop gap, running time and energy consumption. Also, it was able to operate robustly under the changing environmental conditions and saved over 10% of energy compared with conventional ATO controllers.

2.6 Concluding Remarks

This chapter gave a preliminary introduction to fuzzy set theory and fuzzy control systems which will be useful for the following chapters. We do not intend to cover all the previous work in this field, but only describe the topics which are relevant here. From the context of this chapter, we can see that fuzzy logic control is still in its developing stages. Many questions need to be answered, namely, the modeling problem, the model-based controller design, stability analysis etc. There is plenty of research space to be carefully examined and worked on. Hopefully, fuzzy logic control will become a major tool for dealing with some systems which are difficult to control by conventional methods.

Chapter 3

Stability Property of Fuzzy Control Systems

3.1 Introduction

The stability of a fuzzy control system (FCS) has always attracted the attention of researchers who are interested in formal FCS analysis. An interesting question is concerned with the formulation of a definition of stability for an FCS, which can describe its dynamic behavior. The problem is very interesting and also difficult because an FCS cannot be represented in terms of the well-studied differential or difference equations. Moreover, its *fuzzy* nature also makes it very hard to have a precise definition of stability. To date, several methods have been proposed to evaluate the stability of an FCS. Kickert and Mamdani (1978) used the describing function method to evaluate the stability of fuzzy control systems. A linguistic phase plane trajectory for analyzing the stability of fuzzy systems has been proposed by Braae and Rutherford (1979). DeGlas (1984) suggested an invariance principle for continuous-time fuzzy dynamic systems. An energetistic stability criterion has also been proposed by Kiszka, Gupta and Nikiforuk (1985).

In conventional system theory, the stability property of a system is always fully investigated and has top priority in controller design. However, for most of the applications of fuzzy control, the stability property is always assumed *a priori* without detailed analysis. Part of the reasons for this assumption is that there is no commonly accepted method for FCS analysis. Practically, the design of a fuzzy logic controller is based on the knowledge of the process experts. The better the experts' understanding of the process, the better is the performance of an FCS. To explain the stability nature of a large class of fuzzy control systems, we propose a notion, namely the "expert's Lyapunov function". Very often, an expert has his own idea about the stability criterion of a system. The criterion may not be of a crisp mathematical form, but is more likely to be a linguistic expression, such as:

If: the temperature is NOT VERY-HIGH,

and the rate of change of the temperature is NOT LARGE,

then: System is *stable.*

During the construction of a fuzzy logic controller, an expert helps a control engineer to incorporate his implicit notion of stability into the rule base. Although the expert may not be familiar with the term "Lyapunov function", he must have his own stability notion , which is not necessarily quantitative, in mind. This notion of stability is the underlying reason for the stability of a fuzzy control system and will be called the "expert's Lyapunov function". A control engineer can verify the stability of an FCS by computing the expert's Lyapunov functions from the linguistic implication rules. An expert can also heuristically write down a Lyapunov function based on his notion to help develop the rule base.

This chapter is organized as follows: some previous work on the stability analysis of an FCS is given in section 2; a stability theorem and a design method are described in section 3; then the proposed idea is illustrated by an example using an inverted pendulum in section 4 ; finally, the results of the chapter are summarized in the last section.

3.2 Review of Previous Work

We shall discuss some of the approaches proposed by researchers for stability analysis of a fuzzy control system in this section. Although many ideas have been investigated by researchers with different degrees of success, none can give a practically useful description of the stability of an FCS. Despite the unsatisfying results, some innovative ideas merit further study and will be briefly discussed in this section.

3.2.1 Conventional System Approach

Most of the researchers in the field of FCS analysis have conventional control backgrounds. It is therefore very natural for them to extend the concept of stability in conventional system theories to fuzzy control systems. In linear system theories, many properties, including stability, are discussed through the state equations. Following a similar approach, we can represent a fuzzy control system by a fuzzy relational matrix $\tilde{R}$ and discuss its dynamic behavior through the next state mapping.

3.2.1.1 Basic Structure

For a simple dynamic system as shown in Figure 3.2.1, assume that the dynamics of the system can be described by the following discrete fuzzy relational equation:

$$\tilde{X}_{k+1} = \tilde{X}_k \, \tilde{U}_k \circ \tilde{R} \tag{3.2.1}$$

where $\tilde{X}_k$ and $\tilde{X}_{k+1}$ are fuzzy sets defined on a multidimensional finite discrete state space X and $\tilde{U}_k$ is a fuzzy set defined on a multidimensional finite discrete input space U.

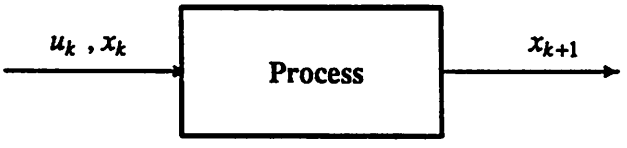

Figure 3.2.1: A simple dynamic system

$\tilde{X}_k$ is the current state; $\tilde{U}_k$ is the current input and $\tilde{X}_{k+1}$ denotes the next state of the process. $\tilde{X}_k \, \tilde{U}_k$ is a fuzzy relation defined in the Cartesian product space $X \times U$ and is defined by

$$\mu_{\tilde{X}_k \tilde{U}_k}(x, u) = \mu_{\tilde{X}_k}(x) \wedge \mu_{\tilde{U}_k}(u) \tag{3.2.2}$$

where μ is the membership function and $\wedge$ denotes the "min" operator. The finite discrete fuzzy

relation $\vec{R}$ is defined on the product space $X \times U \times X$ and is considered as a next state mapping. The next state $\vec{X}_{k+1}$ is calculated from

$$\mu_{\vec{X}_{k+1}}(x_{k+1}) = \underset{x_k, u_k}{\vee} [\mu_{\vec{X}_k \vec{U}_k}(x_k, u_k) \wedge \mu_{\vec{R}}(x_k, u_k, x_{k+1})] \tag{3.2.3}$$

where $\vee$ denotes the "max" operator.

3.2.1.2 Dynamic Behavior of a Fuzzy Control System

Some equalities will be given in this section first, before the examination of the dynamic behavior of (3.2.1).

Theorem 3.2.1

If: $\vec{X}$ is a finite discrete fuzzy set on X and $\vec{Q}$ is a finite discrete fuzzy relation on $X \times X$,

then:

$$(\vec{X} \circ \vec{Q}) \circ \vec{Q} = \vec{X} \circ (\vec{Q} \circ \vec{Q}) \tag{3.2.4}$$

Proof: See Appendix A.

□

Theorem 3.2.2

If: $\vec{X}$ and $\vec{U}$ are finite discrete fuzzy sets on X and U, and $\vec{R}$ is a fuzzy relation defined on $X \times U \times X$,

then:

$$\vec{X}\,\vec{U} \circ \vec{R} = \vec{X} \circ (\vec{U} \circ \vec{R}) = \vec{U} \circ (\vec{X} \circ \vec{R}). \tag{3.2.5}$$

Proof: See Appendix A.

□

Suppose that we consider a constant input $\tilde{U} = \tilde{U}_c$ response of (3.2.1) with an initial condition $\tilde{X} = \tilde{X}_0$, then we can derive

$$\tilde{X}_1 = \tilde{X}_0 \tilde{U}_c \circ \tilde{R} \tag{3.2.6}$$

$$\tilde{X}_2 = \tilde{X}_1 \tilde{U}_c \circ \tilde{R} \tag{3.2.7}$$

$$\cdots$$

$$\cdots$$

$$\tilde{X}_N = \tilde{X}_{N-1} \tilde{U}_c \circ \tilde{R} \tag{3.2.8}$$

From (3.2.6)-(3.2.8), we have

$$\tilde{X}_2 = (\tilde{X}_0 \tilde{U}_c \circ \tilde{R}) \circ (\tilde{U}_0 \circ \tilde{R}) \tag{3.2.9}$$

$$= (\tilde{X}_0 \circ (\tilde{U}_c \circ \tilde{R})) \circ (\tilde{U}_0 \circ \tilde{R}) \tag{3.2.10}$$

$$= (\tilde{X}_0 \circ \tilde{Q}) \circ \tilde{Q} \qquad (\tilde{Q} = \tilde{U}_c \circ \tilde{R}) \tag{3.2.11}$$

Also, by Theorem 3.2.2, we have

$$\tilde{X}_2 = \tilde{X}_0 \circ (\tilde{Q} \circ \tilde{Q}) \tag{3.2.12}$$

$$= \tilde{X}_0 \circ \tilde{Q}^2 \tag{3.2.13}$$

Following the same procedure, we can get

$$\tilde{X}_N = \tilde{X}_0 \circ \tilde{Q}^N \tag{3.2.14}$$

From (3.2.14), the asymptotic behavior of the system depends on the behavior of $\tilde{Q}_N$, as $N \to \infty$. However, it was shown by Thomason (1977) that the powers of any fuzzy relational matrix will either converge to a fixed fuzzy matrix or to cycles with a finite period.

3.2.1.3 The Concept of Stability

To discuss the stability of a system, it is essential to clearly define the meaning of "unstable". In the conventional system theories, "unboundedness" is an unmistakable indication of instability. However, this concept of instability becomes ambiguous when a system is represented by fuzzy sets. From the analysis in section 3.2.1.2, we can see that the final state $\vec{X}_N$ always converges or cycles with a finite period, which is clearly different from the conventional analysis. It becomes obvious that a different kind of stability, i.e. a stability with degrees, should be used to analyze fuzzy control systems. The degrees of stability might be a real-numbered index or even some linguistic statements such as, "very stable", "more or less unstable". In the following paragraph, we shall talk about a stability measure proposed by Tong (1980).

Consider that a state space is partitioned into two non-fuzzy regions, X_I the interior set or the stable region, and X_B the boundary set or the unstable region. The degree of stability of a state $\vec{X}$ is given by

$$\sigma(\vec{X}) = 1 - I(\vec{X}, X_B) \tag{3.2.15}$$

where $I(\vec{X}, X_B)$ denotes the degree to which the fuzzy set $\vec{X}$ is included in the unstable region X_B. The choice of the function I is not obvious and is left open. Two possible candidates were proposed as:

$$I_1(\vec{X}, X_B) = \vee((\mu_{\vec{X}}(x) \wedge \mu_{X_B}(x)) \tag{3.2.16}$$

$$I_2(\vec{X}, X_B) = \wedge([1 - (\mu_{\vec{X}}(x)] \vee \mu_{X_B}(x)) \tag{3.2.17}$$

Examples of the two stability functions are shown in Figure 3.2.2. The degrees of stability are

$$\sigma_1(\vec{X}_1) = 1.0, \quad \sigma_1(\vec{X}_2) = 1 - \alpha, \quad \sigma_1(\vec{X}_3) = 0.0, \quad \sigma_1(\vec{X}_4) = 0.0, \tag{3.2.18}$$

$$\sigma_2(\vec{X}_1) = 1.0, \quad \sigma_2(\vec{X}_2) = 1.0, \quad \sigma_2(\vec{X}_3) = \beta, \quad \sigma_2(\vec{X}_4) = 0.0. \tag{3.2.19}$$

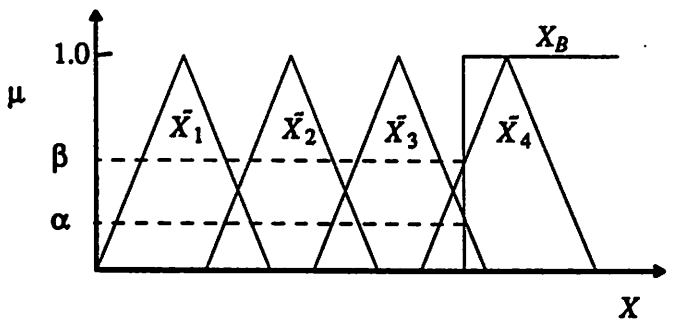

Figure 3.2.2: Stable and unstable states

3.2.1.4 Remarks

The so called "soft" definition of stability which considers different degrees of stability is likely the solution to the stability definition of fuzzy control systems. Tong's stability function demonstrates a possible formulation of such a concept. However, we still need an operational definition which can be incorporated into a practical design method. Moreover, the analysis above also shows its weakness in describing the behavior of a fuzzy dynamic system. The final state is obviously obscured by the fuzzy iterations, which might pose an even more serious problem.

3.2.2 Energetistic Approach

The concept of "energy" is commonly used as a measure of the stability of a dynamic system. One famous example is from Lyapunov function theory which uses a Lyapunov function as a generalized energy function to indicate the stability property of a dynamic system (Vidyasagar (1973)). The energetistic approach by Kiszka, Gupta and Nikiforuk (1985) is based on the assumption that a dynamic system is *stable* if its total energy decreases monotonically until an equilibrium state is reached. In their work, they try to answer questions such as: Is a fuzzy system

stable, unstable, or oscillatory? What causes the instability of a fuzzy system? How can it be removed? A summary of their work is given in this section. Although the questions are very interesting, we are not convinced that the answers have been found.

3.2.2.1 Basic Structure

The basic equation of a fuzzy dynamic system used in this section is the same as in (3.2.1) and is of the form:

$$\vec{X}_{k+1} = \vec{X}_k \circ \vec{U}_k \circ \vec{R}, \quad k = 0, 1, \cdots \tag{3.2.20}$$

where $\vec{X}_k$ and $\vec{X}_{k+1}$ are state variables, which are represented by fuzzy sets, at the time instant k and $k+1$ respectively.

Therefore, the constant input response of the system is also the same as in (3.2.14),

$$\vec{X}_N = \vec{X}_0 \circ \vec{Q}^N \tag{3.2.21}$$

where $\vec{Q} = \vec{U}_c \circ \vec{R}$, and $\vec{U}_c$ is a fuzzy constant.

3.2.2.2 Energetistic Stability Method

It is intuitively true that a dynamic system is stable if its total energy decreases monotonically until an equilibrium state is reached. The argument should also apply to a fuzzy control system since it belongs to a class of nonlinear dynamic systems. However, the question remains as to how to define a "generalized energy function" which is related to the stability of an FCS. Some comparisons and observations were made by Kiszka to conclude that the desired "energy" function of a fuzzy set should depend on its *support* and *shape*. Two fuzzy sets, both named *medium*, are shown in Figure 3.2.3. $\vec{X}$ and $\vec{X}'$ have the same value corresponding to the maximum membership. But $\vec{X}'$ has a wider spread than $\vec{X}$. Hence, it is expected that

$$E(\vec{X}' = medium) > E(\vec{X} = medium) \tag{3.2.22}$$

where E ( ) denotes the energy function of a fuzzy set.

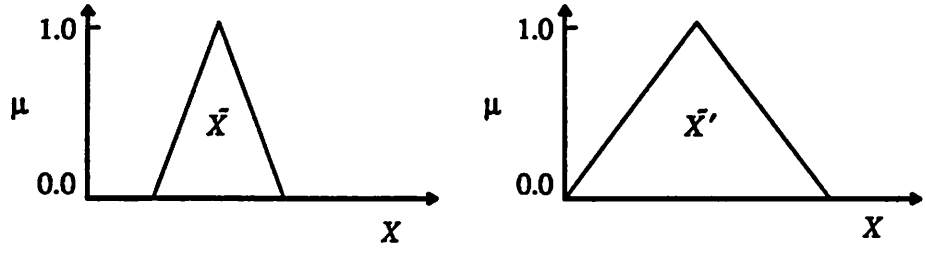

Figure 3.2.3: Two fuzzy sets named *medium*

A generalized energy function is formulated to account for its connections to the support and shape of the membership function of a fuzzy set. The definition is

$$E(\bar{X}) = \frac{1}{n} \sum_{i=1}^{n} w(x_i) f(\mu_{\bar{X}}(x_i)) \tag{3.2.23}$$

where w( ) is a function which relates the support of $\bar{X}$ to the energy function E, f ( ) is a function which relates the shape of the membership function of $\bar{X}$ to E, n is the cardinality of $\bar{X}$, and $\frac{1}{n}$ is a normalization factor.

The definition of (3.2.23) is heuristic and does not have any theoretical justification. Also, it is not clear as to what kind of functions w and f should be. Kiszka simplified (3.2.23) by using the functions

$$w(x) = x \tag{3.2.24}$$

$$f(x) = x \tag{3.2.25}$$

i.e. (3.2.23) becomes

$$E(\bar{X}) = \frac{1}{n} \sum_{i=1}^{n} x_i \, \mu_{\bar{X}}(x_i) \tag{3.2.26}$$

Now consider a fuzzy dynamic system described by the fuzzy relation equation

$$\vec{X}_{k+1} = \vec{X}_k \circ \vec{Q} \tag{3.2.27}$$

which has the solution (3.2.21) when given the initial state $\vec{X}_0$. We can further define the energy function of the state at each time instant k by

$$E(\vec{X}_k) = E(\vec{X}_0 \circ \vec{Q}^k), \qquad k = 1, 2, \cdots \tag{3.2.28}$$

A characteristic function of energy is also given by

$$\Delta E_k = E(\vec{X}_0 \circ \vec{Q}^k) - E(\vec{X}_0 \circ \vec{Q}^{k-1}), \qquad k = 1, 2, \cdots \tag{3.2.29}$$

Based on their intuition and experience, Kiszka and his colleagues developed some rules of stability for a fuzzy dynamic system.

Conjectured Rules of Stability

(1) The fuzzy dynamic system described by (3.2.27) is said to be *stable* if

$$\Delta E_k \le 0, \qquad \text{for } k \to \infty. \tag{3.2.30}$$

(2) The fuzzy dynamic system described by (3.2.27) is said to be *unstable* if

$$\Delta E_k \ge 0, \qquad \text{for } k \to \infty. \tag{3.2.31}$$

(3) The fuzzy dynamic system described by (3.2.27) is said to be *oscillatory* with a period τ if

$$| \Delta E_k | = | \Delta E_{k+\tau} |, \qquad \text{for } k \to \infty. \tag{3.2.32}$$

where $|*|$ denotes the absolute value.

3.2.2.3 Remarks

The approach proposed by Kiszka is very interesting and, in our option, deserves further investigations. However, the value of their work is mostly in the innovative idea. Many questions remain unsolved, such as, what is the physical meaning of the proposed "energy"? how does this energy relate to the stability of an FCS? Besides, it has been proved that the powers of any

fuzzy relational matrix will either converge to a fixed fuzzy matrix or cycles with a fixed period. That means, Kiszka's energy function can never "blow up" like the conventional one. Obviously, the work is far from complete.

3.3 Expert's Lyapunov Function

There have been many successful experimental and industrial applications of fuzzy logic control since it was first proposed by Zadeh (1968). The scales and complexity of some applications are very large, e.g. automatic train control. It is expected that this kind of large scale project needs detailed planning and analysis. One of the important tasks is the stability analysis of the system. However, due to the fact that there is no existing method which can perform satisfactory analysis of an FCS, the design of these industrial fuzzy logic controllers was reportedly done solely by human experts. It would, therefore, give rise to questions like: why are these systems stable? how is the stability incorporated into the system design?

To answer the questions, we propose a notion, namely the "expert's Lyapunov function". The stability of almost all fuzzy logic controllers is guaranteed by the existence of such a notion in the experts' mind. Generally, an expert always has his own understanding of the stability of a system, which may be explicit or implicit. In the process of controller design, an expert will inevitably incorporate his stability concept into the rule base and thus guarantee the stability of the system.

3.3.1 Stability Criterion

To further elaborate the concept of the expert's Lyapunov function, we formulated two criteria for the stability of a fuzzy control system. The stability of an FCS is predicted when the criteria are met. Before stating the stability theorem, we shall present some preliminary definitions (Vidyasagar (1973)).

Definition 3.3.1 Class K Function

A continuous function $\alpha: R \rightarrow R$ is said to belong to *class K* if

(i) $\alpha(\cdot)$ is nondecreasing, (3.3.1)

(ii) $\alpha(0)=0$, and (3.3.2)

(iii) $\alpha(p)>0$ *whenever* $p>0$. (3.3.3)

□

Definition 3.3.2 Locally Positive Definite Function (l.p.d.f.)

A continuous function $V: R_+ \times R^n \rightarrow R$ is a *locally positive definite function* (l.p.d.f.) if there exists a continuous nondecreasing function $\alpha: R \rightarrow R$ such that

(i) $\alpha(0)=0$, $\alpha(p)>0$ *whenever* $p>0$, (3.3.4)

(ii) $V(t,0)=0 \quad \forall\ t \geq 0$, (3.3.5)

(iii) $V(t,x) \geq \alpha(x) \quad \forall\ t \geq 0, \forall\ x \in B_r$, (3.3.6)

where B_r is a ball with radius r and center 0.

□

Definition 3.3.3 Decrescent Function

A continuous function $V: R_+ \times R^n \rightarrow R$ is said to be *decrescent* if there exists a class k function $\beta(\cdot)$ such that

$$V(t,x) \leq \beta(x) \tag{3.3.7}$$

$\forall\ t \geq 0, \forall\ x \in B_r \subset R^n$, for some $r>0$.

□

For a nonlinear autonomous plant, a stablizing fuzzy logic controller in a linguistic rule base form generally has the fuzzy state variables $\vec{x}$, as the only independent variables and can be

expressed as a fuzzy function, $\bar{g}$, of the fuzzy state variables, such as:

$$\bar{u} = \bar{g}(\bar{x}) \tag{3.3.8}$$

By including the fuzzification and the defuzzification operators, a stablizing fuzzy logic controller can be represented by a real-valued function g as:

$$u = g(x) \tag{3.3.9}$$

where x and u are real state and input variables.

The following stability theorems are extensions of the Lyapunov stability theorems in nonlinear system analysis. Given a nonlinear autonomous plant with a stablizing fuzzy logic controller, the complete system is Lyapunov stable if there exists a function, namely the Lyapunov function, which satisfies the conditions in the theorem. The Lyapunov function can actually be considered as a generalized energy function. The convergence of such an energy function to zero indicates the convergence of the state variables and hence the stability of the system.

Theorem 3.3.1 (Stability Theorem)

Consider a nonlinear autonomous plant P and a fuzzy controller C (including the fuzzification and the defuzzification operator) represented by:

$$\dot{x} = f(x,u) \tag{3.3.10}$$

$$u = g(x) \tag{3.3.11}$$

where x is the state variable, $x \in R^n$, u is the input, $u \in R^r$.

If: there exists a continuously differentiable, decrescent l.p.d.f. V such that

$$\frac{d}{dt} V(t,x) \le 0 \tag{3.3.12}$$

$\forall \quad t \ge t_0, x \in X$, where X is the universe of discourse,

then: the equilibrium **0** of the FCS is uniformly stable over $[t_0,\infty)$.

Proof: See Appendix A.

The above theorem claims that an FCS is Lyapunov stable if the criterion in (3.3.12) is satisfied. By giving more constraints on the Lyapunov function V, the asymptotic stability can also be guaranteed by the next theorem.

Theorem 3.3.2 (Asymptotic Stability Theorem)

Consider a nonlinear autonomous plant P and a fuzzy controller C as in (3.3.10)-(3.3.11),

If: there exists a continuously differentiable, decrescent l.p.d.f. V

such that $-\frac{d}{dt}V$ is an l.p.d.f.,

then: the equilibrium 0 of the FCS is uniformly asymptotically stable over $[t_0, \infty)$.

Proof: See Appendix A.

Theorem 3.3.1 and 3.3.2 show that we can prove the Lyapunov stability of a fuzzy control system by finding a suitable Lyapunov function. The statement is true not only for a fuzzy control system but also for a general nonlinear dynamical system. However, the function g in (3.3.10) representing a fuzzy logic controller, is generally a highly nonlinear function. Practically, it is very difficult to find a proper Lyapunov function to verify its stability for a completed fuzzy control system. It would, on the other hand, be much easier if a fuzzy logic controller is built based on a certain Lyapunov function. Therefore, the value of the stability theorems does not purely lie in mathematical proofs but also in the design method described in next section.

3.3.2 Design Method

Consider a nonlinear autonomous plant P described by:

$$\dot{x} = f(x, u) \tag{3.3.13}$$

As mentioned in the previous section, an expert always has his own notion of a stability criterion for the system. A control engineer or an expert himself can find a Lyapunov function based on

this notion to help the design of a stablizing fuzzy logic controller. The choice of the Lyapunov function certainly depends on the experts, however the l–2 norm of the state variables seems to be an intuitively good choice.

Suppose that V is the Lyapunov function devised by an expert based on his knowledge of the controlled system. In order to satisfy the conditions in Theorem 3.3.1, we need to solve the following inequality equation for the admissible class of inputs,

$$\dot{V}(x) = \frac{dV}{dx}\dot{x} \tag{3.3.14}$$

$$= \frac{dV}{dx} f(x, u) \leq 0. \tag{3.3.15}$$

The constrained input surface U_c can be defined by

$$U_c = \{\, u \mid \forall\, x, \frac{dV}{dx} f(x, u) \leq 0 \,\}. \tag{3.3.16}$$

The constrained input surface U_c, is a set of inputs which satisfies the stability criterion of Theorem 3.3.1. The fuzzy input surface, denoted by U_f, is the set of inputs generated by the fuzzy logic controller. In the process of constructing a rule base, an expert can use the constrained input set U_c to compare with the fuzzy input set U_f. If U_f is completely contained in U_c, then the FCS is stable according to Theorem 3.3.1 or Theorem 3.3.2. Also, if there is some mismatch of the two sets, a controller designer can modify the fuzzy logic controller based on the difference of the two sets.

3.4 Example

Consider the inverted pendulum shown in Figure 3.4.1. It consists of a motor-driven cart on which a pole--an inverted pendulum--is mounted with a frictionless ball-bearing pivot. The pole can fall freely about the pivot axis. θ is the angle between the pole and the vertical line; m is mass of the pole; $2l$ is the length of the pole; m_c is the mass of the cart; f is the force applied to the cart; x is the position of the cart. The angular position θ and angular velocity $\dot{\theta}$ of the pole

can be measured by a position sensor and a velocity sensor.

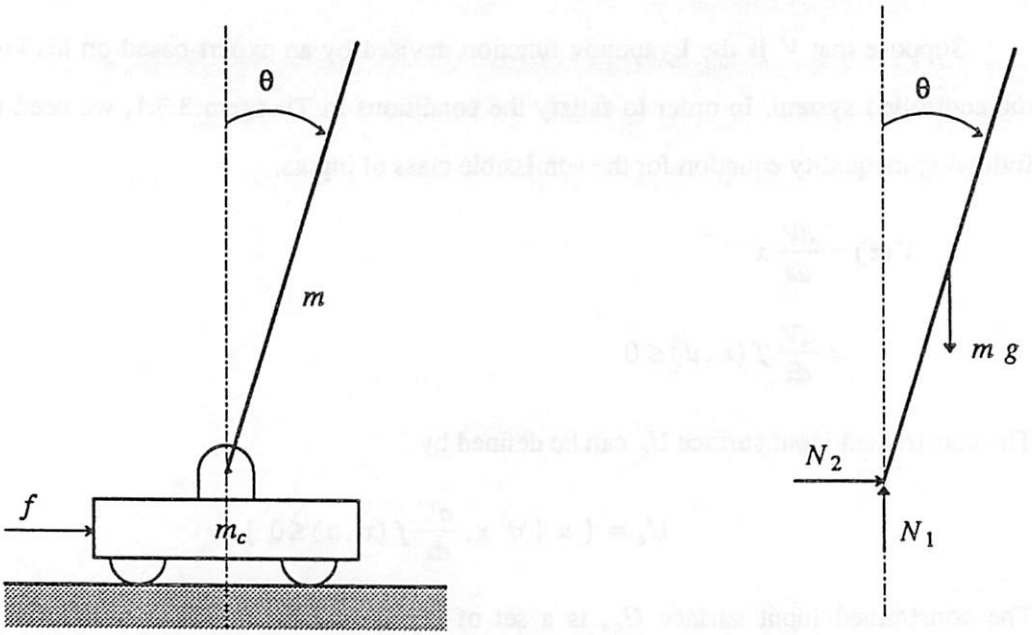

Figure 3.4.1: An Inverted Pendulum

By considering the angular position θ and angular velocity $\dot{\theta}$, a fuzzy logic controller was implemented in a computer program to balance the pole. The rule base of the fuzzy controller consists of 9 linguistic rules, such as:

if θ is PS_1, $\dot{\theta}$ is PM_2, **then** f is PS_3.

where PS_1 is a fuzzy set named *positive small* for θ; PM_2 is a fuzzy set named *positive medium* for $\dot{\theta}$; PS_3 is a fuzzy set named *positive small* for f.

The state trajectories of the fuzzy logic controller with initial conditions $\theta_0 = 0.5\ rad$ and $\dot{\theta}_0 = 0.8\ rad/sec$ are shown in Figure 3.4.2-3.4.4.

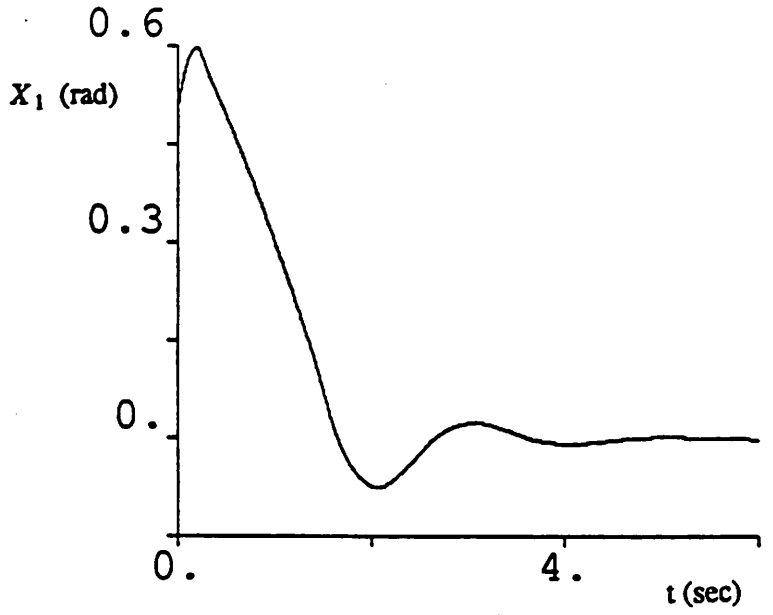

Figure 3.4.2: Pendulum position θ (radians)

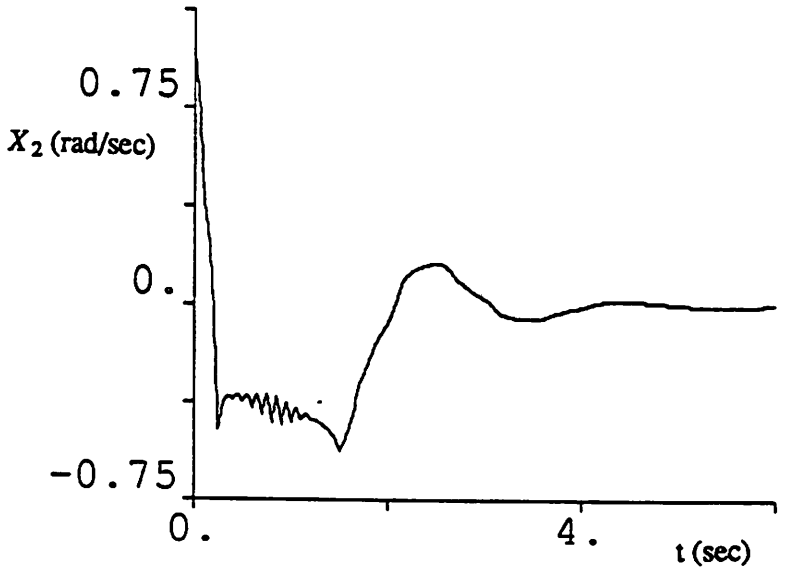

Figure 3.4.3: Pendulum velocity $\dot{\theta}$ (rad/sec)

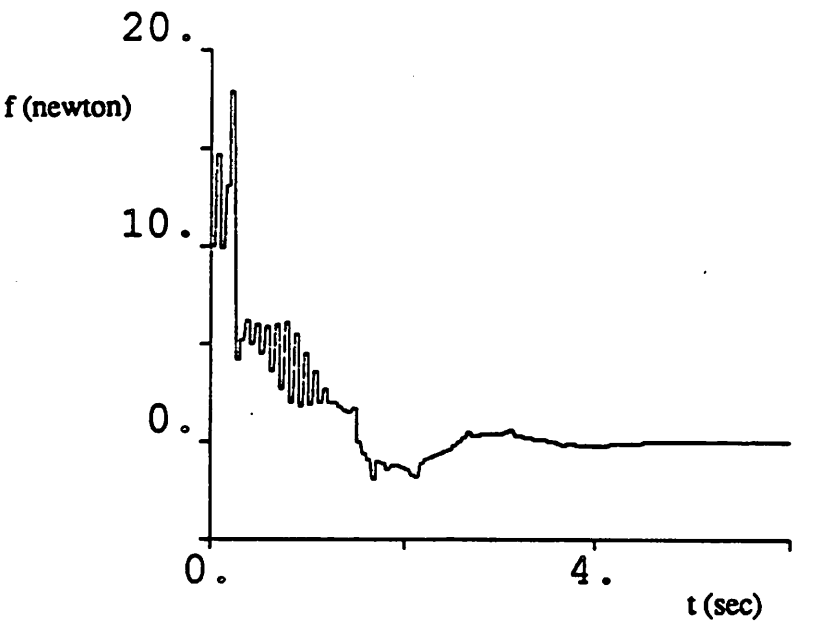

Figure 3.4.4: Fuzzy logic controller output

Assuming the ideal case, i.e. no friction at the joint and no slippage of the cart wheel, etc., the kinematic equations of the pole can be derived as:

$$m_c\ddot{x} + m\ddot{x} + ml\ddot{\theta}\cos\theta - ml\dot{\theta}^2\sin\theta = f \tag{3.4.1}$$

$$\frac{1}{3}ml^2\ddot{\theta} + ml(\ddot{x}\cos\theta + l\ddot{\theta}) - mgl\sin\theta = 0. \tag{3.4.2}$$

By eliminating x from (3.4.1) and (3.4.2), we have

$$\frac{4}{3}l\ddot{\theta} + \frac{1}{m_c+m}(-m\,l\ddot{\theta}\cos\theta + m\,l\dot{\theta}^2\sin\theta + f)\cos\theta - g\sin\theta = 0 \tag{3.4.3}$$

which can be re-written as:

$$(\frac{4l}{3} - m\frac{l}{m_c+m}\cos^2\theta)\ddot{\theta} + m\frac{l}{m_c+m}\sin\theta\cos\theta\,\dot{\theta}^2 - g\sin\theta + \frac{1}{m_c+m}f\cos\theta = 0 \tag{3.4.4}$$

By choosing $x_1 = \theta$ and $x_2 = \dot{\theta}$ as the state variables with the limitation $-\frac{\pi}{2} < x_1 < \frac{\pi}{2}$, the state equations of the system are

$$\dot{x_1} = x_2 \tag{3.4.5}$$

$$\dot{x_2} = \frac{g \sin x_1 - \frac{ml}{m_c + m} \sin x_1 \cos x_1 x_2^2 - \frac{1}{m_c + m} f \cos x_1}{\frac{4}{3} l - \frac{ml}{m_c + m} \cos^2 x_1} \tag{3.4.6}$$

Suppose an expert chooses the following Lyapunov function:

$$V = \frac{1}{2}(x_1^2 + x_2^2) \tag{3.4.7}$$

then the derivative of V with respect to t is

$$\dot{V} = x_1 \dot{x_1} + x_2 \dot{x_2} \tag{3.4.8}$$

$$= x_1 x_2 + x_2 \frac{g \sin x_1 - \frac{ml}{m_c + m} \sin x_1 \cos x_1 x_2^2 - \frac{1}{m_c + m} f \cos x_1}{\frac{4}{3} l - \frac{ml}{m_c + m} \cos^2 x_1} \tag{3.4.9}$$

$$= x_2 (x_1 + \frac{g \sin x_1 - \frac{ml}{m_c + m} \sin x_1 \cos x_1 x_2^2 - \frac{1}{m_c + m} f \cos x_1}{\frac{4}{3} l - \frac{ml}{m_c + m} \cos^2 x_1}) \tag{3.4.10}$$

Since V is a continuously differentiable decrescent function, by Theorem 3.3.1, the system can be stablized if

$$\dot{V} \leq 0. \tag{3.4.11}$$

This implies that

for $x_2 > 0$, we need to have

$$f > \frac{m_c + m}{\cos x_1} (\frac{4}{3} l x_1 - \frac{ml}{m_c + m} x_1 \cos^2 x_1 + g \sin x_1 - \frac{ml}{m_c + m} \sin x_1 \cos x_1 x_2^2) \tag{3.4.12}$$

or for $x_2 < 0$,

$$f < \frac{m_c + m}{\cos x_1}\left(\frac{4}{3}lx_1 - \frac{ml}{m_c + m}x_1\cos^2 x_1 + g\sin x_1 - \frac{ml}{m_c + m}\sin x_1\cos x_1 x_2^2\right). \quad (3.4.13)$$

The input surfaces generated by the fuzzy logic controller and the input surface constrained by (3.4.12) and (3.4.13) are given in Figure 3.4.5 and Figure 3.4.5 respectively.

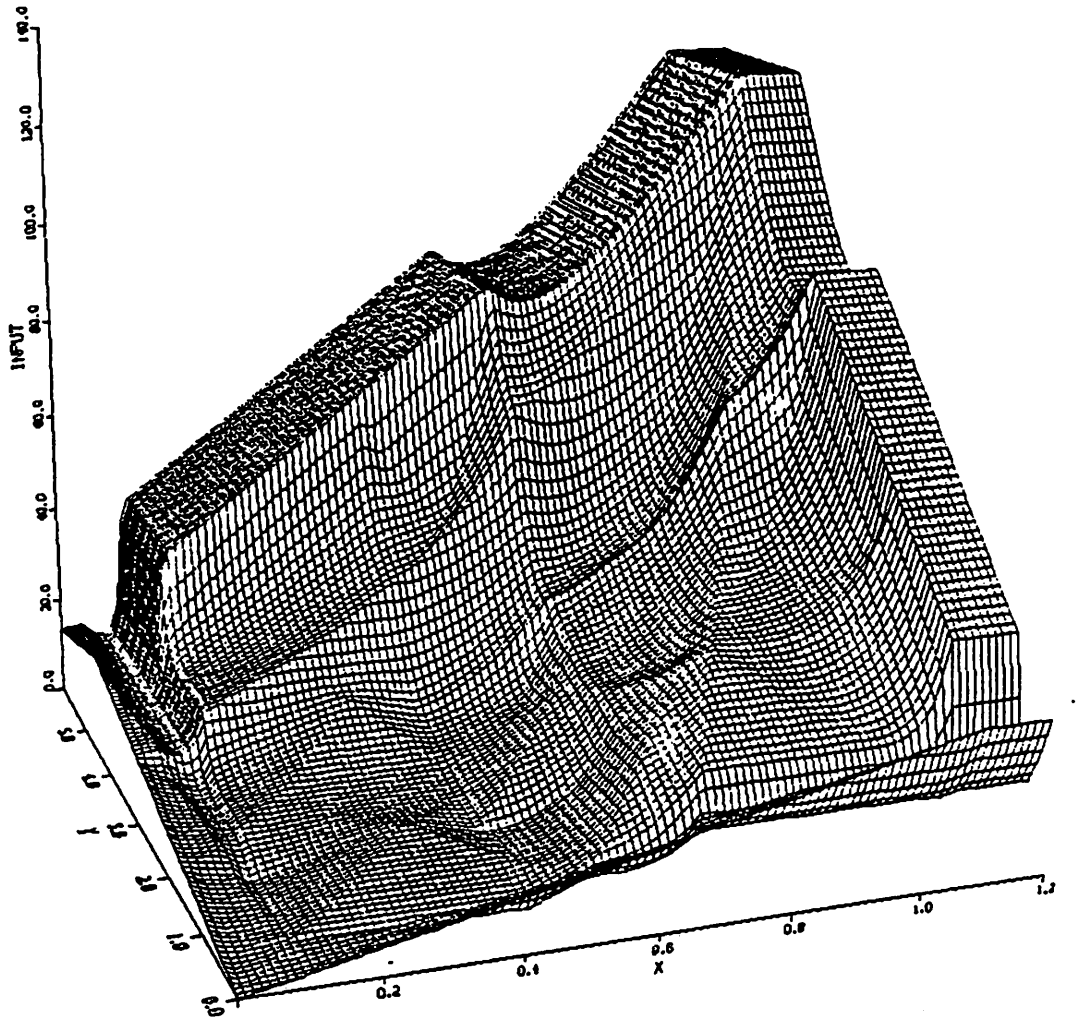

Figure 3.4.5: The controller input surface U_f

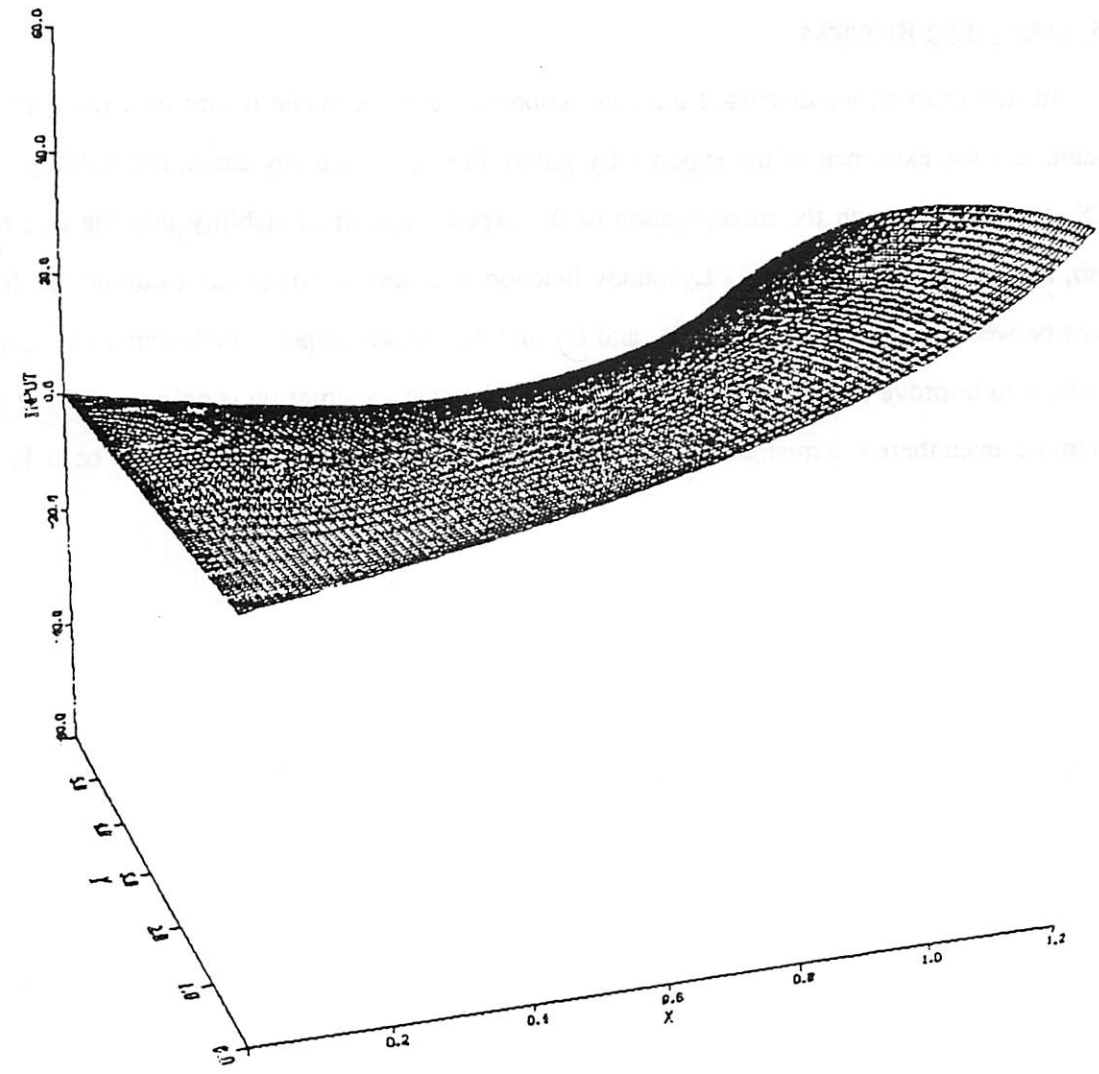

Figure 3.4.6: The constrained input surface U_c

By comparing the two input surfaces, we can easily verify the stability of the FCS by Theorem 3.3.1. At the same time, if there is any inconsistency between the two surfaces, the expert can modify the rule base accordingly.

3.5 Concluding Remarks

In this chapter, we discussed the connection between the stable nature of a fuzzy control system and the existence of the expert's Lyapunov function. In many cases, the stability of an FCS is assured through the incorporation of the expert's notion of stability into the rule base. Also, by heuristically choosing a Lyapunov function, a control engineer can examine the differences between the two input spaces U_c and U_f and then make proper adjustments to the current rule base to improve the stability of the system. Notice that the criterion is only a *sufficient* condition, i.e. even there is a mismatch of U_c and U_f, the fuzzy control system may still be stable.

Chapter 4

Analysis of Fuzzy Dynamical Systems Using Cell-to-Cell Mapping

4.1 Introduction

Control problems are usually formulated and analyzed in a precise mathematical language. The thriving of artificial intelligence, which utilizes human experience in a more relaxed form than the conventional mathematical approach, has attracted more attention to fuzzy logic control recently. Many industrial applications have been carried out since Zadeh's first introduction of fuzzy logic control (1968), for instance, automatic train control (Yasunobu and Miyamoto (1985)) and kiln control (Holmblad and Ostergaard (1982)). However, in spite of these, there are still very few tools for analyzing the global behavior of a fuzzy dynamical system.

The global behavior of a dynamical system means the evolution of the system states corresponding to various initial conditions. Linear algebra has been an indispensable tool for solving these problems for linear dynamical systems. Differential topology has been a useful tool for the case of some nonlinear systems although an unified approach is still not available. For a fuzzy dynamical system, there is not yet a very useful mathematical tool because of its fuzziness, complexity and non-parameterization.

Many researchers have attempted to find a systematic method for analyzing fuzzy dynamical systems. Tong (1980) used an approach of defining system operators as fuzzy relations on the state, output and control spaces. Cumani (1982) considered the system quantities such as states, input and output as fuzzy variables which obey time-evolving possibility distributions governed by Hisdal's Calculus of conditional possibility. Kiszka et al. (1985) formulated an energy function of a fuzzy set and proposed that a fuzzy system is stable if the energy function is monotonically decreasing. However, the results by these researchers failed to give significant descriptions of the dynamics of a fuzzy system.

We believe that the main reason for the previous unsuccessful attempts is the conservativeness of the min-max calculus. The min-max operation (Zadeh (1965)), widely accepted by most researchers in this field, is too conservative to extract the behavior of a fuzzy dynamical system because the operation has the effect of "flattening-out" the distribution of the membership functions. When a fuzzy dynamical system is described in a recursive form, the cumulation of this effect makes the observation of the evolution of the system dynamics almost impossible.

To remedy this problem, different operators which are less conservative were first investigated. The "min" operator is actually the most conservative member in a class of so called triangular or T-norms operators and the max operator is its corresponding T-conorm (Dubois and Prade (1980)). The T-norms satisfy several properties as:

$$T(0,0)=0 \tag{4.1.1}$$

$$T(a,1)=T(1,a)=a \tag{4.1.2}$$

$$T(a,b)\leq T(c,d), \quad \text{if } a\leq c \text{ and } b\leq d \tag{4.1.3}$$

$$T(a,b)=T(b,a) \tag{4.1.4}$$

$$T(a,T(b,c))=T(T(a,b),c) \tag{4.1.5}$$

and has drawn a lot of interest recently, especially in uncertainly reasoning (Bonissone and Decker (1986)). However, most of the T-norms, except for the "min", lack of an important associativity property of their matrix operation when a T-norm is considered to be an "and" operator and its T-conorm to be an "or" operator. This mathematical incompleteness unfortunately prevents the T-norms to be the solution of the aforementioned problem.

Some people may consider the problem unavoidable as the consequence of the fuzziness possessed by the system. However, one must bear in mind that although there is uncertainty in a fuzzy system, an "expert" should be able to tell the "trend" of the system dynamics. The rule base created by the expert may be vague in words but should be descriptive in global behavior. Hence, a methodology should be devised to grasp the evolving trend of the states of a fuzzy dynamical system. It should also be noted that "fuzziness" is different from "randomness". For a

fuzzy dynamical system, "fuzziness" lies in the formulation of a fuzzy controller. We try to imitate human thinkings with of a rule base. Just like an expert system, the expert's knowledge is formulated to give a better way of handling a complex process. Generally, the "uncertainty" or "fuzziness" is from the imprecise knowledge representation of an expert, not from the "randomness" of the process. In another word, once a fuzzy controller is given, the whole system can actually be considered as a deterministic system.

The proposed method applies the concept of cell-to-cell mapping to describe the global behavior of a fuzzy dynamical system. The method of cell-to-cell mapping was developed by Hsu (1980) for analyzing the global behavior of nonlinear dynamical systems. The state space is first partitioned into finite number of disjoined sets called Cell State Space. The differential equation describing the system dynamics is then integrated for a certain time interval and a mapping from the Cell State Space to itself can be determined. By working on this next state cell-to-cell mapping, which contains the information of states evolvement, the periodic motions of the mapping and their domain of attractions can be obtained. In analyzing a fuzzy dynamical system, since our goal is to obtain the evolving trend of the states, the concept of cell-to-cell mapping is particularly useful. We first convert a fuzzy dynamical system to a cell-to-cell mapping, and then extract the global behavior of the system dynamics by a method similar to Hsu's. By working on the cell-to-cell mapping, the problem of fuzziness cumulation is eliminated.

The remainder of this chapter is organized as follows. Section 2 gives a general description of the cell-to-cell Mapping method. Section 3 gives the model of fuzzy dynamical systems and shows the transformation from the original fuzzy mapping to the proposed cell-to-cell mapping and the extraction of the dynamic behavior from the cell-to-cell mapping. Section 4 performs the global analyses of both the real and fuzzy initial state responses. Some definitions of the cellular stability are also described in this section. Section 5 shows the validity of the method by applying it to a simple example. Section 6 gives some further discussions and the concluding section summarizes the results.

4.2 Cell-to-Cell Mapping Method

Consider a dynamical system with a governing equation

$$\dot{\mathbf{x}}(t) = f(t, \mathbf{x}(t)) \tag{4.2.1}$$

where $\mathbf{x}$ is a N-dimensional real vector and f is a real vector function. Assume that f is a periodic function of t with a period τ. We can then integrate (4.2.1) over the period of τ and derive a relation between the state at t and the next state at $t+\tau$. In another word, we can represent the same dynamical system with a different governing equation

$$\mathbf{x}(k+1) = G(\mathbf{x}(k)) \tag{4.2.2}$$

where G is a next-state point-to-point mapping. This kind of point-to-point mapping representation of a dynamical system is called a point map or a Poincare map in the mathematical literature. The method dates back to Poincare (1881) and Birkhoff (1920). There have been consistent mathematical developments since then, for instance, Arnol'd (1963), Smale (1967), Bernussou (1977). Hsu applied this method to analyze a certain class of strongly nonlinear dynamical systems and further developed a methodology, namely the cell-to-cell mapping, in the past decade. His method proved to be very effective in dealing with some peculiar phenomena, such as bifurcation, found in nonlinear dynamical systems.

The motivation to use the cell-to-cell mapping lies in the question of "how fine a scale is one allowed in specifying a state variable", Hsu (1980). There is undoubtly a physical limitation of the accuracy of any measurement. If the "true" values of two readings of a state variable differ by less than the measurement accuracy, there will be no way to distinguish the two values and they have to be considered equal. Moreover, in computation, one is also limited by the numerical roundoffs. Therefore, in the practical world, there is no real continuum of a state variable, but a large number of discrete values for each of the state variables. This point of view leads to the cell-to-cell mapping method.

4.2.1 Cell State Space

Consider again the dynamical system governed by (4.2.1). The coordinate axes of the state variables x_i, $i = 1, \ldots, N$ can be partitioned into a number of intervals with the size h_i, $i = 1, \ldots, N$. The interval z_i of the state variable x_i is defined to be the one containing all the x_i, such that

$$(z_i - \frac{1}{2}) h_i \leq x_i \leq (z_i + \frac{1}{2}) h_i \tag{4.2.3}$$

The N-tuple $(z_1, \ldots, z_N)$ is called a *cell* and denoted by **z**. Every point **x** in the state space belongs to a certain cell **z**. The collection of all the cells, namely the *Cell State Space* is normally denoted by Z. Some basic definitions of the cells are given next.

Definition 4.2.1

A cell $\mathbf{z}'$ is said to be a *contiguous* cell to **z** in the Z_i direction if

$$\mathbf{z}' - \mathbf{z} = +\mathbf{e}_i \text{ or } -\mathbf{e}_i \tag{4.2.4}$$

where e_i is the unit vector in the Z_i direction.

□

Definition 4.2.2

A cell $\mathbf{z}' \neq \mathbf{z}$ is said to be a *neighboring* cell of cell **z** if

$$| z'_i - z_i | \leq 1, \quad \text{for } i = 1, \ldots, n \tag{4.2.5}$$

Therefore, a contiguous cell to **z** must also be a neighboring cell of **z**.

□

4.2.2 Cell-to-Cell Mapping

For the dynamical system given in (4.2.1), we shall further assume that the system is an autonomous (time-invariant) system, i.e.

$$\dot{\mathbf{x}}(t) = f(t, \mathbf{x}(t)) \tag{4.2.6}$$

$$= f(\mathbf{x}) \tag{4.2.7}$$

Therefore, given an initial condition $\mathbf{x}(t_0)$ and a time period τ, the state at $t = t_0 + \tau$ can be calculated by

$$\mathbf{x}(t_0+\tau) = \mathbf{x}(t_0) + \int_{t_0}^{t_0+\tau} \dot{\mathbf{x}}(t)\, dt\,. \tag{4.2.8}$$

Since the system is autonomous, we can assign $t_0 = 0$ to be the initial time. We can also simplify the notations of $\mathbf{x}(t_0)$ to $\mathbf{x}_0$, $\mathbf{x}(t_0+\tau)$ to $\mathbf{x}_\tau$ and reformulate (4.2.8) by

$$\mathbf{x}_\tau = \mathbf{x}_0 + \int_0^\tau \dot{\mathbf{x}}\, dt\,. \tag{4.2.9}$$

Now assume that the state space has been partitioned into a number of cells to form a cell state space Z. For every cell $\mathbf{z} \in Z$, we represent $\mathbf{z}$ by its center point $x_{\mathbf{z}}^c$. Given an initial condition $\mathbf{x}_0 = x_{\mathbf{z}}^c$ for a cell $\mathbf{z}$ and a time period τ, the next state x_τ is

$$x_\tau = x_{\mathbf{z}}^c + \int_0^\tau \dot{\mathbf{x}}\, dt\,. \tag{4.2.10}$$

Then we can define a mapping from the cell $\mathbf{z}$ to the cell $\mathbf{z}'$, if

$$x_\tau \in \mathbf{z}' \tag{4.2.11}$$

The procedure above can be done for every cell in the cell state space. The result will be a next-state cell-to-cell mapping denoted by F, i.e.

$$\mathbf{z}(k+1) = F(\mathbf{z}(k))\,. \tag{4.2.12}$$

This cell-to-cell mapping is constructed to approximate the dynamical behavior of the original system. The smaller the cells are, the better is the approximation.

Before giving the method for extracting the dynamical behaviors from the cell-to-cell mapping, we first give some basic definitions relating to its dynamic properties.

Definition 4.2.3

A cell $\mathbf{z}^*$ which satisfies

$$\mathbf{z}^* = F(\mathbf{z}^*) \tag{4.2.13}$$

is said to be an *equilibrium* cell of the cell-to-cell mapping F.

□

Definition 4.2.4

A sequence of k distinct cells $\mathbf{z}^*(1), \ldots, \mathbf{z}^*(k)$ which satisfy

$$\mathbf{z}^*(m+1) = F^m(\mathbf{z}^*(1)), \quad m = 1, \ldots, k-1 \tag{4.2.14}$$

$$\mathbf{z}^*(1) = F^k(\mathbf{z}^*(1)), \tag{4.2.15}$$

is said to constitute a periodic motion of period k of the cell-to-cell mapping F. The motion is called a *$p-k$ motion* and the cells are called *$p-k$ cells*.

□

Remark: Obviously, an equilibrium cell is a p-1 cell.

Definition 4.2.5

A cell z is said to be r-step removed from a p-k motion if r is the minimum integer such that

$$F^r(\mathbf{z}) = \mathbf{z}^*(j) \tag{4.2.16}$$

where $\mathbf{z}^*(j)$ is one of the p-k cells in a p-k motion.

□

Remark: It is clear that the cell z will be mapped into a p-k cell in r steps and further mappings will remain in the p-k motion.

Definition 4.2.6

For a p-k motion, the set of cells which are r-step or less steps removed from the p-k motion is called the *$r-$step domain of attraction* for the p-k motion. The total domain of attraction or simply the domain of attraction of a p-k motion is its r-step domain of attraction with r going to

infinity.

□

Definition 4.2.7

For a cell-to-cell mapping F, assume the domain of interest in the X_j subspace to be limited in $[\, lb_j, ub_j \,], j = 1,...,n$. Those cells which belong to the domain of interest are called *regular* cells. The region which is outside of the domain of interest is considered as one cell and is called a *sink* cell.

□

After the partitioning of the state space, the cell state space consists of a number of regular cells and a sink cell. Label the regular cells by positive integers $1, 2, \ldots, N_c$ and the sink cell by $N_c + 1$. Since we don't care about the further development of the dynamic system once it enters the sink cell, we have one additional rule for the mapping:

$$F(N_c + 1) = N_c + 1 \tag{4.2.17}$$

That means the sink cell is a p-1 cell and the system will stay in the sink cell once it is mapped into it.

With the number of cells being finite, the evolution of the cell mapping starting with a regular cell z can only lead to three possible outcomes:

(1) Cell z is one of the members of a p-k motion, i.e. cell z will be mapped back to itself in k steps.

(2) Cell z is mapped into a p-k motion in r steps, i.e. cell z belongs to the r-step domain of attraction of a periodic motion.

(3) Cell z is mapped into the sink cell in r steps, i.e. cell z belongs to the r-step domain of attraction of the sink cell.

4.2.3 Grouping Algorithm

We shall process the cell state space to find the desired periodic motions and domain of attractions by a grouping algorithm (Hsu and Guttalu (1985)). The algorithm uses exhaustive search. It will process all the cells to determine the global behavior of a mapping. Every cell z will be assigned three characteristic numbers. A group number $G(\mathbf{z})$ is assigned to every cell in a periodic motion and every cell in its domain of attraction. The step number $S(\mathbf{z})$ of a cell z indicates the number of steps it takes to map z into a periodic cell. If z happens to be a periodic cell then $S(\mathbf{z})=0$. The period of a periodic motion is denoted by a periodicity number. The same periodicity number $P(\mathbf{z})$ is assigned to all the cells in a group.

To distinguish the cells in different stages, we divide them into *virgin cells*, *cells under processing* and *processed cells*. The group number of all the virgin cells is set to zero at the beginning of the algorithm. When a virgin cell is called up to be processed, we change the group number to -1, which means it is a cell under processing. After the process, a cell is assigned a group number and is called a processed cell.

To begin with, since the sink cell is a p-1 cell, it is considered to be the first periodic motion and we set $G(N_c+1)=1$, $S(N_c+1)=0$ and $P(N_c+1)=1$. Then we process the regular cells $\mathbf{z}=1,\ldots,N_c$ sequentially. Let us consider a virgin cell z and the processing sequence as:

$$\mathbf{z}\rightarrow F(\mathbf{z})\rightarrow F^2(\mathbf{z})\rightarrow\cdots\rightarrow F^i(\mathbf{z}) \tag{4.2.18}$$

At each step, there are three possible conditions:

(1) The newly generated image cell $F^i(\mathbf{z})$ is a virgin cell. In this case, we change the group number of the image cell from 0 to -1, which indicates it has become a cell under processing, and proceed to the next image cell $F^{i+1}(\mathbf{z})$.

(2) The newly generated image cell $F^i(\mathbf{z})$ is a processed cell with a positive group number. In this case, the current processing sequence is mapped into a cell with known global properties, i.e. the group number, step number and periodicity number. It's obvious that the sequence of the cells should have the same group number and periodicity number as those

of $F^i(\mathbf{z})$. The step number of the cells in the sequence can be decided by:

$$S(F^j(\mathbf{z})) = S(F^i(\mathbf{z})) + i - j\,, \quad j = 0, 1, \ldots, i-1 \tag{4.2.19}$$

After this, we pick up another virgin cell and start a new processing sequence.

(3) The newly generated image cell $F^i(\mathbf{z})$ is a cell under processing. In this case, a new periodic motion is discovered. All the cells in the sequence are assigned a new group number. Let us assume the reappearing cell to be the $(j+1)$-th cell in the sequence (4.2.18), i.e. $F^i(\mathbf{z}) = F^j(\mathbf{z})$, $j < i$. Then the periodicity number is $(i-j)$ for this periodic motion and is assigned to every cell in the sequence. The step number can also be determined by:

$$S(F^k(\mathbf{z})) = j - k, \quad k = 0, 1, \ldots, j-1, \tag{4.2.20}$$

$$S(F^k(\mathbf{z})) = 0, \quad k = j, j+1, \ldots, i-1. \tag{4.2.21}$$

When everything is done, we again go back to pick up another virgin cell and begin a new process. The process will be performed repetitively until every cell in the cell state space has been processed and the global behavior of the mapping is completely determined.

In summary, the grouping algorithm divides the cell state space into different groups consisting of periodic motions and their domain of attractions. Every cell in the cell state space belongs to a certain group. The step number of each cell also indicates the distance of each cell from its attracting periodic motion.

4.3 Fuzzy Mapping to Cell-to-Cell Mapping

In this section, we shall transform a fuzzy dynamical system into the form of a cell-to-cell mapping so that it could be used to analyze the global behavior of the fuzzy dynamical system. The fuzzy system is originally represented by a fuzzy mapping. The fuzzy mapping is extended to a real mapping by including the fuzzification and the defuzzification blocks in the input and output, respectively. Then the real mapping can be processed to derive the desired cell-to-cell mapping.

4.3.1 Fuzzy Mapping to Real Mapping

Consider a simple time-invariant process represented by the block diagram shown in Figure 3.2.1, where u_k is the input variable at time k and x_k, x_{k+1} are the state variables at time k and $k+1$ respectively. Assume that we can model the process by a fuzzy dynamical equation:

$$\vec{x}_{t_{k+1}} = \vec{f}(\vec{x}_{t_k}, \vec{u}_{t_k}) \tag{4.3.1}$$

where $\vec{f}$ is a fuzzy mapping defined on $X \times U \rightarrow X$, X is a multidimensional state space and U is a multidimensional input space. The state variables $\vec{x}_{t_k}$ and $\vec{x}_{t_{k+1}}$ are fuzzy sets defined on X; the input $\vec{u}_{t_k}$ is a fuzzy set defined on U and $t_k, k = 0,1,\cdots$ are a sequence of time indices which may not be uniformly distributed.

For a closed loop fuzzy control system, the fuzzy logic controller is generally a function of the output or the measurable state variables, i.e. we have

$$\vec{u}_{t_k} = \vec{g}(\vec{x}_{t_k}) \tag{4.3.2}$$

where $\vec{g}$ is also a fuzzy mapping which maps from X to U.

From (4.3.1) and (4.3.2), we get:

$$\begin{aligned} \vec{x}_{t_{k+1}} &= \vec{f}(\vec{x}_{t_k}, \vec{g}(\vec{x}_{t_k})) \\ &= \vec{f}_1(\vec{x}_{t_k}) \end{aligned} \tag{4.3.3}$$

where $\vec{x}_{t_k}, \vec{x}_{t_{k+1}} \in X$ and $\vec{f}_1$ is a fuzzy mapping $X \rightarrow X$, i.e. a fuzzy relation defined on $X \times X$.

Generally, the fuzzy relation $\vec{f}_1$ is of the following rule-based form:

$$\begin{aligned} &\text{if } x_1^{t_k} = \vec{A}_{11}, \ldots, x_n^{t_k} = \vec{A}_{1n}, \text{ then } x_1^{t_{k+1}} = \vec{B}_{11}, \ldots, x_n^{t_{k+1}} = \vec{B}_{1n}, \\ &\text{if } x_1^{t_k} = \vec{A}_{21}, \ldots, x_n^{t_k} = \vec{A}_{2n}, \text{ then } x_1^{t_{k+1}} = \vec{B}_{21}, \ldots, x_n^{t_{k+1}} = \vec{B}_{2n}, \\ &\qquad \cdots \\ &\qquad \cdots \\ &\text{if } x_1^{t_k} = \vec{A}_{m1}, \ldots, x_n^{t_k} = \vec{A}_{mn}, \text{ then } x_1^{t_{k+1}} = \vec{B}_{m1}, \ldots, x_n^{t_{k+1}} = \vec{B}_{mn}. \end{aligned} \tag{4.3.4}$$

where the rule base consists of m *if–then* rules and $\tilde{A}_{ij}$, $\tilde{B}_{ij}$, $i = 1, \ldots, m$, $j = 1, \ldots, n$ are fuzzy variables. In this chapter, we shall mostly use (4.3.3) to represent a fuzzy dynamical system. The detailed description of the inference operation of the fuzzy logic controller is given in Chapter 2 and Zimmermann (1985) and is not repeated here.

The fuzzy mapping $\tilde{f}_1$ defined as a fuzzy relation between two spaces can be extended to a real mapping between the same two spaces by including the fuzzifying and defuzzifying operators. The center block of Figure 4.3.1 is a fuzzy mapping in the form of a rule base and the whole diagram including the fuzzifying and defuzzifying blocks is a real mapping. The fuzzifying block takes in the real inputs and matches them to different fuzzy variables to find corresponding membership values. These will be used as inputs to the fuzzy mapping. The defuzzifying block takes the outputs of the fuzzy mapping, which are in the form of fuzzy numbers, and converts them to real numbers by weighted average or some other method. Practically, we always use the real mapping in our operations, instead of the fuzzy mapping alone, since the measurements and the control inputs must be real numbers. In this paper, we shall differentiate these two mappings by putting a "~" on the fuzzy one(e.g. f_1 is the real mapping of the fuzzy mapping $\tilde{f}_1$).

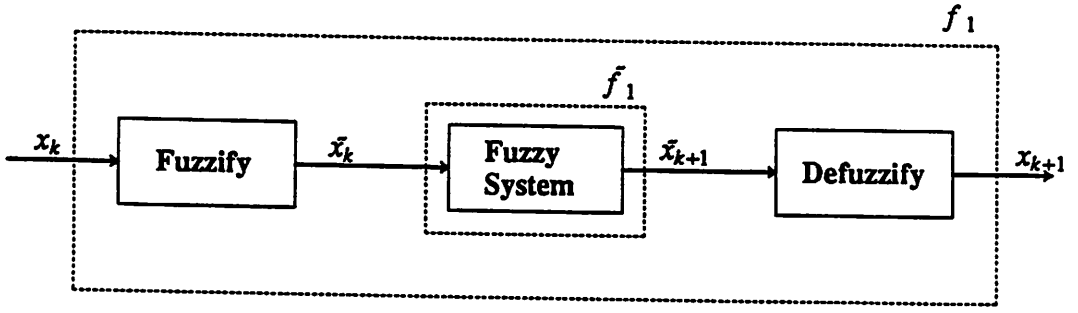

Figure 4.3.1: Relation between a fuzzy mapping and the real mapping

4.3.2 Real Mapping to Cell-to-Cell Mapping

The main idea of the cell-to-cell mapping approach is to transform the infinite numbered point to point real mapping problem into a finite numbered cell to cell mapping problem. The system dynamics can then be easily described once the cell-to-cell mapping is derived.

First of all, our domain of interest is generally bounded in the real case. It is meaningless to consider something which is outside our domain of interest. For example, owing to the current limitation of a robot-arm driving motor, we cannot control the robot arm with a very large current. Without loss of generality, we shall limit our domain of interest for the X_j space in a finite closed interval $[\,lb_j\,,\,ub_j\,],\ j=1,\ldots,n$.

Consider the n-dimensional state space X and the real mapping f_1, obtained from the fuzzy mapping $\tilde{f}_1$. Let us first divide the i-th subspace X_i into numbers of equal intervals with an interval size h_i. An interval will be denoted by an integer z_i if for all the x_i belonging to the interval, we have

$$(z_i - \frac{1}{2})\,h_i \le x_i \le (z_i + \frac{1}{2})\,h_i \tag{4.3.5}$$

The n-tuple $(\,z_1,\ldots,z_n\,)$ is then called a cell and is denoted by z. A point $x \in X$ belongs to z if and only if (4.3.5) holds for all i. Now the state space X can be considered as a collection of cells and be represented by the so called *Cell State Space* Z. The cells in Z are actually lattice points of a Euclidean space with integer-valued Cartesian coordinates. Let $\mathbf{e}_i$ be the unit vector in the direction of the Z_i-axis. A cell z can be represented as

$$\mathbf{z} = \sum_{i=1}^{n} z_i \mathbf{e}_i \tag{4.3.6}$$

After partitioning the state space X into a cell state space Z, we wish to transform the real mapping f_1 into a cell-to-cell mapping F. The procedure for deriving F is described as follows:

(1) For every cell $\mathbf{z} = (z_1, z_2, \ldots, z_n)$, by definition, if $x \in \mathbf{z}$, then (4.3.5) holds for all i, i.e. z can be considered as an n-dimensional cube in X. We shall choose the center point z^c of a cell z to represent the cell, where

$$z_i^c = z_i \; h_i, \qquad i = 1, \ldots, n \tag{4.3.7}$$

(2) We now define the cell-to-cell mapping $F: Z \rightarrow Z$. For all z and its center point z^c at time t_k

$$\text{if } x_{t_{k+1}} = x_{t_k + \Delta t_k} = f_1(x_{t_k}{}^c) \in z^*$$

$$\textit{then } F(z_{t_k}) = z^* \tag{4.3.8}$$

where Δt_k is chosen to be a constant time interval, such as t_m.

To summarize, the cell-to-cell mapping F is constructed by representing the cells in the cell state space Z by their center points and then finding the cells where the images of the center points are located after a certain time period t_m. The mapping f_1 is not limited to bc in the form of differential equations as in Hsu's work. It can be a rule base or a combination of rules and differential equations.

The cell-to-cell mapping F is created by considering the mapping f_1 as a deterministic nonlinear system and does not deal with uncertainty at all. Although there may be more than one image cell of an initial cell z if we consider more points in z, we concentrate only on the one evolved from the center point of z in our work. The multi-image case is very interesting and is also under study by the author but will not be discussed here.

4.3.3 Extracting Dynamical Behaviors from Cell-to-Cell Mapping

After the transformation of a fuzzy dynamical system to its corresponding cell-to-cell mapping is completed, the grouping algorithm (described in section 4.2.3) can be applied to process the mapping to find the different $p-k$ motions and their domain of attractions in the cell-to-cell mapping. The results of the grouping algorithm will give a very clear and complete picture of the behavior of the fuzzy dynamical system. The detailed analysis and a simulation example will be given in the next two sections.

4.4 Fuzzy Dynamical System Analysis

We have incorporated the method of cell-to-cell mapping into a method of fuzzy dynamical system analysis in previous sections. Given a fuzzy dynamical system, we transform it into a real mapping by including the fuzzification and defuzzification operators. Then a cell-to-cell mapping is obtained from this real mapping. The cell-to-cell mapping then undergoes a process to find its periodic motions and the corresponding domain of attractions. After this process, every cell in the cell state space belongs to a certain group which consists of a periodic motion and its domain of attraction. In this section, we shall use the result to obtain the system response for either a real initial state or a fuzzy initial state.

4.4.1 Real Initial State Response

Now consider a real initial state (within our region of interest)

$$x = x_0 \tag{4.4.1}$$

where x is the state variable and $x_0 \in X$ is the initial value of x. We only need to identify the cell z such that $x_0 \in$ z. Then we can easily understand the behavior of the system from the group number, step number, and periodicity number associated with it. Furthermore, a global analysis can be done by computing the domain of attractions of periodic motions with different step numbers. A detailed example will be presented in the next section.

4.4.2 Fuzzy Initial State Response

It is natural to ask at this point about the case when the initial state is a fuzzy variable like "small" or "medium". The answer is not as straightforward as in the real-valued domain case.

First, any sequence of evolving cells must converge to a periodic motion (including the sink cell) because there are only finite number of cells. The main purpose of global analysis is to find those periodic motions and their corresponding domain of attractions. In the real domain case,

we assigned every cell a group number to describe the behavior of the system. In the fuzzy domain, as can be expected, the system behavior will be described by fuzzy group numbers.

For simplicity, We consider only a two dimensional state space $X = X_1 \times X_2$. It can be easily extended to a higher dimension. Suppose that we have a fuzzy initial state:

$$\tilde{x}_1 = \tilde{A}_1 \tag{4.4.2}$$

$$\tilde{x}_2 = \tilde{A}_2 \tag{4.4.3}$$

where $\tilde{x}_1$ and $\tilde{x}_2$ are fuzzy variables and $\tilde{A}_1$ and $\tilde{A}_2$ are fuzzy sets in the subspaces X_1 and X_2. We first define the α-level sets of $\tilde{x}_1$ and $\tilde{x}_2$ by:

$$A_1^{\alpha} = \{ x_1 \in X_1 \mid \mu_{\tilde{x}_1}(x_1) \geq \alpha \} \tag{4.4.4}$$

$$A_2^{\alpha} = \{ x_2 \in X_2 \mid \mu_{\tilde{x}_2}(x_2) \geq \alpha \} \tag{4.4.5}$$

where $A_1^{\alpha} \subset X_1$ and $A_2^{\alpha} \subset X_2$ are crisp sets, not fuzzy sets.

Then we denote the α-level region enclosed by the α-level sets A_1^{α} and A_2^{α} as:

$$R^{\alpha}(x_1, x_2) = \{ (x_1, x_2) \mid x_1 \in A_1^{\alpha}, x_2 \in A_2^{\alpha} \} \tag{4.4.6}$$

Every point in R^{α} has a membership value greater than α in both X_1 and X_2 directions. However, the smallest unit in a cell state space is a cell. We shall not be able to process R^{α} in Z without further modification. One reasonable approach is to take an approximation of R^{α} with a set of cells. Thus, we define the minimum cells set of R^{α} by:

$$R_m^{\alpha}(x_1, x_2) = \min \{ z \in Z \mid \forall\ x \in R^{\alpha}(x_1, x_2), \exists\ z, \textit{such that } x \in z \} \tag{4.4.7}$$

In other words, R_m^{α} is the smallest set of cells which contains R^{α}.

Now consider a decreasing positive α sequence

$$\alpha_1, \alpha_2, \ldots, \alpha_k \quad (\textit{generally } \alpha_1 = 1). \tag{4.4.8}$$

We can easily derive R^{α_i} and $R_m^{\alpha_i}$, $i = 1, \ldots, k$ from (4.4.6)-(4.4.7). The problem will be as simple as in the real domain if all the cells in $R_m^{\alpha_i}$ belong to only one group. Unfortunately, that is not necessarily the case. Let's assume there are $1, 2, \ldots, g$ numbered periodic motions from the

Table 4.4.1: The cell numbers of groups in different minimum cells sets

α/group number	1	2	. . .	g
α_1	$N_{R_m^{\alpha_1}}(1)$	$N_{R_m^{\alpha_1}}(2)$	. . .	$N_{R_m^{\alpha_1}}(g)$
α_2	$N_{R_m^{\alpha_2}}(1)$	$N_{R_m^{\alpha_2}}(2)$	. . .	$N_{R_m^{\alpha_2}}(g)$
.	.		. . .	
.	.		. . .	
.	.		. . .	
α_k	$N_{R_m^{\alpha_k}}(1)$	$N_{R_m^{\alpha_k}}(2)$	. . .	$N_{R_m^{\alpha_k}}(g)$

Table 4.4.2: The cell proportions of groups in different minimum cells sets

α/group number	1	2	. . .	g
α_1	P_{11}	P_{12}	. . .	P_{1g}
α_2	P_{21}	P_{22}	. . .	P_{2g}
.	.		. . .	
.	.		. . .	
.	.		. . .	
α_k	P_{k1}	P_{k2}	. . .	P_{kg}

result of the grouping algorithm. Every cell in our domain of interest belongs to one of the domain of attractions of periodic motions. Therefore, every cell in $R_m^{\alpha_i}$ also belongs to one of the groups. We shall further define:

$$N_{R_m^{\alpha_i}}(i) = \textit{the number of cells in } R_m^{\alpha_i} \textit{ which belong to the } i\textit{-th group}, i = 1, \ldots, g \tag{4.4.9}$$

Table 4.4.1 shows the number of cells belonging to different groups in the minimum cells sets $R_m^{\alpha_i}$, $i = 1, 2, \ldots, k$. The sums of the elements in each row are the total numbers of cells in $R_m^{\alpha_i}$ for $i = 1, \ldots, g$. Table 4.4.2 shows the proportions of the cells of different groups in each minimum cells sets. The sum of the elements in each row is equal to 1. The proportions are defined as:

$$P_{ij} = \frac{N_{R_m^{\alpha_i}}(j)}{\sum_{k=1,\ldots,g} N_{R_m^{\alpha_i}}(k)}, \quad i=1,\ldots,k, \; j=1,\ldots,j \tag{4.4.10}$$

Finally, we shall define the fuzzy group number $\vec{G}$ of the fuzzy initial states $\vec{x}_1$ and $\vec{x}_2$ as follows:

$$\vec{G}(\vec{x}_1, \vec{x}_2) = 1/m_1 + 2/m_2 + \cdots + g/m_g \tag{4.4.11}$$

$$m_j = \max_{i=1,\ldots,k} \min(\alpha_i, P_{ij}) \tag{4.4.12}$$

where α_i and P_{ij} are defined in (4.4.8) and (4.4.10). With fuzzy initial states $\vec{x}_1$ and $\vec{x}_2$, the fuzzy group number $\vec{G}$ denotes the possibility distribution of the system's final destination among different groups. An example will also be shown in section 4.5.

4.4.3 Cellular Stability

In this section, we shall give a stability definition based on the result of the above analysis. Some related definitions are introduced first.

Definition 4.4.1

The *norm* of a cell z with component z_i, $i = 1, \ldots, n$, is defined as

$$\| \mathbf{z} \| = \max_i | z_i | \tag{4.4.13}$$

□

Definition 4.4.2

Assume that S and S' are two sets of cells. The *distance* between S and S', denoted by $D(S, S')$ is given by

$$D(S, S') = \min_{\mathbf{z} \in S, \mathbf{z}' \in S'} \| \mathbf{z} - \mathbf{z}' \| \tag{4.4.14}$$

□

Definition 4.4.2 will also apply to the distance between two cells and the same notation is used when there is no possible confusion.

Definition 4.4.3

Given a set of cells S, the ε-neighborhood of S is defined by

$$N(S, \varepsilon) = \{ \mathbf{z} \mid D(\mathbf{z}, S) \leq \varepsilon \} \tag{4.4.15}$$

□

In the following discussions, we shall assume that vector **0** is the equilibrium point of the fuzzy dynamical system described in (4.3.3). The assumption is justified because one can always transform the original equilibrium point, say $\mathbf{x}_e$, to zero by redefining the state variable as

$$\mathbf{x}' = \mathbf{x} - \mathbf{x}_e \tag{4.4.16}$$

Definition 4.4.4

Given the equilibrium point **0**, a "target set" T_ε is defined to be an ε-neighborhood of **0** for some $\varepsilon \geq 0$, i.e.

$$T_\varepsilon = N(\mathbf{0}, \varepsilon) \tag{4.4.17}$$

□

Definition 4.4.5 Cellular Stable

Given an ε-target set T_ε and a domain set S_D, the fuzzy dynamical system (4.3.3) is *cellular stable* if and only if

(1) there exists a number of $p-k$ motions, such that all their $p-k$ cells belong to the target set T_ε, and

(2) the r-step domain of attraction of all the $p-k$ motions contains the domain set S_D, for some finite r's.

□

From the above theorem, a fuzzy dynamical system is considered to be cellular stable if all the cells in the domain of interest will converge to the target set in finite time. The target set T_ε actually represents an extended equilibrium region. In conventional theories, an equilibrium point is regarded as the target set in the analysis. We loosen the constraints by enlarging the target set from a point to a collection of cells to accommodate the resolution limitation of the cells and the fuzzy nature of a fuzzy dynamical system. To further explain the concept of cellular stability, a stability theorem is given in the following.

Definition 4.4.6

Given a cell-to-cell mapping F, a set of cells S_I is called a positively invariant set of F if

$$F(S_I) \subset S_I, \tag{4.4.18}$$

and is called an invariant set of F if

$$F(S_I)=S_I. \tag{4.4.19}$$

□

Theorem 4.4.1

Consider a fuzzy dynamical system in (4.3.3) and its corresponding cell-to-cell mapping F,

If: S_I is a positively invariant or invariant set of F,

then: there exists at least one $p-k$ motion in S_I and each cell in S_I either belongs to a $p-k$ motion or its domain of attraction.

Proof: Since S_I is a positively invariant or invariant set of F, we have

$$F(S_I) \subset S_I, \quad \text{or } F(S_I)=S_I. \tag{4.4.20}$$

Consider a sequence of cells starting from a cell $\mathbf{z} \in S_I$,

$$\mathbf{z} \to F(\mathbf{z}) \to F^2(\mathbf{z}) \to \cdots \tag{4.4.21}$$

According to (4.4.20), all the cells in the sequence belong to S_I. Since the number of cells in S_I is finite, the sequence will eventually map back to a previous cell in the sequence to form a $p-k$ motion. If the repeating cell, $F^k(\mathbf{z})$ is $\mathbf{z}$, then $\mathbf{z}$ is one of the $p-k$ cells. If not, $\mathbf{z}$ belongs to the domain of attraction of the $p-k$ motion.

□

Theorem 4.4.2 (Cellular Stability Theorem)

Consider a fuzzy dynamical system in (4.3.3) and its corresponding cell-to-cell mapping F with a domain set S_D,

If: there exists a positively invariant or invariant l-neighborhood of $\mathbf{0}$, $N(\mathbf{0}, l)$, $l \geq 0$, such that

$$D(\mathbf{0}, F(\mathbf{z})) < D(\mathbf{0}, \mathbf{z}), \qquad \forall\ \mathbf{z} \in S_D - N(\mathbf{0}, l), \tag{4.4.22}$$

then: there exists a target set $T_\varepsilon \subseteq N(0,l)$, such that the fuzzy dynamical system is cellular stable with the target set T_ε and the domain set S_D.

Proof: From definition 4.4.1 and 4.4.2, the distance function D can only have integer values, such as 0, 1, 2,..... Consider a sequence of cells starting from a cell $\mathbf{z} \in S_D - N(0,l)$,

$$\mathbf{z} \rightarrow F(\mathbf{z}) \rightarrow F^2(\mathbf{z}) \rightarrow \cdots \tag{4.4.23}$$

The distances between **0** and the cells in the (4.4.23)

$$D(0,\mathbf{z}) \rightarrow D(0,F(\mathbf{z})) \rightarrow D(0,F^2(\mathbf{z})) \rightarrow \cdots \tag{4.4.24}$$

can only be non-negative decreasing integers according to (4.4.22). Therefore, there must exist an i such that

$$D(0,F^i(\mathbf{z})) \leq l, \tag{4.4.25}$$

i.e. the sequence (4.4.23) will map into a cell in $N(0,l)$ in finite steps i.

Now, since $N(0,l)$ is a positively invariant or invariant set, by Theorem 4.4.1, there exists at least one $p-k$ motion and each cell in $N(0,l)$ belongs to a certain $p-k$ motion or its domain of attraction. Therefore, all the cells in the domain set S_D either belong to a certain $p-k$ motion in $N(0,l)$ or its domain of attraction. Now, if we choose

$$T_\varepsilon = \textit{an epsilon-neighborhood of } \mathbf{0} \textit{ which contains all the } p-k \textit{ motions in } N(0,l) \tag{4.4.26}$$

then the fuzzy dynamical system is cellular stable according to definition 4.4.5.

□

Theorem 4.4.2 gives a *sufficient* condition for the cellular stability of a fuzzy dynamical system. The dynamical system is cellular stable if the conditions are met. However, it is also possible that a fuzzy system is cellular stable without satisfying (4.4.22).

4.5 Example

In this section, a simple fuzzy dynamical system is analyzed to illustrate the validity of the method. The system is composed of an inverted pendulum and a fuzzy controller. The physical structure of the inverted pendulum is the same as shown in Figure 3.4.1. A free falling pole is mounted on a cart which is controlled by an actuator. The actuator will apply a force f to control the motion of the cart. The control objective is to use the actuator force f to control the motion of the cart such that the pole can be balanced in the vertical position.

The state equations of the inverted pendulum system are shown in (4.5.1) and (4.5.2) with x_1 the angle between the pole and the vertical line and x_2 the angular velocity of the pole.

$$\dot{x}_1 = x_2 \tag{4.5.1}$$

$$\dot{x}_2 = \frac{g \sin x_1 - \cos x_1 [\frac{ml}{m_c + m} x_2^2 \sin x_1 - \frac{1}{m_c + m} f]}{\frac{4}{3} l - \frac{ml}{m_c + m} \cos^2 x_1} \tag{4.5.2}$$

where g is the gravity constant, m is the mass of the pendulum, m_c is the mass of the cart, l is the length of the pendulum and f is the input force to the cart.

A fuzzy controller is constructed by considering the angular position and angular velocity of the pendulum as conditional variables and the force f as reaction variable. The rule base consists of 9 linguistic control rules.

To apply the method of cell-to-cell mapping, a cell state space is to be determined. The region of interest of the state space is taken to be from -1.5 to 1.5 (rad) for the angular position x_1 and from -10.0 to 10.0 (rad/sec) for the angular velocity x_2. The specified region is then partitioned into 101×101 blocks with equal divisions $h_1 = 0.0297$ and $h_2 = 0.1980$. Thus the cell state space contains 10201 regular cells and one sink cell which covers the region outside the region of interest.

Then we need to find the cell-to-cell mapping from dynamical system which comprises of the continuous time plant and the discrete time fuzzy controller, as shown in Figure 4.5.1.

The sampling time t_s of the controller is set to be $10\,ms$. The cell-to-cell mapping is constructed by finding out the image cells of every cell in the cell state space with $t_m = 10\,t_s$, where t_m is the time taken by the initial cell to reach its image cell. The image cell is obtained by integrating (4.5.1) and (4.5.2) with the input u determined by the fuzzy controller. Before we proceed further, we would like to compare of the original real mapping and its corresponding cell-to-cell mapping. A graph of the trajectories of the real mapping and the cell-to-cell mapping is given in Figure 4.5.2. Both trajectories start with the initial condition $x_1 = 0.6\,rad$, $x_2 = 3.0\,rad/sec$. It can be seen that the two trajectories are very close to each other. The trajectory of the real mapping reaches to the vicinity of the origin while the cell-to-cell mapping trajectory converges to a p-8 motion around the origin.

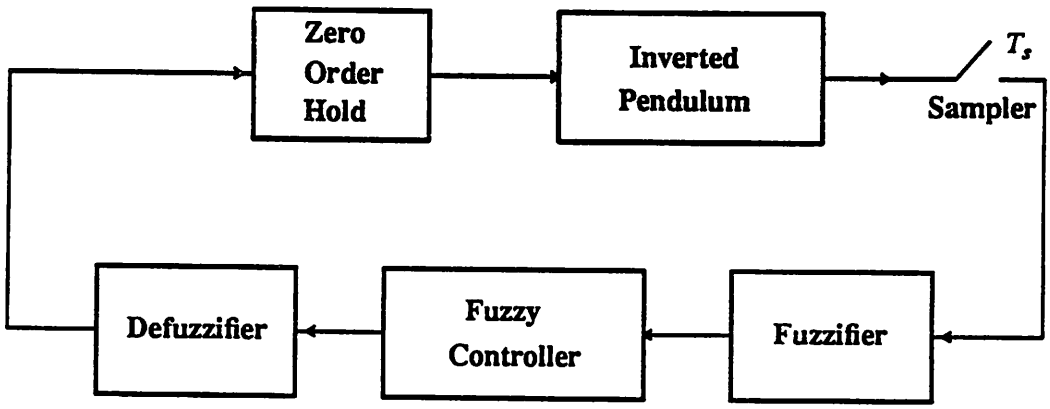

Figure 4.5.1: Block diagram of the inverted pendulum system

By applying the grouping algorithm to the cell-to-cell mapping, the periodic motions and their domain of attractions can be obtained. The result in Figure 4.5.3 shows one p-8 motion and three p-1 motions. The p-8 motion consists of 8 cells around the origin (group 1). There is one p-1 motion at the origin (group 2) and two p-1 motions at (1.4703, 0) (group 3) and (–1.4703, 0) (group 4) respectively. One other p-1 motion which is not shown in Figure 4.5.3 is the sink cell (group 5). The separation of group 1 and group 2 is mainly due to the rough partitions of the cell state space. It does not imply the existence of an inherent limit cycle of the system. However, a

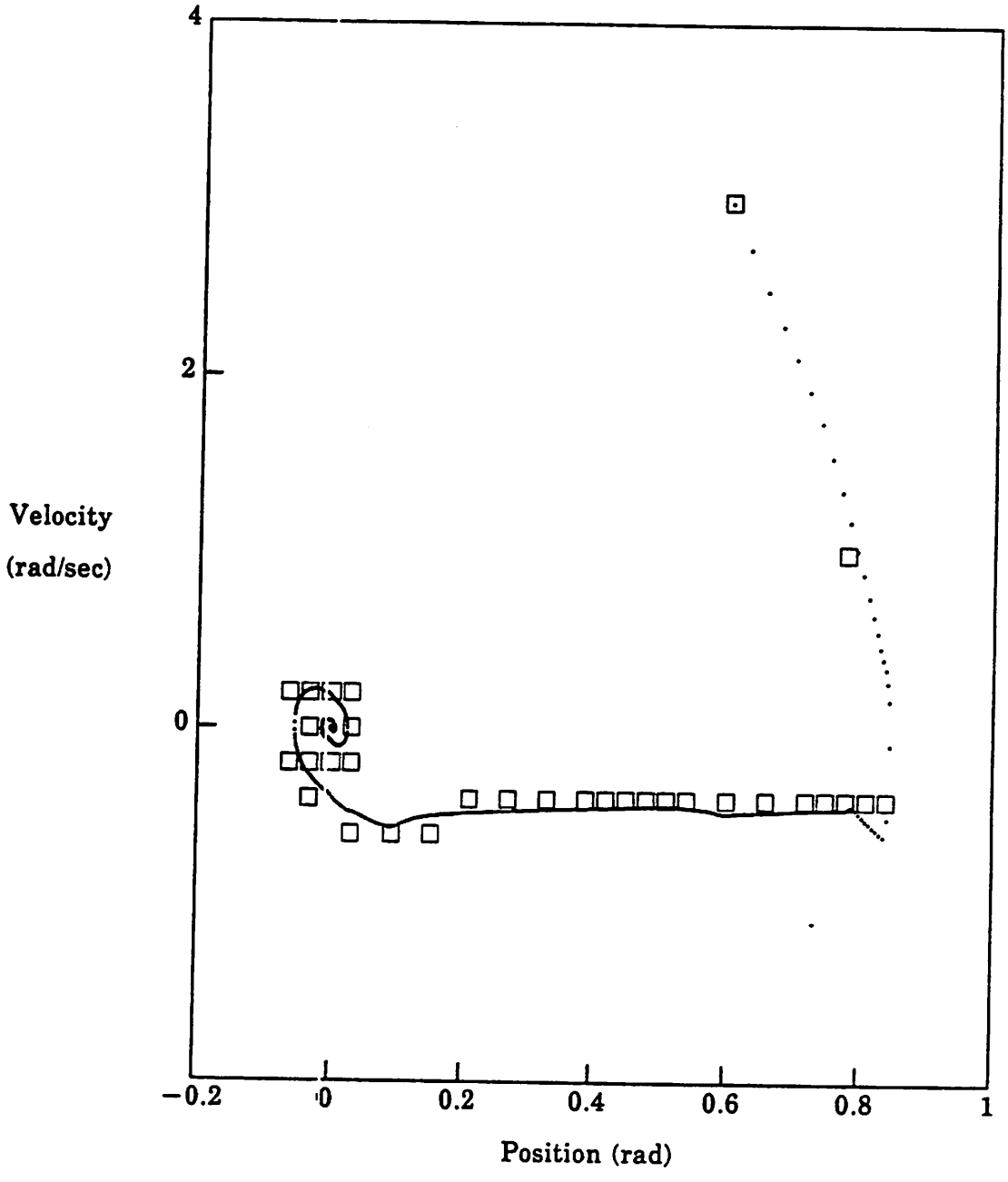

Figure 4.5.2: Trajectories for the inverted pendulum and its cell-to-cell mapping with initial states $x_1 = 0.6, x_2 = 3.0$. (Squares are for the cell-to-cell mapping and the dots are for the real trajectory.)

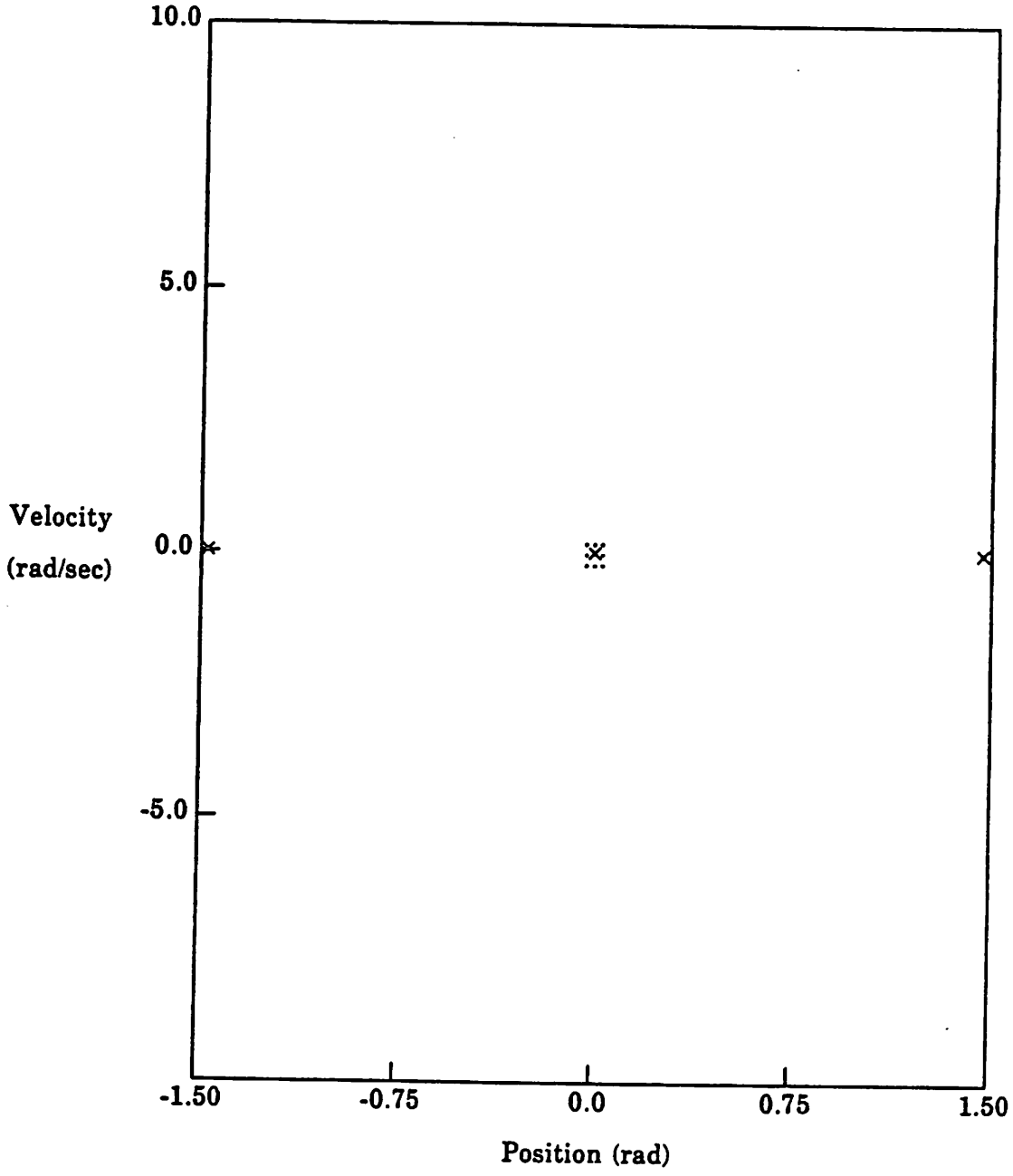

Figure 4.5.3: The four periodic motions of the cell-to-cell mapping. The eight dots around the origin are in group 1, the x at the origin is group 2, the x at (1.47, 0.0) is group 3, and the x at (-1.47, 0.0) is group 4. Group 5 is the sink cell and is not shown in the graph.

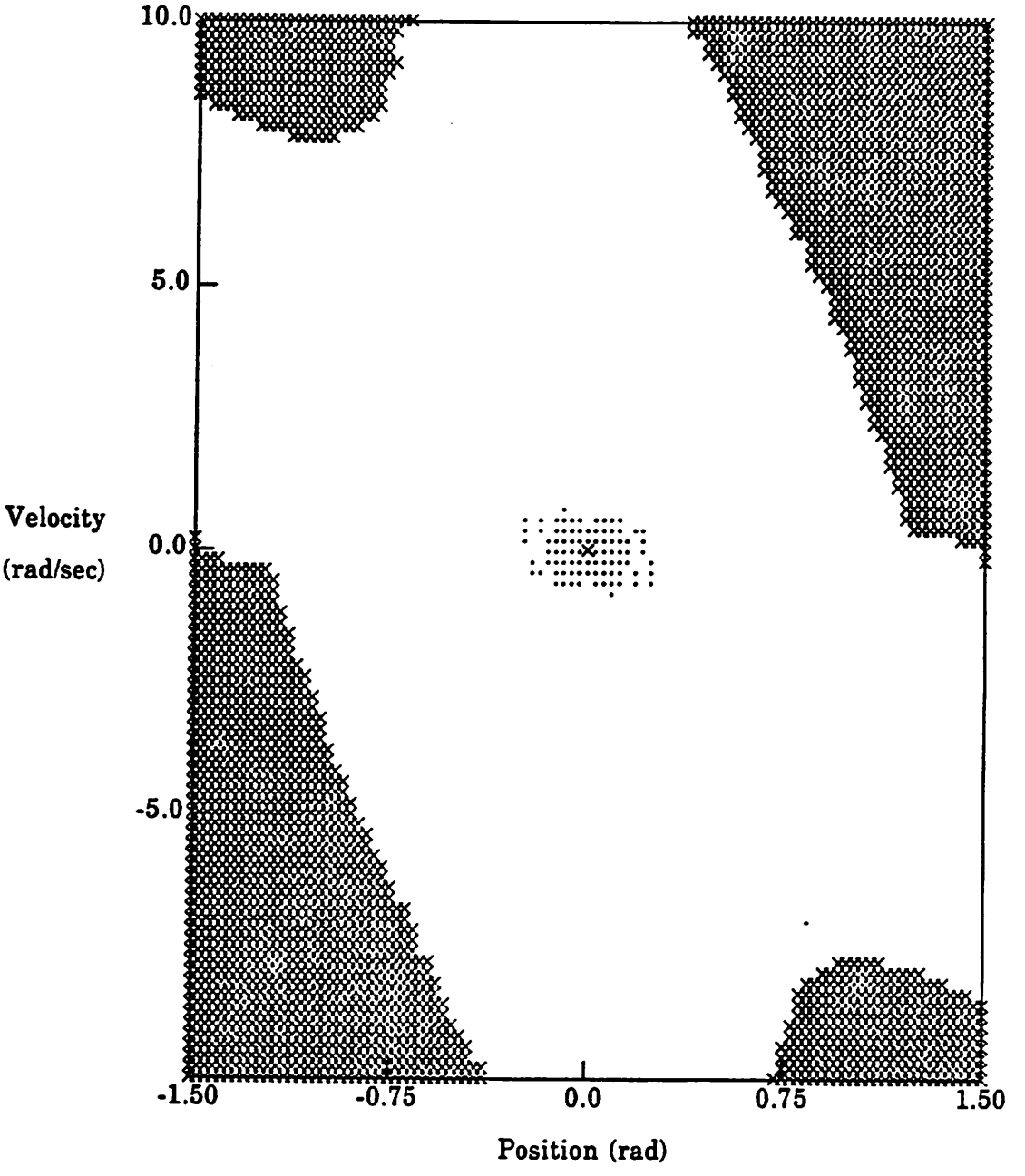

Figure 4.5.4: The 5-step domain of attraction of the periodic motions. The dark area in the corners is the domain of attraction for group 5(the sink cell).

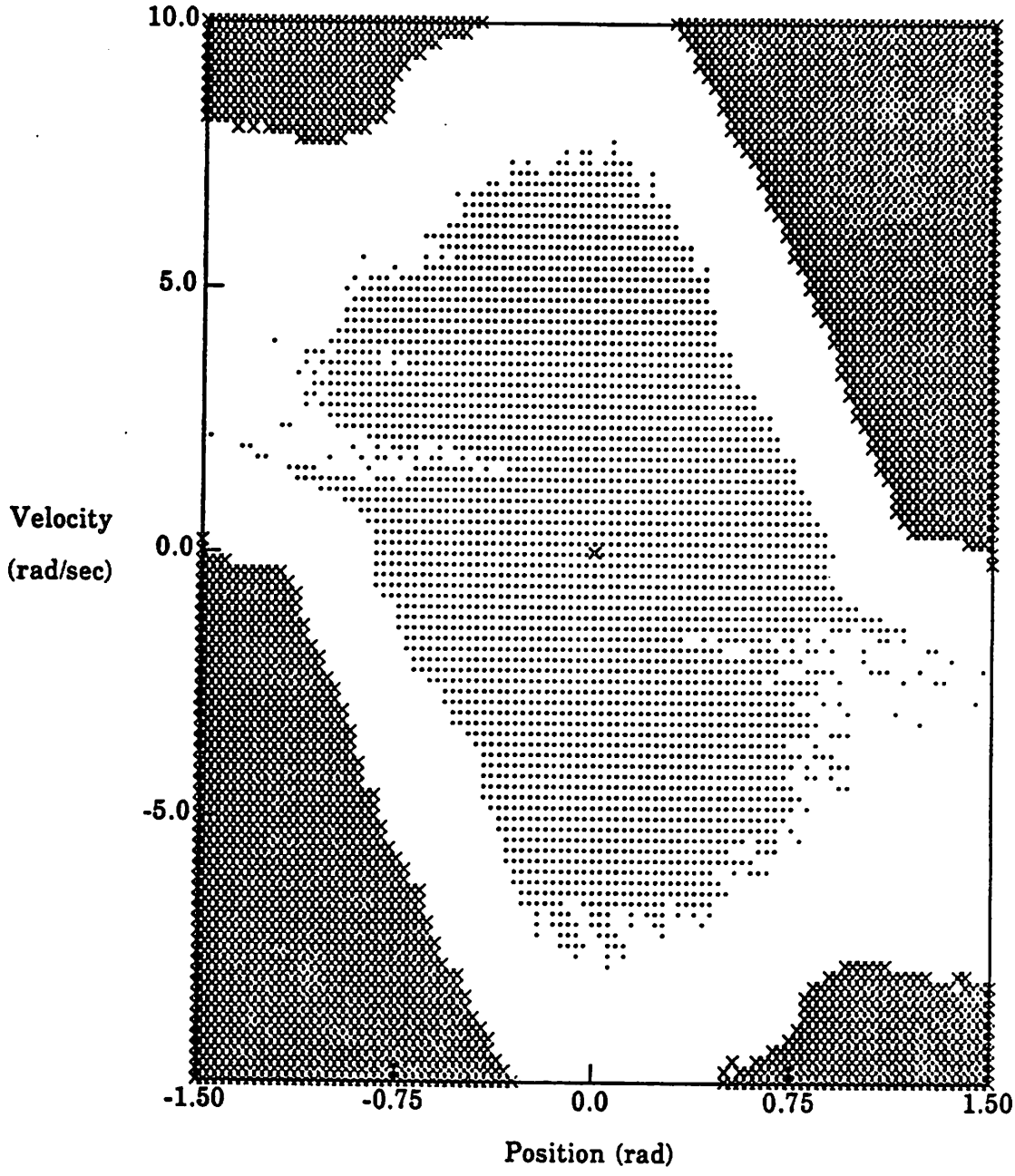

Figure 4.5.5: The 20-step domain of attraction of the periodic motions. The dark area in the corners is the domain of attraction for group 5(the sink cell).

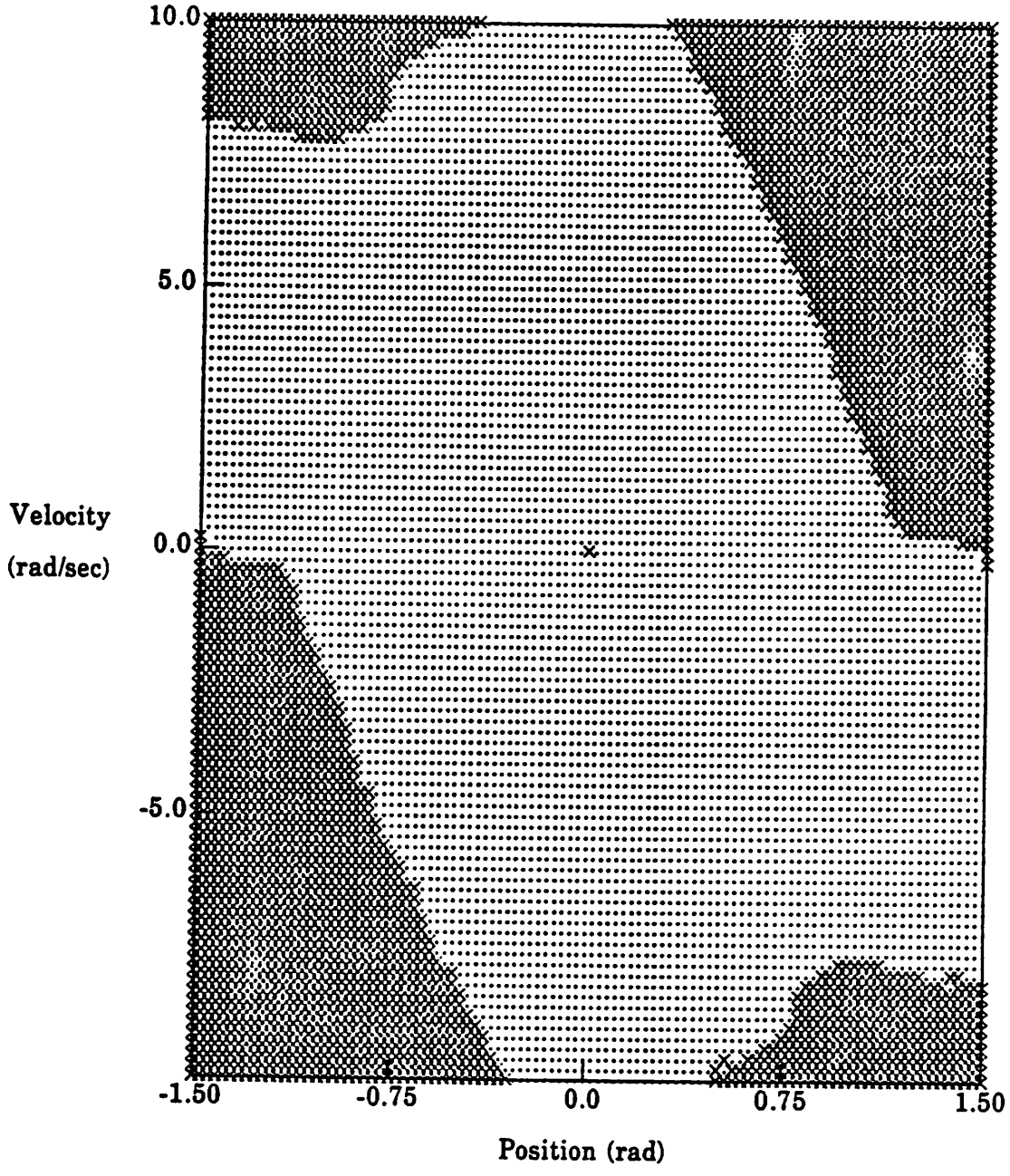

Figure 4.5.6: The domain of attraction of the periodic motions. The dark area in the corners is the domain of attraction for group 5(the sink cell).

finer partitioning for the cell state space could be used, if this were a concern. Group 3 and group 4 are undesirable equilibrium points since our control goal is to drive the trajectory to the origin in the state space. In our simulation, they can be easily removed by modifying one fuzzy control rule. This implies that the method of cell-to-cell mapping can also be used as a computer aided tool for modifying or designing fuzzy controllers. Figure 4.5.4 - 4.5.6 respectively show the 5,20-step and the total domain of attractions of those periodic motions. The convergence rate of an initial cell to its periodic motion can be estimated from the plot of the finite-step domain of attractions.

4.5.1 Real Initial State Response

With the results from the grouping algorithm, we can now discuss the real initial state response of the fuzzy dynamical system. Suppose that we have an initial state $x^0 = (x_1^0, x_2^0) = (0.6, 3.0)$. The first step is to find out which cell $\mathbf{z}^0$ contains x^0. The cell coordinate can be easily derived as $\mathbf{z}^0 = (z_1^0, z_2^0) = (20, 15)$. From the result of the grouping algorithm, we have

$$G(\mathbf{z}^0) = 1, \quad S(\mathbf{z}^0) = 24, \quad P(\mathbf{z}^0) = 8. \tag{4.5.3}$$

The group number 1 is assigned to the p-8 motion around the origin in our simulation. Therefore, the interpretation of the above information is that $\mathbf{z}^0$ will converge to the p-8 motion in 24 steps. Since the time interval for each step is $t_m = 10t_s = 100\,ms$, $\mathbf{z}^0$ will converge in $2.4\,seconds$. Though this prediction in the cell state space can only be considered as a rough estimate in the real state space, it does give us some idea about the system dynamics.

4.5.2 Fuzzy Initial State Response

Now let us assume fuzzy initial states to be $\bar{x}_1 = 0.\bar{7}5$ and $\bar{x}_2 = 5.\bar{0}$, where $0.\bar{7}5$ and $5.\bar{0}$ are fuzzy numbers with membership functions shown in Figure 4.5.7.

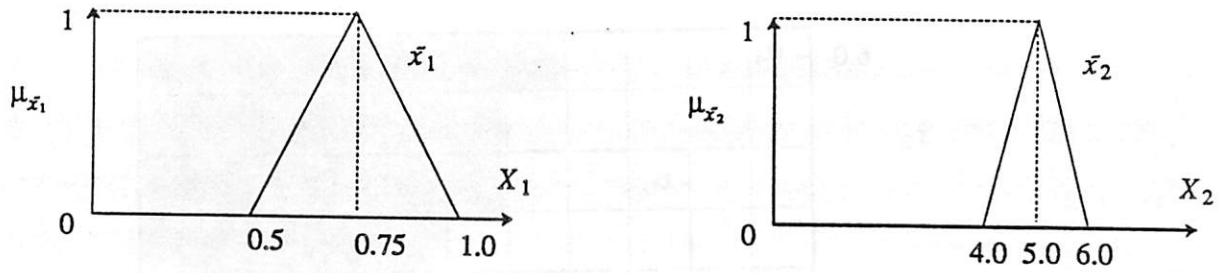

Figure 4.5.7: The membership functions of $\tilde{x}_1$ and $\tilde{x}_2$

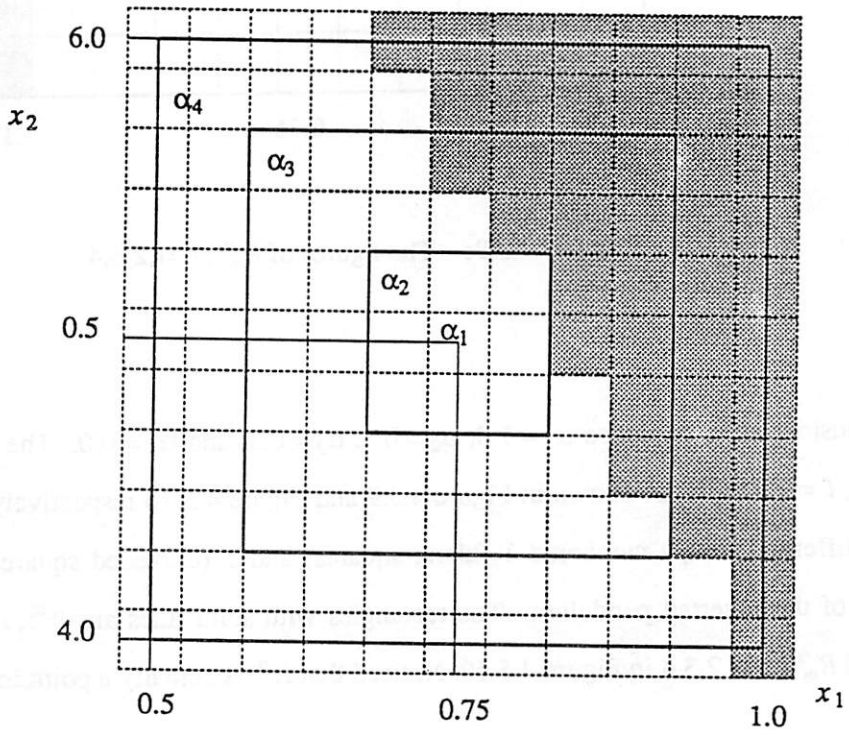

Figure 4.5.8: The figures of R^{α_i}, i=1,2,3,4

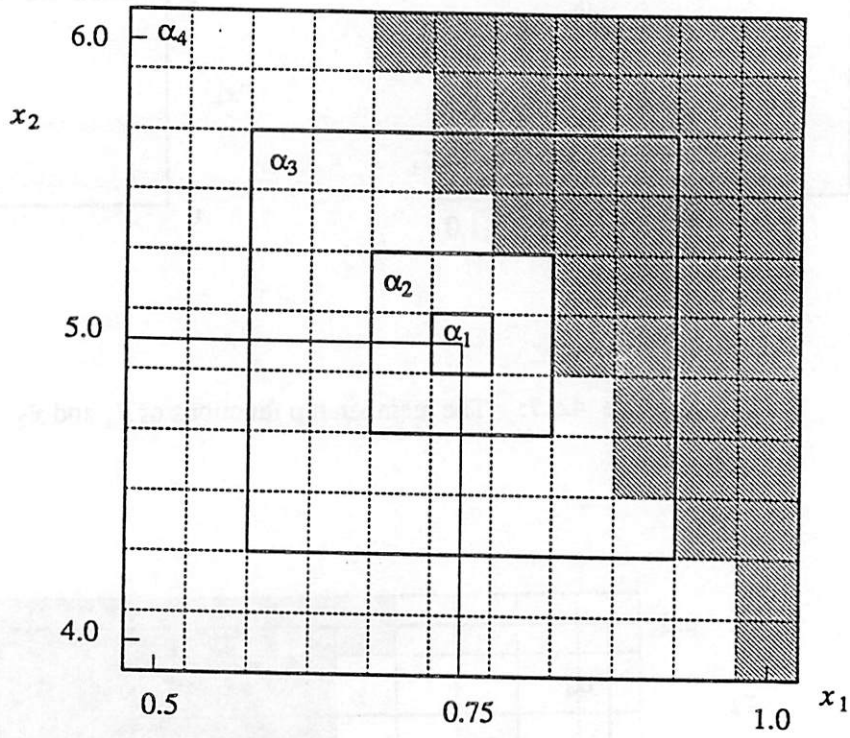

Figure 4.5.9: The figures of $R_m^{\alpha_i}$, i=1,2,3,4

Consider an α sequence $\alpha_1 = 1.0$, $\alpha_2 = 0.7$, $\alpha_3 = 0.3$, and $\alpha_4 = 0.0$. The corresponding R^{α_i} and $R_m^{\alpha_i}$, $i = 1, \ldots, 4$ are shown in Figure 4.5.9 and Figure 4.5.10 respectively. The cells belong to two different groups numbered 1 (blank squares) and 5 (darkened squares) from the global analysis of the inverted pendulum. The rectangles with solid lines are R^{α_i}, i=1,2,3,4 in Figure 4.5.9 and $R_m^{\alpha_i}$, i=1,2,3,4 in Figure 4.5.10. Noticed that R^{α_1} is actually a point in the state space.

Table 4.5.1: Cell numbers in $R_m^{\alpha_i}$, i=1,2,3,4

α/group number	1	2	3	4	5
1.0	1	0	0	0	0
0.7	9	0	0	0	0
0.3	36	0	0	0	13
0.0	79	0	0	0	42

Table 4.5.2: Cell proportions in $R_m^{\alpha_i}$, i=1,2,3,4

α/group number	1	2	3	4	5
1.0	1.0	0	0	0	0
0.7	1.0	0	0	0	0
0.3	0.73	0	0	0	0.26
0.0	0.65	0	0	0	0.35

By counting the cells in the respective regions, we shall have two tables of numbers of cells and proportions as seen in Table 4.5.1 and Table 4.5.2. From (4.4.10)-(4.4.12), we can derive:

$$m_1 = \max_{i=1,2,3,4} \min(\alpha_i, P_{i2}) \tag{4.5.4}$$

$$= \max(1.0, 0.7, 0.3, 0.0)$$

$$= 1.0$$

$$m_5 = \max_{i=1,2,3,4} \min(\alpha_i, P_{i1}) \tag{4.5.5}$$

$$= \max(0.0, 0.0, 0.26, 0.0)$$

$$= 0.26$$

Finally, the fuzzy group number of the fuzzy initial states $\vec{x}_1$ and $\vec{x}_2$ is:

$$\vec{G}(\vec{x}_1, \vec{x}_2) = 1/1.0 + 2/0.0 + 3/0.0 + 4/0.0 + 5/0.26 \tag{4.5.6}$$

which means the system state has a possibility value 1.0 of converging to group 1, a possibility value 0.26 of converging to group 5, and a possibility value 0.0 of converging to any of the last three groups.

4.5.3 Stability Analysis

In Figure 4.5.6, there are a p-1 motion at the origin (group 2) and a p-8 motion around the origin (group 1). The domain of attraction of group 1 covers most of the central area in the figure. The four corners belong to the domain of attraction of group 5, the sink cell. Now, suppose that we choose an ε-neighborhood T_ε of **0** to be

$$T_\varepsilon = N(0, 1) \tag{4.5.7}$$

$$= \{ (0,0), (1,0), (1,1), (0,1), (-1,1), (-1,0), (-1,-1), (0,-1), (1,-1) \}, \tag{4.5.8}$$

and a domain set S_D to be

$$S_D = N(0, 20). \tag{4.5.9}$$

Since T_ε is the union of group 1 and group 2 and all the cells in S_D belong to the domain of attraction of group 1, the fuzzy system of inverted pendulum is cellular stable by definition 4.4.5.

4.6 Discussion

We wish to discuss some important points in this section. In the process of creating the cell-to-cell mapping from a fuzzy mapping, we fixed Δt_k to be a constant t_m and the partition to be equal divisions. However, this is only for the reasons of simplicity. There are other approaches for dealing with the problem:

(1) With the partition fixed, we could choose different Δt_k for different initial cells. The function of a cell-to-cell mapping in this method is to describe the flow of a fuzzy dynamical system. Therefore, we only need to know the relation between a cell and its neighboring cells. In other words, beginning with an initial cell, the process can be terminated once the trajectory enters one of its neighboring cells. A cell-to-cell mapping constructed in this fashion has advantages over the previous approach. For fast dynamic regions, Δt_k will be small, thus avoiding possible big jumps when using fixed Δt_k. For slow dynamic regions, Δt_k will become large so as to find the actual image cell position instead of staying in the same cell and creating the illusion of periodic motions. It is obvious that more details of the system dynamics will be revealed in this approach.

(2) Another approach is to use nonuniform partitions while keeping Δt_k fixed. We can partition the state space according to the speed of the system dynamics in local regions. There will be coarse partitions for slow dynamics regions and fine partitions for fast dynamics regions. The analysis will be more efficient and revealing by using such an approach because unnecessary computations will be avoided.

However, it is still unclear how we can know the speed of the system dynamics beforehand. Hence, the first approach seems to be more implementable than the second.

One other issue to be discussed is the accuracy of the global analysis. Since the cell state space is actually an approximation of the real state space, the prediction of the system dynamics around the boundary of two groups is likely to be unreliable. The accuracy will certainly improve with the cost of computation time if we use smaller cell size of the state space.

4.7 Concluding Remarks

A new approach for the global analysis of fuzzy dynamical systems has been proposed. It applies the cell-to-cell mapping method to obtain the evolving trend of the states of a fuzzy dynamical system. This method solves the problem of uncertainty accumulation by confining uncertainty only in the transformation from the fuzzy mappings to the cell-to-cell mappings. It greatly reduces the uncertainty in the analysis. As a result, we can give a characterizing description of the system dynamics. However, the method can only give an approximate prediction of the behavior of a fuzzy system, which is, on the other hand, quite reasonable owing to the fuzziness inherent by the system.

This cell-to-cell mapping method proves to be very successful in analyzing the global behavior of the inverted pendulum case given in the example. It gives very good, though not precise, description of the system behavior. There remains a question about how we relate the predictions to a real system, i.e. how do we know that the behavior of the cells in the cell state space can really represent the behavior of the states in the real state space? We observed a very consistent behavior of the inverted pendulum and its corresponding cell-to-cell mapping in Fig. 4.4.2. But it is obvious that the phenomenon is not universal. Intuitively, the method will certainly work if we have infinitesimal partitions. In that case, the cell-to-cell mapping will simply converge to the original point-to-point mapping. But it is impractical to have very small divisions of a state space because of the large amount of computation needed. Hence, it is very important to know the conditions on a fuzzy dynamical system such that the cell-to-cell mapping method can be applied. One reasonable conjecture is the "well-behavedness" of a fuzzy dynamical system. The "well-behavedness" may be the continuity of the vector field of the fuzzy dynamical system or some other more restrictive condition. The answer is not clear at this moment and needs further investigation.

Chapter 5

Experimental Comparison of Conventional and Fuzzy Logic Control

5.1 Introduction

Fuzzy logic control has found many practical applications recently. Its idea of transforming the knowledge and/or experience of human operators into a rule-base to perform control actions has proved to be very useful in dealing with systems which are difficult to control by conventional methods. One of the advantages of this approach is that it does not require a mathematical model of the process. All the control rules are formulated based on the operators' knowledge or "rules of thumb". On the other hand, all existing classical control theories are constructed with a certain model representation in mind, such as transfer functions and state equations. Their design procedures require precise prior knowledge of system models. Hence, it becomes extremely difficult when only partial or none of the system information is available.

State feedback control is one of the modern control techniques (Kailath (1980)), which uses a control law

$$u = -\mathbf{k}\mathbf{x}, \tag{5.1.1}$$

or

$$u(i) = -\mathbf{k}\mathbf{x}(i) \tag{5.1.2}$$

in discrete-time cases. u is the input variable of the plant, which is a real number in single-input systems; $\mathbf{x}$ is the state variable which is an $n \times 1$ column vector; $\mathbf{k}$ is the $1 \times n$ row vector of feedback gains. Its formulation is based on the state space representation of the controlled system. Many theories have been developed to design a stablizing state feedback controller (SFC) for an LTI system (Chen (1984)). Its basic idea is to relocate the eigenvalues of a system to the left-half plane of its state space through state feedback. A closed-loop system can be stablized if the

corresponding open-loop system is *stablizable* in the first place. This method is very simple and effective and is widely used in practical situations.

Since fuzzy logic control is not based on conventional mathematical theory, it needs justification through the performance of its practical implementations. It would , therefore, be very interesting to compare the differences between model-based control algorithms, such as state feedback control and AI-oriented fuzzy logic control through some specific experimentation. A hardware setup of a cart-pole system was built under a project at the NASA Ames Research Center to study the implementation of a stablizing fuzzy logic controller and to compare it with a state feedback controller. The control objective was to balance an inverted pendulum in a vertical position by controlling the motion of the cart and at the same time move the cart to a prescribed position. This problem has been studied in many articles and has been used as a benchmark for many nonlinear control algorithms. The results of the experiment postulate the differences between the two different approaches and are described in the rest of the chapter.

This chapter is organized as follows: Section 2 describes the problem formulation and its hardware setup. Section 3 gives the modeling and identification of the system. Section 4 explains the design procedures and the details of the two controllers compared. Section 5 makes a comparison of the fuzzy logic and state feedback controllers based on the experimental data. This is followed by our conclusions.

5.2 Description and Design of the System

The basic physical configuration of the cart-pole system in the experiment is similar to Figure 3.4.1. A cart with four grooved free-turning wheels was placed on a pair of tight-fitting rails. A pole was installed on the cart by fixing one end to the pivot of a low-friction potentiometer through a pole-holder. The potentiometer was mounted firmly on the cart so that the pole could fall freely in a forward-backward direction. The cart-pole combination was driven by a DC motor through an aluminum chain which matched the teeth of the driving pulleys. A 15KΩ low-

friction one-turn potentiometer and a 10KΩ ten-turn potentiometer were mounted on the end of the pole and the shaft of the DC motor respectively , were used to take the readings of the angle of the pole with respect to the vertical line and the cart position on the rails. These two readings were used as the sensor inputs to the controller. Shielded electric wires were also used to protect the measurements from outside noise.

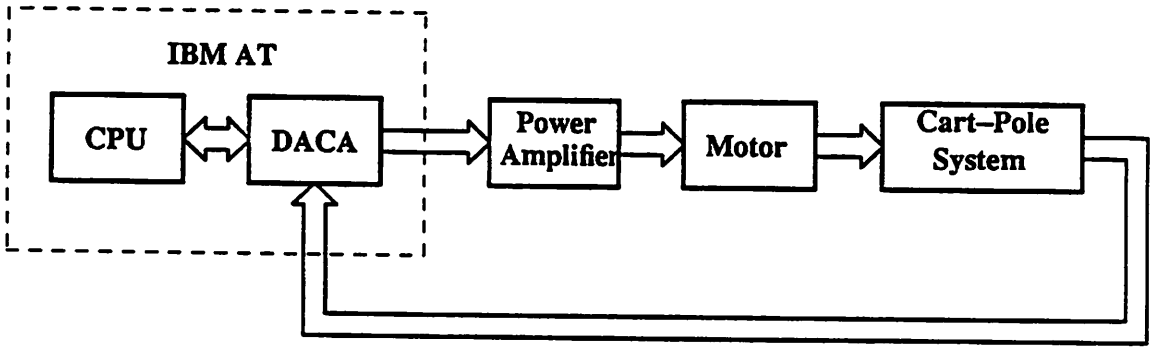

Figure 5.2.1: Block diagram of the cart-pole control system

As shown by the block diagram of the integrated system in Figure 5.2.1, the plant includes the cart-pole system, the DC motor, and the power amplifier which supplies electric current to the motor. The major part of the controller block is an IBM AT with a Data Acquisition and Control Adaptor (DACA) which has two D/A channels and four A/D channels. Two A/D channels are used to take sensor measurements from the two potentiometers and one D/A channel is used to send the controller output to the power amplifier, which will then be amplified to feed to the driving motor. The software controllers are coded in C and this code can be found in Appendix B. The data communication programs are also in C. These are not shown in the block diagram but can also be found in Appendix B.

5.3 System Modeling and Identification

First, system modeling and identification was done with a view to designing a state feedback controller. In this experiment, a fuzzy logic controller was designed, as generally claimed, without a model of the system.

5.3.1 Modeling

The inverted pendulum problem has been studied by many researchers. The governing equations of the cart-pole system are given by the following nonlinear differential equations (Cannon (1967)) (Barto, Sutton and Anderson (1983)):

$$\ddot{\theta} = \frac{g \sin\theta + \cos\theta \left[\frac{-f - m\, l\, \dot{\theta}^2 \sin\theta + \mu_c\, sgn(\dot{x})}{m_c + m} \right] - \frac{\mu_p\, \dot{\theta}}{m\, l}}{l \left[\frac{4}{3} - \frac{m \cos^2\theta}{m_c + m} \right]} \tag{5.3.1}$$

$$\ddot{x} = \frac{f + m\, l\, [\dot{\theta}^2 \sin\theta - \ddot{\theta} \cos\theta] - \mu_c\, sgn(\dot{x})}{m_c + m} \tag{5.3.2}$$

where θ is the angle of the pole with respect to the vertical line, x is the horizontal position of the cart, f is the driving force applied to the cart, g is the acceleration due to gravity, m_c is the mass of the cart, m is the mass of the pole, l is the half-pole length, μ_c is the coefficient of friction of cart on track, and μ_p is the coefficient of friction of pole on cart.

The plant being considered in our experiment included a cart-pole system, a driving DC motor and a power amplifier. Its input was the voltage output of the DACA board, which supplied the power amplifier. The power amplifier was assumed to have a constant gain. Since the angular position and velocity of the DC motor shaft were linearly proportional to the cart position and velocity respectively, the relation between the cart driving force f, the velocity of the cart $\dot{x}$, and the input voltage v was modeled as:

$$f = k\, v - b\, \dot{x} \tag{5.3.3}$$

where k and b are constant coefficients. By considering the coulomb friction between the cart and the rails, the net force applied to the cart f_n can be derived by:

$$f_n = f - f_c \tag{5.3.4}$$

$$= k\,v - b\,\dot{x} - \mu_c\, sgn\,(\dot{x}) \tag{5.3.5}$$

$$= k\,(v - \frac{\mu_c}{k} sgn\,(\dot{x})) - b\,\dot{x} \tag{5.3.6}$$

$$= k\,(v - v_c) - b\,\dot{x} \qquad (v_c = \frac{\mu_c}{k} sgn\,(\dot{x})) \tag{5.3.7}$$

$$= k\,v' - b\,\dot{x} \qquad (v' = v - v_c) \tag{5.3.8}$$

where f_c is the coulomb friction and v_c is its equivalent in terms of the DACA board input voltage.

The pole friction was neglected because of the use of a low-friction potentiometer which linked the pole to the cart, i.e. $\mu_p = 0.0$. Choose θ, $\dot{\theta}$, x, and $\dot{x}$ to be the state variables x_1, x_2, x_3, x_4 and v' to be the input variable. Based on (5.3.1)-(5.3.8), the whole system can be linearized at the origin and represented in a state space form:

$$\dot{\mathbf{x}} = \mathbf{A}_l\,\mathbf{x} + \mathbf{B}_l\,v' \tag{5.3.9}$$

where

$$\mathbf{A}_l = \begin{bmatrix} 0 & & & 0 \\ \frac{3(m_c+m)g}{l(4m_c+m)} & 1 & 0 & \frac{3b}{l(4m_c+m)} \\ 0 & 0 & 0 & 1 \\ \frac{-3mg}{4m_c+m} & 0 & 0 & \frac{-4b}{4m_c+m} \end{bmatrix} \tag{5.3.10}$$

$$\mathbf{B}_l = \begin{bmatrix} 0 \\ \dfrac{-3k}{l(4m_c + m)} \\ 0 \\ \dfrac{4k}{4m_c + m} \end{bmatrix} \tag{5.3.11}$$

5.3.2 System Identification

We first identified the DC motor dynamics by considering the cart-pole system with the pole taken off. The governing equation can be described as:

$$f_n = f - f_c \tag{5.3.12}$$

$$= M\ddot{x} \tag{5.3.13}$$

$$= k(v - v_c) - b\dot{x}, \qquad \left(v_c = \frac{\mu_c}{k} sgn(\dot{x})\right) \tag{5.3.14}$$

where M includes the mass of the cart, the mass of the chain and the equivalent mass of the motor shaft inertial, v_c is the *equivalent voltage* of the coulomb friction. The coulomb friction was first identified to be 1.967 volt by examining the relation between the input voltage and the steady state velocity of the cart as shown in Figure 5.3.1.

From the step response of the cart position (Figure 5.3.2) with the mass M known, k and b were identified to be:

$$k = 0.835, \quad b = 25.0. \tag{5.3.15}$$

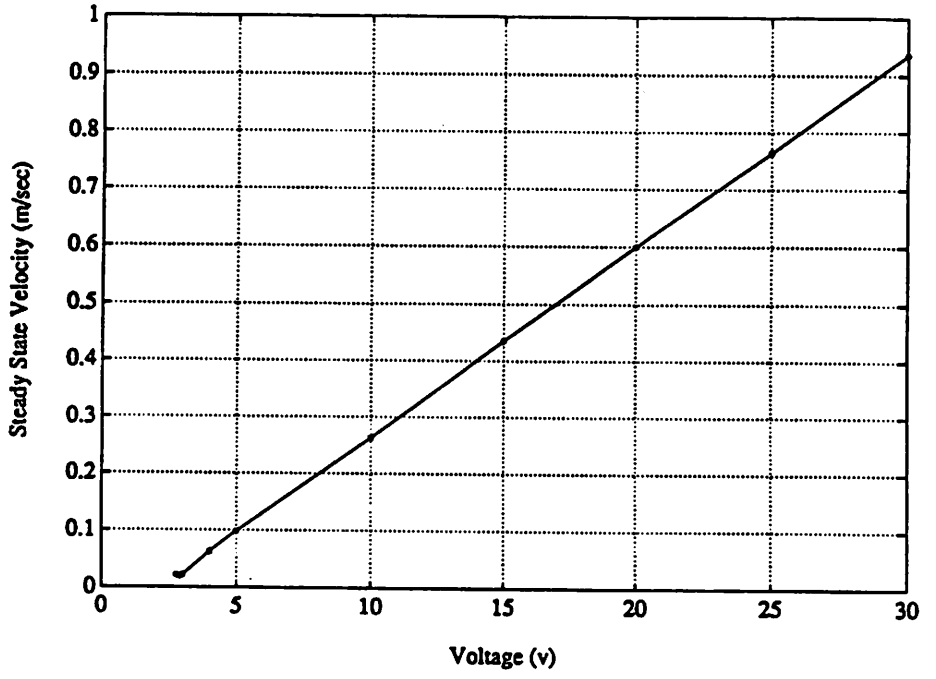

Figure 5.3.1: Steady state velocity *vs.* input voltage

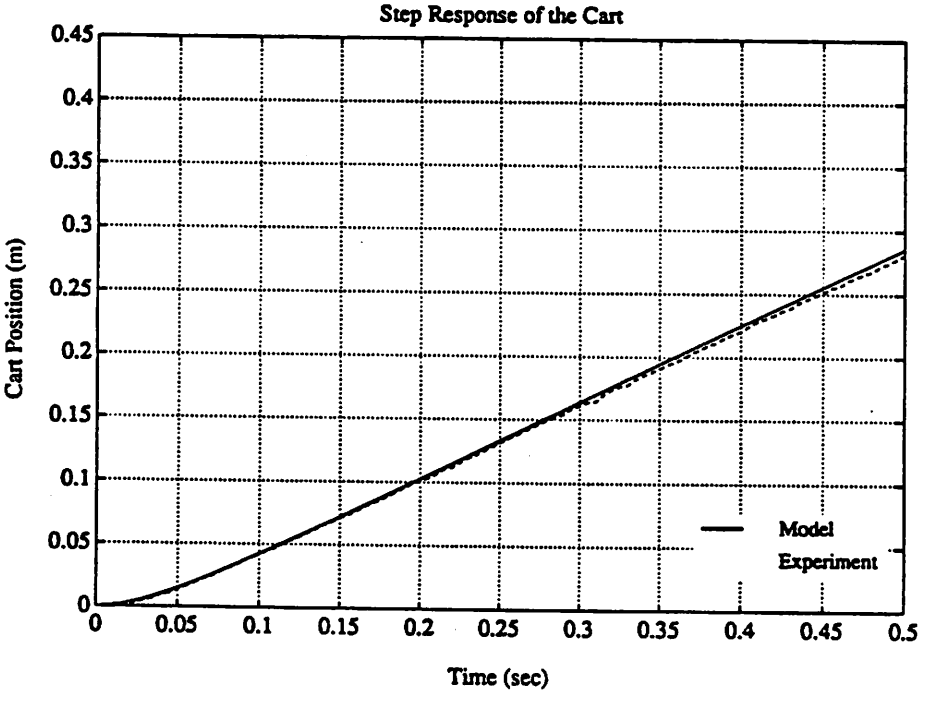

Figure 5.3.2: Step response of the cart system

Finally, the state space representation of the cart-pole system with

half-pole length l: 0.2858 m,

mass of the pole m: 0.0473 kg,

mass of the cart m_c: 0.6983 kg,

is given by

$$\dot{\mathbf{x}} = \mathbf{A}_l \mathbf{x} + \mathbf{B}_l v' \tag{5.3.16}$$

where

$$\mathbf{A}_l = \begin{bmatrix} 0.000 & 1.000 & 0.000 & 0.000 \\ 27.002 & 0.000 & 0.000 & 92.386 \\ 0.000 & 0.000 & 0.000 & 0.000 \\ -0.4896 & 0.000 & 0.000 & -35.205 \end{bmatrix} \qquad \mathbf{B}_l = \begin{bmatrix} 0.000 \\ -3.086 \\ 0.000 \\ 1.176 \end{bmatrix} \tag{5.3.17}$$

In our experiment, the control algorithms were implemented in an IBM AT with a sampling time of 20 ms. Therefore, the identified model was transformed to its discretized form with sampling time 20 ms as:

$$x(k+1) = \Phi x(k) + \Gamma v'(k) \tag{5.3.18}$$

where

$$\Phi = \begin{bmatrix} 1.0054 & 0.0200 & 0.000 & 0.0148 \\ 0.5337 & 1.0054 & 0.000 & 1.3292 \\ -0.0001 & 0.0000 & 1.000 & 0.0144 \\ -0.0070 & -0.0001 & 0.000 & 0.4945 \end{bmatrix} \qquad \Gamma = \begin{bmatrix} -0.0005 \\ -0.0444 \\ 0.0002 \\ 0.0169 \end{bmatrix} \tag{5.3.19}$$

5.4 Controller Design

The design procedures of both the fuzzy logic controller and the state feedback controller are described in detail in this section. Since fuzzy logic control is supposedly based on the knowledge of human experts, a fuzzy logic controller was designed entirely by heuristics and no model of the system was involved. The state feedback controller, on the other hand, was designed according to the identified system model in (5.3.18).

5.4.1 Fuzzy Logic Controller

Balancing a pole on one's palm is intuitively easy. Most people can perform the balancing for a short while. However, it becomes extremely difficult if one wishes to balance the pole on the palm and at the same time hold one's position. Very few people, such as acrobats, can perform it for a long period of time. In this experiment, a fuzzy logic controller was designed to perform the balancing through observations and heuristics. After some trial and error, the policies for balancing the pole were easily acquired as

if the pole is falling to the right, *then* push the cart to the right,

if the pole is falling to the left, *then* push the cart to the left.

The pole-balancing policies above were easy to understand and to derive. However, it became more difficult when the cart position was also considered. After some playing around with a pole, we came up with two general rules for controlling the cart position, i.e.

if the pole is *almost* balanced and the cart is on the right side of center,

then push the cart to the right,

if the pole is *almost* balanced and the cart is on the left side of center,

then push the cart to the left.

At first, it would seem to contradict to our intuition to push the cart to the right (left) direction when it was already in the right (left) position. Given a position of the cart, the trick was to push

the cart in the same direction hard enough so that the pole would fall to the opposite direction. Then the attempt of balancing the pole would move the cart toward the center position.

Four variables were used to describe the system status at each stage and functioned as the feedback signals to the controller. One variable denoted the control action adopted corresponding to a specific system status. The variables were:

θ : angle of the pole with respect to the vertical line

$\dot{\theta}$: angular velocity of the pole

x : horizontal position of the cart on the rails

$\dot{x}$: velocity of the cart

v : voltage applied to the power amplifier

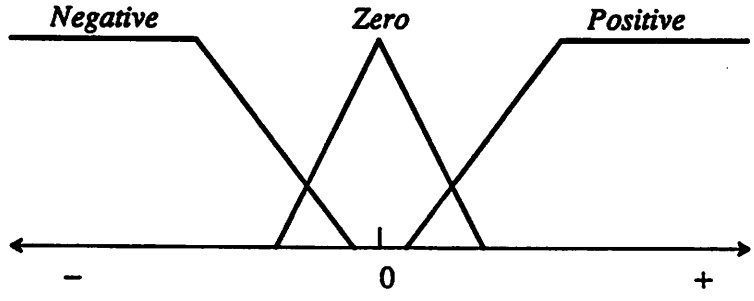

Figure 5.4.1: Fuzzy terms for the state variables

Three labels were used to linguistically define the value of each of the state variables: Positive (PO), Zero (ZE), and Negative (NE). Figure 5.4.1 illustrates the membership functions of these linguistic terms. Note that the three linguistic labels were defined differently for each of the four state variables. Seven labels were used for the input variable v, namely Negative-Small (NS), Negative-Medium (NM), Negative-Large (NL), Zero (ZE), Positive-Small (NS), Positive-Medium (NM), and Positive-Large (NL). The membership functions of these linguistic terms are illustrated in Figure 5.4.2.

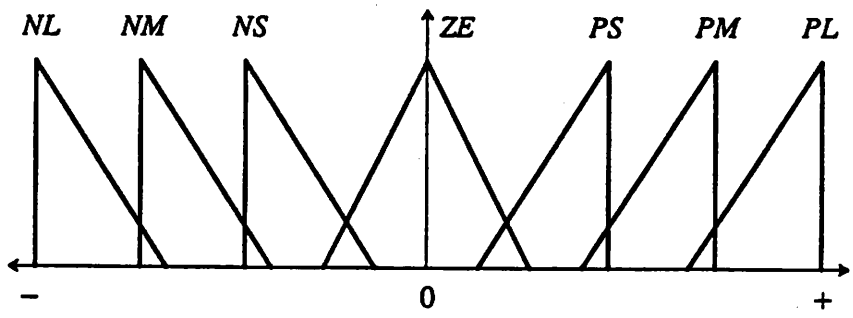

Figure 5.4.2: Fuzzy terms for input voltage v

The control policies were implemented in a rule base consisting of 9 rules for controlling the angular position of the pole and 4 rules for centering the cart position. The complete rule base is described in the following:

Rule-1 : if θ is PO and $\dot{\theta}$ is PO , then v is PL

Rule-2 : if θ is PO and $\dot{\theta}$ is ZE , then v is PM

Rule-3 : if θ is PO and $\dot{\theta}$ is NE , then v is ZE

Rule-4 : if θ is ZE and $\dot{\theta}$ is PO , then v is PS

Rule-5 : if θ is ZE and $\dot{\theta}$ is ZE , then v is ZE

Rule-6 : if θ is ZE and $\dot{\theta}$ is NE , then v is NS

Rule-7 : if θ is NE and $\dot{\theta}$ is PO , then v is ZE

Rule-8 : if θ is NE and $\dot{\theta}$ is ZE , then v is NM

Rule-9 : if θ is NE and $\dot{\theta}$ is NE , then v is NL

Rule-10 : if θ is VS and $\dot{\theta}$ is VS and x is PO and $\dot{x}$ is PO ,
then v is PM

Rule-11 : if θ is VS and $\dot{\theta}$ is VS and x is PO and $\dot{x}$ is PVS, then v is PS

Rule-12 : if θ is VS and $\dot{\theta}$ is VS and x is NE and $\dot{x}$ is NE, then v is NM

Rule-13 : if θ is VS and $\dot{\theta}$ is VS and x is NE and $\dot{x}$ is NVS, then v is NS

where *VS* is Very Small, *PVS* is Positive Very Small, and *NVS* is Negative Very Small. To reduce the computation time in order to balance the pole in real time, we adopted the second type of fuzzy reasoning for our controller operation. Details of the operation were given in Chapter 2.

5.4.2 State Feedback Controller

Based on the model given in (5.3.18)-(5.3.19), a state feedback controller was designed by the pole-placement technique. The input variables $v'(i)$ was a linear combination of the two state variables x_1 and x_2:

$$v'(n) = k_1 x_1(n) + k_2 x_2(n) + k_3 x_3(n) + k_4 x_4(n) \tag{5.4.1}$$

where k_1, k_2, k_3 and k_4 are state feedback gains. The determination of the controller gains was also constrained by the voltage rating of the power amplifier. A stablizing discrete state feedback controller was designed by assigning the poles inside the unit circle at:

$$p_1 = 0.55 + 0.5i \ , \ p_2 = 0.55 - 0.5i \ , \ p_3 = 0.98 + 0.1i \ , \ p_4 = 0.98 - 0.1i \ . \tag{5.4.2}$$

The corresponding state feedback controller has the gains :

$$k_1 = 605 \ , \ k_2 = 19.3 \ , \ k_3 = 352.1 \ , \ k_4 = 40.2 \ . \tag{5.4.3}$$

Therefore, including the friction compensation, the final control law was:

$$v(n) = v'(n) + v_c \, sgn(x_4(n)) \tag{5.4.4}$$

$$= k_1 x_1(n) + k_2 x_2(n) + k_3 x_3(n) + k_4 x_4(n) + v_c \, sgn(x_4(n)) \tag{5.4.5}$$

5.5 Experimental Results and Performance Comparison

Both of the fuzzy logic and the state feedback controller achieved their control objectives in our experiment. The experimental data of the trajectories of the pole angle, cart position, and input voltage are shown in Figure 5.5.1-5.5.6.

Performance Comparison

In our experiment, the fuzzy logic controller succeeded in balancing the pole practically from the beginning of the testing and was tuned to maintain the cart position at the prescribed center point after some parameter adjustments. The state feedback controller also achieved very good performance on both the pole balancing and cart centering, which was really not unexpected. It was accomplished through time-consuming system modeling and identification. A precise mathematical model as in (5.3.19), had to be acquired before the state feedback controller could be designed.

Fuzzy logic control demonstrated its potential in handling difficult control problems in our experiment. It was very interesting to see that a difficult pole balancing and a cart position centering tasks could be done simultaneously simply by implementing the intuitive and linguistic balancing strategies from a person who was really not an "expert". The result by fuzzy logic control, was especially valuable if the amount of effort spent on modeling and identification is also considered. For instance, an estimated 75% time was spent on the modeling and identification in our experiment. Our fuzzy logic controller required minimal time to design and balanced the pole within the first few trials. Its control, though not as precise, was as good as the state feedback controller. Based on the experimental results, the performance comparison of the two different approaches are summarized in the following:

- Design Complexity: The design of a fuzzy logic controller involves much less mathematical calculations than a state feedback controller. It does not require modeling and identification of a controlled process, which can be very difficult and time-consuming if the process is very complex, there is no physical model, and/or human factors are involved. Contrary to

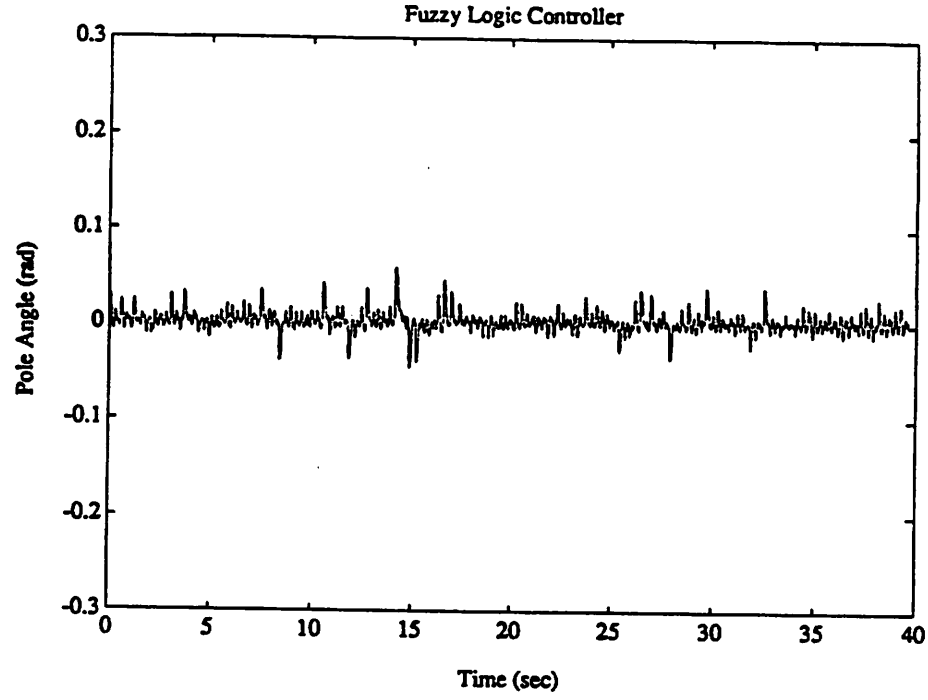

Figure 5.5.1: Pole position for fuzzy logic control

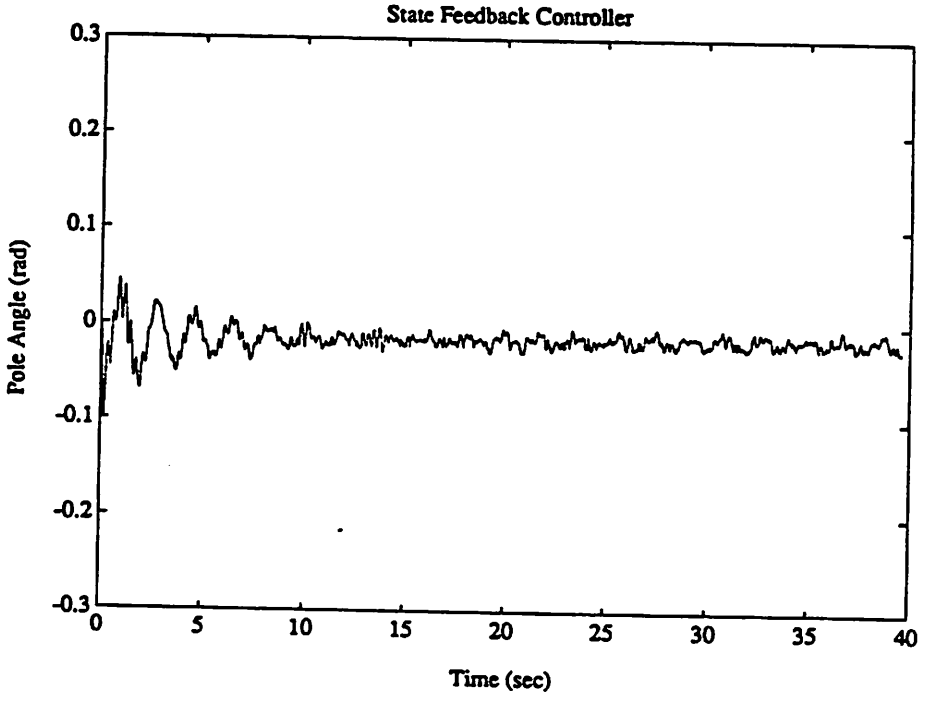

Figure 5.5.2: Pole position for state feedback control

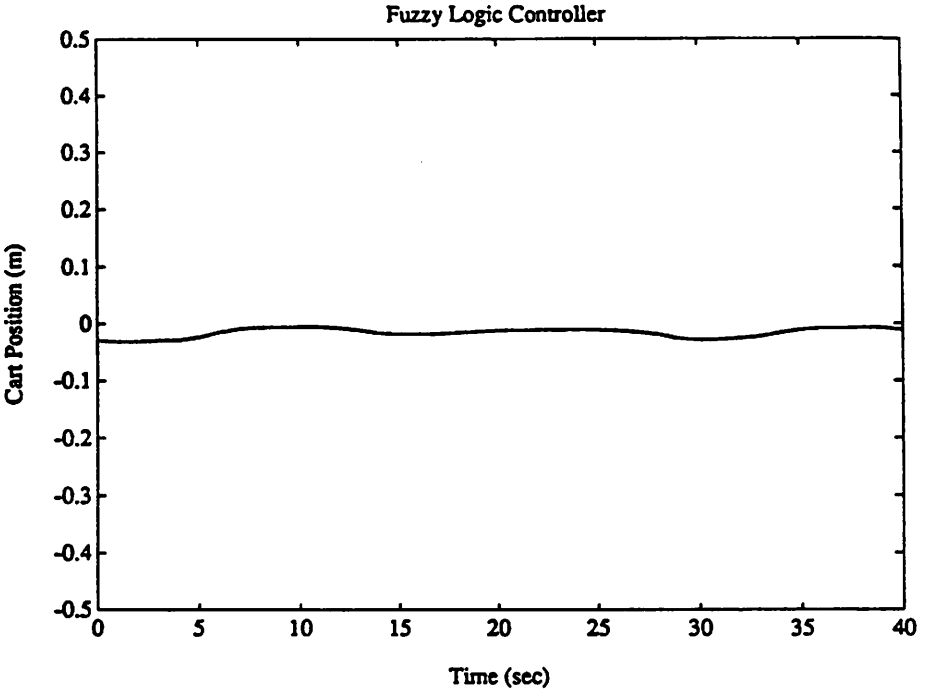

Figure 5.5.3: Cart position for fuzzy logic control

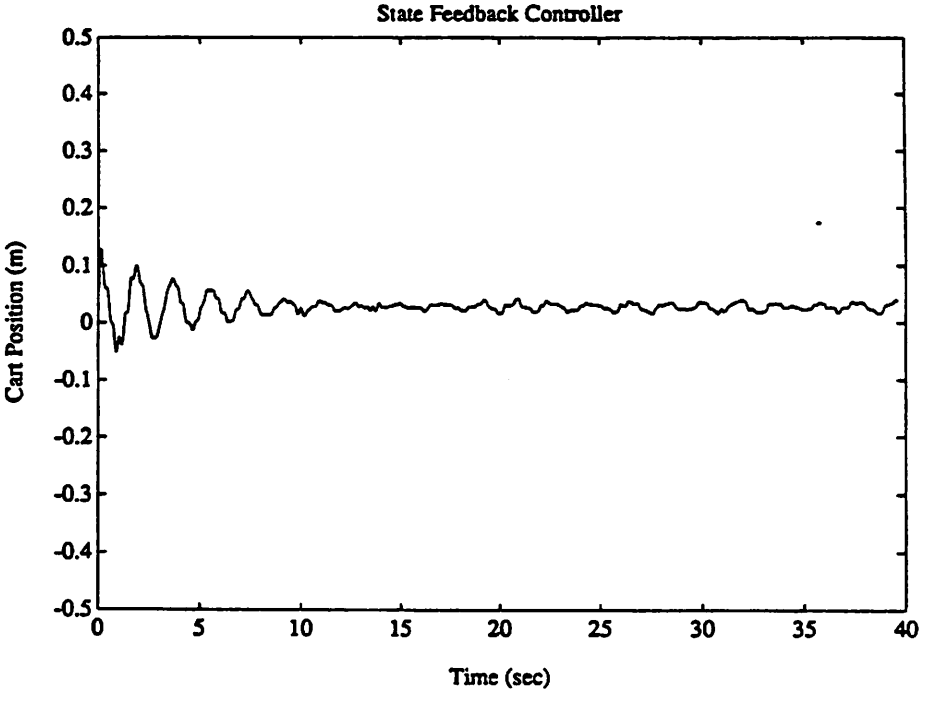

Figure 5.5.4: Cart position for state feedback control

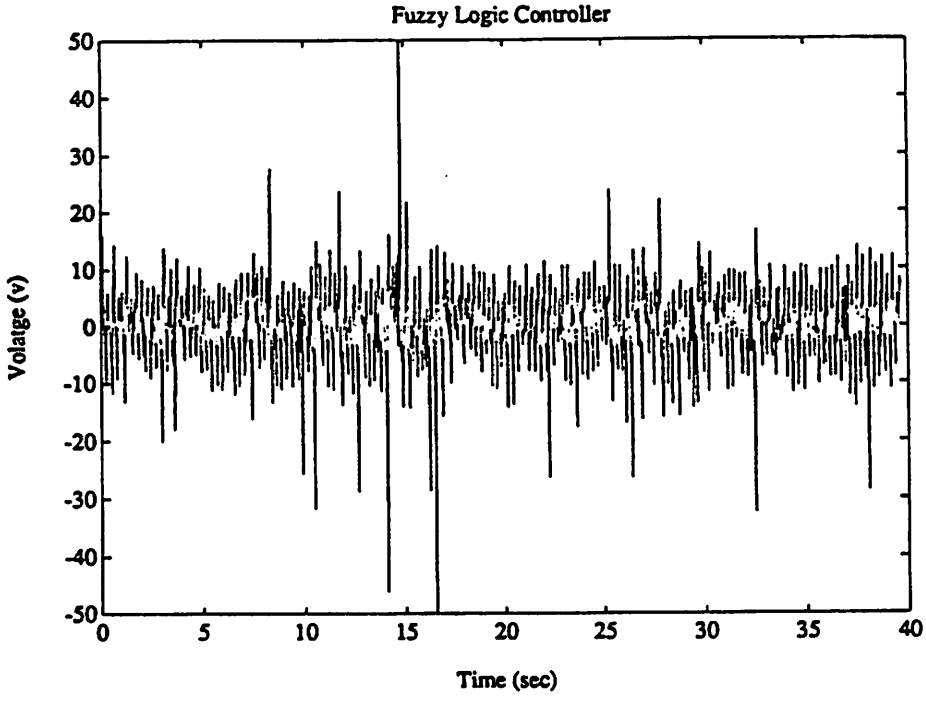

Figure 5.5.5: Applied voltage for fuzzy logic control

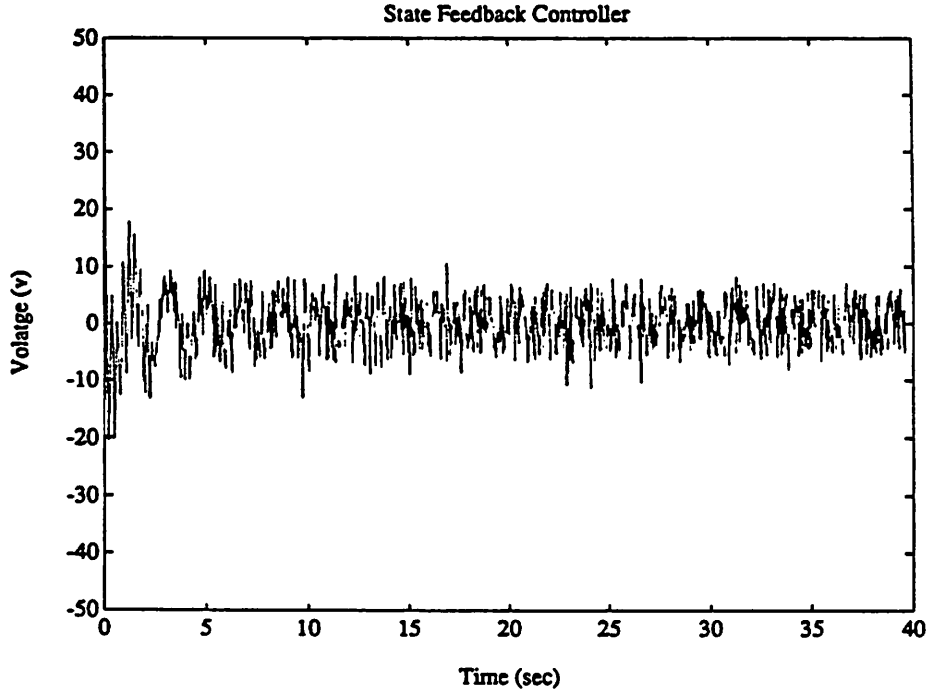

Figure 5.5.6: Applied voltage for state feedback control

fuzzy logic control, state feedback control is a model-based control algorithm. Its design requires complete knowledge of the concerned process. However, this approach is very simple and effective if a well-understood LTI system is considered. Therefore, the design complexity of the two approaches dependents more on the process considered than on the design strategy itself.

- Performance: "Precision" is usually not important for fuzzy logic control. The rule base constructed by human experts normally achieves an *acceptable* performance in a fairly short period of time. It, however, can only reach a control objective approximately due to its *fuzzy* nature. As demonstrated by Figure 5.5.1, the pole position does not converge to zero uniformly but oscillates in its neighborhood. State feedback control benefited from a precise model and has a more precise behavior.
- Stability Region: Since many systems are inherently nonlinear, their identified models are applicable only around their equilibrium points. A model-based state feedback controller will perform correctly when it is operating in an area, namely the stability region, around the equilibria. A fuzzy logic controller, which is nonlinear in design, adopts different control actions in different areas and can often work in a larger stability region than a state feedback controller.
- Robustness: Robustness is one of the acclaimed properties of fuzzy logic control. The design of a state feedback controller does not include the uncertainty in a system, while a fuzzy logic controller based on experts' knowledge has a larger tolerance to the variation of the process parameters. Moreover, we believe that the multiple firing of control rules in a fuzzy logic controller is also partly responsible for the robustness of the system.
- Controller Modification: This is, in our opinion, the main disadvantage in current fuzzy logic controller design. Many parameters are used in the linguistic rules of a fuzzy logic controller. The fine tuning of these parameters to achieve a better system performance is a very time-consuming process. State feedback controller involves much less parameters than its AI counterpart and can be easily fine-tuned.

5.6 Concluding Remarks

Our experimental study of the AI-oriented fuzzy logic control algorithm and the conventional state feedback control scheme verified some of the generally claimed differences between the two approaches. The rule-based fuzzy logic control, benefits from the knowledge of human operators, does not require a precise model of a system. It is generally formulated to control a complex nonlinear process with man-machine interactions. On the other hand, state feedback control, a modern control scheme, is based on a precisely identified model to reallocate the poles of the closed loop system in order to achieve stability. It is originally designed on linear systems and does not result in a large stability region when a highly nonlinear system is involved. In our experiment, fuzzy logic control performed, though not as precise, as well as state feedback control. Most importantly, it had its great advantage of design efficiency. Given a "true" expert and a reasonable amount of effort, we believe that fuzzy logic control can perform better than state feedback control in many cases.

This experiment was conducted in order to compare the differences between fuzzy logic control and conventional state feedback control. In our experiment, the model of the system was derivable and was identified so that the state feedback controller can be designed accordingly. However, there are many different complex situations in the real world. Some of them can be resolved only by conventional control theories, for instance, mechanical machining often requires high precision and is best for mathematically-proved conventional control theories. Some others may be solved by both methods, such as the inverted pendulum problem in our experiment. The choice of either methods depends on the availability of resources and the goals to be achieved. There are, however, some situations which are either impossible or very difficult to be handled by conventional methods, such as the automobile parking problem. In these cases, the AI-oriented fuzzy logic control is more appropriate to be applied to achieve the control goals.

All in all, fuzzy logic control has demonstrated its usability and value in certain types of situations. It was never designed to replace a conventional control algorithm. Instead, it aimed at offering a different approach, which has proved to be very effective, in many situations where

conventional methods fail to achieve better results.

Chapter 6

Conclusions

Most of the current applications of fuzzy logic control are aimed at replacing a human operator with a linguistic rule-based system. The utilization of an operator's knowledge of a specific process in achieving a control goal has proved to be very effective and successful. Due to the fact that human operators are always using *fuzzy* expressions in describing their knowledge, fuzzy set theory and fuzzy logic have played important roles in this approach. However, we wish to point out that the key factor of fuzzy logic control, i.e. representing human operators' knowledge in a *fuzzy* format, is actually the weakest point, as far as analysis is concerned, because of its noncompatibility to the existing precise mathematical world of analysis. Without a formal analysis, all the applications of fuzzy logic control can only be justified through practical implementations. There is no stability analysis before and/or after the building up of a real system. A lengthy process of "trial and error" and the fine tuning of its parameters has to be involved.

In this thesis, we first discussed the stability property of a fuzzy control system and proposed a notion of "expert's Lyapunov function" to explain its seemingly intrinsic stability. Two theorems are formulated to describe its stability criterion. To remedy the disadvantage of lacking of an analytical methodology, this thesis also endeavors to develop an analytical method which can describe the global behavior of a fuzzy dynamical system and indicate its stability properties. The proposed method incorporated an analytical technique, namely cell-to-cell mapping, in constructing the global picture of a fuzzy dynamical system. A thorough analysis of fuzzy dynamical systems using this method was performed. Both the real and fuzzy initial states responses were discussed. Cellular stability was defined and a stability theorem formulated to give a complete description of the analyzed fuzzy system. In order to understand the differences between the AI-oriented fuzzy logic control and conventional control algorithms, such as state feedback control, an experiment was conducted under a project in NASA Ames research center. The experimental

results gave a clearer picture of both the benefits and limitations of fuzzy logic control.

Though fuzzy logic control was successful in the past, we recognize the fact that many future applications may not have human experts available for the controller design. The space station thermal control studied by NASA, for example, involved complex modeling and computations if handled by conventional methods. The problem may possibly be solved by using the technique of fuzzy logic control. Unfortunately, it is obvious that there will be no "expert" for consultation in the design process. As a result, the automation in fuzzy logic controller design has become much more important than it was before. The method proposed in this thesis can also serve the purpose of design automation. Given a process and a controller candidate, our method is capable of determining its closed-loop performance as well as indicating the undesirable rules in the controller for further modifications. The procedure can be repeated until satisfactory control is achieved.

We shall conclude this thesis with several constructive propositions for future fuzzy logic control automation.

(i) *Linguistic model-based fuzzy logic control*: It is the most intuitive and probably the most promising way for design automation. The proposed controller will depend on the linguistic model used. The basic problem can be simplified to the form:
Given a set of implication rules R_1, and a desired control goal R_3, find R_2 such that

$$R_1 \text{ and } R_2 \rightarrow R_3. \tag{6.1}$$

This approach involves qualitative reasoning and is very interested in our opinion.

(ii) *Self-tuning fuzzy logic control*: This approach will be most beneficial to current controller design if succeeded. It can be done by incorporating some optimization and/or learning scheme in a fuzzy logic controller such that the self-tuning can be achieved by evaluating its performance. A combination of fuzzy logic and neural networks may be a good direction for future research.

(iii) *Rule Extraction*: The purpose of fuzzy logic control automation is to reduce the degree of dependency on human experts. This can be achieved by systematically formulating control rules which were given previously by human experts (Chen (1989)). This approach is particularly interesting because it preserves the useful format of fuzzy logic control while eliminating the necessity of human experts.

Appendix A

Proof of Theorem 3.2.1 :

Assume that the cardinality of X is n, i.e. $card(X) = n$. Hence, $\vec{X}$ is a $1 \times n$ row vector and $\vec{Q}$ is a $n \times n$ square matrix. Let LHS denote the left-hand side in (3.2.4) and RHS denote the right-hand side. To simplify the notation, denote $\mu_X(x_i)$ by X_i and $\mu_Q(x_i, x_j)$ by Q_{ij}, $x_i, x_j \in X$. Then, the kth element of LHS is

$$LHS_k = \bigvee_j ([\bigvee_i X_i \wedge Q_{ij}] \wedge Q_{jk}) \tag{A.1}$$

and

$$RHS_k = \bigvee_i (X_i \wedge [\bigvee_j Q_{ij} \wedge Q_{jk}]). \tag{A.2}$$

Since the $\vee$ (max) and the $\wedge$ (min) operators are commutative and distributive, we have

$$LHS_k = \bigvee_j (\bigvee_i [X_i \wedge Q_{ij} \wedge Q_{jk}]) \tag{A.3}$$

$$= \bigvee_{i,j} [X_i \wedge Q_{ij} \wedge Q_{jk}] \tag{A.4}$$

$$= \bigvee_i (\bigvee_j [X_i \wedge Q_{ij} \wedge Q_{jk}]) \tag{A.5}$$

$$= RHS_k \tag{A.6}$$

Q.E.D.

Proof of Theorem 3.2.2 :

Using similar notations as in Theorem 3.2.1 with MHS denoting the middle term in (3.2.5) as well as the definition of $\vec{X}\vec{U}$ in (3.2.2), it is easy to see that

$$LHS_k = \bigvee_{i,j} (XU_{ij} \wedge P_{ijk}) \tag{A.7}$$

$$= \bigvee_{i,j} ([X_i \wedge U_j] \wedge P_{ijk}) \tag{A.8}$$

$$= \underset{i,j}{\vee} (X_i \wedge U_j \wedge P_{ijk}) \tag{A.9}$$

and

$$MHS_k = \underset{i}{\vee} (X_i \wedge [\underset{j}{\vee} U_j \wedge P_{ijk}]) \tag{A.10}$$

$$= \underset{i,j}{\vee} (X_i \wedge U_j \wedge P_{ijk}) \tag{A.11}$$

$$= LHS_k \tag{A.12}$$

and

$$RHS_k = \underset{j}{\vee} (U_j \wedge [\underset{i}{\vee} X_i \wedge P_{ijk}]) \tag{A.13}$$

$$= \underset{i,j}{\vee} (X_i \wedge U_j \wedge P_{ijk}) \tag{A.14}$$

$$= LHS_k \tag{A.15}$$

Q.E.D.

Proof of Theorem 3.3.1 :

From (3.3.10) and (3.3.11), we can have

$$\dot{x} = f(x, u) \tag{A.16}$$

$$= f(x, g(x)) \tag{A.17}$$

$$= f'(x) \tag{A.18}$$

The theorem can then be proved by the Lyapunov Uniform Stability Theorem.

Q.E.D.

Proof of Theorem 3.3.2 :

From (3.3.10) and (3.3.11), we can have

$$\dot{x} = f(x, u) \tag{A.19}$$

$$= f(x, g(x)) \tag{A.20}$$

$$= f'(x) \tag{A.21}$$

The theorem can then be proved by the Lyapunov Asymptotic Stability Theorem.

Q.E.D.

Appendix B

The control and data communication programs used in the inverted pendulum experiment are included in this appendix and are listed in the following:

(1) fuzzy.c : The host program for fuzzy logic control,

(2) sf.c : The host program for state feedback control,

(3) daca.h : The header file for the IBM DACA board,

(4) daca.c : The data acquisition program for DACA board,

(5) pendulum.c : Definitions of input-output variables.

```
/*********************************************************/
/* fuzzy.c:  host program for the real time          */
/*         control of the inverted pendulum system.  */
/*         using fuzzy logic control                 */
/*         NO timer interrupt is used to invoke      */
/*         sample().                                 */
/*         CE line is used to enable DA conversion   */
/*         and hence control the sampling rate.      */
/*********************************************************/

#include <stdio.h>
#include "daca.h"
#define sign(x) ((x >= 0.0) ? 1.0 : -1.0)
#define max(x,y) ( (x > y) ? x : y )
#define min(x,y) ( (x < y) ? x : y )

double exp();
extern float pos_x1();  /*in pendulum.c */
extern float pos_x3();  /*in pendulum.c */
extern float drive();   /*in pendulum.c */
extern int reset();

static float f1,f2,f3,sf1,sf2;
static float x[5];
static float oldx[5];
static float olderx[5];
static float f;
static float samp_time;
static float _null;
static float _tmp;
static float x1[15];
static float x2[15];
static float x3[15];
static float x4[15];
static float lx2[15];
static float lx4[15];
static float w[15];
static float u[15];
static float wu[15];
static float w1, w2, wu1, wu2, ww, uu;
static float mag;
static float weight;
static float a, az, azz, b, bz, bzz;
static float c, cz, d, dz;
static float pos1, zero1, veryz1, neg1;
static float pos2, zero2, veryz2, neg2;
static float pos3, zero3, neg3;
static float pos4, poszero4, zero4, negzero4, neg4;
static float d1, d2, d3, d4, d5, d6, d7, d8, d9, d10;
static float d11, d12, d13, d14, d15, d16, d17, d18, d19, d20;
static float d21, d22, d23, d24, d25, d26, d27, d28, d29, d30;
static float d31, d32;


static int i;
static int j;
static int _num_samp;
static int _choice;
static int _time;
```

```
static char _cmd;

float data1[10000];
float data2[10000];
float data3[10000];
float data4[10000];
float data5[10000];


static char filename[10];
FILE *af, *fopen();

main()
{

        do {

                reset();

                prompt();

                start();

                printf(" Work End! Start storing data.");

                reset();

                store_data();

        } while( !(finish()));

        reset();
}

/*****************************************************************/

prompt()
{

        printf("  DATA INPUT-STORAGE FILE (<= 7 characters)    :-  ");
        scanf("%s", *filename);

        printf("  CONTROLLER GAINS mag:-  ");
        scanf("%f ", &mag);

        printf(" POLICY WEIGHTING weight:-  ");
        scanf("%f ", &weight);

        printf(" Force1 f1:-  ");
        scanf("%f ", &f1);

        printf(" Force2 f2:-  ");
        scanf("%f ", &f2);

        printf(" Force3 f3:-  ");
        scanf("%f ", &f3);

        printf(" SForce1 sf1:-  ");
        scanf("%f ", &sf1);
```

```
        printf(" SForce2 sf2:-   ");
        scanf("%f ", &sf2);

        printf(" SAMPLING TIME:-  ");
        scanf("%f", &samp_time );

        printf(" NUMBER OF NULL ITERATIONS:-  ");
        scanf("%f", &_null );

        printf(" CHOICE NUMBER , 0-fixed samples, 1-continuous   :-  ");
        scanf("%d", &_choice);

        if( _choice == 0 ) {
                printf(" NUMBER OF SAMPLE <1 - 10000>          :-  ");
                scanf("%d", &_num_samp);
        }

}


start()
{
        reset();

        _time = 0;
        getchar();
        printf("  Turn on POWER. HIT RETURN TO START !  ");
        if( _choice != 0 )
        printf("  HIT 'q' after RETURN  TO STOP !  ");
        getchar();


        if( _choice == 0 ) {
                while(_time < _num_samp) {
                        sample();
                }
        }
        else {
                do {
                        _cmd = bdos(0x06,0x00ff) & 0x00ff;
                        sample();
                } while(_cmd != 'q');
        }

}


store_data()
{
        af = fopen(*filename,"w");

        for(i=0;i<_time;i++) {
                fprintf( af,"%5d  %f  %f  %f0,i,data1[i],data3[i],data5[i]);
        }

        fclose(af);
}

int finish()
{
```

```
        printf("  Do you wish to run again   <y/n>    :- ");

        if((char)getchar() == 'y')return(0);

return(1);
}


/************************************************/

sample()
{

/****** start of the controller ***************/

        /*** Parameters ***/

        a = 0.2;
        az = 0.2;
        azz = 0.01;

        b = 1.0;
        bz = 0.5;
        bzz = 0.1;

        c = 1.0;
        cz = 0.1;

        d = 10.0;
        dz = 1.0;

        /*** Sensor readings ***/

        x[1] = pos_x1();
        x[2] = (x[1] - oldx[1])/samp_time;
        x[3] = pos_x3();
        x[4] = (x[3] - oldx[3])/samp_time;

        lx2[2] = (x[2]+oldx[2]+olderx[2])/3.0;
        lx4[4] = (x[4]+oldx[4]+olderx[4])/3.0;

        /*** Membership values ***/

        d1 = max( x[1]/a, 0.0);
        pos1 = min( d1, 1.0 );

        d2 = max(1.0-x[1]/(az), 0.0);
        d3 = max(1.0+x[1]/(az), 0.0);
        zero1 = min ( d2, d3);

        d4 = max(1.0-x[1]/(azz), 0.0);
        d5 = max(1.0+x[1]/(azz), 0.0);
        veryz1 = min ( d4, d5);

        d6 = max( x[1]/(-a), 0.0);
        neg1 = min( d6, 1.0 );

        d7 = max( lx2[2]/b, 0.0);
        pos2 = min( d7, 1.0 );
```

```
d8 = max(1.0-lx2[2]/(bz), 0.0);
d9 = max(1.0+lx2[2]/(bz), 0.0);
zero2 = min ( d8, d9);

d10 = max(1.0-lx2[2]/(bzz), 0.0);
d11 = max(1.0+lx2[2]/(bzz), 0.0);
veryz2 = min ( d10, d11);

d12 = max( lx2[2]/(-b), 0.0);
neg2 = min( d12, 1.0 );

d13 = max( x[3]/c, 0.0);
pos3 = min( d13, 1.0 );

d14 = max(1.0-x[3]/(cz), 0.0);
d15 = max(1.0+x[3]/(cz), 0.0);
zero3 = min ( d14, d15);

d16 = max( x[3]/(-c), 0.0);
neg3 = min( d16, 1.0 );

d17 = max( lx4[4]/d, 0.0);
pos4 = min( d17, 1.0 );

d18 = max(1.0-lx4[4]/(dz), 0.0);
d19 = max(1.0+lx4[4]/(dz), 0.0);
zero4 = min ( d18, d19);

d20 = max( lx4[4]/(-d), 0.0);
neg4 = min( d20, 1.0 );

d21 = max(1.0-lx4[4]/(dz), 0.0);
d22 = max(1.0+lx4[4]/(0.000000001*dz), 0.0);
poszero4 = min( d21, d22);

d23 = max(1.0-lx4[4]/(0.000000001*dz), 0.0);
d24 = max(1.0+lx4[4]/(dz), 0.0);
negzero4 = min( d23, d24);

/*** Rules ***/

x1[1] = pos1;
x2[1] = pos2;
w[1] = min( x1[1], x2[1] );
u[1] = f1 + 100.0*w[1];

x1[2] = pos1;
x2[2] = zero2 ;
w[2] = min( x1[2], x2[2] );
u[2] = f2 + 100.0*w[2];

x1[3] = pos1;
x2[3] = neg2;
w[3] = min( x1[3], x2[3] );
u[3] = 0.0*w[3];

x1[4] = zero1 ;
x2[4] = pos2;
w[4] = min( x1[4], x2[4] );
u[4] = f3*w[4]*(0.5 + 0.5*sign(x[1]));
```

```
x1[5] = zero1 ;
x2[5] = zero2 ;
w[5] = min( x1[5], x2[5] );
u[5] = 0.0

x1[6] = zero1 ;
x2[6] = neg2;
w[6] = min( x1[6], x2[6] );
u[6] = -f3*w[6]*(0.5 - 0.5*sign(x[1]));

x1[7] = neg1 ;
x2[7] = pos2;
w[7] = min( x1[7], x2[7] );
u[7] = -400.0*w[7];

x1[8] = neg1;
x2[8] = zero2;
w[8] = min( x1[8], x2[8] );
u[8] = -(f2) - 100.0*w[8];

x1[9] = neg1;
x2[9] = neg2;
w[9] = min( x1[9], x2[9] );
u[9] = -(f1) - 100.0*w[9];

x1[11] = veryz1 ;
x2[11] = veryz2 ;
x3[11] = pos3;
x4[11] = pos4;
d25 = min(x1[11],x2[11]);
d26 = min(x3[11],x4[11]);
w[11] = min( d25, d26 );
u[11] = 10.0*sf1*w[11];

x1[12] = veryz1 ;
x2[12] = veryz2 ;
x3[12] = pos3;
x4[12] = poszero4 ;
d27 = min(x1[12],x2[12]);
d28 = min(x3[12],x4[12]);
w[12] = min( d27, d28 );
u[12] = 0.5*sf2*w[12];

x1[13] = veryz1 ;
x2[13] = veryz2 ;
x3[13] = neg3 ;
x4[13] = neg4 ;
d29 = min(x1[13],x2[13]);
d30 = min(x3[13],x4[13]);
w[13] = min( d29, d30 );
u[13] = -10.0*sf1*w[13];

x1[14] = veryz1 ;
x2[14] = veryz2 ;
x3[14] = neg3 ;
x4[14] = negzero4 ;
d31 = min(x1[14],x2[14]);
d32 = min(x3[14],x4[14]);
w[14] = min( d31, d32 );
```

```
        u[14] = -0.5*sf2*w[14];


        /*** Defuzzification ***/

        w1 = 0.0;
        w2 = 0.0;
        wu1 = 0.0;
        wu2 = 0.0;

        for( i=1; i < 10; i++) {
                wu[i] = u[i]*w[i];
                w1 = w1 + w[i];
                wu1 = wu1 + wu[i];
        }


        for( i=11; i < 15; i++) {
                wu[i] = u[i]*w[i];
                w2 = w2 + w[i];
                wu2 = wu2 + wu[i];
        }

        ww = w1 + w2;
        uu = mag*(wu1 + weight*wu2);

        if(ww > 0.0)
                f = uu/ww;
        else
                f = 0.0;

        /****** end of controller **********************/

        data1[_time] = x[1];
        data3[_time] = x[3];
        data5[_time] = f;

        drive(f);

        olderx[1] = oldx[1];
        olderx[2] = oldx[2];
        olderx[3] = oldx[3];
        olderx[4] = oldx[4];

        oldx[1] = x[1];
        oldx[2] = x[2];
        oldx[3] = x[3];
        oldx[4] = x[4];

        _time ++;

        for(i=0;i<_null;i++)
        _tmp = (float) (i+1)*(float)i/(float)(i+1);

}
```

```
/*********************************************************/
/* sf.c  : host program for the real time          */
/*         control of the inverted pendulum system.  */
/*         using state feedback control              */
/*         NO timer interrupt is used to invoke      */
/*         sample().                                 */
/*         CE line is used to enable DA conversion   */
/*         and hence control the sampling rate.      */
/*********************************************************/

#include <stdio.h>
#include "daca.h"

double exp();
extern float pos_x1();  /*in pendulum.c */
extern float pos_x3();  /*in pendulum.c */
extern float drive();   /*in pendulum.c */
extern int reset();

static float x[5];
static float oldx[5];
static float v;
static float k1;
static float k2;
static float k3;
static float k4;
static float samp_time;
static float _null;
static float _tmp;
static float _dead_zone;
static float _noise_immunity;
static float _dynamic_friction;

static int i;
static int j;
static int _num_samp;
static int _choice;
static int _time;
static int _mode;

static char _cmd;

float data1[10000];
float data3[10000];
float data5[10000];


static char filename[10];
FILE *a, *fopen();

main()
{

      do {
            reset();

            prompt();

            start();
```

```
            printf(" Work End! Start storing data.");

            reset();

            store_data();

        } while( !(finish()));

        reset();

}



/*************************************************************************/

prompt()
{

        printf("  DATA INPUT-STORAGE FILE (<= 7 characters)    :-  ");
        scanf("%s", *filename);

        printf("  CURRENT CONTROLLER GAINS k1,k2,k3,k4:-  ");
        scanf("%f %f %f %f", &k1, &k2, &k3, &k4 );

       printf("  DYNAMIC FRICTION        :-    ");
       scanf("%f", &_dynamic_friction);

        printf("  SAMPLING TIME:-  ");
        scanf("%f", &samp_time );

        printf("  NUMBER OF NULL ITERATIONS:-  ");
        scanf("%f", &_null );

        printf("  CHOICE NUMBER , 0-fixed samples, 1-continuous    :-  ");
        scanf("%d", &_choice);

        if( _choice == 0 ) {
            printf("  NUMBER OF SAMPLE <1 - 10000>          :-  ");
            scanf("%d", &_num_samp);
        }

}


start()
{
        reset();

        _time = 0;
        getchar();
        printf("  Turn on POWER. HIT RETURN TO START !  ");
        if( _choice != 0 )
        printf("  HIT 'q' after RETURN  TO STOP !  ");
        getchar();


        if( _choice == 0 ) {
            while(_time < _num_samp) {
```

```
                sample();
            }
        }
        else {
            do {
            _cmd = bdos(0x06,0x00ff) & 0x00ff;
            sample();
            } while(_cmd != 'q');
        }

}


store_data()
{
        a = fopen(*filename,"w");

        for(i=0;i<_time;i++) {
            fprintf(a,"%5d  %f  %f  %f0,i,data1[i],data3[i],data5[i]);
        }

        fclose(a);
}

int finish()
{
        printf("  Do you wish to run again   <y/n>     :- ");

        if((char)getchar() == 'y')return(0);

return(1);
}


sample()
{

/****** start of the controller ***************/
/****** Sensor readings ************************/

        x[1] = pos_x1();
        x[2] = (x[1] - oldx[1])/samp_time;
        x[3] = pos_x3();
        x[4] = (x[3] - oldx[3])/samp_time;

        v = k1*x[1]+k2*x[2]+k3*x[3]+k4*x[4]+_dynamic_friction*sgn(x[4]);


/****** end of controller *********************/

        data1[_time] = x[1];
        data3[_time] = x[3];
        data5[_time] = v;

        drive(v);

        oldx[1] = x[1];
        oldx[2] = x[2];
        oldx[3] = x[3];
        oldx[4] = x[4];
```

```
        _time ++;

        for(i=0;i<_null;i++) {
                _tmp = (float) (i+1)*(float)i/(float)(i+1);
        }

}
```

```
/*************************************************************/
/* File :- DACA.H                                            */
/*      Header File for the IBM Data Acqui-                  */
/*      sition Board (DACA).                                 */
/*************************************************************/

/* Device number selection */

#define SEL_DEVICE  0xc2e2          /* select device number */

#define     ANALOG      9           /* analog device number */
#define BINARY      8           /* binary device number */

/* Binary Device (Parallel I/O) */

#define DPORT       0x22e2          /* binary port base address */

#define DPORTLO         DPORT       /* low byte data        */
#define DPORTHI         DPORT+1     /* high byte data       */

/* Analog Device */

#define D_A0            0x0000      /* D/A channel 0        */
#define D_A1            0x0001      /* D/A channel 1    */

#define A_D0            0x0000      /* A/D channel 0    */
#define A_D1            0x0001      /* A/D channel 1        */
#define A_D2            0x0002      /* A/D channel 2    */
#define A_D3            0x0003      /* A/D channel 3    */

#define D_A_SEL         0x12e3      /* Channel select register  */
#define A_D_SEL         0x02e3      /* Channel select register  */

#define STOP_A_D    0x0000          /* Stop conversion          */
#define START_A_D   0x0001          /* Start conversion         */

#define A_D_CNT         0x02e2      /* A/D control register     */

#define A_D_STA         0x02e2      /* A/D status register      */

#define A_D_RDY         0x00f2      /* A/D ready                */

#define A_D_DAT         0x22e2      /* A/D data register        */
#define A_D_LOW         0x22e2      /* A/D low byte             */
#define A_D_HGH         0x22e3      /* A/D high byte        */

#define D_A_DAT         0x32e2      /* D/A data register        */
#define D_A_LOW         0x32e2      /* D/A low byte             */
#define D_A_HGH         0x32e3      /* D/A high byte        */


/* Unit Conversions */

#define VOFFSET         2048        /* Analog Binary Offset     */

#define DIG_VLTS    10.0/2048.0 /* Digital to volts     */
#define VLTS_DIG    2048.0/10.0 /* Volts to digital     */
```

```
/*****************************************************************/
/* File :- DACA.C                                                */
/*      Data Acquisition program for the IBM                     */
/*      Data Acquisition Board (DACA).                           */
/*****************************************************************/

#include <stdio.h>
#include "daca.h"


/*****************************************************************/
/*    DIGITAL-ANALOG CONVERSION                                  */
/*****************************************************************/

float fda0(value)
float value;
{
float actuator;
int output;

        actuator = value;

        if(actuator >= 2047.0) actuator = 2047.0;
        if(actuator <= -2048.0) actuator = -2048.0;  /* actuator limit check */

        outportb(SEL_DEVICE,ANALOG);  /* select analog I/O */

        outportb(D_A_SEL,D_A0);  /* select D/A channel */

        output = (int)(actuator + (float)VOFFSET);  /* convert to digital */

        outportb(D_A_LOW,(output & 0x00ff));
        outportb(D_A_HGH,((output >> 8) & 0x00ff));  /* output value */

return(actuator);
}

float fda1(value)
float value;
{
float actuator;
int output;

        actuator = value;

        if(actuator >= 2047.0) actuator = 2047.0;
        if(actuator <= -2048.0) actuator = -2048.0;  /* actuator limit check */

        outportb(SEL_DEVICE,ANALOG);  /* select analog I/O */

        outportb(D_A_SEL,D_A1);     /* select D/A channel */

        output = (int)(actuator + (float)VOFFSET);  /* convert to digital */

        outportb(D_A_LOW,(output & 0x00ff));
        outportb(D_A_HGH,((output >> 8) & 0x00ff));  /* output value */

return(actuator);
}
```

```
int da0(value)
int value;
{
int actuator;
int output;

        actuator = value;

        if(actuator >= 2047) actuator = 2047;
        if(actuator <= -2048) actuator = -2048;      /* actuator limit check */

        outportb(SEL_DEVICE,ANALOG);        /* select analog I/O */

        outportb(D_A_SEL,D_A0);             /* select D/A channel */

        output = actuator + VOFFSET;  /* convert to digital */

        outportb(D_A_LOW,(output & 0x00ff));
        outportb(D_A_HGH,((output >> 8) & 0x00ff));   /* output value */

return(actuator);
}


int da1(value)
int value;
{
int actuator;
int output;

        actuator = value;

        if(actuator >= 2047) actuator = 2047;
        if(actuator <= -2048) actuator = -2048;      /* actuator limit check */

        outportb(SEL_DEVICE,ANALOG);        /* select analog I/O */

        outportb(D_A_SEL,D_A1);             /* select D/A channel */

        output = actuator + VOFFSET;  /* convert to digital */

        outportb(D_A_LOW,(output & 0x00ff));
        outportb(D_A_HGH,((output >> 8) & 0x00ff));   /* output value */

return(actuator);
}


float vda0(value)
float value;
{
float actuator;
int output;

        actuator = value;

        if(actuator >= 10.0) actuator = 10.0;
        if(actuator <= -10.0) actuator = -10.0;      /* actuator limit check */
```

```
        outportb(SEL_DEVICE,ANALOG);        /* select analog I/O */

        outportb(D_A_SEL,D_A0);             /* select D/A channel */

        output = (int)(actuator * VLTS_DIG + VOFFSET); /* convert to digital */

        outportb(D_A_LOW,(output & 0x00ff));
        outportb(D_A_HGH,((output >> 8) & 0x00ff));  /* output value */

return(actuator);
}


float vda1(value)
float value;
{
float actuator;
int output;

        actuator = value;

        if(actuator >= 10.0) actuator = 10.0;
        if(actuator <= -10.0) actuator = -10.0;      /* actuator limit check */

        outportb(SEL_DEVICE,ANALOG);        /* select analog I/O */

        outportb(D_A_SEL,D_A1);             /* select D/A channel */

        output = (int)(actuator * VLTS_DIG + VOFFSET); /* convert to digital */

        outportb(D_A_LOW,(output & 0x00ff));
        outportb(D_A_HGH,((output >> 8) & 0x00ff));  /* output value */

return(actuator);
}


/*************************************************************/
/*      ANALOG-DIGITAL CONVERSION                            */
/*************************************************************/

float fad0()
{
int input;
float position;

        outportb(SEL_DEVICE,ANALOG); /* select analog device */

        outportb(A_D_CNT,STOP_A_D);   /* stop conversion */
        outportb(A_D_SEL,A_D0);    /* select A/D channel 0  */

        outportb(A_D_CNT,START_A_D); /* start conversion */
        outportb(A_D_SEL,A_D0);    /* select A/D channel 0  */

        while((inportb(A_D_STA) & 0x00ff) != A_D_RDY);
                                /* wait for complete conversion */

        outportb(A_D_CNT,STOP_A_D);   /* stop conversion */
```

```
        outportb(A_D_SEL,A_D0);    /* select A/D channel 0  */

        input = inportb(A_D_LOW) & 0x00ff; /* input low byte */

        input |= ((inportb(A_D_HGH) & 0x000f) << 8);
                                   /* input high byte and combine */

        position = ((float)(input - VOFFSET));

return(position);
}

float fad1()
{
int input;
float velocity;

        outportb(SEL_DEVICE,ANALOG); /* select analog device */

        outportb(A_D_CNT,STOP_A_D);   /* stop conversion */
        outportb(A_D_SEL,A_D1);    /* select A/D channel 1  */

        outportb(A_D_CNT,START_A_D); /* start conversion */
        outportb(A_D_SEL,A_D1);    /* select A/D channel 1  */

        while((inportb(A_D_STA) & 0x00ff) != A_D_RDY);
                                   /* wait for complete conversion */

        outportb(A_D_CNT,STOP_A_D);   /* stop conversion */
        outportb(A_D_SEL,A_D1);    /* select A/D channel 1  */

        input = inportb(A_D_LOW) & 0x00ff; /* input low byte */

        input |= ((inportb(A_D_HGH) & 0x000f) << 8);
                                   /* input high byte and combine */

        velocity = ((float)(input - VOFFSET));

return(velocity);
}


float fad2()
{
int input;
float position;

        outportb(SEL_DEVICE,ANALOG); /* select analog device */

        outportb(A_D_CNT,STOP_A_D);   /* stop conversion */
        outportb(A_D_SEL,A_D2);    /* select A/D channel 2 S*/

        outportb(A_D_CNT,START_A_D); /* start conversion */
        outportb(A_D_SEL,A_D2);    /* select A/D channel 2 S*/

        while((inportb(A_D_STA) & 0x00ff) != A_D_RDY);
                                   /* wait for complete conversion */

        outportb(A_D_CNT,STOP_A_D);   /* stop conversion */
        outportb(A_D_SEL,A_D2);    /* select A/D channel 2 S*/
```

```
        input = inportb(A_D_LOW) & 0x00ff; /* input low byte */

        input |= ((inportb(A_D_HGH) & 0x000f) << 8);
                                /* input high byte and combine */

        position = ((float)(input - VOFFSET));

return(position);
}

float fad3()
{
int input;
float velocity;

        outportb(SEL_DEVICE,ANALOG); /* select analog device */

        outportb(A_D_CNT,STOP_A_D);   /* stop conversion */
        outportb(A_D_SEL,A_D3);    /* select A/D channel 3  */

        outportb(A_D_CNT,START_A_D); /* start conversion */
        outportb(A_D_SEL,A_D3);    /* select A/D channel 3  */

        while((inportb(A_D_STA) & 0x00ff) != A_D_RDY);
                                /* wait for complete conversion */

        outportb(A_D_CNT,STOP_A_D);   /* stop conversion */
        outportb(A_D_SEL,A_D3);    /* select A/D channel 3  */

        input = inportb(A_D_LOW) & 0x00ff; /* input low byte */

        input |= ((inportb(A_D_HGH) & 0x000f) << 8);
                                /* input high byte and combine */

        velocity = ((float)(input - VOFFSET));

return(velocity);
}


int ad0()
{
int input;
int position;

        outportb(SEL_DEVICE,ANALOG); /* select analog device */

        outportb(A_D_CNT,STOP_A_D);   /* stop conversion */
        outportb(A_D_SEL,A_D0);    /* select A/D channel 0  */

        outportb(A_D_CNT,START_A_D); /* start conversion */
        outportb(A_D_SEL,A_D0);    /* select A/D channel 0  */

        while((inportb(A_D_STA) & 0x00ff) != A_D_RDY);
                                /* wait for complete conversion */

        outportb(A_D_CNT,STOP_A_D);   /* stop conversion */
        outportb(A_D_SEL,A_D0);    /* select A/D channel 0  */

        input = inportb(A_D_LOW) & 0x00ff; /* input low byte */
```

```
        input |= ((inportb(A_D_HGH) & 0x000f) << 8);
                                /* input high byte and combine */

        position = input - VOFFSET;

return(position);
}

int ad1()
{
int input;
int velocity;

        outportb(SEL_DEVICE,ANALOG); /* select analog device */

        outportb(A_D_CNT,STOP_A_D);    /* stop conversion */
        outportb(A_D_SEL,A_D1);    /* select A/D channel 1  */

        outportb(A_D_CNT,START_A_D); /* start conversion */
        outportb(A_D_SEL,A_D1);    /* select A/D channel 1  */

        while((inportb(A_D_STA) & 0x00ff) != A_D_RDY);
                                /* wait for complete conversion */

        outportb(A_D_CNT,STOP_A_D);    /* stop conversion */
        outportb(A_D_SEL,A_D1);    /* select A/D channel 1  */

        input = inportb(A_D_LOW) & 0x00ff; /* input low byte */

        input |= ((inportb(A_D_HGH) & 0x000f) << 8);
                                /* input high byte and combine */

        velocity = input - VOFFSET;

return(velocity);
}

int ad2()
{
int input;
int position;

        outportb(SEL_DEVICE,ANALOG); /* select analog device */

        outportb(A_D_CNT,STOP_A_D);    /* stop conversion */
        outportb(A_D_SEL,A_D2);    /* select A/D channel 2  */

        outportb(A_D_CNT,START_A_D); /* start conversion */
        outportb(A_D_SEL,A_D2);    /* select A/D channel 2  */

        while((inportb(A_D_STA) & 0x00ff) != A_D_RDY);
                                /* wait for complete conversion */

        outportb(A_D_CNT,STOP_A_D);    /* stop conversion */
        outportb(A_D_SEL,A_D2);    /* select A/D channel 2  */

        input = inportb(A_D_LOW) & 0x00ff; /* input low byte */

        input |= ((inportb(A_D_HGH) & 0x000f) << 8);
```

```
                                /* input high byte and combine */

        position = input - VOFFSET;

return(position);
}

int ad3()
{
int input;
int velocity;

        outportb(SEL_DEVICE,ANALOG); /* select analog device */

        outportb(A_D_CNT,STOP_A_D);    /* stop conversion */
        outportb(A_D_SEL,A_D3);    /* select A/D channel 3  */

        outportb(A_D_CNT,START_A_D); /* start conversion */
        outportb(A_D_SEL,A_D3);    /* select A/D channel 3  */

        while((inportb(A_D_STA) & 0x00ff) != A_D_RDY);
                                /* wait for complete conversion */

        outportb(A_D_CNT,STOP_A_D);    /* stop conversion */
        outportb(A_D_SEL,A_D3);    /* select A/D channel 3  */

        input = inportb(A_D_LOW) & 0x00ff; /* input low byte */

        input |= ((inportb(A_D_HGH) & 0x000f) << 8);
                                /* input high byte and combine */

        velocity = input - VOFFSET;

return(velocity);
}


/*********************************************************/
/* FUNCTION :                                            */
/*                                                       */
/*      reset() - reset ad-da registers                  */
/*                                                       */
/*********************************************************/

int reset()
{

        outportb(SEL_DEVICE,ANALOG);  /* select analog I/O */
        outportb(D_A_SEL,D_A0);
        outportb(D_A_LOW,0x00);
        outportb(D_A_HGH,0x08);             /* reset D/A0  */

        outportb(D_A_SEL,D_A1);
        outportb(D_A_LOW,0x00);
        outportb(D_A_HGH,0x08);             /* reset D/A1  */

}
```

```
/*************************************************************/
/* File :- PENDULUM.C                                        */
/*        Define input-output varaible names                 */
/*************************************************************/

#define PI 3.141592653

extern float fad0();
extern float fad1();
extern float fda0();


/*************************************************************/
/* Calibration for angle:    0: 7.22 volts                   */
/*                          90: 11.0 volts                   */
/*                         -45: 5.15 volts                   */
/*************************************************************/

float pos_x1()
{
      float p;
      if ( (fad0()/2047.0 - 7.22/10.0) >= 0.0)
      p = PI / 2.0 * (fad0()/2047.0 - 7.22/10.0)/((11.0-7.22)/10.0) ;
      else
      p = PI / 4.0 * (fad0()/2047.0 - 7.22/10.0)/((7.22-5.15)/10.0) ;
      return(p);
}

/*************************************************************/
/* Calibration for position:  0.0 cm: 6.75 volts             */
/*                         -0.527 cm: 2.05 volts             */
/*************************************************************/

float pos_x3()
{
      float p;
      p = 0.527/(6.75 - 2.05)*(fad1()/2047.0 - 6.75/10.0)*10.0 ;
      return(p);
}


float drive(value)
float value;
{
      float p;

      p = (value/30.0)*2047.0;
      return(fda0(p));
}
```

References

Arnol'd, V. D., "Small Denominators and Problems of Stability of Motion in Classical and Celestial Mechanics", *Russian Mathematical Survey*, Vol. 18, pp.85-191, 1963.

Baldwin, J. F. and Pilsworth, B. W., "Axiomatic Approach to Implication for Approximate Reasoning using Fuzzy Logic", *Fuzzy Sets and Systems*, Vol. 3, pp.193-219, 1980.

Bandler, W. and Kohout, L. J., "Semantics of Implication Operators and Fuzzy Relational Products", *Int. J. Man-Machine Studies*, Vol. 12, pp.89-116, 1980.

Bar-Shalom, Y. and Tse, E., "Concepts and Methods in Stochastic Control", *Control Dynam. Syst.: Adv. Theory Appl.*, Vol. 23, pp.99-172, 1976.

Basar, T., "Solutions to a Class of Nonstandard Stochastic Control Problems with Active Learning", *IEEE Trans. of Automatic Control*, Vol. 33, No. 12, pp.1122-1129, 1989.

Barto, A. G., Sutton, R. S., and Anderson C. W., "Neuronlike Adaptive Elements that Can Solve Difficult Learning Control Problems", IEEE Trans. on Systems, Man, and Cybernetics, Vol. SMC-13, pp.834-846, 1983.

Bartolini, G. et al., "Development of Performance Adaptive Fuzzy Controllers with Applications to Continuous Casting Plants", *Cybernetics and Systems Research*, Trapple, R. Ed., North-Holland, 1982.

Bellman, R. E. and Zadeh, L. A., "Decision Making in a Fuzzy Environment", *Management Science*, Vol.17, No.4, pp.141-164, 1970.

Berenji, H., Chen, Y.-Y., Lee, C.-C. and Murugesan, S., "The Inverted Pendulum Control using Approximate Reasoning" - A short video presentation, *Uncertainty in AI workshop, American Association for Artificial Intelligence Conference*, St. Paul, 1988.

Berenji, H., Chen, Y.-Y., Lee, C.-C. and Murugesan, S., "An Experiment-based Comparative Study on Fuzzy Control", *American Control Conference*, Pittsburgh, 1989.

Bernard, J. A., "The Construction and Use of a Knowledge Base in the Real-Time Control of Research Reactor Power", *Proc. 6th Power Plant Dynamics, Control, and Testing Symp.*, Knoxville, TN, 1986.

Bernussou, J. "Point Mapping Stability", *Pergamon Press*, New York, 1977.

Birkhoff, G. D., "Surface Transformations and Their Dynamical Applications", *Acta Math.*, Vol. 43, pp.1-119, 1920.

Bonissone, P. P. and Decker K. S., "Selecting uncertainty calculi and granularity: an experiment in trading-off precision and complexity", *Uncertainty in Artificial Intelligence*, Kanal L. N. and Lemmer J. F. (Editors), Elsevier Science Publishers B. V. (North-Holland), pp.217-247, 1986.

Braae, M. and Rutherford, D. A., "Selection of parameters for a fuzzy logic controller", *Fuzzy Sets and Systems*, Vol. 2, pp. 185-199, 1979.

Braae, M. and Rutherford, D. A., "Theoretical and linguistic aspects of the fuzzy logic controller", *Automatic*, Vol. 15, pp. 553-577, 1979.

Buckley, J. J., "The multiple judge, multiple criteria ranking problem: A fuzzy set approach", *Fuzzy Sets and Systems*, Vol. 13, pp.25-38, 1984.

Chang, S. S. L. and Zadeh, L. A., "On Fuzzy Mapping and Control", *IEEE Trans. Syst., Man, Cybern.*, Vol. SMC-2, pp.30-34, 1972.

Cannon, R. H., "Dynamics of Physical Systems", *McGraw-Hill*, 1967.

Chen, C. T., "Linear System Theory and Design", *CBS College Publishing*, 1984.

Chen, Y.-Y., "Stability Analysis of Fuzzy Control-- A Lyapunov Approach", *Proceedings of 1987 IEEE Syst., Man, Cybern., Annual Conference*, Vol. 3, pp.1027-1031, 1987.

Chen, Y.-Y., "The Analysis of Fuzzy Dynamic Systems using Cell-to-Cell Mapping", *Proceedings of 1988 IEEE Syst., Man, Cybern., Annual Conference*, Shengyang, 1988.

Chen, Y.-Y. and Tsao, T.-C., "A New Approach for The Global Analysis of Fuzzy Dynamical Systems", *Proceedings of the 27th IEEE Conference on Decisions and Control*, Austin,

1988.

Chen, Y.-Y. and Tsao, T.-C., "A Description of the Dynamic Behavior of Fuzzy Systems", *IEEE Transaction of Systems, Man, and Cybernetics*, Vol. 19, No. 4, July, 1989.

Chen, Y.-Y., "Rules Extraction for Fuzzy Control Systems", submitted to *Conference of Decision and Control*, 1989.

Chiu, H. M. and Hsu, C. S., "A Cell Mapping Method for Nonlinear Deterministic and Stochastic Systems-Part I: Examples of Application", *ASME J. of Applied Mechanics*, 1986.

Cumani, A., "On a Possibility Approach to the Analysis of Fuzzy Feedback Systems", *IEEE Trans. Syst., Man, Cybern.*, Vol. SMC-12, pp.417-422, 1982.

Czogala, E. and Pedrycz, W., "On Identification in Fuzzy Systems and its Applications in Control Problems", *Fuzzy Sets and Systems*, Vol. 6, No. 1, pp.73-83, 1981.

Czogala, E. and Pedrycz, W., "Fuzzy Rule Generation for Fuzzy Control", *Cybernet. Systems*, Vol. 13, No. 3, pp.275-294, 1982.

Dubois, D. and Prade, H., "Fuzzy Sets and Systems, Theory and Applications", *Academic Press*, 1980.

Flanagan, M. J., "On the Application of Approximate Reasoning to Control of the Activated Sludge Process", *Proc. of the Joint Automatic Control Conference*, San Francisco, 1980.

Fukuzaki, T. et al., "Knowledge Based System for Control Rod Programming of BWRs", *J. of Nuclear Science and Technology*, Vol. 25, No. 2, pp.120-130, 1988.

deGlas, M., "Invariance and stability of fuzzy systems", *J. Mathematical Analysis and Applications*, Vol. 99, pp. 299-319, 1984.

Gupta, M. et al, "Controllability of Fuzzy Control Systems", *IEEE Trans. Systems, Man, and Cybernet.*, Vol. SMC-16, No. 4, pp.576-582, 1986.

Gupta, M. et al, "Multivariable Structure of Fuzzy Control Systems", *IEEE Trans. Systems, Man, and Cybernet.*, Vol. SMC-16, No. 5, pp.638-656, 1986.

Hogle, P. A. and Bonissone, P., "A Fuzzy Algorithm for Path Selection in Autonomous Vehicle Navigation", *Proc. 23rd IEEE Conference on Decision and Control*, Las Vegas, 1984.

Holmblad, L. P. and Ostergaad, J., "Control of a Cement Kiln by Fuzzy Logic", *Fuzzy Information and Decision Process*, pp.389-399, 1982.

Hong. Z., "The Application of Fuzzy and Artificial Intelligence Method in the Building Up Blast Furnace Smelting Process Model", *Industrial Application of Fuzzy Control*, Sugeno, M. ed., North-Holland, 1985.

Hsu, C. S., "A Theory of Cell-to-Cell Mapping Dynamical Systems", *ASME J. of Applied Mechanics*, Vol.47, pp.931-939, 1980.

Hsu, C. S. and Guttalu, R. S., "An Unravelling Algorithm for Global Analysis of Dynamical Systems: An Application of Cell-to-Cell Mappings", *ASME J. of Applied Mechanics*, Vol.47, pp.940-948, 1980.

Hsu, C. S., "A Generalized Theory of Cell-to-Cell Mapping for Nonlinear Dynamical Systems", *ASME J. of Applied Mechanics*, Vol.48, pp.634-642, 1981.

Hsu, C. S., Guttalu, R. S. and Zhu, W. H., "A Method of Analyzing Generalized Cell Mappings", *ASME J. of Applied Mechanics*, Vol.49, pp.885-894, 1982.

Hsu, C. S., "A Probabilistic Theory of Nonlinear Dynamical Systems Based on the Cell State Space Concept", *ASME J. of Applied Mechanics*, Vol.49, pp.895-902, 1982.

Hsu, C. S., "A Discrete Method of Optimal Control Based upon the Cell State Space Concept", *J. of Optimization Theory and Applications*, Vol. 46, No. 4, 1985.

Hsu, C. S. and Chiu, H. M., "A Cell Mapping Method for Nonlinear Deterministic and Stochastic Systems-Part I: The Method of Analysis", *ASME J. of Applied Mechanics*, 1986.

Kailath, T., "Linear Systems", *Prentice-Hall*, 1980.

Kandel, A., "Fuzzy Mathematical Techniques with Applications", *Addison-Wesley*, 1986.

Kasai, Y. and Morimoto, Y., "Electrically Controlled Continuously Variable Transmission (ECVT-II)", *Proc. of the Int. Congress on Transportation Electronics*, Dearborn, 1988.

Kickert, W. and Van Nauta Lemke H. R., "The Application of Fuzzy Set Theory to Control a Warm Water Process", *Automatica*, Vol. 12, No. 4, pp.301-308, 1976.

Kickert, W. and Mamdani, E., "Analysis of a fuzzy logic controller", *Fuzzy Sets and Systems*, Vol. 1, pp.29-44, 1978.

King, P. J. and Mamdani, E. H., "The Application of Fuzzy Control System to Industrial Processes, *Automatica*, Vol. 13, No. 3, pp.235-242, 1977.

King, P. J., "Fuzzy Logic Control of a Cement Kiln Precalciner Flash Furnace", *Proc. of the IEEE Conference on Applications of Adaptive and Multivariable Control*, Hull, U. K., 1982.

Kiszka, J., Gupta, M. and Nikiforuk, P., "Energetistic stability of fuzzy dynamic systems", *IEEE Systems, man and Cybernetics*, Vol. SMC-15, No. 6, pp. 783-792, 1985.

Kiszka, J., Gupta M. and Nikiforuk P., "Energetistic Stability of Fuzzy Dynamic Systems", *IEEE Trans. Syst., Man, Cybern.*, Vol. SMC-15, No. 6, pp.783-792, 1985.

Kokawa, M., "Heuristic Approach to Pump Operations using Multi-Valued Logic", *Fuzzy Information and Decision Process*, Gupta, M. et al., North-Holland, 1982.

Kumar, P. R. and Varaiya P., "Stochastic Systems: Estimation, Identification, and Adaptive Control", *Prentice-Hall*, 1986.

Kumar S. R. and Majumder, D. D., "Fuzzy Logic Control of a Nonlinear Multivariable Steam Generating Unit using Decoupling Theory", *IEEE Trans. Systems, Man, and Cybernet.*, Vol. SMC-15, No. 4, pp.539-558, 1985.

Larkin, L. I., "A Fuzzy Logic Controller for Aircraft Flight Control", *Proc. of the 23rd Conference of Decision and Control*, Las Vegas, 1984.

Mamdani, E. H., "Applications of Fuzzy Algorithms for Control of Simple Dynamic Plant", *Proc. IEE*, Vol. 121, No. 12, pp.1585-1588, 1974.

Mamdani, E. H. and Assilian, S., "An Experiment in Linguistic Systhesis with a Fuzzy Logic Controller", *Int. J. Man-Machine Studies*, Vol. 7, No. 1, pp.1-13, 1975.

Mizumoto, M. "Note on the Arithmetic Rule by Zadeh for Fuzzy Conditional Inference", *Cybernetics and Systems*, Vol. 12, pp.247-306, 1981.

Murakami, S., "Application of Fuzzy Controller to Automobile Speed Control System", *Proc. IFAC Symp. on Information, Knowledge Representation and Decision Analysis*, Marseille, 1983.

Murayama, Y. et al., "Optimizing Control of Diesel Engine", *Proc. of the 1st Int. Conference on Fuzzy Information Processing*, Hawaii, 1984.

Ostergaard, J. J., "Fuzzy Logic Control of a Heat Exchanger Process", *Fuzzy Automata and Decision Process*, Gupta, M. M. et al., North-Holland, 1977.

Pappis, C. P. and Mamdani, E. H., "A Fuzzy Logic Controller for a Traffic Junction", *IEEE Trans. Systems, Man and Cybernet.*, Vol. SMC-7, No. 10, pp.707-717, 1977.

Poicare, H., "Sur les courbes definies par les equations differentielles", *J. Math. Pure Appl.*, Series 3, Vol. 7, pp.275-422, 1881.

Procyk, T. J. and Mamdani, E. H., "A Linguistic Self-Organizing Process Controller", *Automatica*, Vol. 15, No. 1, pp.15-30, 1979.

Rutherford, D. A., a Carter, G. A., "A Heuristic Adaptive Controller for a Sinter Plant", *Proc. of the 2nd IFAC Symposium on Automation in Mining, Mineral and Metal Processing*, Johannesburg, R.S.A., 1976.

Sakai, Y. and Ohkusa, K., "A Fuzzy Controller in Turning Process Automation", *Industrial Application of Fuzzy Control*, Sugeno, M. ed., North-Holland, 1985.

Scharf, E. M., "The Application of a Fuzzy Controller to the Control of a Multi-Degree-of-Freedom Robot Arm", *Industrial Application of Fuzzy Control*, Sugeno, M. ed., North-Holland, 1985.

Sinha, N. K. and Wright, J. D., "Application of Fuzzy Control to a Heat Exchanger Process", *Proc. of 16th IEEE Conf. on Decision and Control*, New Orleans, 1977.

Smale, S., "Differentiable Dynamical Systems", *Bulletin of the American Mathematical Society*, Vol. 73, pp.747-817, 1967.

Sugeno, M. and Takagi, T., "A New Approach to Design of Fuzzy Controller", *Advances in Fuzzy Sets Possibility Theory and Applications*, Wang. P. P. et al., Plenum, New York, 1983. Sciences, Vol. 36, pp.59-83, 1985.

Sugeno, M., "An introductory survey of fuzzy control", *Information Sciences*, Vol. 36, pp.59-83, 1985.

Sugeno, M. and Kang, G.T., "Fuzzy Modeling and Control of Multilayer Incinerator", *Fuzzy Sets and Systems*, Vol. 18, pp.329-346, 1986.

Sugeno, M., "An Introductory Survey of Fuzzy Control", *Information Sciences*, Vol. 36, pp.59-83, 1985.

Takagi, T. and Sugeno, M., "Derivation of Fuzzy Control Rules from Human Operator's Control Actions", *Proc. of the IFAC Symposium on Fuzzy Information, Knowledge Representation and Decision Analysis*, Marseille, 1983.

Takagi, T. and Sugeno, M., "Fuzzy Identification of Systems and its Applications to Modeling and Control", *IEEE Trans. Systems, Man, and Cybernet.*, Vol. SMC-15, No. 1, pp.116-132, 1985.

Tomason, M. G., "Convergence of Powers of a Fuzzy Matrix", *J. of Mathematical Analysis and Applications*, Vol. 57, pp.476-480, 1977.

Tong, R. M., "Synthesis of Fuzzy Models for Industrial Process", *Int. J. Gen. Systems*, Vol. 4, pp.143-162, 1978.

Tong, R. M., "Analysis and Control of Fuzzy Systems using Finite Discrete Relations", *Int. J. Control*, Vol. 27, No. 3, pp.431-440, 1978.

Tong, R. M., "Some Properties of Fuzzy Feedback Systems", *IEEE Trans. Syst., Man, Cybern.*, Vol. SMC-10, pp.327-330, 1980.

Tong, R. M., "A Retrospective View of Fuzzy Control Systems", *Fuzzy Sets and Systems*, Vol. 14, pp.199-210, 1984.

Umbers, I. G. and King, P. J., " An Analysis of Human-Decision Making in Cement Kiln Control and the Implication for Automation", *Int. J. Man-Machine Studies*, Vol. 12, No. 1, pp.11-23, 1980.

Uragami, M., Mizumoto, M., and Tanaka, K., "Fuzzy Robot Controls", *J. of Cybernet.*, Vol. 6, No. 1-2, pp.39-64, 1976.

Vidyasagar, M., Nonlinear Systems Analysis, *Prentice-Hall*, 1978.

Willaeys, D., Mangin P., and Malvache, N., "Use of Fuzzy Sets for Systems Modeling and Control, Application to the Speed Control of a Strongly Perturbed Motor", *Proc. of the 5th IFAC/IFIP Int. Conference on Digital Computer Applications to Process Control*, The Hague, 1977.

Yasunobu, S. and Miyamoto, S., "Automatic Train Operation System by Predictive Fuzzy Control", *Industrial Application of Fuzzy Control*, pp.1-18, 1985.

Yonekura, M., "The Application of Fuzzy Sets Theory to the Temperature of Box Annealing Furnaces using Simulation Techniques", *Proc. of the 8th IFAC World Congress*, Tokyo, 1981.

Zadeh, L. A., "Fuzzy Sets", *Information and Control*, pp.338-353, 1965.

Zadeh, L. A., "A Rationale of Fuzzy Control", *J. of Dynamic Systems, Measurement and Control*, 94, Series G, pp.3-4, 1972.

Zadeh, L. A., "Outline of a New Approach to the Analysis of Complex Systems and Decision Process", *IEEE Trans. Syst., Man, Cybern.*, Vol. SMC-3, pp.28-44, 1973.

Zimmermann, H. -J., Fuzzy Set Theory and its Application, *Kluwer-Nijhoff Publishing*, 1985.