A Dissertation Submitted to the University of Fukui for the Degree of

Doctor of Engineering

福井大学審査

学位論文　**[博士（工学）]**

Fuzzy Logic Controllers: Optimization Issues on Design and Rule Base Reduction Algorithms

(ファジィロジックコントローラ：設計上の最適化問題とルールベースの削減アルゴリズム)

September, 2013

Pintu Chandra Shill

This is to certify that the thesis entitled “Fuzzy Logic Controllers: Optimization Issues on Design and Rule Base Reduction Algorithms” is approved for the degree of Doctor of Engineering.

_____

Prof. Hiroshi Ikeda, Member

_____

Prof. Tomohide Naniwa, Member

_____

Prof. Kazuyuki Murase, Supervisor

_____

Prof. kouki Nagamune, Member

_____

Prof. Yasutake Takahashi, Member

ABSTRACT

The majority of Fuzzy Logic Controllers (FLCs) to date are working on the basis of expert knowledge derived from heuristic knowledge of experienced operators. Traditional fuzzy logic controller has inferior adaptability due to invariable membership functions (MFs) parameters and fixed type rule set. These manual coded traditional FLCs use only expert knowledge base. Therefore, they do not perform well in complex problems, especially when there are a large number of input variables. Therefore, an optimization tool is highly desirable to automatically build the fuzzy knowledge base that maximizes the controller performance. In this thesis, we first develop a multi-criteria optimization tool using new hybrid genetic algorithm (HGAs) for automatic fuzzy knowledge base (FKB) acquisition. Our approach consists of two phases: first phase involves selection and definition of fuzzy control rules as well as adjustment of membership functions parameters, while the second phase performs an optimal selection of membership function types corresponding to fuzzy control rules. Learning both parts concurrently represents a way to improve the accuracy of the FLCs to minimize the errors. Furthermore, sensitivity of the FLC has been analyzed and compared for various types of membership functions. It has been argued that the performance of FLCs greatly depends on the parameters as well as types of membership functions. Thus, the aforementioned HGAs are a viable solution for designing an efficient adaptive FLCs system.

On the other hand, a serious problem limiting the applicability of standard fuzzy controllers is the rule-explosion problem; that is, the number of rules increases exponentially with the number of input variables to the fuzzy controller. We propose automatic design methods with rule base size reduction for fuzzy logic controllers (FLCs) through genetic algorithms (GAs). This is done by optimizing the FLCs membership functions and automatically generates the fuzzy rules and at the same time applies the genetic reduction technique to determine the minimum number of fuzzy rules required in building the fuzzy models. In the rule base reduction process, the redundant rules are removed by setting their all consequent weight factor to zero and merging the conflicting rules during the learning process. Optimizing the FLCs MFs with learning and reducing rule base concurrently represents a way to maximize the performance of FLCs. The control algorithm is successfully tested for intelligent control of two degrees of freedom inverted pendulum.

Moreover, an efficient tool to deal with the ‘rule explosion’ problem is the hierarchical system by which a fuzzy system can be decomposed into a number of hierarchically connected low-dimensional systems. In this thesis, we also propose an automatic way of evolving hierarchical fuzzy logic controller. Firstly, genetic programming (GP) is employed to identify the optimum and appropriate hierarchical architecture and secondly, genetic algorithm (GA) employed to tune the fuzzy sub controllers (SC) involve in hierarchical controllers. The proposed control algorithms

coupled both GP and GA optimizations. The automatically generated hierarchical fuzzy controller consists of a number of low-dimensional fuzzy systems in a hierarchical form.

Interval type-2 fuzzy sets are able to model and minimize the numerical and linguistic uncertainties associated with the inputs and outputs of fuzzy logic controller (FLC). In the fourth part of the thesis, an adaptive interval type-2 FLC is proposed for trajectory control of mobile robot with unstructured dynamical uncertainties. Quantum genetic algorithm is employed to tune type-2 fuzzy sets and rule sets simultaneously for effective design of interval type-2 FLCs. Traditional fuzzy logic controllers (FLCs), often termed as type-1 FLCs using type-1 fuzzy sets, have difficulty in modeling and minimizing the effect of uncertainties present in many real time applications. Therefore, manually designed type-2 FLCs have been utilized in many control process due to their ability to model uncertainty and it relies on heuristic knowledge of experienced operators. The type-2 FLC can be considered as a collection of different embedded type-1 FLCs. However, manually designing the rule set and interval type-2 fuzzy set for an interval type-2 FLC to give a good response is a difficult task. The purpose of our study is to make the design process fully automatic for building a type-2 FLCs. The type-2 FLCs exhibit better performance for compensating the large amount of uncertainties with severe nonlinearities.

Finally, the above proposed control algorithms are successfully tested and evaluated for intelligent control of well known benchmark applications namely backing up a truck, trailer-and-cab, and trajectory control of nonholonomic robots and two degrees of freedom inverted pendulum. The simulation studies exhibit competing results with high accuracy that demonstrates the effective use of the proposed control algorithms.

I dedicate this dissertation

to my parents,

wife and

my daughter

ACKNOWLEDGEMENT

In my thesis, I have done research work related to soft computing technique and this could not have been possible without the kind assistance of some people to whom I would like to express my gratitude. First and foremost, I thank the almighty for giving me the strength and ability to complete this study.

Firstly, I express deep sense of appreciation, reverence, and sincere gratitude to my supervisor Professor Kazuyuki Murase, department of System Design Engineering, University of Fukui, Japan, for introducing me this promising research area. I am indebted to him for his unlimited encouragement, proper guidance, help, invaluable advice, and support throughout the duration of the thesis. Special thanks to you very much.

Secondly, I would like to give special thanks Professor Yochiro Maeda, Osaka Institute of Technology, Japan, for giving valuable comments to encourage me about this thesis before starting this study. Special thanks to Md Monirul Islam, Muhammad Aminul Haque Akhand and Md. Faijul Amin for their kind assistant, proper guidance and encouragement throughout this study. Thanks to you very much.

Thirdly, I am grateful to the Japanese Government for the financial support I received from the Ministry of Education, Culture, Sports, Science and Technology (MEXT). I would like to give thanks all of researcher in the field of soft computing and robotics whose research and journal papers help me for completing my research work. Thanks to all very much.

Fourthly, I would like to say endless thanks my parents for providing support and the occasional welcome distraction while I worked on this thesis. I am greatly indebted to my wife, sisters and brothers also for their encouragement, supports and sacrifices throughout my life.

Last but not least, I would like to give thanks to all of my laboratory members and friends at University of Fukui for their spontaneous support in validating and discussing the ideas presented in this research work. I wish to express my gratitude to all those who have one way or another helped me in making this study a success.

September, 2013
Shill Pintu Chandra

TABLE OF CONTENTS

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Preamble

Nowadays fuzzy logic controllers (FLCs) are increasingly used in numerous practical applications in control [1-3], prediction [4], classification [5], inference [6], and decision making [7]-[10]. Specifically, they have been used for designing a robust controller that can yield satisfactory performance and deal with uncertainty and imprecision. FLCs works based on the fuzzy knowledge base (FKB) but the generation of the FKB of a fuzzy rule-based system (FRBS) presents several difficulties because the FKB depends on the concrete application, and this makes the accuracy of the FRBS directly depend on its composition. Many approaches have been proposed to automatically learn the FKB from numerical information. Most of them have focused on the rule base (RB) learning, using a predefined data base (DB) [11]-[13]. Several approaches adjust the membership function definitions and do not modify the number of linguistic terms in each fuzzy partition since the RB remains unchanged [14]. For this reason, an optimization tool is highly desirable to tune FKB components concurrently.

In fuzzy control the number of fuzzy rules required increases in an exponential manner and this phenomenon is known as the curse of dimensionality or combinatorial rule explosion problem. Then a reduction of the rule base becomes necessary. It is thought that dense rule bases should be reduced, so that only the minimal necessary number of rules remains, still containing the essential information in the original base. Although several techniques are used for this purpose, such as: the boolean method [15], neural networks [16], the decoupling approach [17], and clustering methods [18] but none of them fully satisfactory. This problem is avoided here by the introduction of a new, evolutionary based method for synthesizing the fuzzy control.

On the other hand, type-2 fuzzy sets and fuzzy logic controller have been used in classification of coded video streams [19], co-channel interference elimination from nonlinear time-varying communication channels[20], connection admission control[21], control of mobile robots[22], decision making[23][24], equalization of nonlinear fading channels[25]-[28], learning linguistic membership grades[29], pre-processing of data in medical diagnosis [30]-[32], quality control[33], relational database[34], survey processing[21][35]-[37], time series forecasting[38], and transport scheduling[39]. The authors [40] applied GAs to search for optimal uncertain means and its extent of interval type-2 gaussian MFs for the chaotic time series prediction. Although many reasonable results have been obtained by using GAs, the discussion of stable and optimal learning of type-2 fuzzy sets with rule base has not been established in type-2 FLCs (T2FLCs).

1.2 Contributions of the thesis

Indeed, the contributions brought together in this thesis can be divided into four main parts. The first part deals with the evolutionary tuning and learning of fuzzy knowledge base (FKB), second part deals with a rule base reduction and tuning algorithm is proposed as a design and optimization

tool for the knowledge-based fuzzy control of a nonlinear system, third part deals with the automatic designing technique of hierarchical fuzzy system and four part deals with the adaptive type-2 fuzzy logic system.

1) Firstly, we propose a method for designing fuzzy logic controllers (FLCs) based on hybrid genetic algorithms (HGAs) with a view to make the design process fully automatic, without requiring any human expert and numerical data. Our approach consists of two phases: first phase involves selection and definition of fuzzy control rules as well as adjustment of membership functions parameters, while the second phase performs an optimal selection of membership function types corresponding to fuzzy control rules. Learning both parts concurrently represents a way to improve the accuracy of the FLCs to minimize the errors. It has been argued that the performance of FLCs greatly depends on the parameters as well as types of membership functions. Thus, the aforementioned HGAs are a viable solution for designing an efficient adaptive FLCs system. In order to evaluate the effectiveness of the proposed method for optimal design of the FLCs, the proposed approach is applied to a well-known benchmark controller design task, car like robot system. Simulation results illustrates that proposed optimization approach can find optimal fuzzy rules and their corresponding membership functions types with a high rate of accuracy. The new HGAs optimized adaptive FLCs outperforms not only a passive control strategy but also human-designed FLCs, a neural coded controller with clustering and a neural fuzzy control algorithm.

2) Secondly, rule base size reduction and tuning algorithm is proposed as a design tool for fuzzy logic controllers (FLCs) through real and binary coded coupled genetic algorithms (GAs). The adaptive schema is divided into two phases: the first phase is concerned with optimizing the FLCs membership functions and learning and reducing phase, automatically generate the fuzzy rules and at the same time applies the genetic reduction technique to determine the minimum number of fuzzy rules required in building the fuzzy models. In the second phase, the redundant rules are removed by setting their all consequent weight factor to zero and merging the conflicting rules during the learning process. The first and second phases are carried out by the real and binary coded coupled GAs. Optimizing the FLCs MFs with Learning and reducing rule base concurrently represents a way to maximize the performance of a FLCs. The control algorithm is successfully tested for intelligent control of two degrees of freedom inverted pendulum. Finally, the simulation studies exhibit competing results with high accuracy that demonstrates the effective use of the proposed algorithm.

3) Thirdly, we propose an automatic way of evolving hierarchical fuzzy Logic controller which is a combination of cascaded fuzzy modules. The control algorithm works in a two steps process simultaneously: Firstly, genetic programming (GP) is employed to identify the optimum hierarchical architecture and secondly, genetic algorithm (GA) employed to tune the fuzzy sub controllers (SC) involve in hierarchical controllers. The proposed control algorithms coupled both GP and GA optimizations. The fine tuning of the antecedent and consequent parameters of fuzzy control rules and their corresponding membership functions (MFs) of SCs encoded in the structure is accomplished using genetic algorithms (GAs). In this method, the total number of rules increases only linearly with the number of input variables. The automatically generated hierarchical fuzzy controller consists of a number of low-dimensional fuzzy systems in a hierarchical form. The proposed hierarchical control algorithm is evaluated using well known benchmark applications namely trailer-and-cab, a nonlinear motion control of nonholonomic robots. Simulation results are

presented at different operating conditions and under various disturbances to verify the effectiveness of developed adaptive small size hierarchical controller.

4) Fourthly, a type-2 Fuzzy logic controller adapted with quantum genetic algorithm, referred to as type-2 quantum fuzzy logic controller (T2QFLC), is presented in this article for trajectory tracking of mobile robot with unstructured dynamical uncertainties. Quantum genetic algorithm is employed to tune type-2 fuzzy sets and rule sets simultaneously for effective design of interval type-2 FLCs. Traditional fuzzy logic controllers (FLCs), often termed as type-1 FLCs using type-1 fuzzy sets, have difficulty in modeling and minimizing the effect of uncertainties present in many real time applications. Therefore, manually designed type-2 FLCs have been utilized in many control process due to their ability to model uncertainty and it relies on heuristic knowledge of experienced operators. The type-2 FLC can be considered as a collection of different embedded type-1 FLCs. However, manually designing the rule set and interval type-2 fuzzy set for an interval type-2 FLC to give a good response is a difficult task. The purpose of our study is to make the design process automatic. The type-2 FLCs exhibit better performance for compensating the large amount of uncertainties with severe nonlinearities. Furthermore, the adaptive type-2 FLC is validated through a set of numerical experiments and compared with QGA evolved type-1 FLCs, traditional and neural type-1 FLCs.

1.3 Thesis Structures

This thesis is organized as follows: Chapter 2 reviews literature relevant to type-1 and type-2 fuzzy logic system with their fuzzifier, inference engine, defuzzifier and type reducer with examples. This chapter also describes how various types of membership functions effects on inference engine performance. A fuzzy knowledge base (FKB) optimization model with new hybrid genetic algorithms will be defined in chapter 3. Chapter 4 discusses an evolutionary fuzzy rule base reduction technique with learning of membership function. Chapter 5 describes a novel method of generating hierarchical fuzzy system. Chapter 6 describes new methods for building intelligent systems using type-2 fuzzy logic and evolutionary computation. Chapter 7 presents conclusion and guidelines about future research works.

CHAPTER 2

FUZZY LOGIC CONTROLLER

2.1 Introduction

Type-1 FLCs will be first reviewed with its fuzzifier, inference engine, rule base and defuzzifier. An example of two rules type-1 FLCs will show different defuzzification results via various type MFs. Then general type-2 FLCs will be described more detail in similar way.

2.2 Type-1 Fuzzy Logic Controllers

FLCs comprise four principal components, which are fuzzifier, FKB, fuzzy inference engine, and defuzzifier as depicted in Fig.1. The FLCs works as follows: the crisp inputs are first fuzzified into, in general, input fuzzy sets (however, we will consider only singleton fuzzification) which then activate the inference engine and the FKB to produce output fuzzy sets. The defuzzifier can then defuzzify fuzzy outputs from the inference engine in order to produce crisp outputs. In the following subsection, each of the four principal components will be described in detail to show how the fuzzy mathematical and logic principles are used in FLCs.

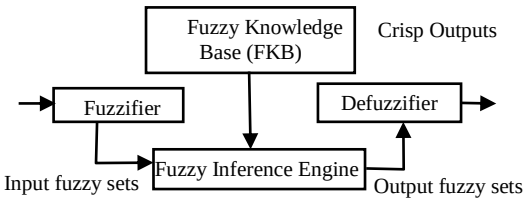


Fig. 2.1. Type-1 Fuzzy Logic Controller

2.2.1 Fuzzifier

The fuzzifier maps a crisp input vector with p inputs $x = (x_1, \ldots . x_p)^T \in X_1 \times X_2 \times \ldots \ldots \times X_p \equiv X$ into input fuzzy sets, A_x. However, we will use the most frequently used singleton fuzzification method as it is fast to compute and thus suitable for real time operation. Input fuzzy sets have only a single point of nonzero membership in the singleton fuzzification[36].

2.2.2 Fuzzy Inference Engine

The fuzzy inference engine combines rules and gives a mapping from fuzzy sets in the input universe of discourse $U \in R^n$ to fuzzy sets in the output universe of discourse $V \in R$ based on the fuzzy logic principle. In the inference engine, multiple antecedents in the rules are connected using AND operation, and the degree of membership in the input sets are combined using those in the output sets using sub-star composition, described in detail in [41] . Multiple rules are combined then by using a join operation.

2.2.3 Defuzzification Process

Finally, the defuzzification process is used to transfer fuzzy sets into a crisp value, y_c. There are several defuzzifier methods in the literature. Centre of gravity (COG) is one of the most popular simple methods for defuzzification process. One of the important advantages of the COG method is that all activated membership functions of the consequents (all active rules) take part in the defuzzification process [42]. The COG method works based on the following equation for transferring fuzzy scheme into a crisp value [43]:

$$Z_c = \frac{\int_y \mu_A(z) z dz}{\int_y \mu_A(z) dz} \qquad (2.1)$$

where z and $\mu_A(z)$ are a output fuzzy variable and its membership function, calculated from equation (2.7) and equation (2.8), due to the consequent fuzzy rules, respectively. $\mu_A(z)$ value greatly depends on the shape, type and distribution of MFs. Based on the above definition, it can be shown that the controller output depends on the type of input output fuzzy sets, i.e., the MFs form (shape and type) of fuzzy sets.

There are many existing defuzzifiers so far. For engineering applications, the criterion for the choice of a defuzzifier is computational simplicity, such as maximum, centroid, center-of-sums, center average (also called the height defuzzifier [44][45]), modified height, and center of sets. Some major defuzzifiers are briefly overviewed as:

2.2.3.1 Height Defuzzifier

Height defuzzifier[46] is most popular used in type-1 fuzzy set as defined:

$$y_h = \frac{\sum_{l=1}^{M} \bar{y}^l \mu_{B^l}(\bar{y}^l)}{\sum_{l=1}^{M} \mu_{B^l}(\bar{y}^l)} \qquad (2.2)$$

Where l is rule number and $\bar{y}^l$ is the point having maximum membership in the l^{th} output fuzzy set μ_{B^l}, and its membership grade in the l^{th} output fuzzy set is $\mu_{B^l}(\bar{y}^l)$ that can be expressed as

$$\mu_{B^l}(\bar{y}^l) = \mu_{G^l}(\bar{y}^l) * \left[\mu^l_{F_1}(x_1^{'}) * \ldots . * \mu^l_{Fn}(x_n^{'})\right] \qquad (2.3)$$

where $\mu_{G^l}(\bar{y}^l$ is the value of MFs grade of consequent set, $*$ is the t-norm, which is normally the product or minimum. Hence, only minimum t-norm is considered for height defuzzifier. However it does not consider the entire shape of the consequent MF, i.e., it only uses the center of the support, $\bar{y}^l$, of the consequent MF. So the modified height defuzzifier is proposed to improve the height defuzzifier.

2.2.3.2 Modified Height Defuzzifier

Under the framework of type-1 fuzzy set, the way to handle consequent uncertainty using the modified height defuzzification with spread measure (δ^l) was discussed in [44][47]. To handle uncertainty, it is a bit artificial but this is the only way in type-1 fuzzy logic systems.

$$y_{mh} = \frac{\sum_{l=1}^{M} \bar{y}^l \mu_{B^l}(\bar{y}^l) / \delta^{l^2}}{\sum_{l=1}^{M} \mu_{B^l}(\bar{y}^l) / \delta^{l^2}} \tag{2.4}$$

Where δ^l is the spread measure/standard deviation of the *l*th consequent set. For triangular and trapezoidal membership functions, δ^l could be the length of its base, whereas for Gaussian membership functions, δ^l could be the length of its base, whereas for Gaussian membership functions, δ^l could be its standard deviation.

2.2.3.3 Centroid Defuzzifier

Firstly centroid fuzzifier [48] must combine all the output type-1 fuzzy sets using union (t-conorm), $B = \bigcup_{l=1}^{M} B^l$, and then find the centroid of this set as

$$y_c(x) = \frac{\sum_{i=1}^{N} y_i \mu_B(y_i)}{\sum_{i=1}^{N} \mu_B(y_i)} \tag{2.5}$$

Where the output set is discretized into N points. Obviously the $y_c(\vec{x})$ is a function of $\vec{x}$ because $\mu_B(y_i)$ is also a function of FLS input $\vec{x}$. Due to computing the union of all output sets, usually it becomes more difficult to implement than above two fuzzifiers.

2.2.3.4 Center of Sets Defuzzifier

For center-of-Sets defuzzifier[48][36], each rule consequent set is replaced by its singleton centroid c^l whose amplitude equals the firing strength. The output can be expressed as

$$y_{\cos}(x) = \frac{\sum_{l=1}^{M} c^l T_{i=1}^{n} \mu_{F_i^l}(x_i)}{\sum_{l=1}^{M} T_{i=1}^{n} \mu_{F_i^l}(x_i)} \tag{2.6}$$

where $T_{i=1}^{n} \mu_{F_i^l}(x_i)$ is the degree of membership value of output fuzzy set where the input are i=1 to n. If each consequent is symmetric, normal, and convex, then $c^l = \bar{y}^l$ and $\mu_{G^l}(\bar{y}^l) = 1$ for singleton case; the results leads to $y_{\cos}(\vec{x}) = y_h(\vec{x})$. However, the consequents are not all symmetric that the center-of-sets defuzzifier will derive more appropriate results of defuzzification than height defuzzifier.

2.2.3.5 Example 2.1

Example 2.1: A type-1 FLC has 3 rules whose antecedents and consequents MFs are shown in Figure 2.2 (a)-(c). Each rule has two antecedents in Figure (a)-1 (a)-2, (b)-1 (b)-2, and c(1) (c)-2, respectively. Suppose that for the particular inputs $x = 3$ and $x = 5$, by using min t-norm for firing

strength selection and product t-norm for inference, the fired output MFs are shown in Figure 2.2 (a)-3, (b)-3 and (c)-3. Then using t-conorm all outputs are combined in Figure 2.2 (d). The numbers 0.9, 0.8, 0.2 indicate the firing strength of each of consequent sets. The output of four defuzzifiers, i.e. y_h, y_{mh}, y_c and $y_{\cos}$, for this example are listed in Figure 2.2(d). For the modified height defuzzifier δ^l is set equal to the standard deviation of the 1th consequent set, i.e. 0.4, 0.2, and 0.2 from left to right in Figure 2.2 (d), respectively. Note that for the 3rd consequent set centered at 5, height $\bar{y}^3 = 5$ but centroid $c^3 = 4.8436$; hence, the outputs of the height and center of sets defuzzifiers are slightly different.

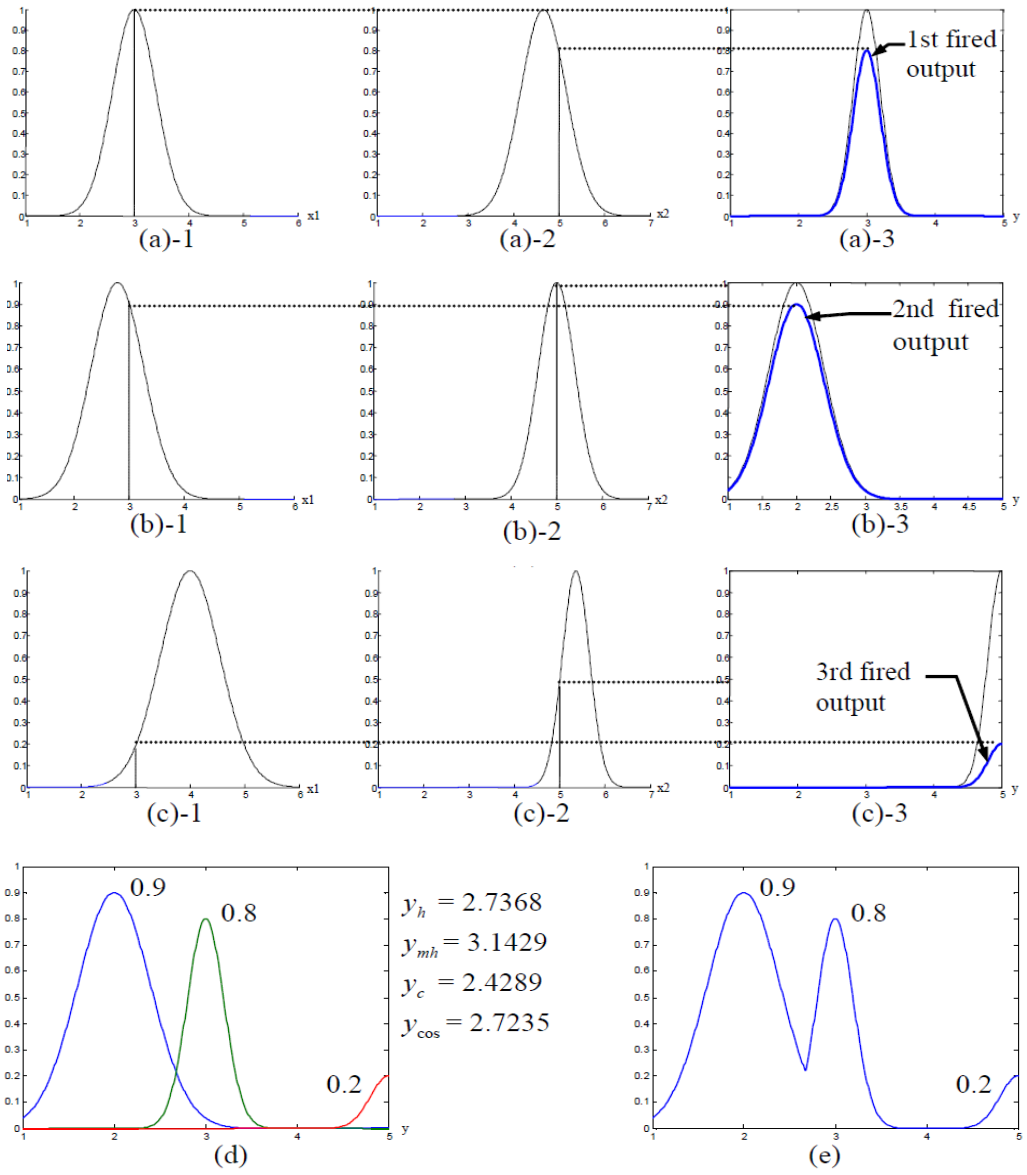


Fig. 2.2. two antecedents of the first rule are shown in (a)-1 and (a)-2 and the consequents (upper solid curve) with its fired output are shown in (a)-3. The two antecedents of second rule are shown in (b)-1 and (b)-2, and the consequents with its fired output are shown in (b)-3. The two antecedents of third rule are shown in (c)-1 and (c)-2, and the consequents with its fired output are shown in (c)-3. Figure (d) shows all fired outputs together. To compute y_c, first it need to combine all outputs as shown in (e).

Details computing for height defuzzifier y_h and modified height defuzzifier y_{mh} are described as bellows:

$$y_h = \frac{0.9 \times 2 + 0.8 \times 3 + 0.2 \times 5}{0.9 + 0.8 + 0.2} = 2.7368$$

$$y_{mh} = \frac{0.9 \times 2 / 0.4^2 + 0.8 \times 3 / 0.2^2 + 0.2 \times 5 / 0.2^2}{0.9 / 0.4^2 + 0.8 / 0.2^2 + 0.2 / 0.2^2} = 3.1429$$

To compute y_c, we need first combine all the output type-1 fuzzy sets shown in Figure 2.2 (e) and discretized into 20 points, then using (2.11) to derive the defuzzified result: $y_c = 2.4289$

To compute $y_{\cos}$, each rule consequent in Figure 2.2 (d) can have its centroid c^l, i.e., $c^1 = 2.0068$, $c^2 = 3$ and $c^3 = 4.8436$, then using (2.12) to derive $y_{\cos}$ as below:

$$y_{\cos} = \frac{0.9 \times 2.0068 + 0.8 \times 3 + 0.2 \times 4.8436}{0.9 + 0.8 + 0.2} = 2.7235$$

2.2.4 Fuzzy Inference Using Different Types of MFs

In this section, we describe a fuzzy inference system with respect to the different type, shape and distribution of MFs. How different MFs types can change the fuzzified and defuzzified value is also discussed in this section. The following (Fig. 2.3) simple numerical example illustrates that the controller output varies according to the type, shape and distribution of MFs.

2.2.4.1 Different Types of MFs

In this section two types of MFs are considered for the convenience of the explanation. The most commonly used triangular and gaussian typed MFs are often used in the conventional fuzzy reasoning rules. They are formally described as follows:

$$F_j^i(x_j) = \begin{cases} 1 - \dfrac{2\left|x_j - a_{ji}\right|}{b_{ji}} & , a_{ji} - \dfrac{b_{ji}}{2} < x_j < a_{ji} + \dfrac{b_{ji}}{2} \\ 0 & , \text{otherwise} \end{cases} \tag{2.7}$$

$$F_j^i(x_j) = \exp\{-(x_j - a_{ji})^2 / b_{ji}\} \tag{2.8}$$

2.2.4.2 Fuzzy Inference w.r.t type, shape and distribution of MFs

Fuzzification of the fuzzy inference system is the process of changing a real value into a fuzzy value. Fuzzification of a real-valued variable is done with intuition, experience and analysis of the set of rules and their associated MFs, i.e fuzzy sets. The different type of MFs defined in equation (2.7) and (2.8) produces different fuzzified value for the same real value as shown in example (Fig. 2.3).

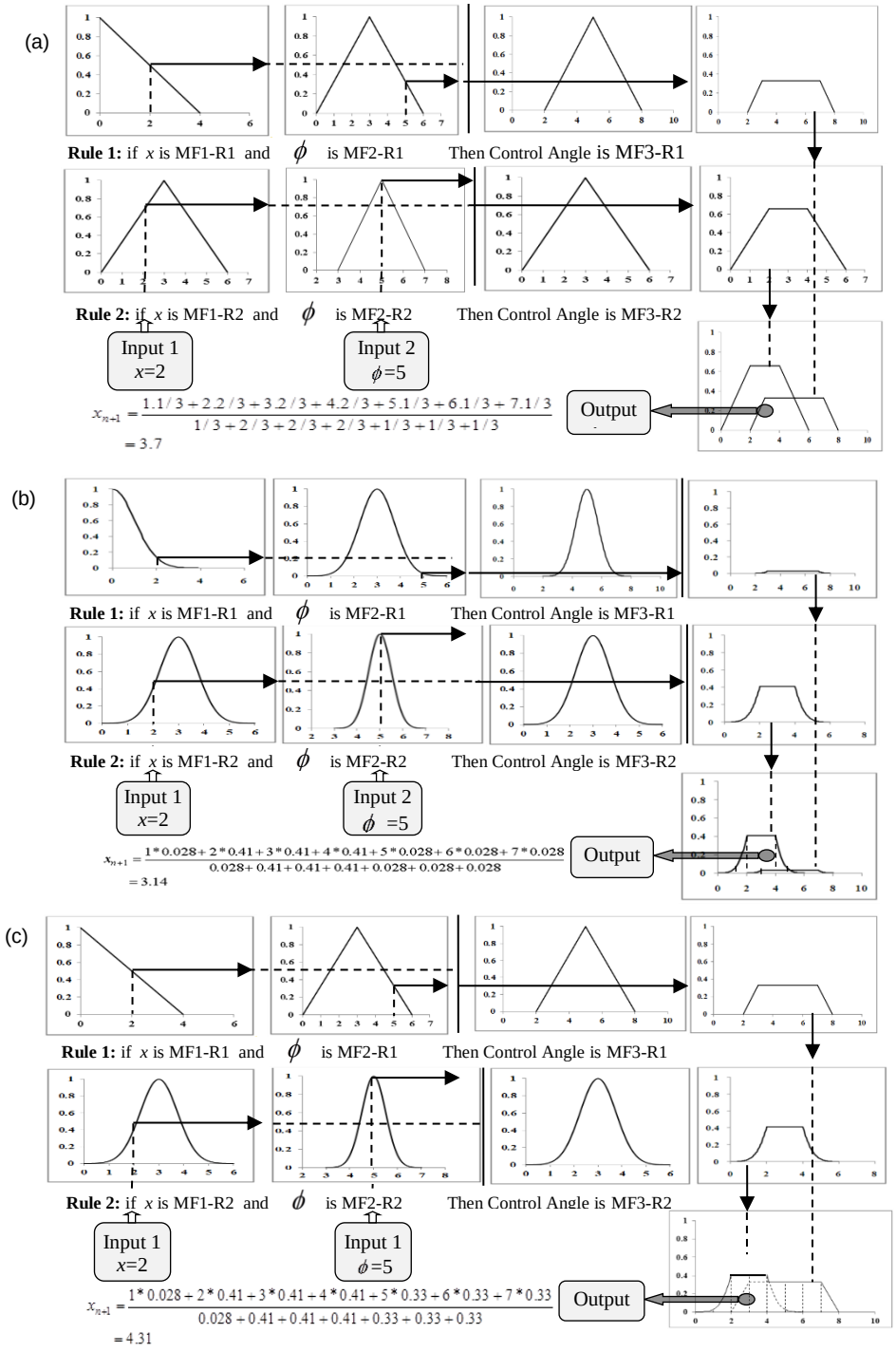


Fig. 2.3. Fuzzy inference system for representing rules by (a) Triangular MFs (b) Gaussian MFs (c) Mixed(one rule is triangular and another one gaussian) MFs

The example is considered for the controller output of two rules system with triangular (i.e., two rules MFs type are triangular), trapezoidal and mixed MFs (i.e., one rule MFs type is gaussian and another rule MFs is triangular).

The fuzzification of the input variable should be realistic. Experience and different procedures as well as the use of different MFs type should be followed while designing a large fuzzy system for the realistic and accurate output. The wrong fuzzification of the input variable(s) might cause instability and error in the system.

From the above example (Fig. 2.3) we have shown that fuzzified and defuzzified value vary according to the MFs type. So, optimal MFs type selection for the rule base is also a crucial issue while designing a controller. Fuzzification or defuzzification process parameters will change the final output of fuzzy controller. In this study, the effect of MFs type on fuzzification and defuzzification process to construct the fuzzy model and controller is also investigated (Chapter 3) as well as selection and definition of fuzzy rules.

2.3 Type-2 Fuzzy Sets and FLCs

The concept of a type-2 fuzzy set was introduced by Prof. Zadeh [49] in 1975. A type-2 fuzzy set is defined by MFs as shown in Fig. 2.4. The fuzzy grade of that is a fuzzy set in the closed interval [0, 1] rather than a point in [0, 1]. A type-2 fuzzy set, denoted as $\tilde{A}$, is characterized by a type-2 MF $\mu_{\tilde{A}}(x,u)$ [45], where $x \in X$ and $u \in J_X \subseteq [0,1]$, i.e.,

$\tilde{A} = \{((x,u), \mu_{\tilde{A}}(x,u)) \mid \forall_x \in X \quad \forall_u \in J_x \subseteq [0,1]\}$ in which $0 \le \mu_{\tilde{A}}(x,u) \le 1$. A can also be expressed as follows [24]:

$A = \int_{x \in X} \int_{u \in J_x} \mu_A(x,u)/(x,u) \quad J_x \in [0,1]$ where $\mu_A(x,u)$ is the secondary grade of MFs, $\iint$ denotes union over all admissible x and u. J_x is called primary membership of x, where $J_x \in [0,1]$ for $\forall_x \in X$ [24]. The uncertainty in the primary memberships of a type-2 fuzzy set $\tilde{A}$, consists of a bounded region that is called the footprint of uncertainty (FOU)[24]. It is the union of all primary memberships [50].

Now how to construct a type-2 fuzzy set. Let the upper membership function be y=x-2 and the lower membership function be y'=2x-5. When x_1=2.8, we get the values of these two functions as 0.6 and 0.8 respectively. So the interval J_{x_1} = [0.6, 0.8]. Then, correspondingly, the secondary membership function should be also in this interval.

2.3.1 Example 2.2

Example 2.2: Figure 2.5 shows a type-2 fuzzy membership function $\mu_{\tilde{A}}(x,u)$ with the discrete x and u. $X = \{1, 2, 3, 4, 5\}$ and $U = \{0, 0.2, 0.4, 0.6, 0.8\}$.

The secondary membership function

at $x = 1$ is $\mu_{\tilde{A}}(1) = 1/0 + 0.35/0.2 + 0.25/0.4 + 0.4/0.6 + 0.15/0.8$ and

at $x = 3$ is $\mu_{\tilde{A}}(1) = 0.25/0.6 + 0.4/0.8$.

The primary memberships are

$J_1 = J_2 = J_4 = J_5 = \{0, 0.2, 0.4, 0.6, 0.8\}$ and

$J_3 = \{0.6, 0.8\}$

The shaded area is FOU.

2.3.2 Type-2 FLC

A type-2 FLC comprises five components, which are fuzzifier, knowledge base (KB) consisting of rule base (RB) and database (DB), fuzzy inference engine, type- reducer and defuzzifier as depicted in Fig.2.6.

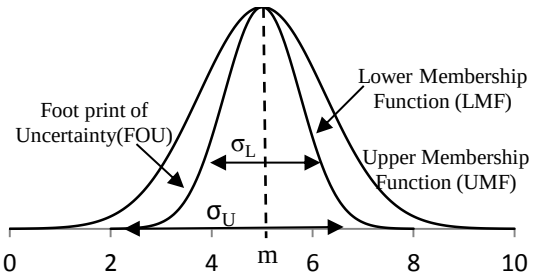


Fig. 2.4. An Interval Gaussian type-2 fuzzy set where σ_L and σ_U are minimum and maximum resultant widths respectively

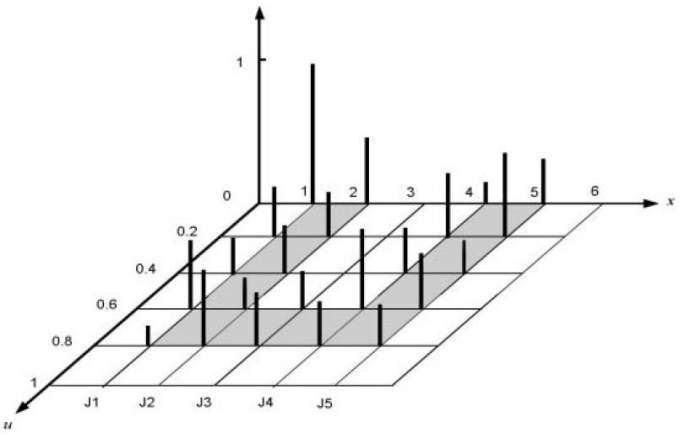


Fig. 2.5. Type-2 Membership Function with Primary and Secondary MFs Value

2.3.1.1 Fuzzifier

Since the input is in crisp normalized values, a fuzzification operator *fuzz* is used to fuzzify it in fuzzy form. The fuzzifier maps a crisp input vector with p inputs $x = (x_1, \ldots . x_p)^T \in X_1 \times X_2 \times \ldots \ldots \times X_p \equiv X$ into input fuzzy sets, $\tilde{A}_x$ [51][52]. However, we will use most frequently used singleton fuzzification method as it is fast to compute and thus it is suitable for mobile real time operation. In the singleton fuzzifier, fuzzy set $\tilde{A}$ has only a single point of non-zero membership with support x_i, where $\mu_{\tilde{A}}(x,u) = 1$ for $x = x_i$ and $\mu_{\tilde{A}}(x,u) = 0$ for $x \neq x_i$ which input measurement x is perfect crisp.

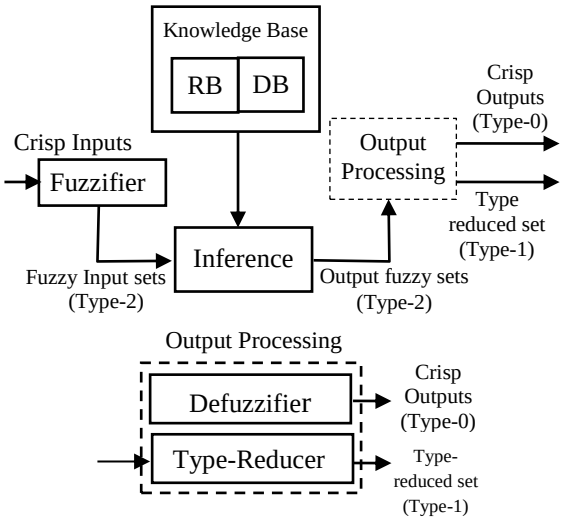


Fig. 2.6. Interval Type-2 FLCs

2.3.1.2 Rule base

The rules will remain the same as in type-1 FLCs but the antecedents and consequents will be represented by interval type-2 fuzzy sets [52]. Like most FLCs [53], the FLC discussed here applies the concepts of fuzzy implication and the compositional rules of inference for approximate reasoning. Suppose that we need to design a multiple-input-multiple-output (MIMO) mobile robot type-2 FLC having p inputs $x_1 \in X_1,.....,x_p \in X_p$ and c outputs $y_1 \in Y_1,.....y_c \in Y_c$ with ith fuzzy rule of the form:

$$R^i_{MIMO}: \text{IF } x_1 \text{ is } \tilde{F}^i_1 .....\text{and} x_p \text{ is } \tilde{F}^i_p, \quad \text{THEN}$$
$$y_1 \text{ is } \tilde{G}^i_1 .....y_c \text{ is } \tilde{G}^i_c, \quad i = 1,...,M \tag{2.9}$$

Where $\tilde{F}^i_1,.....\tilde{F}^i_p$ and $\tilde{G}^i_1,.....\tilde{G}^i_c$ are the antecedent and consequent MFs associated with the linguistic p input variables and c output variables, respectively, and M is the number of rules in the rule base.

2.3.1.3 Fuzzy Inference Engine

The fuzzy inference engine combines rules and gives a mapping from type-2 fuzzy sets in the input universe of discourse $U \in R^n$ to type-2 fuzzy sets in the output universe of discourse $V \in R$ based on the fuzzy logic principle. The structure of ith type-2 rule is having one output $y_k \in Y_k$:

$\tilde{R}^i: \tilde{F}^i_1 \times \tilde{F}^i_2 \times .....\times \tilde{F}^i_p \rightarrow \tilde{G}^i_k$ and the type-2 fuzzy relation can be expressed by membership function as:

$$\begin{aligned} \mu_{R^i}(\vec{x}, y) &= \mu_{F^i_1 \times F^i_2 \times ..... \times F^i_n \rightarrow G^i}(\vec{x}, y) \\ &= \mu_{F^i_1}(x_1) \prod .....\prod \mu_{F^i_n}(x_n) \prod \mu_{G^i}(y) \end{aligned} \tag{2.10}$$

Where $\prod$ denotes meet operation. The membership grades in the input type-2 fuzzy sets are combined with those in the output type-2 fuzzy sets using the extended sup–star composition; multiple rules are combined using the Join operation. They are defined and explained in a greater detail in [54]-[55].

2.3.1.3.1 Sub-Star Composition

If R is a fuzzy relation in U×V, and x is a fuzzy set in U, then the "sub star composition" asserts that the fuzzy set y in V induced by x is given by

$$y=x \text{ o } R$$

where (xoR) is the substar composition of x and R. If star represents the minimum operator, then this definition reduces to zadeh's composition rule of inference. The extended sup-star composition can be obtained simply by extending the type-1 sup-star composition by replacing type-1 membership functions by type-2 membership functions, the sup operation with Join operation and the t-norm operation with the Meet operation.

2.3.1.3.2 Fuzzy Relation Form

A fuzzy relation is characterized by the same two items as a fuzzy set as shown in Fig. 2.7. First is a list containing element and membership grade pairs, $\{\{v_1, w_1\}, R_{11}\}, \{\{v_1, w_2\}, R_{12}\}, \ldots\ldots \{\{v_n, w_m\}, R_{nm}\}$. Note that the elements of the relation are defined as ordered pairs, $\{v_1, w_1\}, \{v_1, w_2\}, \ldots\ldots \{v_n, w_m\}$. These elements are again grouped with their membership grades, $\{R_{11}, R_{12}, \ldots\ldots R_{nm}\}$, which are values that range from 0 to 1, inclusive.

The second item characterizing fuzzy relations is the universal space. For relations, the universal space consists of a pair of ordered pairs, $\{\{V_{min}, V_{max}, c_1\}, \{W_{min}, W_{max}, c_2\}\}$. The first pair defines the universal space to be used for the first set under consideration in the relation, and the second pair defines the universal space for the second set. The following is an example showing how fuzzy relations are represented in this package.

Universal spaces for fuzzy sets and fuzzy relations are defined with three numbers in this package. The first two numbers specify the start and end of the universal space, and the third argument specifies the increment between discrete elements. Here is an example.

Fuzzy Relation [{{{1, 1}, 0.1}, {{1, 2}, 0.5}, {{2, 1}, 0.9}, {{2, 2}, 0.7}, {{3, 1}, 0.4}, {{3, 2}, 1}}, Universal Space→{{0, 3, 1}, {0, 2, 1}}]

Assuming that V and W are two collections of objects, an arbitrary fuzzy set R, defined in the Cartesian product VxW, will be called a fuzzy relation in the space VxW.

R is thus a function defined in the space VxW, which takes values from the interval [0, 1].

$R : V \times W \rightarrow [0, 1]$

In the case where V = W, we have a binary fuzzy relation on a single set V.

We can start our discussion by considering a countable collection of objects.

$V = \{v_i\}$, i = 1, 2, ...

$W = \{w_j\}$, j = 1, 2, ...

A fuzzy relation R can be represented in the following way:

R = $\{\{\{v_i, w_j\}, R(v_i, w_j)\}\}$, i = 1, 2, ... ; j = 1, 2, ...

A matrix or graphic interpretation is a more convenient way to study fuzzy relations. We examine fuzzy relations in this notebook in a manner analogous to that introduced earlier for fuzzy sets.

Step-1:
Let V = {1, 2, 3} and W = {1, 2, 3, 4}.
A fuzzy relation R in VxW has the following definition.

Step-2:

Input: R = FuzzyRelaton[{{{1,1},1},{{1,2}, 0.2},{{1,3}, 0.7}, {{1,4},0}, {{2,1},0.7}, {{2,2},1}, {{2,3},0.4} {{2,4},0.8},{{3,1},0}, {{3,2}, 0.6},{{3,3},0.3},{{3,4},0.5}}, UniversalSpace→{{1,3},{1,4}}];

This relation can be represented in the following two forms: as a membership matrix

Step-3:

Input:To Membership Matrix[R]//Matrix Form

Output//MatrixForm= $\begin{bmatrix} 1 & 0.2 & 0.7 & 0 \\ 0.7 & 1 & 0.4 & 0.8 \\ 0 & 0.6 & 0.3 & 0.5 \end{bmatrix}$ as a graph

Step-4:

Input: FuzzyPlot3D[R, AxesLabel→{"V", "W", "R"}, Boxed→False]

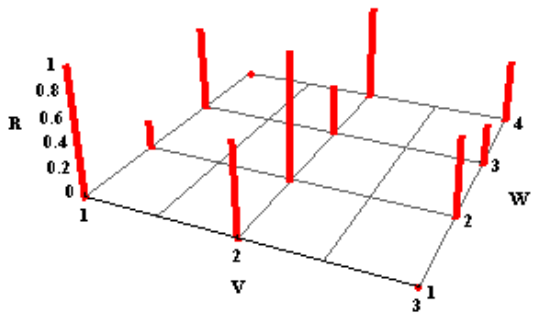


Fig. 2.7. Fuzzy Relation

Note that the elements of the fuzzy relation are defined as ordered pairs; v_i is the first and w_j the second element of an ordered pair $\{v_i, w_j\}$. The membership grades of the elements are represented by the heights of the vertical lines at the corresponding elements.

2.3.1.4 Type- Reduction

Type-reduction is that when an interval type-2 fuzzy sets is reduced to an interval-valued type-1 fuzzy set and then these type reduced sets are defuzzified to obtain crisp outputs. In this paper, we use centroid type reduction due to its reasonable computational complexity.

Three major type reduction methods are described as follows:

2.3.1.4.1 Centroid Type- Reduction

Similar to the centroid defuzzifier of type-1 (2.5), the union of type-2 fuzzy sets firstly requires computing the join of their secondary membership functions; i.e. to compute the secondary membership function $\mu_{\tilde{B}}(y)$ from $\tilde{B} = \bigcup_{l=1}^{M} \tilde{B}^l$, as:

$$\mu_{\tilde{B}}(y) = \coprod_{l=1}^{M} \mu_{\tilde{B}^l}(y) \qquad \forall_y \in Y \tag{2.11}$$

where $\mu_{\tilde{B}}(y)$ is the secondary membership function for the l[th] rule, and it depends on many factors such as join, meet and embedded sets. $\coprod$ is the join operator. The centroid type reduction calculates the centroid of $\tilde{B}$. Then extension form type-1 centroid defuzzifier (2.5) to type-2 centroid type reduction can be expressed as

$$y_c(x) = \int_{\theta_1 \in J_{y1}} \cdots \int_{\theta_N \in J_{yN}} [f_{y1}(\theta_1) * \cdots * f_{yN}(\theta_N)] / \frac{\sum_{i=1}^{N} y_i \theta_i}{\sum_{i=1}^{N} \theta_i} \tag{2.12}$$

where $*$ is the t-norm, which is normally the product or minimum, where θ_i is also the only one element that can be chosen from each interval J_{xi}. However, as analyzed by Karnik and Mendel [36], by using product t-norm, most value of the $[f_{y1}(\theta_1) * \cdots * f_{yN}(\theta_N)]$ will be very small, which cause troubles on practical computation. Hence, only minimum t-norm is considered for centroid type-reduction. Obviously, the centroid type reduced output $y_c(x)$ is a type-1 fuzzy set. Here i=1,….,*N*. For different FLS inputs, different values of $y_c(x)$ will be derived. Similarly the sequence to compute this process, the y-domain is discretized into N points $y_1, \ldots y_N$ and then J_{y_i} is discretized into a suitable number of points T_i (i=0,1,…*N*). Total number of computations is $\prod_{i=1}^{N} T_i$. However, this process needs to compute $\mu_{\tilde{B}}(y)$ firstly (i.e. combined from all output sets to form one $\tilde{B}$) that is high computation intensively. Note that the Centroid type reduction here must use minimum t-norm to perform [36].

Example:

Assume that X= {3, 6}. A type-2 fuzzy set Ã is given as follows:
Ã= (0.5/0.6+1/0.7)/3+ (0.2/0.8+1/0.5)/6. (82)
According to the type-2 fuzzy set, we can get the centroid of the set as the following type-1fuzzy set:
$y_c(x)$=(0.5*0.2)/[(3*0.6+6*0.8)/(0.6+0.8)]+(0.5*1)/[(3*0.6+6*0.5)/(0.6+0.5)]+(1*0.2)/[(3* 0.7+6 *0.8)/(0.7+0.8)]+(1*1)/[(3*0.7+6*0.5)/(0.7+0.5)]
=0.1/(4,71)+0.5/(4.36)+0.2/(4.6)+1/(4.25).

2.3.1.4.2 Height Type- Reduction

The extension from type-1 height defuzzifier (2.2) to type-2 height type reduction can be expressed as

$$y_h(\underset{\rightarrow}{x}) = \int_{\theta_1 \in J_{\bar{y}^1}} \cdots \int_{\theta_N \in J_{\bar{y}^M}} [f_{\bar{y}^1}(\theta_1) * \cdots * f_{\bar{y}^M}(\theta_M)] / \frac{\sum_{l=1}^{M} \bar{y}^l \theta_l}{\sum_{l=1}^{M} \theta_l} \tag{2.13}$$

where $l = 1,\ldots,M$. The $\bar{y}^l$ is the point having maximum membership in the l[th] output set and θ_l , $J_{\bar{y}^l}$, and $f_{\bar{y}^l}(\forall l)$ are associated with $\mu_{\tilde{B}^l}(\bar{y}^l)$. The sequence to obtain $y_h(\underset{\rightarrow}{x})$ is firstly to choose $\bar{y}^l$ from each rule output, then discretize the primary membership of each $\mu_{\tilde{B}^l}(\bar{y}^l)$ into a suitable number of points M_l, where $l = 1,\ldots,M$, i.e. rule number. Totally there will be $\prod_{l=1}^{M} M_l$ computations. Compare to centroid type reduction, the difference is that the discretized number of points on the horizontal axis is using the number of rules M instead of N [36].

2.3.1.4.3 Modified Height Type- Reduction

The extension from type-1 modified height defuzzifier (2.4) to type-2 modified height type reduction can be expressed as

$$y_{mh}(\underset{\rightarrow}{x}) = \int_{\theta_1 \in J_{\bar{y}^1}} \cdots \int_{\theta_N \in J_{\bar{y}^M}} [f_{\bar{y}^1}(\theta_1) * \cdots * f_{\bar{y}^M}(\theta_M)] / \frac{\sum_{l=1}^{M} \bar{y}^l \theta_l / \delta^{l^2}}{\sum_{l=1}^{M} \theta_l / \delta^{l^2}} \tag{2.14}$$

where all symbols have same meaning as (2.13)[36]. The only difference between the modified height type reduction and the height type reduction is that each output set secondary membership function, $\mu_{\tilde{B}^l}(\bar{y}^l)$, in the modified height type reduction is scaled by $1/\delta^{l^2}$.

2.3.1.4.4 Center of Sets Type- Reduction

Similar to center of sets defuzzifier (2.6) of type-1 FLCs the extension to type-2 center of sets type reduction needs to replace each type-2 consequent set, $\tilde{G}^l$, by its centroid, $C_{\tilde{G}^l}$ (a type-1 set); and finds a weighted average of these centroids. The firing strength corresponding to the l[th] rule is $\prod_{i=1}^{n} \mu_{\tilde{F}_l}(x_i)$, indicated by W_l , i.e., using meet operation for type-2 to replace $T_{i=1}^{n} \mu_{F_i^l}(x_i)$ of type-1 center of sets defuzzifier in (2.6). W_l is also a type-1 set. Then the center-of-sets centroid can be depicted by a generalized centroid expression as

$$y_{\cos}(\underset{\rightarrow}{x}) = \int_{v_1 \in C_{\tilde{G}^1}} \cdots \int_{v_M \in C_{\tilde{G}^M}} \int_{w_1 \in W_1} \cdots \int_{w_M \in W_M} T_{l=1}^{M} \mu_{C_{\tilde{G}^l}}(v_1) * T_{l=1}^{M} \mu_{w_l}(w_1) \Big/ \frac{\sum_{l=1}^{M} v_l w_l}{\sum_{l=1}^{M} w_l} \tag{2.15}$$

To obtain $y_{\cos}(\underset{\rightarrow}{x})$, a practical sequence is as follows:

1. Discretize its output space Y and computing its centroid $C_{\tilde{G}^l}$ for each consequent.
2. Compute the firing strength W_l for each rule.

3. Discretize the domain of each type-1 fuzzy set $C_{\tilde{G}^l}$ and W_l into a suitable number of points as N_l and M_l (l=1,…*M*), respectively.
4. Enumerate all the possible combinations. The total number of combinations will be $\prod_{l=1}^{M} M_l N_l$.
5. Compute the center-of-sets type reduction using (2.15) with M $C_{\tilde{G}^l}$ and W_l.

2.3.1.4.5 Example 2.3

Example 2.3: A type-2 FLC consists of the antecedents with type-2 fuzzy sets and the consequents with type-1 fuzzy sets. The membership functions of consequents are same as example 2.1 shown in Figure 2.2. Suppose that the firing strength $\underline{f^l}$ and $\bar{f}^l$ of the operation in the input and antecedents can be derived to fire consequent part. Then, by using product inference and t-norm, the fired rule output MFs can be shown as Figure 2.8(a). At each point $y \in [1,5]$, these three fired output sets can be described as:

- $\tilde{B}^1$: Two primary memberships, one is 0.9×exp(-1/2((y-2)/0.4)^2), and the other one is 0.8×exp(-1/2((y-2)/0.4)^2). The corresponding secondary grades are 1 and 0.8.
- $\tilde{B}^2$: Two primary memberships, one is 0.8×exp(-1/2((y-3)/0.2)^2), and the other one is 0.6×exp(-1/2((y-3)/0.2)^2). The corresponding secondary grades are 1 and 0.5.
- $\tilde{B}^3$: Two primary memberships, one is 0.2×exp(-1/2((y-5)/0.2)^2), and the other one is 0.1×exp(-1/2((y-5)/0.2)^2). The corresponding secondary grades are 1 and 0.4.

To compute height type reduction y_h, we firstly choose two maximum memberships $\bar{y}^l$ from each rule output and its corresponding primary memberships of each $\mu_{\tilde{B}^l}(\bar{y}^l)$, i.e. the first rule: [1/0.9]/2 and [0.5/0.8]/2, the second rule: [1/0.8]/3 and [0.8/0.6]/3, and the third rule: [1/0.2]/5 and [0.4/0.1]/5. From (2.12), the height type reduction need to consider each of the eight possible type-1 embedded sets and computes height defuzzification on them to get the crisp output in this type-reduced sets and computes height defuzzification on them to get the crisp output in this type-reduced set, e.g. first consider the situation where the fired consequents sets have primary memberships equal to 0.9, 0.6 and 0.2. The corresponding point in the type-reduced set is calculated as $(1\times 0.8\times 1)/(\frac{0.9\times 2+0.6\times 3+0.2\times 5}{0.9+0.6+0.2}) = 0.8/2.7059$. Next, consider that the point having maximum membership in the type-reduced set is calculated as $(1\times 1\times 1)/(\frac{0.9\times 2+0.8\times 3+0.2\times 5}{0.9+0.6+0.2}) = 1/2.7368$. The complete height type-reduced set is shown in Figure 2.8 (b). For the centroid type reduction (2.12), a combined output set is shown in Figure 2.8(c). As simple as this type-2 set discretized to 21 sample points (dot lines), there are 2^{21}=2,097,152 embedded type-2 sets for which the centroid computational procedure must be performed. The result of centroid type-reduced set is shown in Figure 2.8(d). For the center of sets type reduction, similarly to height type reduction, it need to derive the centroid of each consequent then compute each of the eight possible combinations by firing strength and centroids. Table 2.1 summarizes the results of four type reductions. The last column "centroid" means the center of gravity of the type reduced set; it can be used as defuzzified value of the type-reduced set. Note that the type reduced set here is not an interval type fuzzy set.

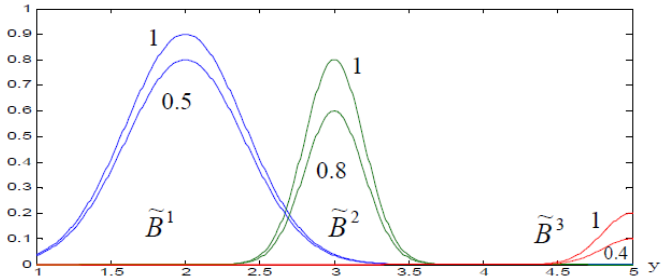
2.3.1.5 Defuzzification

After the type-reduction stage, Defuzzify the type reduced interval set $y_c(\vec{x})$, determined by its left most y_l and right most point y_r using the average of y_l and y_r. Hence the defuzzified crisp output is

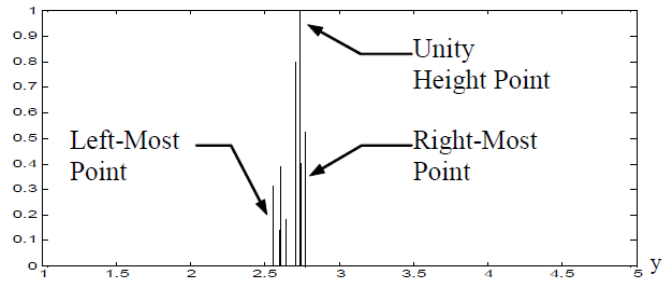
$$Y(\vec{x}) = \frac{y_l + y_r}{2}$$

Table 2.1. THE RESULTS OF FOUR TYPE REDUCTIONS

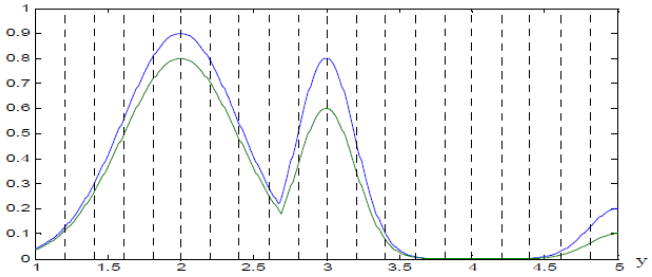
Type-Reduced Set	Left-Most Point	Right-Most Point	Width	Unity Height point	Centroid
Height	2.5625	2.7778	0.2153	2.7368	2.6985
Modified Height	2.9730	3.2000	0.2270	3.1429	3.1125
Centroid	2.3403	2.5114	0.1711	2.4729	2.4321
Center of Sets	2.5527	2.7604	0.2077	2.7204	2.6832



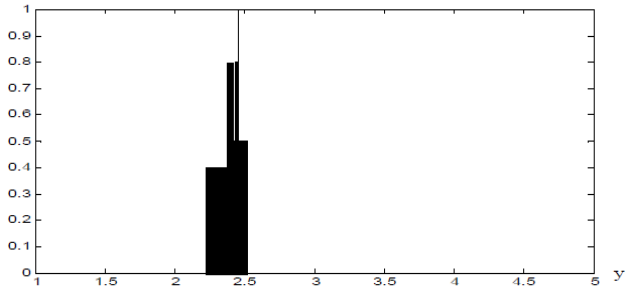
(a)



(b)



(c)



(d)

Fig. 2.8. The fired outputs are shown in (a), and the height type-reduced set is shown in (b). The combined fired output set for centroid type reduction is shown in (c) and its type-reduced set is shown in (d).

CHAPTER 3

HYBRID GENETIC FUZZY SYSTEM

This chapter presents design and optimization model of type-1 fuzzy logic controller using new hybrid GAs in detail.

3.1 Introduction

The vast majority of FLCs developed to date are based on the expert knowledge base derived from imprecise heuristic knowledge base. Such heuristic knowledge base can be collected and delivered by experienced human experts (e.g. decision maker, designer, process planner, machine operator). In most cases, FLCs are used when there is an expert knowledge base available to manually construct the fuzzy knowledge base (FKB) and an important nonlinearity of the modeled process. The manual construction of an FKB requires the evaluation of each proposition made by the expert to measure its performance. If the controller performance is not satisfactory, the expert modifies (i.e., fuzzy set repartition, change rule base, change the type of MFs) the FKB. Such manual approach is very time-consuming and tedious as it requires trial and error method and completely dependent on the expert's intuition, experiences and his understanding of the problem's behavior. Moreover, this process does not guarantee to generates optimal or near optimal FKB [56]-[57]. Therefore, an optimization method is highly desirable to automatically build the FKB that maximize the controller performance.

In this study, we consider new hybrid (real and binary coded coupled) GAs as an optimization tool for the automatic generation of FKBs. The FKBs contain two cooperative but distinct parts; (1) Parameters of fuzzy sets defining the fuzzy control rules and (2) Type of MFs defined on the shape of fuzzy set. They are distinct in such a way that first one deals with real numbers while the second one uses integer numbers to define the type of MFs. For this matter, we consider a hybrid GAs where the binary part is mapped into a string of integers.

3.2 Review of Existing Methods and differences to proposed Method

A sustainable number of approaches have been proposed to automatically learn the FKB. Most of them have focused on the rule base learning, one part of FKB, from the numerical data using a predefined another parts (number of labels, working ranges and MFs types for each linguistic variable) of FKB[58]-[61]. This operation mode makes the FKB have a significant influence on the controller performance. Moreover, our design approach does not require any set of input-output numerical data training pairs. Some methods used genetic algorithms (GAs)[62-63] tabu search (TS),[57] particle swarm optimization (PSO),[64]-[65] and self-organizing feature map (SOFM)[66] as an optimization tool to improve parameter optimization problem. Currently, GAs,[67] are considered to be among the most effective and widely used global search techniques. As a result, GAs has been extensively applied to identify fuzzy systems, mainly with the objective of increasing accuracy, thus leading to the so-called genetic fuzzy systems [68-69].

GAs was used by Arslan and Kaya [62] in the determination of MFs whereas their controller used predefined rule base. They applied GAs to design a fuzzy logic control system having a single input and output. We employed HGAs to concurrently learn parameters of MFs, rule base and MFs types for designing fuzzy logic control system with arbitrary number of input and output variables. Bagis [57] proposed a method based on TS algorithm for the determination of MFs only. Bai and Chen [56] proposed an automatic method for students' evaluation task. Its purpose was to automatically construct the grade MFs of lenient-type grades, strick-type grades, and normal-type grades of fuzzy rules, respectively. The system performs fuzzy reasoning to infer the scores of students based on the constructed grade-MFs. Their proposed method used predefined rule base. Meredith, Karr & Krishna [70] presented an approach based on GAs to the tuning of MFs in FLCs for a helicopter. Wong[64] et al. applies PSO to determine appropriate MFs of the fuzzy system (FS) automatically and presented a motion control structure with a distance fuzzy controller and an angle fuzzy controller for the two-wheeled mobile robot. Yang and Bose[66] presented a method for generating fuzzy MFs with unsupervised learning using SOFM. The SOFM approach is a two-step procedure: firstly, it generates the proper clusters and secondly it generates the fuzzy MFs according to the clusters in the first step. They applied this method to pattern recognition. These methods differ mostly in the order or the selection of different MF types, shapes, width and distribution on the performance of a FLC. Moreover, other differences between the previous approaches lie mainly in the type of coding and the way rule set and the shape and width of MFs are optimized.

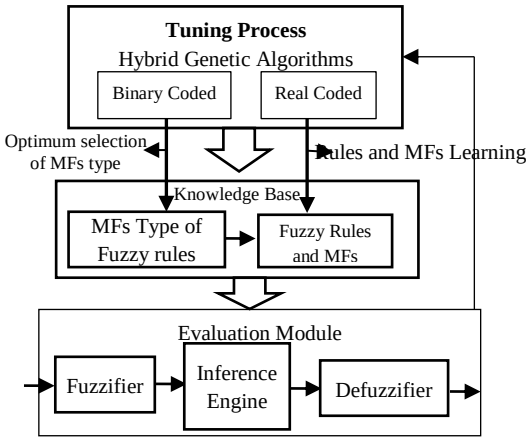


Fig. 3.1. Combined FLCs and hybrid GAs Architecture

3.3 Designing Optimal FLCs Using Hybrid GAs

The first step in designing an integrated FLCs and HGAs architecture (depicted in Fig. 3.1) is to decide which parts of the FKB are subject to be optimized by HGAs. FKB of a fuzzy logic control system is made up of two principal components, i.e., the parameters of the fuzzy rule base, constituted by the collection of fuzzy rules with MFs shape and MFs type associated with the linguistic labels of each fuzzy rule. In this case, both of these two components of FKB are adapted using HGAs. In the developed architecture, the real coded part of HGAs is employed for generating desirable fuzzy control rules with parameters of MFs and at the same time the binary coded part of HGAs is employed for optimum selection of MFs type of each control rule.

The learning of FKB components concurrently for optimal design of FLCs consists of the following steps (Fig. 3.2):

Step 1: Identify the variables (Inputs, states, outputs) of the control process, type of fuzzy inference engine, fuzzification and defuzzification type and the MFs type to be used.
Step 2: Partition the input and output universe of discourse or the interval spanned by each variable into a number of fuzzy regions (fuzzy sets) and assign a fuzzy MFs i,e linguistic level of each region.
Step 3: Encode the input output spaces fuzzy regions into real valued-strings. At the same time encode the MFs type representing integer number (Fig. 3.5) into bit- string (0 or 1). For example, encode 0 to 00, 1 to 01, 2 to 10 and 3 to 11 to select the gaussian, left-triangular, right-triangular and triangular MFs respectively. Each chromosome contains real and binary valued string.
Step 4: Use HGAs in two different ways: phase-1(real coded part) and phase-2 (binary coded part) as self-learning adaptive methods to generate a set of fuzzy control rules with adjustment of MFs shape and select the optimum MFs type of each rule respectively.
Step 5: Use newly generated fuzzy rules and their corresponding MFs type to determine the performance of FLCs and assign a fitness value. At the same time calculate the importance of chromosomes according to the performance of criteria.
Step 6: Check the termination condition. If termination criterion is not met go to Step 3.
Step 7: Determine a mapping from the input space to the output space based on the combined fuzzy rule base and their corresponding MFs type using a defuzzifying procedure.

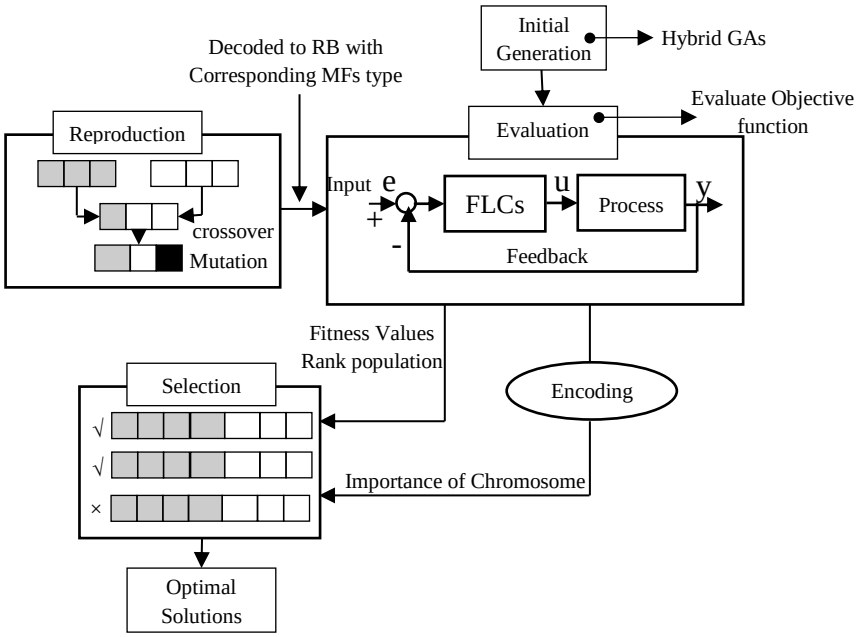


Fig. 3.2. Optimization of Fuzzy Logic Controller by Genetic Algorithms

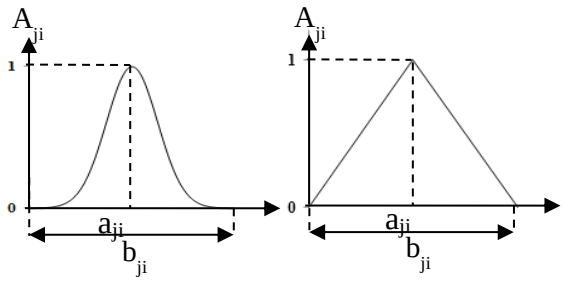


Fig. 3.3. Optimizing Parameters of MFs

3.3.1 Optimization of FLCs

In this study, the FKB of FLCs is tuned using new hybrid GAs (Fig. 3.2) by simultaneously optimizing the design objectives. The design variables of the FLCs are the parameters of the input and output MFs of each rule defined by generalized MFs described in Eq. (2.2) and (2.3), and parameters selecting the type of MFs.

3.3.1.1 Rule Optimization parameters

In this study, we use different MFs type such as triangular, gaussian defined by the equation (2.2), (2.3) where x_j the crisp value. We can see that triangular and gaussian MFs needs a domain of definition x_j and two real parameters a_{ji} and b_{ji} to be defined (Fig. 3.3). These tow real parameters defines the MFs shape to process the user inputs and the fuzzy inference system (FIS) that can provide an "appropriate" output through fuzzy reasoning and inference mechanisms. For this reason, we encode each chromosome in a real gens chromosome for optimizing the parameters of MFs of each fuzzy control rule. The MFs fully defines the fuzzy set.

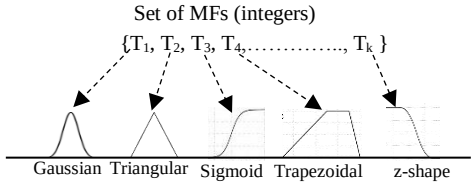


Fig. 3.4. Coding of a MFs type

3.3.1.2 Optimal MFs Type Selection

In this study, genetic MFs type selection technique try to select the optimal MFs for each rule separately while maintaining and improving the system performance. MFs type selection is a preprocessing step when designing a conventional fuzzy controller and every rule used one MFs type (i.e., either gaussain or triangular). The MFs types are coded as a set of integers where each integer in the set represents a one MFs type (see Fig. 3.4). In this case, we used binary coded GA where the binary value is mapped into a string of integers (see Fig. 3.5). In the binary coding, 2 bits resolution for every parameter was used. In this way, the binary part of HGAs selects the best combination of MFs type for the rule base of FLCs.

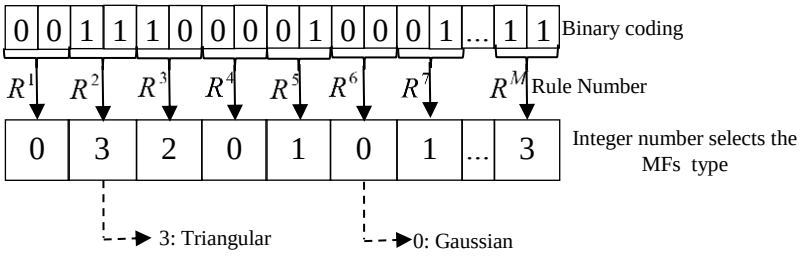


Fig. 3.5. Membership function Selection Methodology

3.3.2 Encoding Schema

We present a new hybrid encoding schema mixing the real and binary coding feature to represent the potential solution. Each genetic representation is a FLCs potential solution. Encoding is the genetic representation for translating the FKB parameter space into a certain space in which the HGAs can operate. The FKB are divided into parameters of MFs and fuzzy rules with their corresponding MFs types. So, the *genotype* C corresponds to several independent sets of reals and a set of integers:

$$C \equiv \{C_R, C_T\}$$

where C_R and C_T are the genotypes of the fuzzy rules and their corresponding MFs type respectively. The fuzzy rules representing fuzzy sets, FS_i in each variable can be characterized by two parameters a_{ji} and b_{ji} as shown in Fig. 3.2. All of the information represented by the FKB parameters is encoded in a structure called a chromosome or string. Each chromosome is divided into two parts (Fig. 3.6): (C_R) defines the fuzzy rules and MFs parameter and (C_T), is used for select the MFs type. (C_R) and (C_T) coding schema are as follows:

C_R: The (C_R) part is a sequence of real genes for the input and output variable, which identify the fuzzy rules and parameters of MFs.

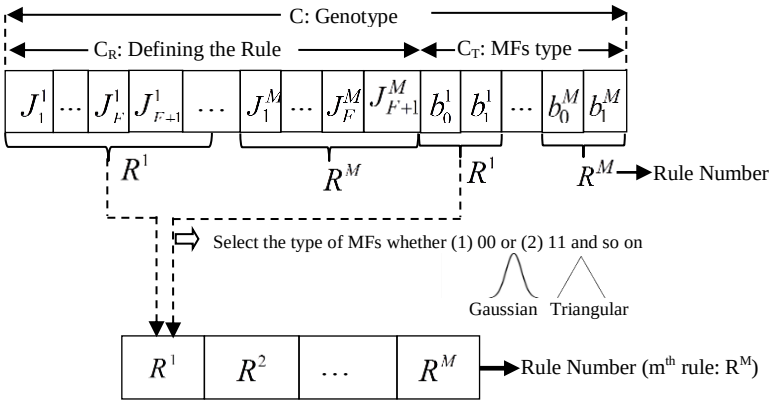


Fig. 3.6. Composing of Gene

(C_T): The (C_T) part is a sequence of binary bits which identify the MFs type for each rule individually. The type of MFs such as trapezoidal or triangular or gaussian in the antecedent and consequent with linguistic meanings can be selected automatically by two consecutive bit b_1 and b_2. The parameter b_1 and b_2 are encoded into the gene with a binary-valued representation.

The new HGAs have been developed for the adjustment of the two interrelated FKB components and seek the optimum FLCs that presents the minimum objective error. A vector represents the chromosome of binary bits and real numbers are constructed, which embodies the FKB parameters.

3.3.3 Evolution Strategy

The evolution strategy leads from the initial population of FLCs to the optimal FLCs, is described as follows:

Selection: After the evaluation of the initial randomly generated population, the GAs begins the creation of the new FLCs generation. FLC-chromosomes from the parent population are selected in pairs to replicate and form offspring FLC-chromosomes. The FLCs-chromosome selection mechanism for reproduction is that *N* parents and theirs generated offspring is combined to select the best *N* FLCs-chromosomes to take part in the population of next FLCs generation.

Crossover: When two strongest chromosomes are selected for reproduction, their vectors are combined in order to produce two new FLC using genetic operators. The main GAs operators used are a crossover and a mutation and are applied with varying probabilities. So, if a probability test is passed crossover takes place. If the probability test fails, the produced children are identical replications of their parents.

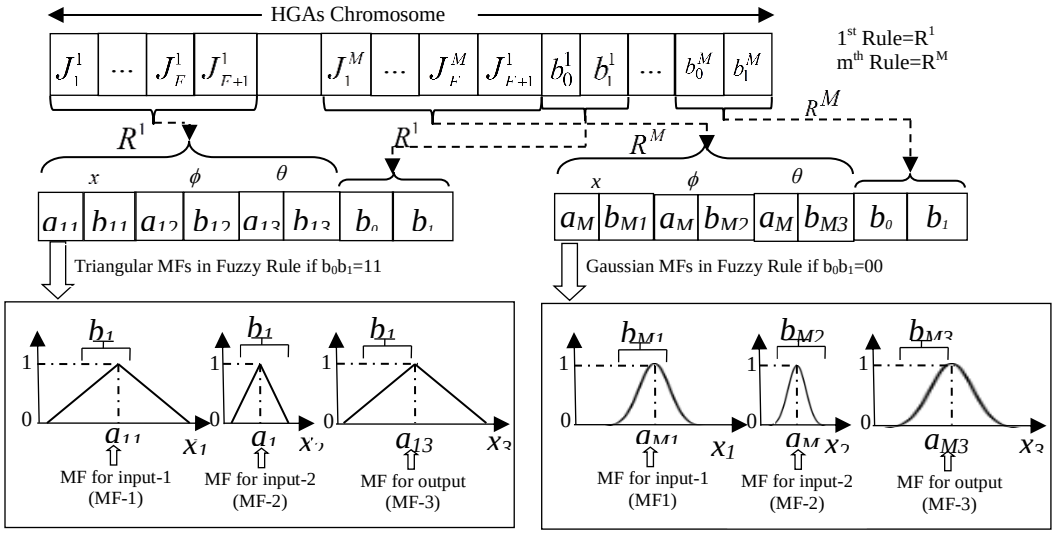


Fig. 3.7. Example of HGAs chromosome (C_R, C_T) coding scheme where C_R, define the control rules as well as shape of MFs and C_T, define the MFs type. Here 1st rule MFs type Triangular and mth rule MFs type Gaussian.

In the binary GA a uniform crossover operator has been used. In the real GA, the max-min-arithmetical (MMA) operator was used[66] as a crossover operator. If C_u^t and C_v^t are to be crossed four possible new individuals (I) are created.

$$\begin{aligned} I_1^{t+1} &= \alpha I_v^t + (1-\alpha) I_u^t \\ I_2^{t+1} &= (1-\alpha) I_v^t + \alpha I_u^t \\ I_3^{t+1} &\text{ with } I_{3k}^{t+1} = \min\{c_k, c_k^{'}\} \\ I_4^{t+1} &\text{ with } I_{4k}^{t+1} = \min\{c_k, c_k^{'}\} \end{aligned} \tag{3.1}$$

A constant parameter α is set equal to 0.5 for our experiments. The two individual with higher fitness are inserted in the new population replacing the parents, producing the new generation.

Mutation: In the binary encoding every parameter of the offspring chromosomes undergoes a probability test and if it is passed,[67] the mutation operator alters that parameter using Michalewicz's nonuniform mutation operator[72]. The operator is described below:

If $C_u^t = \{P_1, \ldots\ldots P_k \ldots . P_L\}$ is a FLCs- chromosome vector with length L and P_k is an FLC parameter that is chosen for a mutation, the new output P_k^{mut} will be an after mutation

$$P_k^{mut} = \begin{cases} P_k + \Delta(t, P_{kr} - P_k) \ \ if\ r = 0 \\ P_k - \Delta(t, P_k - P_{kl}) \ \ if\ r = 1 \end{cases} \tag{3.2}$$

where r is the random parameter, (P_{kr}, P_{kl}) the domain of parameter P_k $(k \in 1,2,\ldots\ldots L)$ and a function $\Delta(t,x)$ returns a value in the range $(0,x)$ such that the probability of the value returned being close to 0 increases with t.

$$\Delta(t,x) = x.(1 - \eta^{(1-t/T)^c}) \tag{3.3}$$

where η is the random floating point number in the interval $[0,1]$; t is the current generation; T is the maximum number of generation; c is a parameter determines the degree of dependence on the number of generations. For our experimental analysis, c was chosen equal to 5. Using this selection method and crossover operators will tend to cause the algorithms to converge.

In this system Mutation probability is dynamically changed in the range from 0.04% to 0.24% per bit and 1% to 10% per real parameter during the generations. The adaptive parameter control for crossover rate is introduced here with the ranges from 40% to 90% per chromosome. Figure 3.7 shows an example of integrating MFs parameters, fuzzy rule sets and their type of MFs sets selection parameters, in a chromosome. Then this chromosome is decoded and represented the fuzzy rules with their corresponding MFs type. In Fig. 3.7 1st rule (R^1) MFs type is triangular and m^{th} rule (R^M) MFs type is gaussian.

3.4 Problem Statement, definitions and Notation

In this section, we shall first describe the kinematic model of a car-like robot and then analyze the fundamental properties of the corresponding system from a control view point. The main challenge for the car like robot control problem is to design a controller to successfully back up a car like robot to a loading dock from any initial states.

3.4.1 Modeling and Analysis of car like robot

The problem, to back up a simulated car like robot to a loading dock in a planner parking lot is depicted in Fig. 3.8 and the involved notations. The fuzzy logic control system must find incrementally a path to the loading dock, independently of the initial states (x_0, y_0, ϕ_0) of the car like robot. In the proposed kinematics model of car-like robot, let (x_f,y_f) and (x_r,y_r) be the coordinates of the front-wheel and rear-wheel center of the car-like robot, respectively.

A state vector, (x_r,y_r) and the angle ϕ of the car like robot with the horizontal line define the actual state of the car like robot. The motion of the car like robot can be described by the following set of equations:

$$\frac{d\theta}{dt} = \frac{u \tan\phi}{L}$$
$$x(t+1) = x(t) + b * \cos(\phi(t+1)) \quad (3.4)$$
$$y(t+1) = y(t) + b * \sin(\phi(t+1))$$
$$\phi(t+1) = \phi(t) + \theta(t)$$

where θ is the steering angle (control action). The discussed objective of this study is how to design a suitable fuzzy control system which can make efficient strategies to back the car-like robot into the correct state of the parking space under the mentioned equations and constraints.

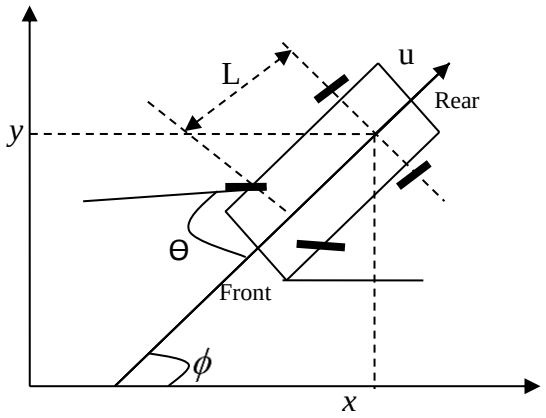


Fig. 3.8. A car like robot model

A fuzzy logic controller receives this state variable as input and calculates the appropriate steering angle $\theta = f(x, \phi)$ that drives the robot toward the goal. The controller have to make the robot reach at the loading dock at a right angle ($\phi = 90^0$) and to align the state (x, y) of the robot with the desired location (x_d, y_d). The car like robot should move backward by some fixed distance b at every stage. Assuming enough clearance between the car like robot and the loading dock we can ignore the y position of coordinate at the time of output (steering angle θ) calculation. There is no predefined path for each car like robot location and therefore the optimum steering angle is unknown. So, at every stage our proposed approach should calculate the steering angle $\theta = f(x, \phi)$ to back up the car like robot to a loading dock from any state and initial car like robot angle in the loading zone. The loading zone corresponding to the plane {0, 0}×{100, 100}.

This experiment should be regarded as an example of highly nonlinear complex problems. The fuzzy logic controller must find the correct function that maps points from the three dimensional input-space to the appropriate output variable, continuously from the initial state until the loading dock, and for all combinations of the initial state.

We assume a constant velocity u and the system variables of the car-like robot are restricted by physical limits as follows:

$$0 \le x_{robot} \le 100$$
$$-90^0 \le \phi_{robot} \le 270^0 \quad (3.5)$$
$$-30^0 \le \theta_{robot} \le 30^0$$

An important characteristic of fuzzy models, FM, is the partitioning of the input and output space of system variables (input, output) into fuzzy regions called linguistic terms using fuzzy sets[68]. The

number of linguistic terms is application (problem) dependent. For backing up a truck control problem, divide the domain regions for x, ϕ and θ into 5, 7 and 7 regions respectively.

Table 3.1. *x*, LINGUISTIC VARIABLE HAS 5 LINGUISTIC TERMS AND THEIR OPERATING RANGE

Range	Linguistic Terms/MFs
$x_{robot} <= 22.5$	LE
$22.5 < x_{robot} <= 32.5$	LE and LC
$32.5 < x_{robot} <= 42.5$	LC
$42.5 < x_{robot} <= 52.5$	LC and CE
$52.5 < x_{robot} <= 55$	CE
$55 < x_{robot} <= 57.5$	CE and RC
$57.5 < x_{robot} <= 70$	RC
$70 < x_{robot} <= 80$	RC and RI
$x_{robot} < 80$	RI

The fuzzy linguistic terms for x_{robot} are: LE (Left), LC (Left Center), CE (Center), RC (Right Center), and RI (Right) and the space $0 \le x_{robot} \le 100$ has been divided into five non-uniform (Not equal) intervals and the result of MFs analysis for the variable x_{robot} are summarized in the table-3.1:

Table 3.2. ϕ, LINGUISTIC VARIABLE HAS 7 LINGUISTIC TERMS AND THEIR OPERATING RANGE

Variable Range	Linguistic Terms/MFs
$\phi_{robot} <= -80$	RB
$-80 < \phi_{robot} <= -5$	RB and RU
$-5 < \phi_{robot} <= 42.5$	RU
$42.5 < \phi_{robot} <= 60$	RU and RV
$60 < \phi_{robot} <= 70$	RV
$70 < \phi_{robot} <= 80$	RV and VE
$80 < \phi_{robot} <= 90$	VE
$90 < \phi_{robot} <= 100$	VE and LV
$100 < \phi_{robot} <= 120$	LV
$120 < \phi_{robot} <= 135$	LV and LU
$135 < \phi_{robot} <= 170$	LU
$170 < \phi_{robot} <= 190$	LU and LB
$\phi_{robot} < 190$	LB

The fuzzy linguistic terms for ϕ_{robot} are: RB (Right Below), RU (Right Upper), RV (Right Vertical), VE (Vertical), LV (Left Vertical), LU (Left Upper), LB (Left Below) and the Space $-90^0 \le \phi_{robot} \le 270^0$ has been divided into seven non-uniform intervals. The result of MFs analysis for the variable ϕ_{robot} is summarized in the following table-3.2:

For the output θ the linguistic terms are NL (Negative Large), NM (Negative Medium), NS (Negative Small), ZE (Zero), PS (Positive Small), PM (Positive Medium), PB (Positive Large) and the Space $-30 \le \theta_{robot} \le 30$ has been divided into seven non-uniform intervals. The result of active MFs analysis for the variable θ_{robot} is summarized in the table-3.3:

Table 3.3. θ , LINGUISTIC VARIABLE HAS 7 LINGUISTIC TERMS AND THEIR OPERATING RANGE

Table 3. θ , Linguistic variable has 7 Linguistic terms and their operating range	
Variable Range	MFs
$\theta_{robot} <= -25$	NB
$-25 < \theta_{robot} <= -16$	NB and NM
$-16 < \theta_{robot} <= -13.5$	NM
$-13.5 < \theta_{robot} <= -4$	NM and NS
$-4 < \theta_{robot} <= -5$	NS
$-5 < \theta_{robot} <= 0$	NS and ZE
$0 < \theta_{robot} <= 1$	ZE
$1 < \theta_{robot} <= 5$	ZE and PS
$5 < \theta_{robot} <= 6$	PS
$6 < \theta_{robot} <= 12$	PS and PM
$12 < \theta_{robot} <= 16$	PM
$16 < \theta_{robot} <= 24$	PM and PB
$\theta_{robot} < 24$	PB

3.5 Simulation Results and Discussions

In this section we report the simulation results carried out with our proposed control algorithm described in the previous section. We obtained the fuzzy control rules and their corresponding MFs type from the best chromosomes of HGAs after 100 generation.

TABLE 3.4. FUZZY CONTROL RULES AND THEIR CORRESPONDING MFS TYPE

θ		x				
		LE	*LC*	*CE*	*RC*	*RI*
ϕ	*RB*	^{1}*PS/G*	^{8}*ZE/T*	^{15}*NS/G*	^{22}*NM/LT*	^{29}*NM/G*
	RU	^{2}*NS/LT*	^{9}*PM/T*	^{16}*PM/LT*	^{23}*PM/G*	^{30}*NS/G*
	RV	^{3}*NM/T*	^{10}*NS/RT*	^{17}*PB/T*	^{24}*NB/G*	^{31}*ZE/G*
	VE	^{4}*NM/G*	^{11}*NM/G*	^{18}*ZE/T*	^{25}*ZE/G*	^{32}*NM/T*
	LV	^{5}*NB/G*	^{12}*PS/G*	^{19}*PB/RT*	^{26}*NS/T*	^{33}*PM/T*
	LU	^{6}*PM/G*	^{13}*NM/T*	^{20}*NB/T*	^{27}*NB/T*	^{34}*NB/LT*
	LB	^{7}*PB/T*	^{14}*PB/T*	^{21}*PS/G*	^{28}*PS/G*	^{35}*NB/T*
Consequent Value/MFs Type, G:Gaussian, T: Triangular, LT: Left Triangular, RT: Right Triangular						

At the beginning of the simulation, the consequent part and MFs type of each rule is empty i.e., each entry of FAM (Fuzzy Associative Memories) cell is empty. To get the consequent part in terms of linguistic language form and MFs type of each rule from the developed optimization tool, we generate 20 successful control trajectory, whose initial states (x, y, ϕ) are (25, 30, 60^0), (80, 30, 150^0), (0, 10, -90^0).. etc.

For all the given initial states the HGAs converges to a set of fuzzy rules with their corresponding MFs type. During the simulation, we generate the twenty sets of fuzzy rules (FAM) from twenty different initial states. Then these rule bases have been combined together to form the FAM as

shown in table-3.4 using fuzzy amalgamation (FA) methodology[69, 2] FA method works as follows: Firstly, generate an empty new FAM. Secondly, combine and compare two tables (FAM): if there exists entries in the cells of first table (FAM) are the same with the corresponding entries in the second table (FAM) and vice versa then add the cell entries into corresponding cells of new table (FAM). On the other hand, for the entries in the cells of two tables are different then compute the average (mean) of the two different entries using the following formula:

$$Numeric-Value\ \ (NV) = \frac{1}{n}\sum_{i=1}^{n} N_i \tag{3.6}$$

where n is the number of combining tables, N_i is the center numeric values of fuzzy rule tables. Find the linguistic term (fuzzy set) for NV where it has highest membership value, and return that linguistic term as a result of two different entries into the corresponding cell of new table (FAM).
After the simulation the HGAs discover a compact set of fuzzy control rules and their corresponding MFs type as shown in table-3.4. Figure 3.9 shows an example of MFs form of fuzzy rules where rule 6 MFs type is gaussian and rule 32 MFs is type triangular.

Control Surface: The MFs and rules are design tools that give opportunity to model a control surface, a convenient way to examine a two input/one-output control strategy and controller properties. The fuzzification of the input variables and the defuzzification process determine the shape of the output control surface. The defuzzification process affects the total control surface since a single operation is applied to all rules in the rule matrix. The fuzzification process divides the inputs and outputs into ranges and the MFs are applied to each range. As the input values change from one range to another and if the MFs are different type, the shape of the contour changes around the range where the change has occurred. It is obvious that using this attributes one can more precisely fulfill a quality criterion in full operational range. The control surface (Fig. 3.10) generated by the applied FLCs while using (a) triangular (b) Gaussian (c) Mixed MFs.

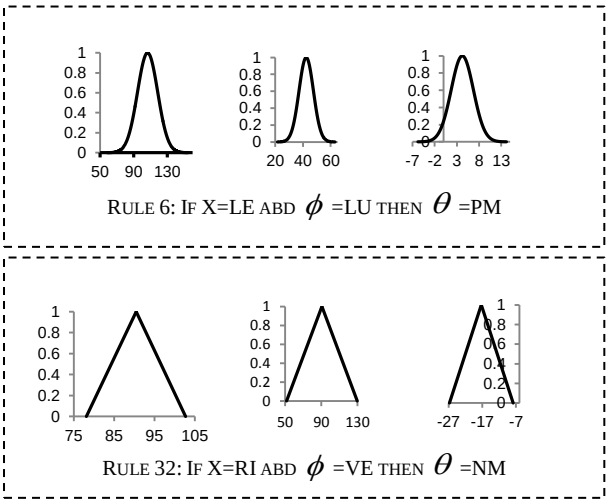
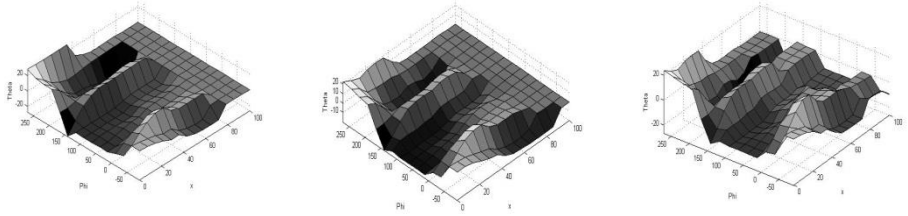


Fig. 3. 9. MFs form of Fuzzy Rules (a) Gaussian (b) Triangular

After applying the FAM and their corresponding MFs (fuzzy sets) type, we obtain the some control trajectories of car like robot in the work place from different initial states as depicted in Fig. 3.11.

For any initial state (x_0, y_o, ϕ_0) is given to back up the robot to hit the loading dock at right angle $\phi_f = 90^0$ and final state at (x_f, y_f) = (50, 100). The desired trajectory can be that the robot moves along the target line or coincide with the target line and then goes along y-axis to arrive at the goal.



(a) (b) (c)

Fig. 3.10. Control Surface generated by applied FLCs (a) Used Triangular MFs (b) Used gaussian MFs (c) Used mixed MFs(In a rule base some rules MFs type is triangular, some rules MFs type is gaussian and so on)

3.5.1 Sensitivity Analysis while selecting different MFs type

In this study, fuzzy rule base is represented by mixed MFs type, some fuzzy rules in rule base are represented by triangular MFs, and some are represented by gaussian MFs etc. while other existing control method used only one MFs type (i.e. either gaussian or triangular) for the whole rule base. In our control algorithm, the simple triangular, left-triangular, right-triangular or gaussian MFs are often used, but may not be the best choice and the basic effect is stated as follows. We have shown that the controller performance is improved greatly when using the mixed MF type i.e., some rules in the rule base used gaussian, some used triangular and so on.

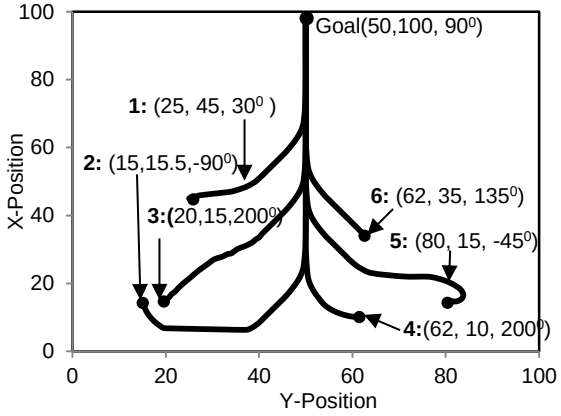


Fig. 3.11. Car like robot trajectories

In this case, fitness function is actually an objective function that is used to determine which solution within a population is better. In evolution algorithm, the solution space consists of a population of chromosomes where each chromosome is one solution that can be evaluated by the fitness function. Finally, the chromosomes can be ranked by calculating the fitness function value. In this study, chromosomes are sorted according to the fitness error and select the best one among all chromosomes of each generation. The fitness error is calculated by the equation 3.7. Figure 3.12 shows the genetic progress of average fitness score of (a) controller with triangular MFs type, (b) controller with gaussian MFs type, and (c) proposed controller, adaptive selection of mixed MFs type. The minimized average trajectory errors after 100 generations for three cases (a, b, c) are 1.16185, 1.14905, and 1.09342 respectively. From this figure we have shown that when the solution

is near the optimum point, only small improvement (sometime the improvement is insignificant) is achieved in each generation.

We also analyzed the performance of our control algorithm with mixed MFs and fuzzy controller with triangular or gaussian MFs type. The performances are compared using the value of docking errors and the trajectory errors as criteria (shown in Fig. 3.13 and Fig. 3.14).

The trajectory error is the ratio of the distance of the actual trajectory of the car like robot divided by the straight distance from initial state to final state.

$$\text{Trajectory Error} = \frac{\text{Distance of Actual Trajecory}}{\text{Straight distance from initial state to final state}} \tag{3.7}$$

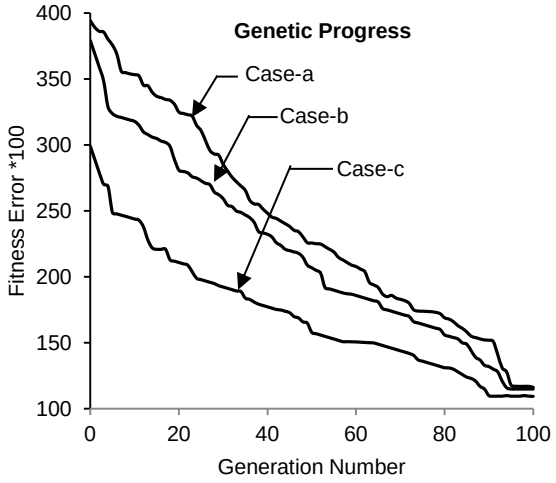


Fig. 3.12. The best of generation Fitness score (Average Trajectory error in percentage)

Figure 3.13 represents the comparison of trajectory errors between the, (a) controller with triangular MFs and (b) controller with Gaussian MFs (c) proposed controller with mixed MFs type. From the Fig. 3.13, we can obtain the average trajectory error of the three controllers (a, b, c) are 1.196, 1.141 and 1.098, respectively. With this trajectory error comparison, we can see the performance of the controller (c) is superior to that of (b) by 0.043 and to that of (a) by 0.098.

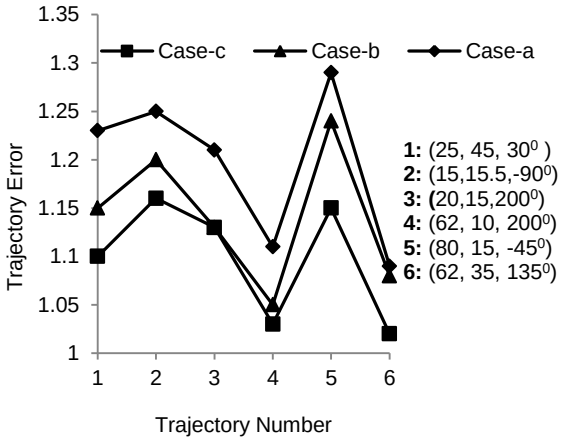


Fig. 3.13. Comparison of trajectory error between three controller (a) Controller with triangular MFs, (b) Controller with Gaussian MFs, and (c) Controller with Mixed MFs (Optimal selection of MFs type through optimization tool).

The docking error is the Euclidean distance between the desired final state $[x_d, y_d, \phi_d]$ and the actual final state $[x_d, y_d, \phi_d]$.

$$\text{Docking Error} = \sqrt{(x_d - x_f)^2 + (y_d - y_f)^2 + (\phi_d - \phi_f)^2} \tag{3.8}$$

Figure 3.14 shows the comparison of docking errors of (a) controller with triangular MFs, (b) controller with gaussian MFs, and (c) the proposed controller. From this figure, we can obtain the average docking error of the three controllers (a, b, c) are 1.035, 0.91 and 0.71, respectively. With this docking error comparison, we can show the performance of the controller (c) is superior to that of (b) by 0.20 and to that of (a) by 0.325.

Finally, we have shown that the shape, distribution and type of MFs have a significant influence on the behavior of fuzzy logic control systems. Both the improvement and the deterioration of the systems' performance due to the changes in the MFs type have been reported also.

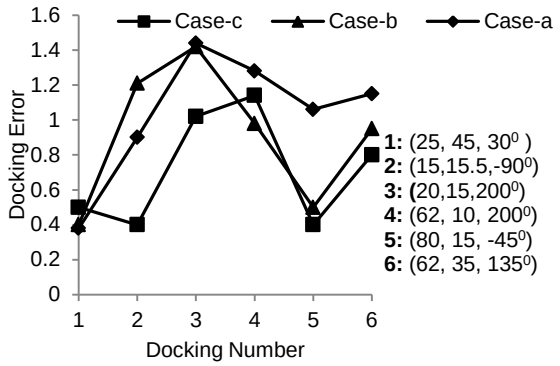


Fig. 3.14.Comparison of docking error between three controller (a) Controller with triangular MFs, (b) Controller with Gaussian MFs, and (c) Controller with Mixed MFs (Optimal selection through HGAs).

3.5.2 Evaluating the work

The car like mobile robot (truck), taken from [75, 76, 3] is a typical problem in nonlinear motion control of nonholonomic systems. It is a notable example that is generally used as benchmark problem for the evaluation of new control algorithms and as such it has been well analyzed [75, 76, 3].

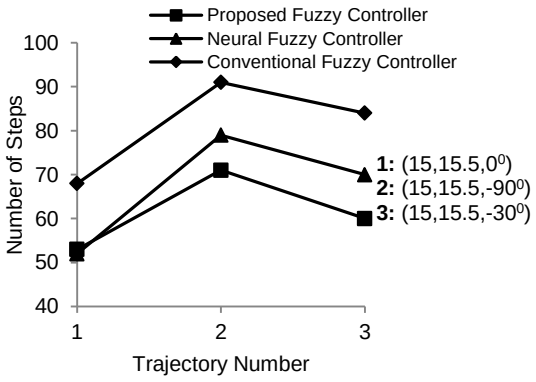


Fig. 3.15. Comparison of trajectory steps between three controllers (a) Conventional Fuzzy Controller, (b) Neural Controller with clustering [3], and (c) Proposed Fuzzy Controller.

Recently Li and Li in[3] have presented the fuzzy control system based on hybrid clustering method and neural network for trajectory tracking of a car like robot (truck). In their two stage work, structure identification and parameter identification are used to construct fuzzy logic control algorithm. The clustering method is applied to construct an initial fuzzy model to determine the number of fuzzy rules from the intuitionistic-desired trajectories. The clusters are automatically generated and the data are appropriately classified through clustering method. Then neural network is applied to obtain a more precise fuzzy model in the parameter identification for the car like robot control. The clustering method (off-line approach) is one of the most promising techniques when input/output samples are available for the system. It is not possible, however, to generate an initial fuzzy model without such type of input output data.

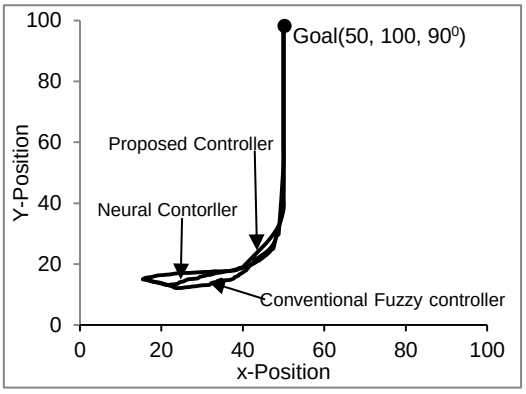


Fig. 3.16. Comparison of trajectory for the initial states (15, 15.5, 0^0) between three controllers (a) Conventional Fuzzy Controller, (b) Neural Controller with clustering[3], and (c) Proposed Fuzzy Controller.

The number of required trajectory steps of the car like robot from given states to the loading zone controlled by the conventional fuzzy system, the neural fuzzy controller with clustering[3] and our proposed control algorithm are given in Fig. 3.15 and Fig. 3.16 (graphical trajectories) for the initial state (15, 15.5, 0^0).

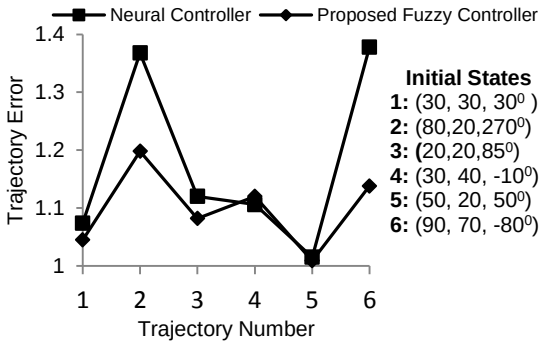


Fig. 3.17. Comparison of trajectory error between Neural Controller [76] and our proposed Fuzzy Controller.

The number of required trajectory steps for a neural controller with clustering is smaller than the conventional method but larger than our approach. In Fig. 3.16, we can see the proposed controller backs the robot up more smoothly than neural controller [3] and conventional controller [3] shown. Figures 3.15 and Fig. 3.16 show our proposed control algorithm performance to be better than that of existing approaches[3]. It both takes fewer steps to arrive at the goal and shows smoother trajectories.

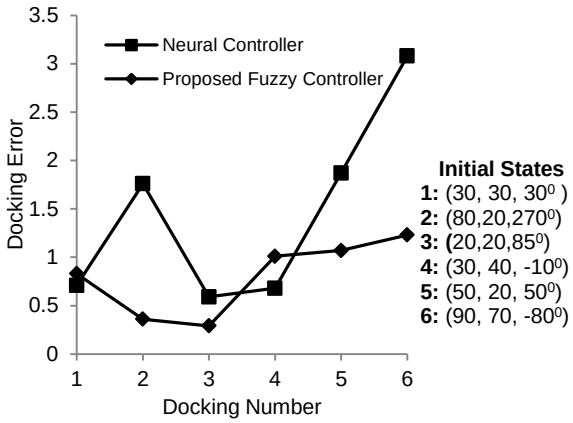


Fig. 3.18. Comparison of docking error between Neural Controller[76] and our proposed Fuzzy Controller.

We also compared the accuracy of the proposed controller against the existing neural controller [76] in terms of a docking error and a trajectory error.

Figure 3.17 shows the comparison of trajectory error between proposed fuzzy controller and neural controller[76]. The average trajectory error for the neural controller[76] and the proposed fuzzy controller are 1.1768 and 1.0986 respectively. Figure 3.18 shows the comparison of docking error between proposed fuzzy controller and the neural controller[76]. The average docking error for neural controller and proposed fuzzy controller are 1.4483 and 0.7983 respectively.

Figure 3.19 shows the comparison of sample robot (truck) trajectory for three initial states between proposed fuzzy controller and neural controller[76]. With this trajectory error (Fig. 3.17), docking error (Fig. 3.18) and trajectory (Fig. 3.19), we can show that the average performance of the fuzzy controller is superior to that of neural controller[76] by 0.0782 and 0.65 respectively.

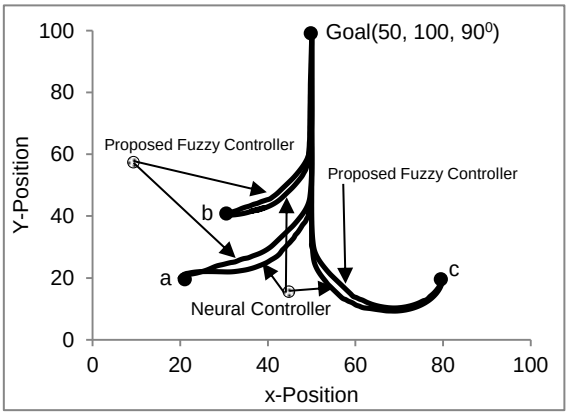


Fig. 3.19 Comparison of robot (truck) trajectory between Neural Controller[76] and our proposed Fuzzy Controller for the initial states (a) (20, 20, 85^0) (b) (30, 40, -10^0) and (c) (80, 20, 270^0)

3.6 Conclusions

This study presented new HGAs to optimize parameters of MFs of linguistic values, optimized linguistic rules and their corresponding MFs type concurrently for building a fuzzy controller. The new HGAs improve accuracy and stability of the outcomes by improving the degree of cooperation (i.e., optimal MFs type for rule) between DB and RB. In our simulation, we found that the fuzzy

controller is sensitive not only to how the rules are created but also to how the type of MFs are assigned to each rule. We analyzed the behavior of proposed method by applying it on the car like robot and compared the outcomes to[3] and[76] which are two common existing approaches in this field. For the car like robot, the controller is able to generate an appropriate control action so that the next state is appropriate in an overall trajectory sense and irregular or ineffective paths are not generated. Actually more tests also showed that HGAs adapts fuzzy control rules and their corresponding MFs type to continuously improve system control and the control results are successful and our method is encouraging. Simulation results illustrate that the our proposed generated fuzzy controller through HGAs performs well, better than the neural controller[3] with clustering and neural controller[76], as judged on the basis of a docking error, a trajectory error, trajectory steps and sample robot trajectory.

CHAPTER 4

OPTIMIZE FUZZY SYSTEM WITH RULE BASE REDUCTION

This chapter describe the rule reduction techniques of fuzzy logic controller. This chapter also describes some optimization criteria for designing a fuzzy logic controller.

4.1 Introduction

A fuzzy logic systems (FLSs) is a universal approximator that maps a crisp input vector into a crisp scalar output where the heart of a FLSs is a linguistic rule base. The number of rules increases exponentially as the number of inputs increases [77]. In some cases when the number of inputs is high as a large parameter number equivalently, it produces a serious computation problem. Fuzzy systems that can handle large number of inputs as well as large number of fuzzy rules may be hard to design and have large storage consumption, high computation complexity and poor convergency in parameter tuning.

In addition to the inconsistency level of rule pairs, it is also meaningful to search for the excess, irrelevant, erroneous or redundant rules in the rule base. The redundant rules do not contribute to the performance of the controller. For this reason, redundant rules are remove from the rule base is necessary. Moreover, irrelevant, redundant, erroneous and conflictive rules in a rule base perturb the controller performance when they coexist with others. Therefore, it makes sense to minimize the rule set by removing the irrelevant, erroneous and redundant rules from the original rule base and merging the conflicting rules, especially when performing fuzzy inference calculations in real time.

4.2 Review of Existing Methods and differences to proposed Method

In recent years, a sustainable number of methods have been developed towards fuzzy model generation and simplification as well as rule base size reduction in fuzzy systems [78]-[90]. Some methods focus on selection of important rules from the rule base that contribute to the inference mechanism, some methods are proposed to eliminate the redundant rules based on some criteria or merger of rules that share some common property. Chao and Chen [78] proposed a fuzzy rule-base simplification method on the basis of similarity analysis. Some fuzzy similarity measures on the basis of triangular membership function were proposed to eliminate redundant fuzzy rules and combine similar input linguistic terms.

Setnes et al. [79] proposed a rule-base simplification approach using set-theoretic similarity measures to reduce the number off fuzzy sets in the models. Yen and Wang [80] presented a method to select a set of important fuzzy rules from a given rule-base using the orthogonal transformation-based methods. Moreover, rule reduction has been addressed in [78-81] using orthogonal transformation or singular value decomposition. In these methods, fuzzy systems are modeled based on a linear regression, a rule firing strength matrix is constructed. The decision whether rules are eliminated or retained is done according to the effective rank of the firing matrix or the evaluation of the rule contributions for ordering. In these methods, a pre-determined number off fuzzy rules is required to build initial fuzzy model.

Jin [82] proposes an effective approach to data-based fuzzy modeling of high-dimensional systems. In order to remove the redundancy in the rule-base a distance based similarity measure has been used to define the similarity of fuzzy sets. On the other hand, genetic algorithms and evolutionary methods have been also successfully used for rule reduction [83]-[85]. In [83][86], the authors present a method to extract rules that contribute the most to the inference system using genetic algorithms. On the other hand, a sustainable number of clustering methods have been proposed to reduce the rule base by pruning the undesired (redundant/erroneous) rules and merging the similar rules [87]-[89]. In [79][90] use a similarity measure to merge the similar rules.

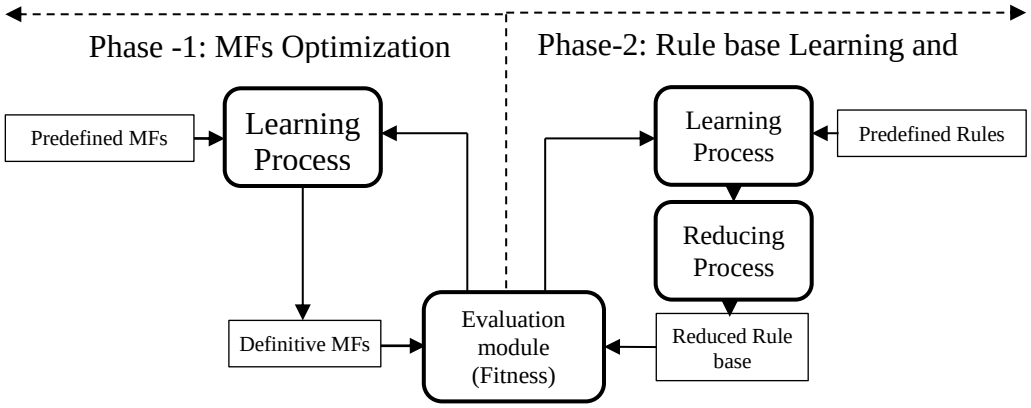


Fig. 4.1. Schema for discovery and reducing fuzzy rules and tuning suitable MFs

The above methods have two steps: first, the generation of an original rule base and removal of the redundant or conflicting rules from the original rule base. The general problems associated with this type of methods are high computational complexity in order to achieve good performance and rather conservative results. But very few study has been done on rule base size reduction techniques that preserve the inference, i.e., the outputs of the original and the reduced rule bases are identical. This study proposes a novel rule base size genetic reduction technique for a restricted class of fuzzy systems that preserves the inference.

4.3 Adaptive Model: Some Optimization issues and Rule Learning and Reducing

In this study, the binary and real coded coupled GAs based adaptive scheme is considered for discovering and reducing useful fuzzy rule base size and generate suitable MFs. The control algorithm is divided into two phases as depicted in Fig. 4.1:

(1) A genetic process to learn the MFs.
(2) A genetic method to learn and reduce fuzzy control rules.

We propose a genetic learning process for getting the MFs together with a reducing process for getting the optimal number of rules for building the fuzzy controller to maximize the controller performance.

4.3.1 Genetic Rule Learning and Reduction

Fuzzy logic systems (FLSs), also called fuzzy inference systems (FIS), are both intuitive and numerical systems that maps crisp inputs into a crisp output. Every FLSs consists of a rule base. The j^{th} rule of the fuzzy controller is of the following format:

$$R^j = IF\ x_1\ is\ A^j_{1k_1}\ and\ x_2\ is\ A^j_{2k_2}\ .....Then\ y\ is\ B_m \tag{4.1}$$

where ($k_n = 1,2,3,.....l_n$) and $I_1, I_2, I_3,.....l_n$ is the number of linguistic variables for the 1st, 2nd, 3rd … nth crisp input. $m = 1,2,3,4....l_m$ where l_m is the number of linguistic variable for the output variable.

The number of fuzzy rules increases exponentially with the number of input variables and their corresponding linguistic variables. If $(x_1, x_2, ...x_n)$ is the inputs variables and $(I_1, I_2, ...I_n)$ are their corresponding number of linguistic terms then the number of candidate rules is $r = I_1 \times I_2 \times I_3 \times .....I_n$. The control action derived from the consequent part of the fuzzy rules.

In the fuzzy model, the consequent value of fuzzy rule is either constant or linear combination of input variables. If the value is constant consequence and related to the number linguistic variables of output variables, then the total number of candidate rules increases by $q = r \times I_m$ for a FLSs. Then the implementation of such controller requires a huge amount of computation time for each step time in order to compute the appropriate control action to be applied to the controller.

Generalizing the fuzzy control rules (eqn 4.2) by adding the weight factor, w^j_m, determines whether the rule is included in the rule base or not as follows:

$$R^j = IF\ x_1\ is\ A^j_{1k_1}\ and\ x_2\ is\ A^j_{2k_2}\ .....Then\ y\ is\ w^j_m B_m \tag{4.2}$$

The weight factor, w^j_m, is binary value 0 or 1. In this study we consider three factors to determine whether the rule is included in the rule base or not. The factors are as follows:

If $\sum_{m=1}^{I_m} w^j_m = 0$ i.e there is no consequent of j^{th} rule then exclude the j^{th} rule from the candidate rule base.

If $\sum_{m=1}^{I_m} w^j_m = 1$ i.e., there is only one consequent of j^{th} rule then include the j^{th} rule directly to the rule base.

If $\sum_{m=1}^{I_m} w^j_m = 2$ or more i.e., there are two or more consequent of j^{th} rule then combines the two or more consequences into one consequence using the following equation:

$$Numeric - Value(NV) = \frac{1}{n}\sum_{i=1}^{n} N_i \tag{4.3}$$

where n is the number of consequent, N_i is the center numeric values of fuzzy rule consequence. Find the linguistic variable for NV where it has the highest membership value, and return that linguistic variable as a result of different consequences.

Table 4.1. CONSEQUENT OF RULES ENCODING TECHNIQUE

Rule-1			Rule-2			Rule-3			Rule-4			Rule-5			Rule-6			Rule-7			Rule-8			Rule-9		
N	Z	P	N	Z	P	N	Z	P	N	Z	P	N	Z	P	N	Z	P	N	Z	P	N	Z	P	N	Z	P
1	0	1	0	0	0	0	0	0	1	0	0	1	0	1	0	0	0	1	1	1	0	0	1	0	0	0

In this case a binary coded genetic schema is applied. Each sub chromosome is a vector of binary numbers whose size is the number of candidate rules, *r*. The vector contains the binary weights w_m^j , determines the consequence of the rule. Then a decoding schema of sub chromosome as well as reducing method is shown in Fig. 4.2. For illustration, you wish to design a control system for a problem that has two inputs and one output. In order to clarify the basic ideas of our proposed approach, we have chosen two inputs x_1, x_2 and one output y . It is also mentioned that you can chose higher number of input and output variables. In order to design a controller, at first divide the state space input x_1 x_2 variable and output y variable spaces are divided into fuzzy regions and assign a fuzzy MFs of each region. In our running example, we assume that we divide the domain regions for x_1, x_2 and y into 3 regions.

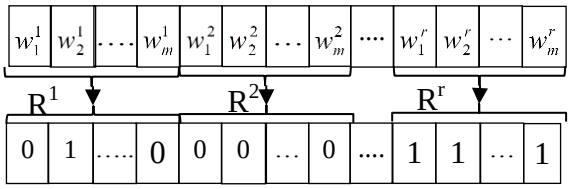


Fig. 4.2. Rule Reducing Technique

The linguistic variables for x_1 x_2 and y are N (Negative), Z (Zero) and P (Positive). There are 9 candidate fuzzy rules, and the fuzzy rule matrix (FRM) can be formed as 3×3. The probable consequent value of each rule can be N, Z or P. If we apply our proposed method then it reduces the rule base. In this case the weighting factor w_m^j is used as the learning parameter to tune the fuzzy rules and reduce the undesired rules. The weighting factor written a bit string depicted in table 4.1. One of the complexities at the rule base reduction level is to merge the conflicting rules, having same antecedents value but different consequent value. Several methods have been developed in the literature in different ways to merge the conflicting rules [91]-[94]. In this study, we propose a simple technique to reduce the computation complexity for merging the conflicting rules. Figure 4.3 shows an example of merging technique of conflicting rules with different consequent values. After the evolution process of GAs the bit string is decoded into fuzzy production rules as well as reduced rules as shown in table 4.2.

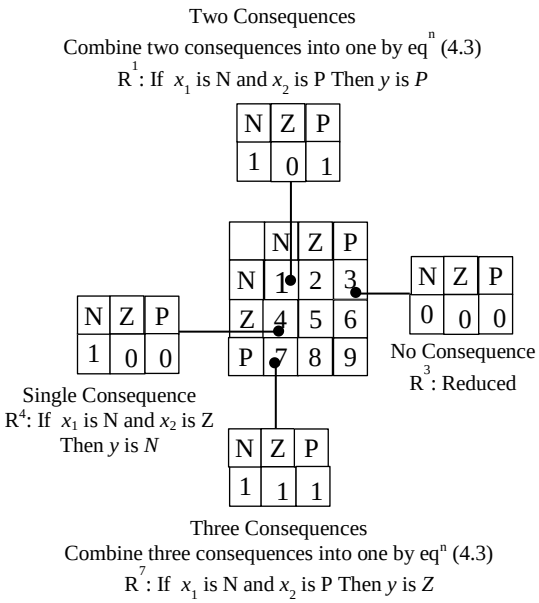


Fig. 4.3. Consequents combination Technique

4.3.2 Genetic Tuning process to obtain MFs

In the proposed learning method of suitable MFs, we will use evolutionary algorithms called CHC algorithm [95]. The CHC algorithm is a nontraditional genetic algorithm which combines a conservative selection strategy that always preserves the best individuals found so far with a radical, highly disruptive recombination operator that produces offsprings which are maximally different from both parents. This GAs presents a good trade-off between exploration and exploitation, being a good choice in problems with complex search spaces.

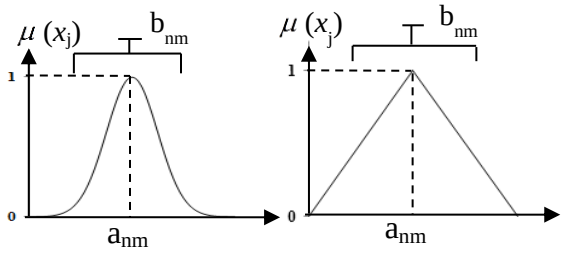


Fig. 4.4. Parameters defining the MFs

4.3.2.1. MFs Parameter Codification and Initial Gene

In this case a real coded genetic schema is applied. Each chromosome is a vector of real numbers whose size is kept the same as the length of the vector with size $n\times(m\times 2)$, where n is the number of items with m linguistic terms. A linguistic term denoted by the fuzzy set can be uniquely defined by its two parameters, a_{ji} and b_{ji} as shown in Fig. 4.4. The Gaussian shape has been adopted for all the MFs in the fuzzy model. The gaussian shape is chosen because of its simple shape; membership functions of any other form, for example, triangular or trapezoidal, can be considered as well. Then a sub chromosome has the following form (Fig. 4.5):

Table 4.2. REDUCED RULE BASE FOR FLC

	Antecedents				Consequents		
Rule No.		Input (x_1)	Operator	Input (x_2)		Output (y)	Remarks
R^1	If	x_1=N	and	x_2=N	Then	y=P	Active
R^2	If	x_1=Z	and	x_2=N	Then	Reduced	
R^3	If	x_1=P	and	x_2=N	Then	Reduced	
R^4	If	x_1=N	and	x_2=Z	Then	y=N	Active
R^5	If	x_1=Z	and	x_2=Z	Then	y=P	
R^6	If	x_1=P	and	x_2 =Z	Then	Reduced	
R^7	If	x_1=N	and	$x2$=Z	Then	y=Z	Active
R^8	If	x_1=Z	and	x_2 =Z	Then	y=P	Active
R^9	If	x_1=P	and	x_2=Z	Then	Reduced	

4.3.3 Genetic optimization process Both Rules and MF*s*

The chromosome is composed of two main sub chromosomes containing a) MFs parameters of antecedents and consequents and b) Parameters that are used for both rule tuning and rule reducing. The previous section described the structure of the overall system in detail. The previous section also presented the parametrization of a FRBS that includes the codification of MFs, control rules and reducing process. Referring to the previous section, it is clear that each chromosome is composed of two sub chromosome, sub chromosome 1 and sub chromosome 2. Sub chromosome 1 is a vector of real valued numbers and it contains MFs parameters while sub chromosome 2 is a vector of binary valued numbers.

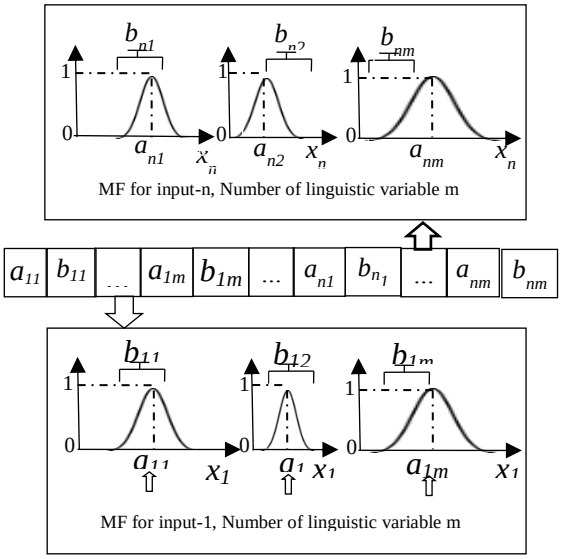


Fig. 4.5. MFs Learning coding schema example

In the present section, a genetic optimization process is described whose goals are to assign the antecedents and consequents MFs of the input and output variables and to obtain the reduced rule

base. From the overall point of view, the objective of the optimization process is to reduce the rule base as well as to optimize the MFs parameters.
To optimize the chromosome 1 i.e., real valued vector that defines the MFs parameters, a nontraditional genetic algorithms called CHC algorithm [95] is used. The four main components of the algorithm are as follows:

4.3.3.1. An elitist selection

After the evaluation of the initial randomly generated population, the CHC algorithm begins the creation of the new FLCs generation. The chromosomes of the current population are merged with the offspring population obtained from it. Then the best individuals are selected to take part in the population of next new population.

4.3.3.2. Crossover

A highly disruptive crossover, HUX, which crosses over exactly of the non-matching alleles, where the bits to be exchanged are chosen at random without replacement. In this way, we guarantee that the two offsprings are always larger fitness value measured by genetic distance than their two parents, thus maintaining not only a high genetic diversity in the new population but also decreasing the risk of premature convergence. The highly disruptive recombination operator, HUX, replaces classical mutation plus crossover in a GAs. This crossover assures ergodicity of the procedure [96].

In each generation, chromosomes are sorted according to the fitness error and select the best one among all chromosomes of each generation.

4.3.3.3. An incest prevention mechanism

Parent population on each generation is randomly chosen without replacement and paired for mating during the reproduction process. However, not all these couples are allowed to cross over. Before mating, the Hamming distance between the potential parents is calculated and if a half of this distance does not exceed a difference threshold D, they are not mated and no offspring coming from them is included in the offspring population. The threshold is usually initialized depending on the chromosome length. According to [95], if L is the chromosome length the threshold should be D = L/4. If no new offspring is obtained and included in the population in one generation, the difference incest threshold *d* is decremented by one. The effect of this mechanism is that only when diverse potential parents are mated, the diversity required by the difference threshold automatically decreases as the population naturally converges.

4.3.3.4. A restart Process

This substitutes the usual GAs mutation, which is only successfully applied when the population has converged. The difference threshold test is considered as the stagnation of the search measure. The stagnation of the search happens when it has dropped to zero and several generations have been run without introducing any new individual in the population.
Then the population is reinitialized (diverged) by using the best M, in this case M=1, individuals as the first chromosome of the new population and generating the remaining M-1 chromosomes by randomly flipping a percentage called divergent rate (DR), in this case DR=0.35, in their bits belonging to the best global individual.

The general parameters that affecting the CHC evolution process are as follows: Population size (PS), number of individuals in the population. Divergence rate (DR), percentage of the best global

solution found used as a pattern to construct the new population in the restart stage. The values are in [0, 1]. Difference threshold (D), degree of maximum similarity allowed between two individuals in the population in the crossover process. The values range of D is [0, L/2], where L is the length of the individual. Best individuals (M), number of best individuals considered when the weakest population is reinitialized in order to increase diversity.

The learning process of MFs works based on the real coded CHC algorithm. Extend the classical binary coded CHC algorithms to deal with real coded chromosomes that increase the coarseness of the search space. New real coded extension process maintains existing basis as much as possible. The main genetic operations of real coded CHC algorithm are same as the classical binary coded CHC algorithms [97].

In both cases, the elitist selection and the restart are exactly the same. The incest prevention mechanism works with a different crossover rate. The detailed description of incest prevention mechanism is in [97]. Incest prevention mechanism recombination of between two parents is only performed if the distance between two parents is not very similar, i.e. if the distance between them is higher than a parameter d, the incest threshold.

4.3.3.5 BLX-a crossover operator

For real coded CHC algorithms, the BLX-α crossover is considered instead of the HUX one [19]. In the following, we describe the BLX-α crossover operator in detail. The parameter α is used to make this crossover as disruptive as desired. Let $C_1 = (c_1^1, c_2^1, \ldots\ldots c_n^1)$ and $C_1 = (c_1^2, c_2^2, \ldots\ldots c_n^2)$ be two real-coded parents to be crossed where *n* is the function dimensionality. This crossover generates an offspring $O = (o_1, o_2, o_3 \ldots\ldots o_n)$ where each o_i is generated randomly within the following specified interval: $[C_{\min} - I\alpha, C_{\max} - I\alpha]$ where $C_{\min} = \min(C_i^1, C_i^2)$, $C_{\max} = \max(C_i^1, C_i^2)$ and $I = C_{\max} - C_{\min}$.

4.4 Simulation Results and Discussions

In this section, simulation results are presented in order to illustrate the effectiveness of the proposed control algorithms.

4.4.1 Application to the control of Two degree of freedom Inverted Pendulum

Being an under-actuated mechanical system that is inherently open loop and unstable with highly non-linear dynamics, the inverted pendulum systems are a perfect test-bed for the design of a wide range of classical and contemporary control techniques. However, their nonlinear dynamics present a challenging control problem, since traditional linear control approaches do not easily apply.

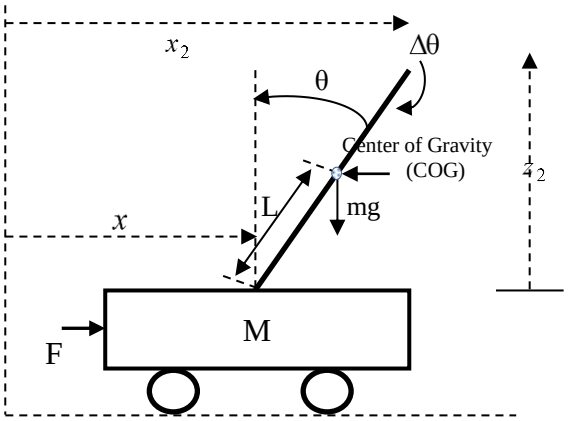


Fig. 4.6. Cart-pole typed inverted pendulum system

This control application is a popular demonstration of using feedback control to stabilize an open-loop unstable system with fewer control inputs than the degrees of freedom. The cart-pole task involves a balancing pole hinged to a motion less cart that travels left or right along a straight bounded track as shown in Fig. 4.6. The pole is free to rotate only in the vertical plane of the cart and track. There are no sidelong resultant forces on the pole and it remains balanced as shown in Fig. 4.7.

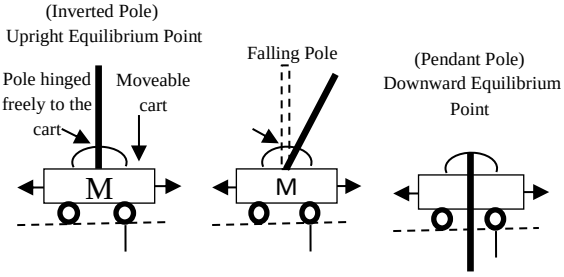


Fig. 4.7. Equilibrium Points

The control objective is to apply a sequence of left or right forces of fixed magnitude to the wheeled cart so that it swings up the pendulum from its natural pendant position and stabilizes in the inverted position, once it reaches the upright equilibrium point (Fig. 4.7). The cart must also be homed to a reference position on the rail. The force $F \in$ [-10, 10] newton's is applied to the cart and a zero magnitude force is not permitted. The total kinetic energy of the system is simply the sum of the kinetic energies of each mass. The kinetic energy T_C of the cart is

$$T_c = \frac{1}{2} M(\dot{x})^2 \tag{4.1}$$

The pole can move in both the horizontal and vertical directions. So the kinetic energy T_p of the pole is

$$T_p = \frac{1}{2} m((\dot{x}_2)^2 + (\dot{z}_2)^2) \tag{4.2}$$

From the free bodied diagram x_2 and z_2 are equal to

$$x_2 = x + L(\sin\theta) \tag{4.3}$$

$$z_2 = L(\cos\theta) \tag{4.4}$$

$$\dot{x}_2 = \dot{x} + L(\cos\theta)(\dot{\theta}) \tag{4.5}$$

$$\dot{z}_2 = -L(\sin\theta)(\dot{\theta}) \tag{4.6}$$

The total kinetic energy, T, of the system is equal to the

$$T = T_c + T_p \tag{4.7}$$

Substitute Eq. (4.1), (4.2), (4.5) and (4.6) in (4.7)

$$T = \frac{1}{2}M(\dot{x})^2 + \frac{1}{2}m(\dot{x})^2 + \frac{1}{2}mL^2(\dot{\theta})^2 + mL(\dot{x})(\cos\theta)(\dot{\theta}) \tag{4.8}$$

The potential energy, E of the system is stored in the pendulum so

$$E = mgz_2 \tag{4.9}$$

Substitute Eq. (4.4) in (4.9)

$$E = mgL(\cos\theta) \tag{4.10}$$

The Lagrangian function is

$$P = T - E \tag{4.11}$$

$$P = \frac{1}{2}(M+m)(\dot{x})^2 + mL(\cos\theta)(\dot{x})(\dot{\theta}) + \frac{1}{2}mL^2(\dot{\theta})^2 - mgL(\cos\theta) \tag{4.12}$$

The state space variables of the system are x and θ, so the lagrange equations are

$$\frac{d}{dt}\left(\frac{\partial P}{\partial \dot{x}}\right) - \frac{\partial P}{\partial x} = F \tag{4.13}$$

$$\frac{d}{dt}\left(\frac{\partial P}{\partial \dot{\theta}}\right) - \frac{\partial P}{\partial \theta} = 0 \tag{4.14}$$

The dynamics of the cart-pole system are modeled as follows by solving Eq. (4.13) and (4.14).

$$\dot{x}_1 = x_2 \tag{4.15}$$

$$\dot{x}_2 = \frac{((M+m)g(\sin x_1) - mL(\sin x_1)(\cos x_1)x_2^2 - (\cos x_1)F)}{L((M+m) - m\cos^2 x_1)} \tag{4.16}$$

$$\dot{x}_3 = x_4 \tag{4.17}$$

$$\dot{x}_2 = \frac{mL(\sin x_1)x_2^2 - mg(\sin x_1)(\cos x_1)}{((M+m) - m\cos^2 x_1)} + \frac{F}{((M+m) - m\cos^2 x_1} \tag{4.18}$$

Let the four state variables are

$x_1 = \theta$; angle of the pole with respect to the vertical axis.

$x_2 = \dot{\theta}$; angular velocity of the pole with respect to the vertical axis.

$x_3 = x$; position of the cart.

$x_4 = \dot{x}$; velocity of the cart.

F_1= Force applied to control the inverted pendulum angle.

$F_{2=}$ Force applied to control the inverted pendulum cart.

$F=F_1+F_2$; Total Force applied to control the inverted pendulum.

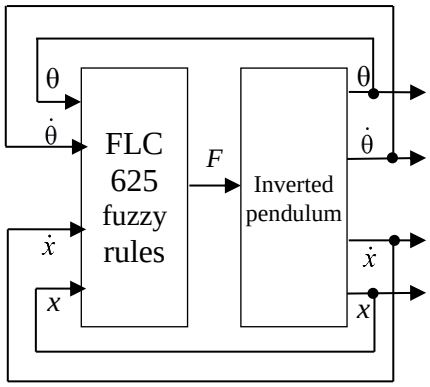


Fig. 4.8. Fuzzy Logic Control Loop with 625 fuzzy rules

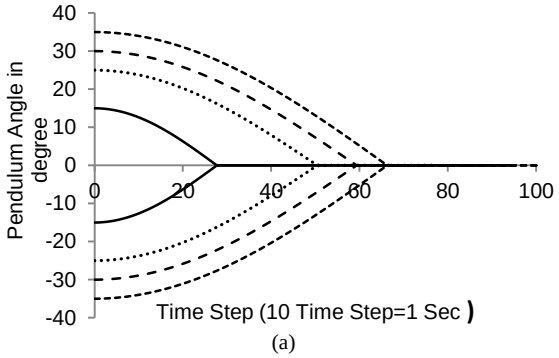
4.4.2 Automatic Design of FLCs of two degrees of freedom inverted pendulum

The main purpose of the fuzzy logic control of the system is to reset the pendulum angle θ and cart position x from every position to the desired one, specified in our case by $[x,\theta]^T = [0,0]^T$. If each state variable i is divided into n_i =5 linguistic terms then the number of rules is 5^4 =625 fuzzy rules. The general structure of the rule is

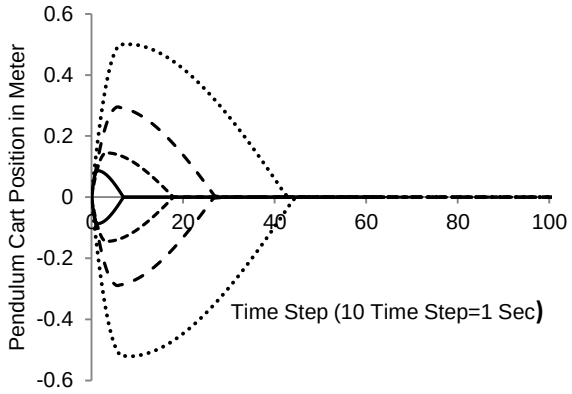
$$\textit{If } (\theta \textit{ is } A_1^i) \textit{ and } (x \textit{ is } A_2^i) \textit{ and } (\dot{\theta} \text{ is } B_1^i) \text{ and } (\dot{x} \text{ is } B_2^i)$$
$$\text{Then } (F_1 = C_1^i) \text{ and } (F_2 = C_2^i)$$

In this case we applied GA as a successful optimization method which has been shown effective in improving the tuning the parameters and learning the fuzzy rule. The fuzzy logic control loop is shown in Fig. 4.8. The details of simulation results are presented in Fig. 4.9 to validate the model in Fig. 4.8. The first one (Fig. 4.9) depicts the evolution of pendulum angle θ for different initial condition varying between -40^0 and $+40^0$: $\theta(0) = \{-35, -30^0 - 25^0, -15^0, 35^0, 30^0, 25^0, 15^0\}$. And $x(0)$ is to fixed 0^0. The second one (Fig. 10) is the evolution of x, $\theta(0)$ is fixed to 0^0.

In this case, our proposed algorithms generate a relative large rule base (625 fuzzy rules). If we choose the less number of linguistic levels (fuzzy sets) for each state variable then the rule base size is reduced. But this less number of fuzzy sets selections will affect the simulation performance.

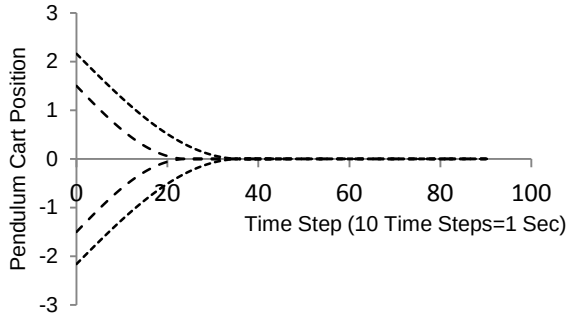


(a)

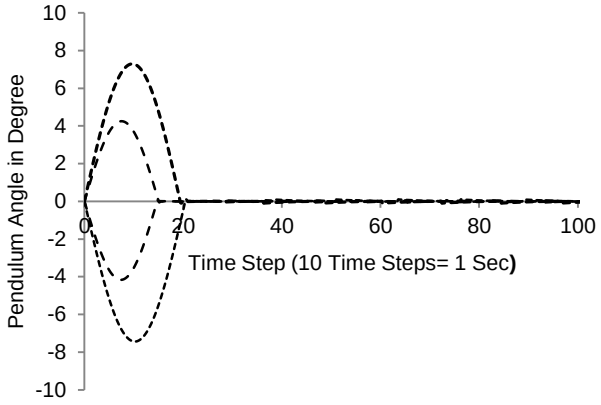


(b)

Fig. 4.9. (a) Evolution of pendulum angle θ for different initial condition varying between -40^0 and $+40^0$ while using 625 rules (b) Pendulum cart position is little bit swing up and stabilizing.



(a)



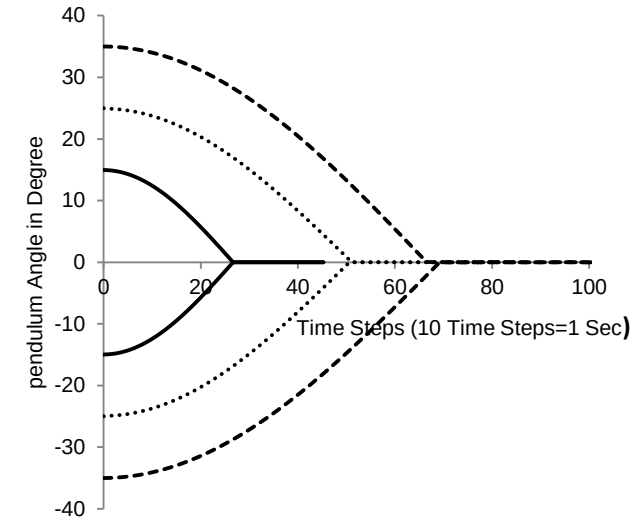
(b)

Fig. 4.10. (a) Evolution of pendulum cart position *x* for different initial condition varying between -3m and +3m while using 625 rules (b) Pendulum angle is little bit swing up and finally stabilizing.

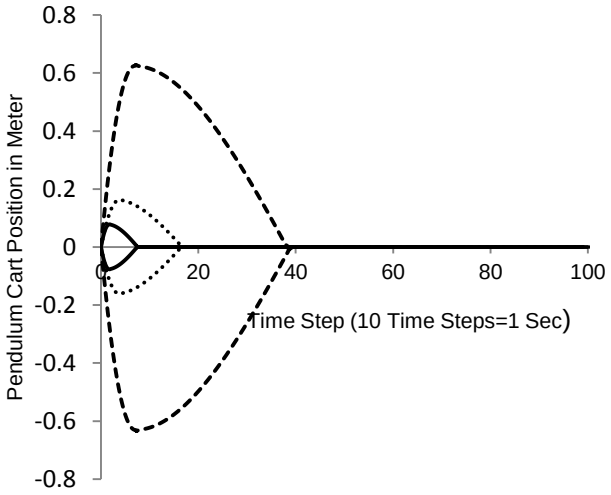
4.4.3 Genetic Reduction of Rule base of FLCs

In order to solve the rule explosion problem we apply our genetic rule base reduction technique, and then it takes 187 instead of 625 rules for the control of two degrees inverted pendulum. The first one (Fig. 4.11) depicts the evolution of pendulum angle θ for different initial condition varying between

-40^0 and $+40^0$: $\theta(0) = \{-35, -30^0 - 25^0, -15^0, 35^0, 30^0, 25^0, 15^0\}$. And $x(0)$ is to fixed 0^0. The second one (Fig. 12) is the evolution of x, $\theta(0)$ is fixed to 0^0.

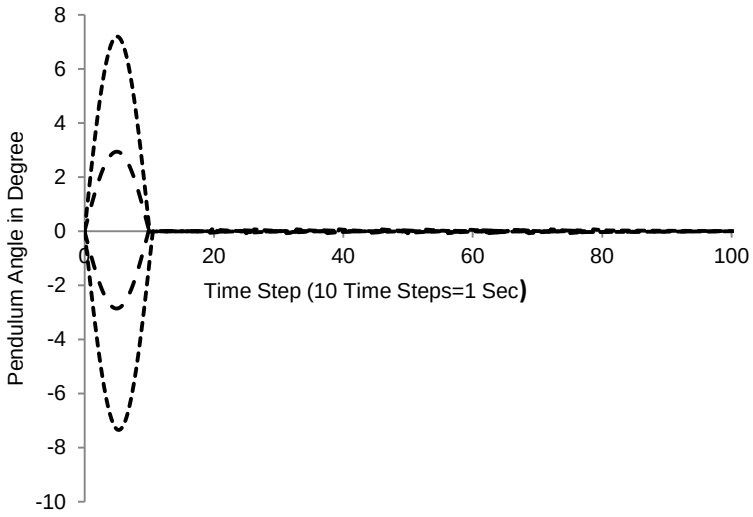


(a)

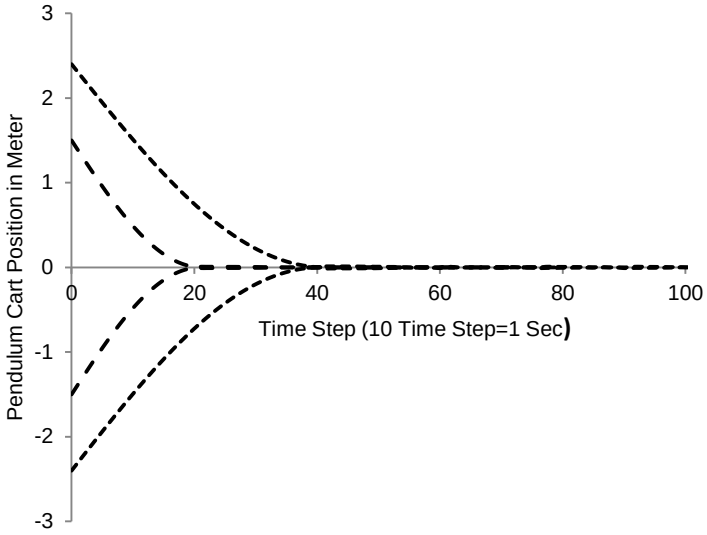


(b)

Fig. 4.11. (a) Evolution of pendulum angle θ for different initial condition varying between -40^0 and $+40^0$ while using 187 rules (b) Pendulum cart position is little bit swing up and stabilizing.



(a)



(b)

Fig. 4.12. (a)Evolution of pendulum cart position *x* for different initial condition varying between -3m and +3m while using 625 rules (b) Pendulum angle is little bit swing up and finally stabilizing.

4.4.4 Genetic Reduction of Rule base with decoupling approach of FLCs

To solve this explosion of fuzzy rules, at first, we decompose a complex single-block fuzzy controller (Fig. 4.8) into a decoupled FLCs depicted in Fig. 4.13. In this case one can consider the pendulum cart and rod independent from the other, yielding smaller automatic designed fuzzy logic controllers, FLC_1 and FLC_2. Here, FLC_1 is applied to control the pendulum angle and FLC_2 is used to control the pendulum cart.

Table 4.3. REDUCED RULE BASE WITH 16 RULES FOR FLC_1

Rule No.		Antecedents			Consequents	
R^1	If	θ=NL	and	$\dot{\theta}$ =NH	Then	F_1=ZE
R^2	If	θ =ZE	and	$\dot{\theta}$ =NH	Then	F_1=NH
R^3	If	θ =PL	and	$\dot{\theta}$ =NH	Then	F_1=NL
R^4	If	θ =PH	and	$\dot{\theta}$ =NH	Then	F_1=ZE
R^5	If	θ =NH	and	$\dot{\theta}$ =NL	Then	F_1=PH
R^6	If	θ =NL	and	$\dot{\theta}$ =NL	Then	F_1=ZE
R^7	If	θ =ZE	and	$\dot{\theta}$ =NL	Then	F_1=NH
R^8	If	θ =NL	and	$\dot{\theta}$ =ZE	Then	F_1=PL
R^9	If	θ =ZE	and	$\dot{\theta}$ =ZE	Then	F_1=PH
R^{10}	If	θ =PL	and	$\dot{\theta}$ =ZE	Then	F_1=ZE
R^{11}	If	θ =PH	and	$\dot{\theta}$ =ZE	Then	F_1=NH
R^{12}	If	θ =NL	and	$\dot{\theta}$ =PL	Then	F_1=ZE
R^{13}	If	θ =ZE	and	$\dot{\theta}$ =PL	Then	F_1=NH
R^{14}	If	θ =PL	and	$\dot{\theta}$ =PL	Then	F_1=ZE
R^{15}	If	θ =ZE	and	$\dot{\theta}$ =PL	Then	F_1=PL
R^{16}	If	θ =NL	and	$\dot{\theta}$ =PL	Then	F_1=ZE

Table 4.4. REDUCED RULE BASE WITH 14 RULES FOR FLC_2

Rule No.		Antecedents			Consequents	
R^1	If	x=NL	and	$\dot{x}$ =NH	Then	F_2=ZE
R^2	If	x =ZE	and	$\dot{x}$ =NH	Then	F_2=NL
R^3	If	x =PL	and	$\dot{x}$ =NH	Then	F_2=PH
R^4	If	x =NH	and	$\dot{x}$ =NL	Then	F_2=ZE
R^5	If	x =ZE	and	$\dot{x}$ =NL	Then	F_2=NH
R^6	If	x =NH	and	$\dot{x}$ =ZE	Then	F_2=NL
R^7	If	x =NL	and	$\dot{x}$ =ZE	Then	F_2=ZE
R^8	If	x =ZE	and	$\dot{x}$ =ZE	Then	F_2=PL
R^9	If	x =PL	and	$\dot{x}$ =ZE	Then	F_2=ZE
R^{10}	If	x =PH	and	$\dot{x}$ =ZE	Then	F_2=NH
R^{11}	If	x =NL	and	$\dot{x}$ =PL	Then	F_2=NL
R^{12}	If	x =ZE	and	$\dot{x}$ =PL	Then	F_2=ZE
R^{13}	If	x =PL	and	$\dot{x}$ =PL	Then	F_2=ZE
R^{14}	If	x =ZE	and	$\dot{x}$ =PH	Then	F_2=PH

Fuzzy control method is a good controller for controlling the single-input-single-output (SISO) system. It means that only one input could be controlled by a fuzzy controller in a time. However, the Fuzzy alone cannot be used successfully to control the cart's pendulum and the pendulum's angle in the same time. The results display that the cart moves in the negative direction with a constant velocity when an impulse force is applied to move it. However, in this study, by applying 2 fuzzy controllers for the system, the inverted pendulum was stabilized successfully. 'fuzzy controller 1- FLC_1' and 'fuzzy controller 2- FLC_2' were used to control the angle of the pendulum and the position of the cart respectively. As a conclusion, the linear model of inverted pendulum system can be stabilized successfully by applying two fuzzy controllers simultaneously if they are tuned properly. Both of the controllers were tuned simultaneously so that the control variable of the controllers is appropriately determined. The control variables (F_1 and F_2) were summed together to become a force (F) to the system.

Thus, the task of fuzzy logic control is similar to that in the previous section: moving from every position to a given final one given by $x_d = 0^0$ for the first sub controller (FLC_1) and $\theta_d = 0^0$ for the second one (FLC_2). Our proposed genetic learning and reducing technique is applied for generating and reducing the two fuzzy rule bases, one for controlling the pendulum cart and another one for the controlling the pendulum angle. For FLC_1 the fuzzy control rule base with 16 rules can be represented as a linguistic matrix shown in table 4.3 where the input variables θ, $\dot{\theta}$ and output variable F_1 with the linguistic level sets {NH: Negative High, NL: Negative Low, ZE: Zero, PL: Positive Low, PH: Positive High}, {NH, NL, ZE, PL, PH} and {NH, NL, ZE, PL, PH} respectively. For FLC_2 the fuzzy control rule base with 14 rules can be represented as a linguistic matrix shown in table 4.4 where the input variables x, $\dot{x}$ and output variable F_2 with the same linguistic level sets as FLC_1.

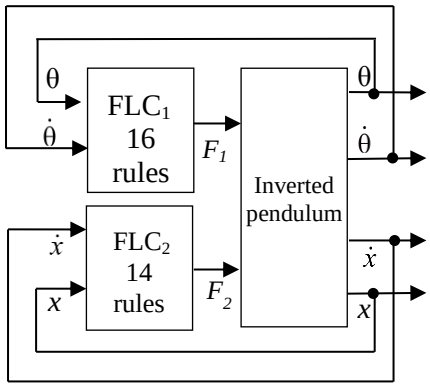
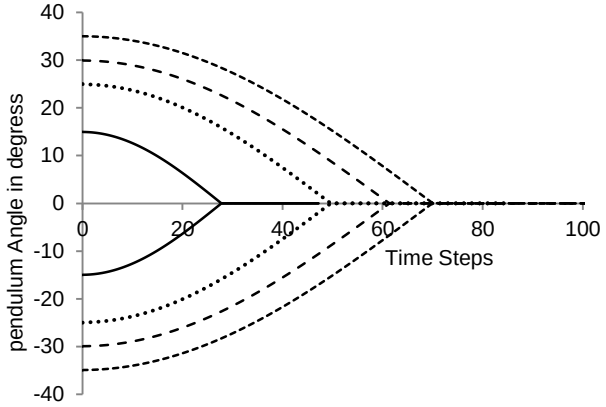


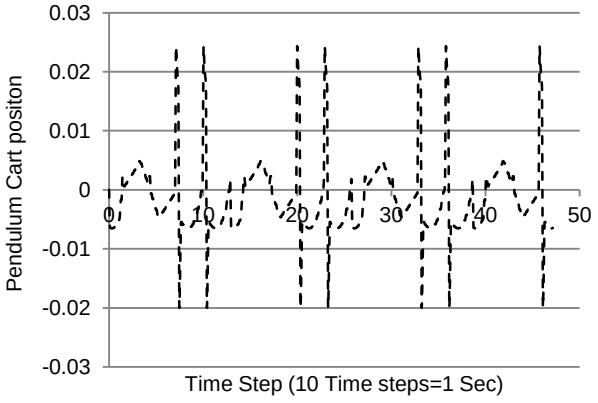
Fig. 4.13. Decoupled Fuzzy Logic Control Loop

The rule base can be established on a two dimensional space like Table 4.3 and Table 4.4. It is seen that the total number of rules is greatly reduced compared with conventional FLCs described in the previous section. In other words, a conventional fuzzy system has four states to be controlled and the range of each state variable is divided into five fuzzy sets. The number of rules forming the knowledge database is 5^4 =625. With the present decoupled method, only two variables need to be fuzzified, and only (16+14)=30 rules are necessary to establish the knowledge database.

Referring to the previous section, we have shown that the obtained fuzzy rule base with 625 rules for the conventional controller and obtained fuzzy rule base with 30 rules for the decoupled controller. In this section, we apply our genetic reduction technique to reduce the obtained rule base. The resulted fuzzy rule base contains 16 rules to control the pendulum angle and 14 rules to control the pendulum cart.



(a)



(b)

Fig. 4.14. (a) Evolution of pendulum angle θ for different initial condition varying between -40^0 and +40^0 while $x(0)$ is fixed 0 but (b) Pendulum cart position is little bit swing up after the stabilization for all cases.

The details of simulation results are presented in Fig.4.14 to validate our control algorithm for learning and reducing and get the optimal fuzzy model in Fig. 4.13. Figure 4.14 depicts the evolution of pendulum angle θ for different initial condition varying between -40^0 and + 40^0: $\theta(0) = \{-35, -30^0 - 25^0, -15^0, 35^0, 30^0, 25^0, 15^0\}$. And $x(0)$ is to fixed 0. In this case, the initial pendulum angle is positively or negatively big, and the pendulum cart position stands up ($x(0) = 0$), the fuzzy logic controller starts the pendulum angle control first. The pendulum angle is moved toward the desired direction such that the pendulum cart is swing up in a little bit and the pendulum system is completely stabilized within few seconds.

The second one (Fig. 4.15) is the evolution of cart position x for different initial condition varying between +3m and -3m while $\theta(0)$ is fixed to 0^0. Since at the control beginning the cart position is big and the pendulum stands up, the fuzzy logic controller starts the cart position control first. The cart is moved further toward the desired direction ($x_d = 0$ m) such that the pendulum angle is inclined to about $\theta < \pm 7^0$. After that, the pendulum angular control takes the highest priority order

over the pendulum cart position control, and the fuzzy logic controller moves the cart towards the desired position so that the pendulum angle is rotated towards the upright position. As a result the inverted pendulum system is completely stabilized within some 4.2 (42 time steps) seconds.

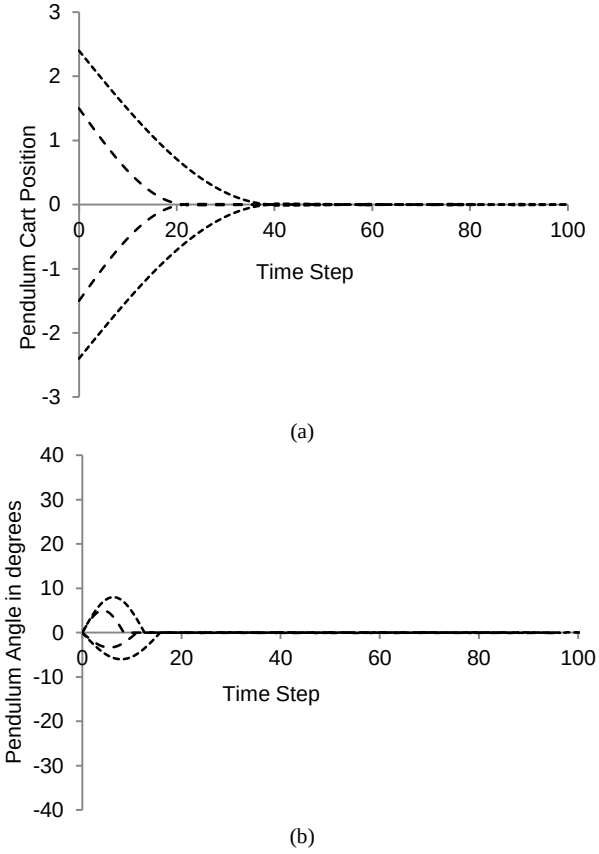


Fig. 4.15. (a) Evolution of pendulum cart position *x* for different initial condition varying between -3m and +3m while $\theta(0)$ is fixed 0 but (b) Pendulum angle is little bit swing up and finally stabilizing.

We evaluate the proposed control algorithms by using its error rate. For quantifying the errors, we use three different performance criteria to analyze the rise time, the oscillation behavior and the behavior at the end of transition period. These three criteria are: *Integral of Square Error* $(ISE = \int_0^\infty [et]^2 dt)$ *Integral of the Absolute value of the Error* $(IAE = \int_0^\infty |e(t)| dt)$ and *Integral of the Time multiplied by Square Error* $(ITSE = \int_0^\infty t[et]^2 dt)$. Figure 16 show the oscillation, a rise time and errors at the end of transition period that the system using our control algorithms.

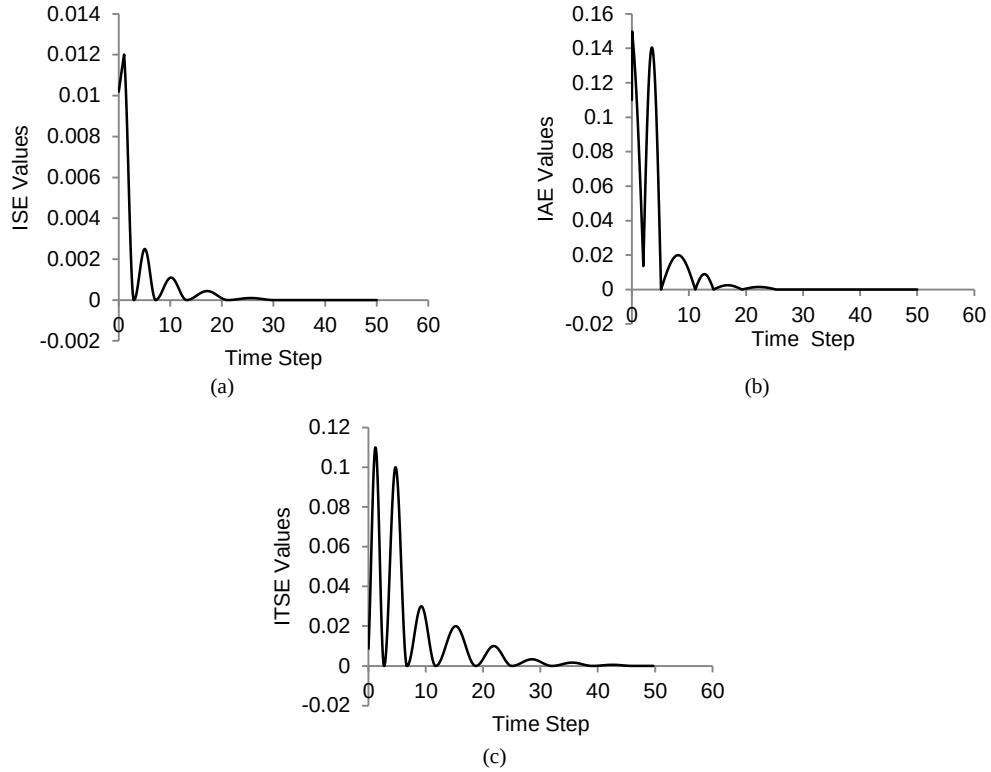
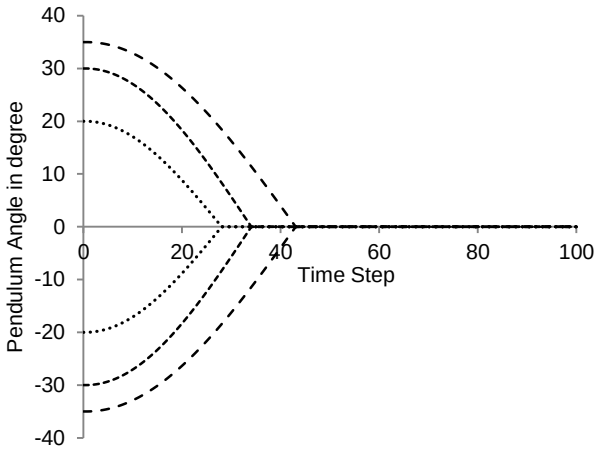


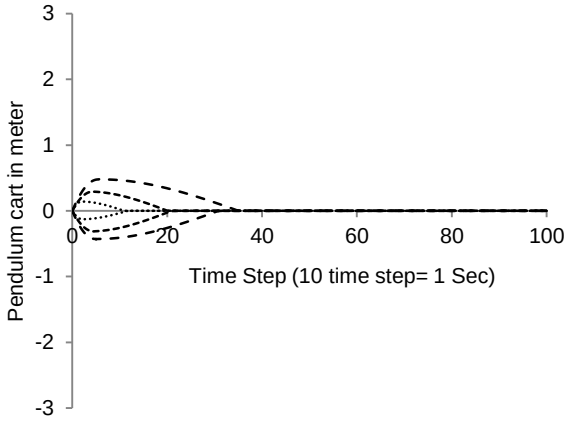
Fig. 4.16. Various error values of the tuned FLCs with reduced rule base for the initial pendulum angle θ =0.1 rad while $x(0)$ is fixed to 0 (a) Integral of square Error (ISE): Rise time of the system response is short. (b) Integral of the absolute Error (IAE): The system is less oscillatory for the tuned FLC before becoming stable. (c) Integral of the time multiplied by square error (ITSE): The error at the end of transition period is less important for the tuned FLC

4.4.5 Generalization ability of proposed controller

In order to illustrate the generalization ability of the proposed controller, the pendulum length is changed while the other parameters remained fixed. Figure 4.17 shows the simulation results where the pendulum length is 0.2m. Since the pendulum length is rather short, a little amount of cart oscillate can cause the pendulum to rotate because the pendulum has high natural frequency. Therefore, even though pendulum cart tends to oscillate a little, the pendulum is balancing upright in a short period of time. The complete successful stabilization time in this case is 4.2 secs.



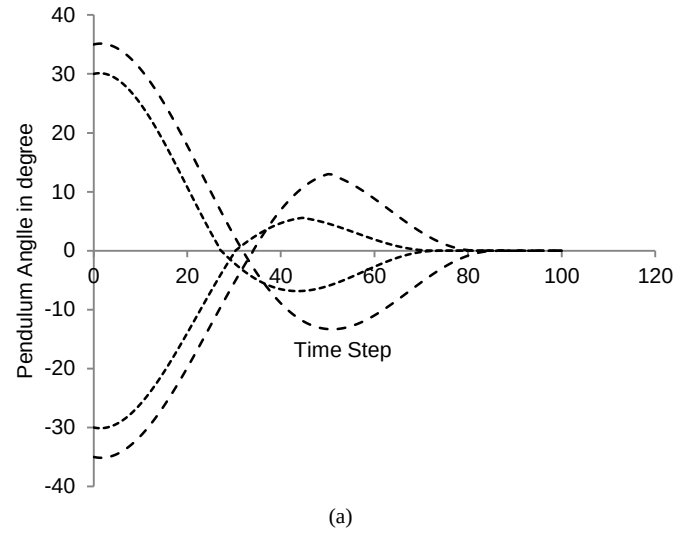
(a)



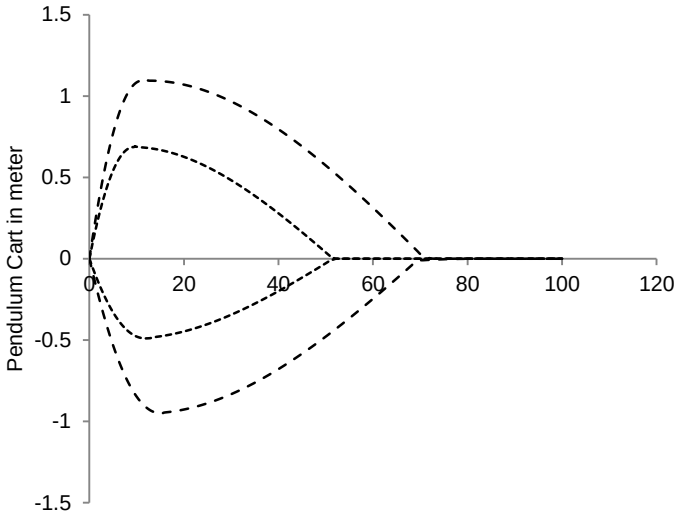
(b)

Fig. 4.17. Control result of pendulum (a) angle θ for different initial condition varying between -40^0 and $+40^0$ while $x(0)$ is fixed 0 (b) pendulum cart when pendulum length is 0.2 m

When the pendulum length is long, the pendulum has the small natural frequency of oscillation and a big momentum. In this situation, the cart has to move for a long distance for balancing the pendulum. Figure 4.18 shows an example where the pendulum length is 2.2m. The complete swing up and successful stabilization time in this case is 8.2 sec.

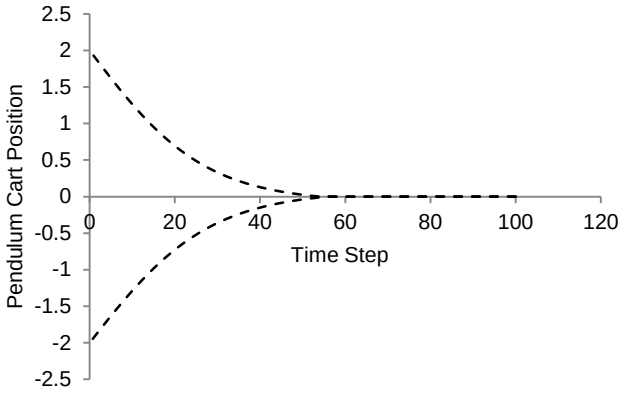


(a)

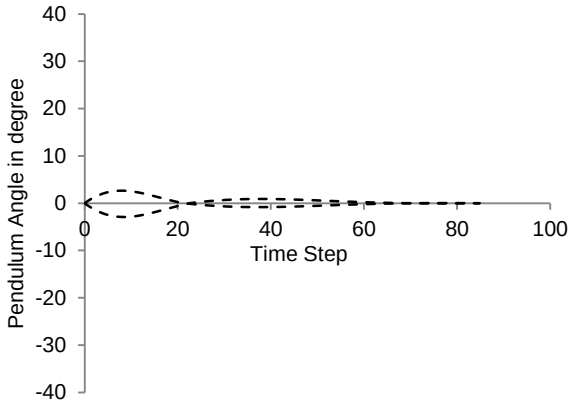


(b)

Fig. 4.18. (a) Evolution of pendulum angle θ for different initial condition varying between -400 and +400 while $x(0)$ is fixed 0 but (b) Pendulum cart position is swing up and finally stabilized.



(a)



(b)

Fig. 4.19. (a) Evolution of pendulum cart position *x* for different initial condition varying between -3m and +3m while $\theta(0)$ is fixed 0 but (b) Pendulum angle is little bit swing up and finally stabilizing when the pendulum length is 2.2m

From Figs. 4.18 - 4.19 it is shown that control results changes due to the pendulum length. When the pendulum length is short, the pendulum balanced smoothly within a short period of time with small cart oscillation. When the pendulum length is long enough, the pendulum balanced smoothly within a long period of time with large cart oscillation.

4.4.6 Evaluating our study

In order to evaluate our new genetic reduction and learning based control algorithm**,** we compare it against well-known reinforcement learning method called SANE (Symbiotic, Adaptive Neuro-Evolution) [98] and other existing methods [99]-[103]. The control system for cart-pole inverted system is swing up and stabilized within a few seconds or time steps in several studies including [93]. The closer observation of the study in [98] reveals that the pendulum angle is swing up and stabilized within 46 time steps for the best case and 4461 for the worst cases from initial pole angles $\pm 15^0$ degrees and random cart positions ±2.4 meters. For the same initial condition, the number of time steps required to stabilize the cart-pole system is 31 for best case and 2123 is for the worst case while using our control algorithm and reduced rule base.

Since most of the classical fuzzy system technique will cause the exponential increase in complexity, Horikawa et al. [104], Lin and Lee [105], Partricar and Provence[106] and Takagi and

Hayashi [103] proposed a methods incorporating neural network into Fuzzy system. In comparison with the fuzzy-neural hierarchical controller [99], it took more than 12.0 s to realize asymptotic stabilization with some offset left when the pendulum angle is 30^0. Kawaji [100] found in his proposed technique that it was difficult to achieve complete successful stabilization even after 20.0 s. Kyung and Lee [101] presented a fuzzy controller, whose rule base was derived from three neural networks. Although the fuzzy controller can stabilize an inverted pendulum system in about 8.0s, it needs 396 rules. Sakai and Takahama [103] applied a nonlinear optimization method to train a fuzzy controller for stabilization. However, the controller spent more than 200.0 s on stabilizing an inverted pendulum system. Pan [102] presented an optimal and robust variable structure systems controller and his controller needed about 8.0 s to finish complete successful stabilization. On the other hand the proposed fuzzy controller build by the optimal number of fuzzy rules can stabilize completely a wide range of the inverted pendulum systems with 9.0 s. In future, we will compare it against the well-known traditional controllers, proportional derivative (PD) and proportional integral derivative (PID) controllers.

4.5 Conclusions

This study presents the automatic design method to construct the optimal fuzzy logic controller including the genetic rule base learning and reduction technique in order to realize the optimal control of nonlinear systems. The main contributions of our study are as follows: Firstly, the proposed control algorithm is able to automatically generate the fuzzy sets and secondly, the automatic rule generation and genetic rule base size reduction schema at the same time. Thirdly, rule base size reduction method use in proposed decoupling approach and its application to two degrees of freedom inverted pendulum. The advantage of using this decoupled control architecture and rule base reduction technique is that the number of rules used in the fuzzy knowledge base has been reduced substantially also. Simulation results are given illustrating the efficient use of the proposed algorithm and a comparison with an existing controller was made to show the effectiveness of the method.

CHAPTER 5

ADAPTIVE HIERARCHICAL FUZZY SYSTEM

5.1 Introduction

During the design of fuzzy logic controllers (FLCs) one of the most important factors is how to reduce the number of involved fuzzy control rules and their corresponding computation requirements. The size of the rule base increases exponentially with the number of input variables and their corresponding linguistic variables [107]. If $(x_1, x_2, \ldots\ldots x_n)$ is the inputs variables and $(I_1, I_2, \ldots\ldots I_n)$ are their corresponding number of linguistic terms then the number of candidate rules is $r = I_1 \times I_2 \times I_3 \times \ldots\ldots I_n$.

Many researchers have already observed this difficulty and various approaches have been investigated to overcome the problem. One approach is to reduce the number of fuzzy sets (MFs) that each input involve. However, this may reduce the accuracy and efficiency of the system [6]. As an alternative, the number of rules in the rule base can also be trimmed if it is known that some rules are never used [108]. In certain circumstances, the rule base size reduction process to design FLCs yields unsatisfactory results [109]. The most obvious is to limit the number of inputs that the system is using. Again, this may reduce the accuracy of the system, give the unsatisfactory performance and in many cases, render the system being modeled unusable. Another most important idea of using hierarchical structure in designing a FLCs has been investigated [110]-[112] also. Raju and Zhou presented a layer based hierarchical fuzzy control system in [110]-[111] where they used fixed hierarchical structure. Their hierarchical controller works from layer to layer where the most important systems variables are selected for the first layer, the next most important variables for the second layer and so on. Yager [112] presented a hierarchical control structure called hierarchical prioritized structure (HPS) where more specific fuzzy rules to override the more general ones. Chung el al. [113] has presented a hierarchical fuzzy controller to reduce the fuzzy rule base. In their work computational time is related to the number of layer due to the fuzzification and defuzzification is performed at every layer.

On the other hand, a number of researcher focuses on automatically finding the proper structure and parameters of fuzzy logic system using evolutionary programming (EA) [114], genetic algorithm (GA) [115]-[116] and tabu search [117] and so on. Karr, Freeman, and Meredith proposed a combination of fuzzy logic and a GA [118]. A GA finds fuzzy rules using the payoff for the success/failure of its actions. A GA was applied to identification of the hierarchical structure of the fuzzy model [119] from given input-output pairs of data. Matsushita, et al.[120] and Furuhashi, et al. [120] have applied a GA to selection of input variables in hierarchical models. This method is very effective in the case where the plant has a strong nonlinearity. Increasingly, genetic algorithms (GAs) are used to optimize the parameters of fuzzy logic controllers (FLCs). Although GAs

provides a systematic design approach, the optimization process is generally performed off-line using a plant model.

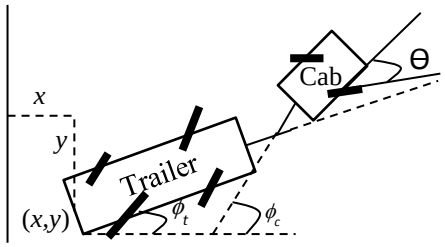


Fig. 5.1. The Trailer-Cab System

But the key factors to ensure successful HFLC design includes identifying and characterizing suitable hierarchical structure, selecting the inputs for each fuzzy sub-model of hierarchical fuzzy model, and optimizing the parameters of antecedent and consequent for a fuzzy model. This study presents a new systematic technique *for* designing a hierarchical fuzzy model. In our proposed method, evolutionary mechanism is used in two phases simultaneously: first phase involves identify the optimal hierarchical structure using the genetic programming (GP), while the second phase performs an optimal tuning of antecedent and consequent parameters of fuzzy control rules and their corresponding MFs. The fine tuning of the antecedents and consequents rule's parameters and their corresponding MF's parameters encoded in the structure is accomplished using genetic algorithm (GA).The main idea of this paper is in the usage of evolutionary mechanism for selecting the fuzzy sets and for constructing a hierarchical architecture of fuzzy model automatically. This study differences between the previous approaches lie mainly adaptive selection of hierarchical architecture through genetic programming (GP). Comparing to the existing controllers our main objective is not the reduction of inputs (existing backing up a truck controller ignore one inputs, y,[3]), but the simplest possible controller with the lowest possible number of fuzzy rules.

5.2 Problem Statement: Trailer-And-CAB

The trailer-and-cab reverse parking control problem is a standard test application for complex control approaches, with its complexity originating from the non-linear nature of the problem. In the majority of control approaches, the primary objective is to back up a simulated trailer-and-cab to a loading dock in planner parking lot as first as possible, depicted in Fig. 5.1. The four state variables ϕ_t, ϕ_c, x and y exactly determine the trailer-and-cab position. The angle ϕ_t specifies the angle of the trailer with the horizontal. The cab part is characterized by angle ϕ_c between its onward direction and x-axis and its dimensions are 2×2m. The coordinate pair (x, y) specifies the position of the rear center of the trailer in the plane. Length and width of the trailer are 4 and 2 meters, respectively.

The current implementation of the trailer-and-cab controller uses the following set of dynamic equations:

$$x(t+1) = x(t) - B\cos(\phi_t(t))$$
$$y(t+1) = y(t) - B\sin(\phi_t(t))$$
$$\phi_t(t+1) = \phi_t(t) - \arcsin\left(\frac{A\sin(\phi_t(t)) - \phi_t(t)}{l_t}\right)$$
$$\phi_c(t+1) = \phi_c(t) - \arcsin\left(\frac{b\sin(\theta)}{l_c + l_t}\right)$$

where $A = b\cos(\theta)$, $B = A\cos(\phi_c(t) - \phi_t(t))$ and r is the distance covered by the wheels of cab part in one time step (0.1s), l_c is the length of cab and l_t is the length of trailer. The goal is to make the trailer arrive at the loading dock at a right angle $(\phi_t = 90^0)$ and to align the position (x, y) of the trailer with the desired loading dock (x_f, y_f) and the trailer with cab is allowed to move backward by some fixed distance b at every stage. The loading zone corresponded to the plane $[0,-50]\times[100,50]$ and (x_f, y_f) equaled (0, 0). At every stage our proposed approach should produce the output angle θ and angle ϕ_t of the trailer with horizontal that backs up the trailer to the loading dock from any initial state. The control goal is to synthesize a fuzzy controller:

$$\theta = f(x, y, \phi_t, \phi_c)$$

In the trailer-and-cab driving state, one condition that the angle between the trailer and cab must not exceed a predefined threshold during the riding. The jackknife phenomenon occurs when reaching this threshold value regardless of the turning angle. The turning angle is the angle between the trailer and the cab increases whenever driving backward.

The variable ranges were as follows:

$$0 \le x \le 100$$
$$-50 \le y \le 50$$
$$-90^0 \le \phi_t \le 270^0$$
$$-90^0 \le \phi_c \le 90^0$$
$$-30^0 \le \theta \le 30^0$$

5.3 Hierarchical fuzzy system: Architectute Design and Methodology

5.3.1 Hierarchical Fuzzy System

A Hierarchical Fuzzy Systems (H-FS) not only provide a more complex and flexible architecture for modelling nonlinear systems, but can also consequently relax the rule explosion phenomenon. Fig. 5.2 depicts the typical structure of hierarchical fuzzy system where the input variables are put into a collection of low-dimensional fuzzy logic units (FLUs) and the outputs of the FLUs are used as the input variables for the FLUs in the next layer. According to them, the number of fuzzy rules that are employed in the hierarchical fuzzy system (HFS) is shown to be proportional to the number of input variables. In the hierarchical architecture, the number of rules will increase linearly whereas it is exponential in conventional counterpart. In hierarchical fuzzy systems the number of rules is altered by decomposing the fuzzy system to a set of simpler fuzzy subsystem connected in a hierarchical manner as shown in Fig. 5.2. Its main objective is to implement the equivalent control functionality with layer based hierarchical architecture of simple fuzzy controller. The simple fuzzy controllers involve lower number input variables which lead to a smaller number of fuzzy rules.

5.3.1.1 Structure Identification

To define the structure of hierarchical controller i.e., number of sub-controller, how the sub-controllers are connected to each other, define the number of inputs for each sub controller are very difficult task for designing a hierarchical fuzzy system. In this study we automatically generate the hierarchical structure through genetic programing (GP).

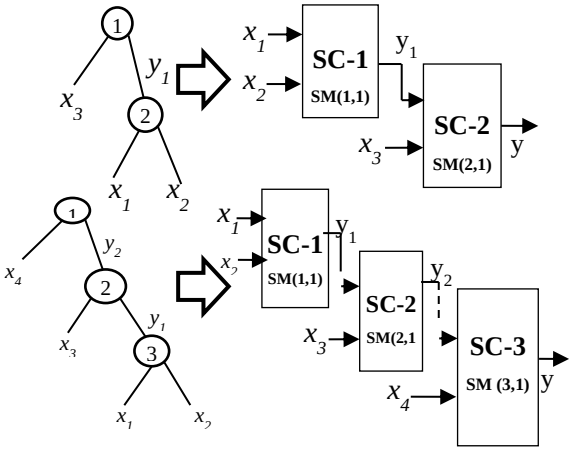


Fig. 5.2. Typical Tree and their corresponding Hierarchical Architecture

5.3.1.2 Genetic Modeling of Hierarchical Structure

Genetic programming is a powerful search technique which belongs to the class of the evolutionary algorithms. Inspired by the darwin's theory, these algorithm works by simulating the evolution of structures by means of fitness-based natural selection and genetic operators, such as reproduction, crossover and mutation. In GP individuals are computer programs, generally represented as trees. In this work, the trees represent a fuzzy controller, in which hierarchical structure, encoded into trees, as illustrated in Fig. 5.2.

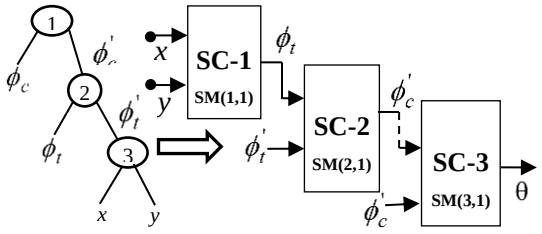


Fig. 5.3. Optimal Hierarchical Fuzzy Logic Architecture for trailer-and-cab

5.3.1.3 Encoding

A tree structure based encoding schema is used for identify the hierarchical optimal structure of hierarchical fuzzy system. Figure 5.2 shows the hierarchical tree schemas are encoded into their corresponding hierarchical fuzzy system. The tree encoded into individual is (1, 2, 3, x_1, x_2, x_3, x_4) of GP represents a potential architecture of hierarchical fuzzy logic controller (HFLC) where (1,2,3) are non-leaf nodes define the sub-model and (x_1, x_2, x_3, x_4) are leaf nodes define the input variables. Then the optimized tree through GP is decoded into hierarchical architecture.

In a HFLC structure the most influential parameters are chosen as the system variables in the first level, the next most important parameters are chosen as the system variables in the second level, and so on. In this hierarchy, the first level gives an approximate output which is then modified by the second level rule set. This procedure can be repeated in succeeding levels of hierarchy. The number of rules in a complete rule set is reduced towards a linear function of the number of variables by the hierarchy. The conventional fuzzy logic controller knowledge-based control - would presently fail because there is no explicit knowledge on how to drive the trailer-and-cab and thus we cannot synthesize the controller function. In this way, decomposing this control tasks into several parts that could logically designed through hierarchical architecture. Figure 5.3 shows the automatically selected smaller size hierarchical architecture for the trailer-and-cab controller.

Now we describe how the above hierarchical controller works. The sub controller, SC-1, takes position coordinates (*x, y)* as its inputs and determines the expected trailer angle, $\phi_t^{'}$, for this position. The second fuzzy logic controller, SC-2, in the hierarchical architecture creates a mapping $\phi_t^{'} - \phi_t \rightarrow (\phi_t - \phi_c)^{'}$. Because the input of the fuzzy logic controller (SC-2) the error of the trailer angle; it can be regarded as a proportional controller that determines the angle difference of cab and trailer parts that is necessary to obtain the expected angle of the trailer. Finally, SC-3 is used to determine the appropriate steering angle, i.e., generate the control action.

In this section, we describe an illustrative example through the Fig. 5.3 to show how the overall output of the hierarchical fuzzy model is computed. The overall output of hierarchical fuzzy model is calculated from layer to layer. Assume that each input variable domain is divided into two intervals depending on the fuzzy set distribution; a particular input value belongs to two consecutive fuzzy sets. In this paper, it is proposed to use gaussian/triangular membership functions is defined by

$$Gaussian\ (x : m, \sigma) = \exp\left(- \frac{(x-m)^2}{\sigma^2} \right)$$

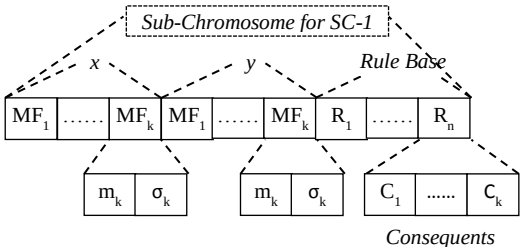


Fig. 5.4. Sub-chromosome coding schema for SC-1

In our proposed hierarchical model, the output $y_i = \phi_t^{'}$ of the sub model (SC-1) unit of first layer is computed. Assume that the input space of *x* and *y* divided into two non-equal intervals and can be represented by two fuzzy sets A_{11}, A_{12} and B_{21}, B_{22}, respectively. The consequent parameters of a

rule base are $c_{ij}^0, c_{ij}^1, c_{ij}^2$ where *i=1,2 and j=1,2* respectively. Therefore, the structure of fuzzy control rules of this sub model is as follows:

$$R_{ij}: if\ x\ is\ A_{1i}\ and\ y\ is\ B_{2j}\ Then$$

$$y_{ij} = c_{ij}^0 + c_{ij}^1 x + c_{ij}^2 y\ \ for\ i = 1,2\ and\ \ j = 1,2$$

The sub model 1 output can be calculated as follows:

$$y(\phi_t^{'}) = \frac{\sum_{i=1}^{2}\sum_{j=1}^{2}\sigma_{ij} y_{ij}}{\sum_{i=1}^{2}\sum_{j=1}^{2}\sigma_{ij}}$$

Where $\sigma_{ij} = \mu_{A_{1i}}(x)\mu_{B_{2j}}(y)$ for i=1, 2 and j=1, 2.

Secondly, the overall output of the second fuzzy model (SC-2) is computed. It has also two inputs ϕ_t, and $\phi_t^{'}$, the output of the first sub model (SC-1). The used fuzzy sets for variables are: $B_{11}, B_{12}, B_{21}, B_{22}, B_{31}$ and B_{32} , respectively. The parameters in the consequent parts in the rule base are $d_{ij}^0, d_{ij}^1, d_{ij}^2$, and d_{ij}^3 (*i*=1, 2 and *j*=1, 2).

The complete rule base of the sub model (SC-2) is described as follows:

$$R_{ij}: if\ \phi_t\ \ is\ B_{1i}\ and\ \phi_t^{'}\ is\ B_{2j}\ Then$$

$$z_{ij}(\phi_c^{'}) = d_{ij}^0 + d_{ij}^1\phi_t\ + d_{ij}^2\phi_t^{'}\ \ for\ i = 1,2\ and\ \ j = 1,2$$

The overall output of the sub-model (SC-2) is determined by the following equation:

$$z(\phi_c^{'}) = \frac{\sum_{i=1}^{2}\sum_{j=1}^{2}\sigma_{ij} z_{ij}}{\sum_{i=1}^{2}\sum_{j=1}^{2}\sigma_{ij}}$$

Where $\sigma_{ij} = \mu_{B_{1i}}(\phi_t\ )\mu_{B_{2j}}(\phi_t^{'})$ for *i*=1, 2 and *j*=1, 2.

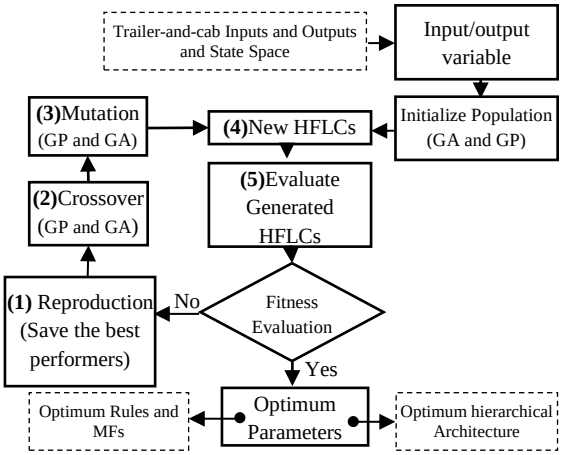


Fig. 5.5. The HFLC and Evolutionary cycle for optimization

In this way, the final control action θ can be calculated. In a traditional FLC, all the rules are used in a single step, when calculating the output of the controller. In our proposed hierarchical approach, the system input variables and consequent rules are divided into several levels in such a way that the most influential variables are chosen in the first level, the next most important variables are chosen in the second level, and so on. On the other hand, the output variable of each level is selected as the input variable in the following level.

5.3.2 Optimization of Fuzzy sub controllers/models

After selecting the appropriate hierarchical architecture, number of sub-controllers with their corresponding input variables through the evolutionary mechanism. Then tune the antecedents and consequents parameters of fuzzy control rules with their corresponding MFs of each sub-controller through GA.

5.3.2.1 Coding of Rule base and their Corresponding MFs

The coding scheme allows a flexible representation of fuzzy rules and their corresponding MFs in a genetic string. The consequent parameters of each rule base and their corresponding MFs are coded as real numbers as shown in Fig. 5.4.

5.3.3 Number of Rules Determination

For multi-level hierarchical control architecture with *L* levels of rules, *n* system variables, *m* fuzzy sets per variable and n_k variables (including the output variable in the previous level) in the k^{th} level of the hierarchy, the total number of rules is determined as follows:

$$T = \sum_{i=1}^{L} m^{n_k}$$

with $n_1 + \sum_{i=2}^{L} (n_k - 1) = n$. If each level used the constant number of input variable i.e., $n_1 = n_2 = n_3 = \ldots\ldots n_L$, the total number of rules is determined as follows:

$$T = \left(\frac{n-N}{N-1} + 1 \right) * m^N$$

with $N = n_1 = n_2 = \ldots\ldots = n_L$

In our proposed hierarchical approach, we have shown that the number of rules in a complete rule base is a linear function of the number of variables while in a conventional one it is an exponential function of number of variables.

5.3.4 Evolutionary Mechanism

In this section, an automatic design method for hierarchical fuzzy controllers is presented. The hierarchical structure is created and optimizes if-then rule's parameters for creating a compact fuzzy rule base of fuzzy models. The rule parameters encoded in the structure is accomplished using evolutionary algorithm. GAs[95] has been used as an adaptive technique to search for the optimum parameters for fuzzy membership functions and/or to match the suitable consequent fuzzy sets to rule premises (optimization of rules).

Initially, the GP starts with a randomly generated hierarchical architecture and GAs starts with a randomly generated fuzzy control rules and their corresponding membership functions, then GP

tries to improve the hierarchical architecture and GA further tries to fine tunes the fuzzy control rules and their corresponding membership functions. GP then goes back to improve the hierarchical architecture structure and the GAs goes back to improve the fuzzy control rules parameters with their corresponding membership functions. This procedure continues until a satisfactory optimum results for designing a hierarchical fuzzy controllers with hierarchical architecture is found or a termination criterion is met.

In this paper, we use general GP algorithms for hierarchical structure identification but use real coded GA [95] for optimizing the sub-controllers. The GA[95] is a nontraditional genetic algorithm which combines a conservative selection strategy that always preserves the best individuals found so far with a radical, highly disruptive recombination operator that produces offsprings that are maximally different from both parents. This GAs presents a good trade-off between exploration and exploitation, being a good choice in problems with complex search spaces.

The design procedure of the evolutionary hierarchical fuzzy system, as shown in Fig. 5.5, may be carried out as follows:

5.3.4.1 Initialization

All the individuals (GP and GA) in populations are initialized. At the start of the process, the initial populations comprise a set of individuals that are scattered widely throughout the search space through randomly initialization. Thus, the use of heuristic knowledge is minimized.

5.3.4.2 Evolution of Control performance

Individuals are evaluated. Each individual of GP in the population decides a hierarchical architecture and parallel each individual of GA in the population decides a combination of the fuzzy control rules and their associated MFs. A model is identified using only the decided parameters through the individual.

A fitness function is used for the evaluation of the individual. In this study, the *evaluation* is performed through weighted sum of position and angle errors. Trailer-and-cab only moves with a fixed speed 2 m/s. Backing of the trailer with cab is considered successful if the following criterion (3) is met:

$$\left|\phi_c - \phi_t\right| \le 60$$

To evaluate the performance of a hierarchical fuzzy system we use a weighted sum of position and angle errors, trajectory and docking errors given below. The weighted sum of position and angle errors is as follows:

$$\varepsilon_c = \varepsilon_x + \varepsilon_y + 0.0267\varepsilon_\phi$$

where

$$\varepsilon_x = \left|x_f - x(T_f)\right|$$
$$\varepsilon_y = \left|y_f - y(T_f)\right|$$
$$\varepsilon_\phi = \left|\phi_f - \phi_t(T_f)\right|$$

Where T_f is the duration of the backing.

After each individual is evaluated and associated with its fitness, the current population undergoes the reproduction process to create the next generation of the population.

Generally, the overall fitness of the next generation population improves. After the evaluation of control performance of the initial randomly generated population i.e. the potential solution of each sub-controller, the GP and GAs begins the creation of the new generation i.e., new hierarchal FLCs generation. The chromosomes of the current population are merged with the offspring population obtained from it. Then the best individuals are selected to take part in the population of next new population.

5.3.4.3 Evaluation, reproduction, crossover, and mutation

Each population generated in this manner then goes through a series of evaluation, reproduction, crossover, and mutation.

The design procedure is stopped when the termination condition is reached. Otherwise, the procedure from (1)-(5) in Fig. 5.5 is repeated. After the evolution process, the generated final population would consist of highly fit parameters that can provide good optimal solutions. In this manner, the evolutionary hierarchical fuzzy model with small size hierarchical architecture is obtained.

Table 5.1. (a) FUZZY RULE MATRIX FOR SC-1 (b) FUZZY RULE MATRIX FOR SC-2

(c) FUZZY RULE MATRIX FOR SC-3

TABLE 5.1 (a).

ϕ_t'		x				
		LE	LC	CE	RC	RE
y	LE	[1]LB	[2]VE	[3]RU	[4]VE	[5]RB
	LC	[6]LU	[7]ZE	[8]RB	[9]RU	[10]LU
	CE	[11]VE	[12]VE	[13]LU	[14]RU	[15]VE
	RC	[16]VE	[17]LB	[18]LU	[19]LB	[20]RB
	RE	[21]RU	[22]RB	[23]VE	[24]VE	[25]LB
Linguistic Level- LE: Left End, LC: Left Corner, CE: Center, RC: Right Corner, RE: Right End						

TABLE 5.1 (b).

ϕ_c'		ϕ_t'				
		LB	LU	VE	RU	RB
ϕ_t	LB	[1]ZE	[2]PO	[3]PO	[4]ZE	[5]NE
	LU	[6]PO	[7]NE	[8]PO	[9]ZE	[10]NE
	VE	[11]NE	[12]ZE	[13]ZE	[14]NE	[15]PO
	RU	[16]ZE	[17]ZE	[18]ZE	[19]PO	[20]NE
	RB	[21]PO	[22]NE	[23]ZE	[24]ZE	[25]PO
Linguistic Level: LB: Left Below, LU: Left Upper, VE: Vertical, RU: Right Upper, RB: Right Below						

TABLE 5.1 (c)

θ		ϕ_c'		
		NE	ZE	PO
ϕ_c	NE	VE	LB	RU
	ZE	RB	VE	VE
	PO	VE	LU	RB
Linguistic Level: NE: Negative, ZE: Zero, PO: Positive				

5.4 Simulation Results And Discussion

In order to test the designed hierarchical fuzzy logic controller, the trailer with cab is backed up to the loading dock from several different initial states. The selection of initial positions covers the whole loading zone and includes some really difficult starting states for all the given initial states the proposed approach converges to some fuzzy control rules. The following are fuzzy control rules for sub controllers Table 5.1(a), 5.1(b), 5.1(c).) The corresponding control surface of the fuzzy rule matrix for the sub-controller 1(SC-1) with two input variables *x* and *y* and an output variable $\phi_t^{'}$ is shown in Fig. 5.6.

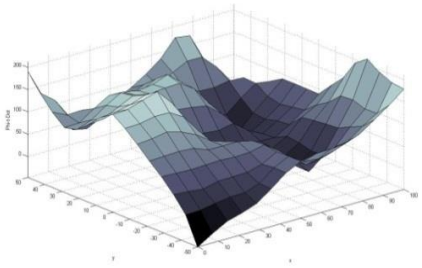
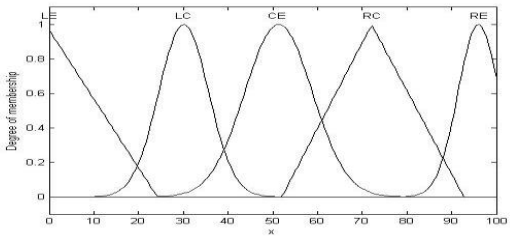
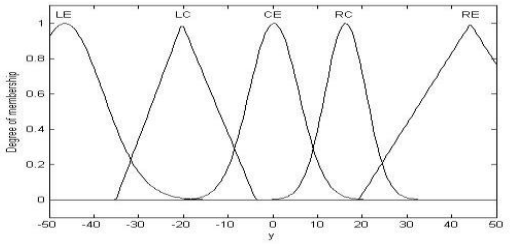


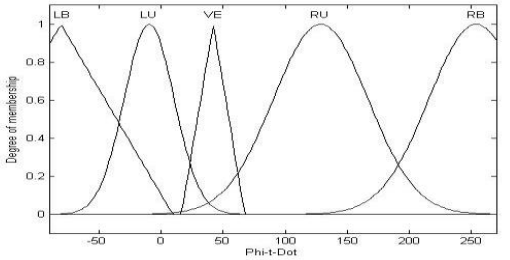
Fig. 5.6. The corresponding control surface of the fuzzy rule matrix for the sub-controller (SC-1)



(a)



(b)



(c)

Fig. 5.7. Generated Optimal Fuzzy sets for the SC-1

For building the proposed fuzzy sub controller-1, (SC-1), optimize the 5 fuzzy sets for the input variable *x*, *y* and output variable $\phi_t^{'}$ through GA as shown in Fig. 5.7. In this study, we select mixed

type of MFs where some fuzzy sets are triangular and some are gaussian to get the better performance. In the same way build the sub-controler-2 (SC-2) and 3 (SC-3) but their number of fuzzy sets for the input variables ($\phi_c = 3$ and $\theta = 5$) are different.

Some interesting backing trajectories are shown in Figs. 5.8. In this case the initial states being sampled from: $x \in \{20,40\}$, $y \in \{-50,50\}$, $\phi_t \in \{-90^0, 90^0\}$ $\phi_c \in \{-30^0 - +30^0\}$ and $\phi_d \in \{0^0\}$. Actually more tests also show the control results are successful and our method is encouraging. In Figure 5.9 we show the control action θ generated by our proposed hierarchical fuzzy controller for trajectories initial states (x, y, ϕ_t, ϕ_c): $(20, -50, -90^0, -10^0)$ and $(20, 50, 90^0, 10^0)$. In this figures, we can see that the control action using our proposed hierarchical fuzzy controller is smoother.

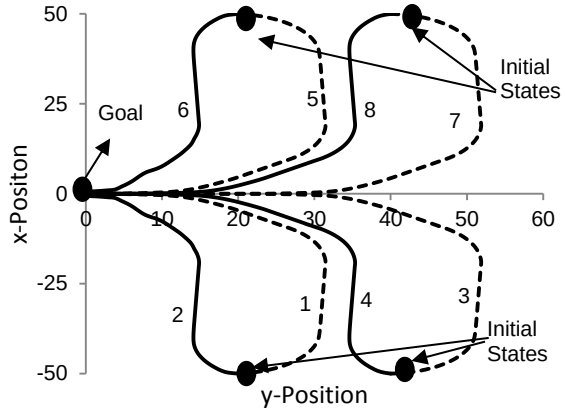
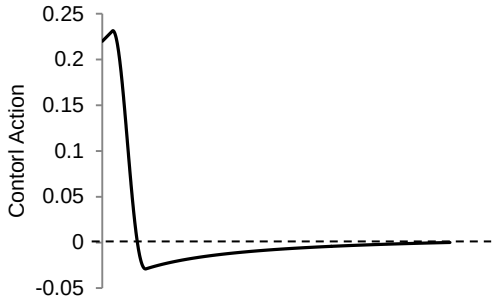
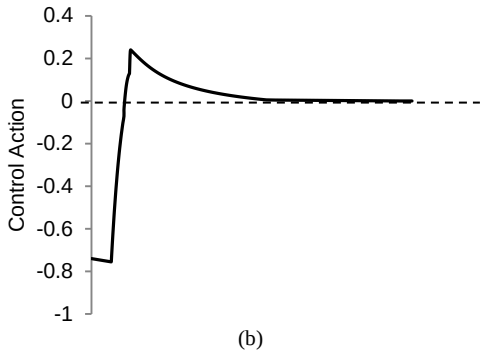


Fig. 5.8. The trajectories of the back of the trailer-cab while being backed up from the eight corners used to compute fitness during evolution. The dashed lines correspond to ϕ_t = -90^0 and the solid lines correspond to ϕ_t = 90^0.



(a)



(b)

Fig. 5.9. Control action θ for trajectory initial state (x, y, ϕ_t, ϕ_c) (a) $(20, -50, -90^0, -10^0)$ (b) $(20, 50, 90^0, 10^0)$

From the experimental studies we show that our control algorithm converges faster and reach the parking space relatively quickly and efficiently measured by the docking error and trajectory error. The trajectory error is the ratio of the distance of the actual trajectory of the trailer-and-cab robot divided by the straight distance from initial state to final state.

$$\text{TrajectoryError} = \frac{\text{Distance of ActualTrajecory}}{\text{Straight distance from initial state to final state}}$$

The docking error is the Euclidean distance between desired final state $[x_d, y_d, \phi_d]$ and actual final state $[x_d, y_d, \phi_d]$.

$$\text{DockingError} = \sqrt{(x_d - x_f)^2 + (y_d - y_f)^2 + (\phi_d - \phi_f)^2}$$

Figure 5.10 shows the trajectory error for the eight different initial states where the trajectory error is depends on the ϕ_t whether it is positive or negative.

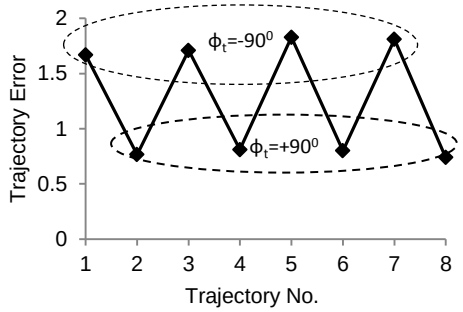


Fig. 5.10. Trajectory Error of the back of the trailer-cab while being backed up from the eight corners.

Figure 5.11 shows the docking error and ε_c for the eight different initial states.

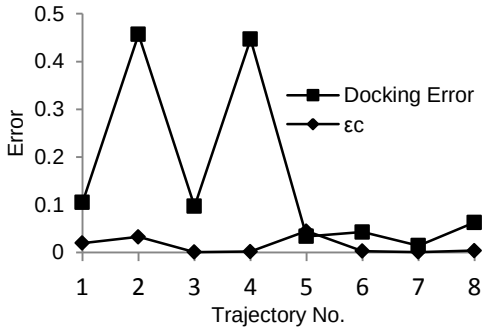


Fig. 5.11. Docking Error and ε_c of the back of the cab-trailer while being backed up from the eight corners

5.5 Conclusions

This paper presents a representation and computation of hierarchical fuzzy logic controller for the classical trailer with cab reverse parking control problem. The GP employed to identify the optimal hierarchical architecture, selection of input variables for each sub-controller and fuzzy control rule parameters with their corresponding membership functions were optimized through the nontraditional GAs and these tasks were successfully achieved. The proposed method automatically generates more precise and concise small sized hierarchical controller than the one by the conventional method. Each sub-controller in the hierarchical fuzzy logic controller has a smaller number of input variables with the lowest possible number of fuzzy rules. Therefore, it does not need many data as well as control rules for the description of the input-output relationships i.e., number of fuzzy rules in the subspace. Simulation results are shown that generated small size hierarchical controller, have small number of rules, are effective and converge faster to identify the nonlinear system and demonstrate the effective use of the proposed control algorithms.

CHAPTER 6

OPTIMIZE TYPE-2 FUZZY LOGIC CONTROLLER

6.1 Introduction

For most fuzzy logic control problems, the most important issue is to determine the parameters that define the type-2 MFs. Because of this, the type-2 MFs optimization problems can be converted to parameter optimization problems. These parameters are generally based on the expert KB that is derived from heuristic knowledge of experienced control engineers and/or generated automatically.

Mendel [36] and Hagras [122] have shown that the type-1 fuzzy logic systems (FLSs) may be unable to model and minimize the effect of uncertainties that prevails' in the real world applications. One restriction is that a type-1 fuzzy set is certain in the logic where the membership grade for each input is a crisp value. On the other hand, interval type-2 FLCs (that use interval type-2 fuzzy sets, characterized by fuzzy MFs) can handle the uncertainties.

GA was used by Martínez [123] in optimization of type-2 FLC. He applied GA to design FLC for the control of the perturbed autonomous wheeled mobile robot. Melin and Castillo [124] proposed a method based on type-2 fuzzy sets and neural networks called neuro-fuzzy to learn the parameters of the fuzzy system for intelligent control of nonlinear dynamic plants. Tan [125] used GA to optimize the parameters of FLCs. His proposed approach used mixed (type-1 and type-2) fuzzy sets for real time control.

In this study, quantum optimization can be regarded as a new optimization tool that can be used to overcome the limitations of trial and error approach as it is a quantum computing technique and finds the optimal solutions effectively and efficiently by combinatorial searching the large and complex solution space. So, QGA can be employed as a powerful search method to perform tasks such as generation of fuzzy rule base (RB), optimization of fuzzy RB, generation of MFs, and tuning of MFs types.

6.2 System Definition

A mobile robot has to move from an initial position to the target (dock) by avoiding collisions with a single stationary obstacle in optimal path. It may have to move along a straight path or take a turn depending on the current situations in order to generate a collision-free path. The problem is taken from [75]. Figure 6.1 depicts the simulated geometry for the robot and loading dock schematically, in which mobile robot is moving among single stationary obstacles, in the same workspace. The control system must find incrementally a path to the loading dock, independently of the initial position of the robot.

The path planning of the mobile robot is determined by the three input variables x, y and ϕ, (considered as a point mass), where x and y are the cartesian co-ordinates of the mobile robot and ϕ is the robot direction angle relative to the horizontal axis x. The output variable is the control steering signal θ. Thus controller is a function of state variables

$$\theta = f(x, y, \phi)$$

As a first investigation, let us assume that there exists enough clearance between the robot, the walls and the obstacle in the workspace so that we can ignore the *y*-position co-ordinate of the robot. The co-ordinate *y* will be re-introduced into the discussion shortly and simplifying the controller function to:

$$\theta = f(x, \phi)$$

The state spaces of two inputs are $-115^0 \le \phi \le 295^0$ & $0 \le x \le 100$, and one output θ within [-40^0, 40^0]. At every stage, the simulated mobile robot only moved forward until it hits the border of the loading dock. The final states (x_f, ϕ_f) will be equal or close to (10, 90^0). The robot kinematics model is described by the following dynamic equations.

$$x_{t+1} = x_t + \cos(\phi_t + \theta_t) + \sin(\phi_t)\sin(\theta_t),$$
$$y_{t+1} = y_t + \sin(\phi_t + \theta_t) - \sin(\phi_t)\sin(\theta_t)$$
$$\phi_{t+1} = \phi_t - \sin^{-1}\left(\frac{2\sin(\theta_t)}{l}\right)$$

Where *l* is the length of the robot, we assume *l*=4. Eq. (1) will be used to derive the next state when present state and control are given.

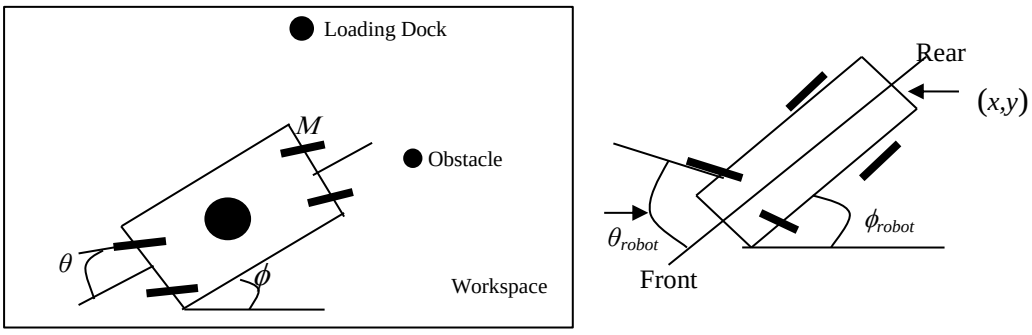


Fig. 6.1. Mobile Robot and loading dock illustration

This experiment should be considered as an example of highly nonlinear complex problems. For obvious reasons such controller does not perform very well if the distance between the truck position and the loading dock is small.

6.3 Hybrid Q-Fuzzy Controller: Design And Methodology

QGA is a probability optimization algorithm based on the concept of the quantum computing. In quantum computing, the smallest unit of information is called Q-bit. A Q-bit can be represented as

$$\alpha|0\rangle + \beta|1\rangle \leftrightarrow \begin{pmatrix} \alpha \\ \beta \end{pmatrix}$$

Where $\alpha^2 + \beta^2 = 1$. $|\alpha|^2$ indicates the probability of finding the Q-bit in "0" state and $|\beta|^2$ indicates the probability of finding the Q-bit in "1" state. A Q-bit may be in "1" state, in "0" state or in a linear superposition of the two states.

A Q-bit individual as a string of *m* Q-bits is defined as

$$\begin{bmatrix} \alpha_1 & \alpha_2 & \dots & \alpha_m \\ \beta_1 & \beta_2 & \dots & \beta_m \end{bmatrix}$$

Where $|\alpha_i|^2 + |\beta_i|^2 = 1$, $i = 1, 2, 3 \ldots m$. A Q-gate which is a quantum mutation gate is used to speed up the convergence. It is defined as:

$$U(\Delta\theta_i) = \begin{bmatrix} \cos(\Delta\theta_i) & -\sin(\Delta\theta_i) \\ \sin(\Delta\theta_i) & \cos(\Delta\theta_i) \end{bmatrix}$$

where $\Delta\theta_i$, $i = 1, 2, 3 \ldots, m$, is the rotation angle of a Q-bit towards the "0" state or "1" state depending on its sign. After applying Q-gate, the Q-bit should satisfy the normalization condition $|\alpha'|^2 + |\beta'|^2 = 1$, where $|\alpha'|^2$ and $|\beta'|^2$ are the values of updated Q-bit.

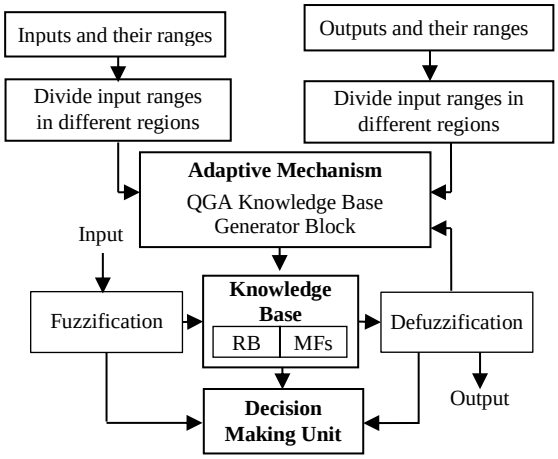


Fig. 6.2. Integration of type-2 FLCs and QGA

A quantum-type-2 fuzzy system depict in Fig. 6.2 is a fuzzy system that uses QGA to determine type- 2 fuzzy sets and fuzzy control rules. In this paper, we employed QGA to optimize the parameters of the MFs of Type-2 FLC; we consider using Gaussian Interval Type-2 MFs to each one of our three variables. At the same time, we also employed QGA for the selection and definition of RB of type-2 FLC.

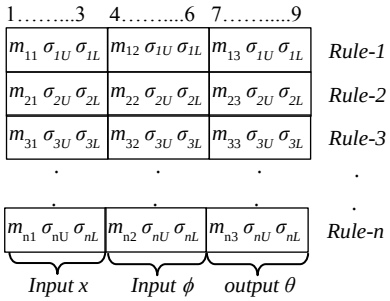


Fig. 6.3. QGA coding schema of type-2 FLC

6.3.1 Encoding

Two input variable x, ϕ and one output variable θ are used to produce fuzzy rules and their corresponding MFs. For this reason these variables are encoded in a chromosome of QGA. The domains for x, ϕ and θ are divided into 5, 7 and 7 regions respectively. The linguistic terms (MFs) for each of the input and output variables are used to describe them. The rule base contains

total 35 rules. Each rule includes the real value of *x*, ϕ and θ. Each variable (input and output) is divided into three parts: center (m), lower MFs width (σ_L), and upper MFs width (σ_U). So the rule base looks like (Fig. 6.3):

An important characteristic of fuzzy models, FM, is the partitioning of the input and output space of system variables (input, output) into fuzzy regions using fuzzy sets. The range of x is divided into five non-uniform intervals [0, 32.5], [32.5, 47.5], [47.5, 52.5], [52.5, 67.5], and [67.5, 100] and they are represented by five linguistic terms LE, LC, CE, RC and RE respectively [31]. The range of ϕ is divided into seven non-uniform regions [-115, -27.5], [-27.5, 46], [46, 86.5],[86.5, 98.5] , [98.5, 146], [146, 216], [216, 295] and then they are represented by seven linguistic terms NL, NM, NS, ZE, PS, PM, and PL respectively. Similarly seven divided regions of the range of θ, [-40, -28], [-28, -12.5], [-12.5, -2.5], [-2.5, 2.5], [2.5, 12.5], [12.5, 28], [28, 40] are represented by linguistic terms NB, NM, NS, ZE, PS, PM and PB [125].

In this study five and seven gaussian type-2 fuzzy sets were used to partition the input spaces *x* and ϕ respectively and seven gaussian type-2 fuzzy sets for output spaces [125]. The rule base, then, contains thirty-five (7×5) rules to account for every possible combination of input fuzzy sets. The fuzzy control if then rules are of the form: If *x* is ({LE, LC, CE, RC, and RE}) and ϕ is ({NL, NM, NS, ZE, PS, PM, and PL}) then θ is ({NB, NM, NS, ZE, PS, PM, and PB}), output is one of the type-2 fuzzy sets used to partition the output space. 315 genes are used to represent the rule set. Therefore, we need to encode a total of 315 parameters for each individual of our population. In order to make this encoding schema we design a chromosome of 315 consecutive real genes. Figure 6.3 show a schematic of the chromosome structure to our quantum-type-2 FLS optimization approach.

6.3.2 Real Coded Quantum Genetic Algorithm

In this paper, real coded quantum genetic algorithm (RCQGA) is used as an optimization process.

6.3.2.1 Representation

Each individual is represented by real coded triploid chromosome which can be defined as follows:

$$\begin{pmatrix} R_1 \cdots R_i \cdots R_n \\ \alpha_1 \cdots \alpha_i \cdots \alpha_n \\ \beta_1 \cdots \beta_i \cdots \beta_n \end{pmatrix}$$

where $(R_i\ \alpha_i\ \beta_i)^T$, $i = 1, 2 \ldots n$ is the i^{th} allele of real coded triploid chromosome and R_i is the i^{th} rule in fuzzy rule base. $(\alpha_i, \beta_i)^T$ is a pair of probability amplitudes of one quantum bit and satisfies normalization condition $|\alpha|^2+|\beta|^2 = 1$, n is the length of real-coded triploid chromosome which is 35.

Table 6.1. OPTIMIZED MFS PARAMETERS (OPTIMAL SOLUTION)

MFs of *x*																		
LE			LC			CE			RC			RE						
m	σ_U	σ_L	m	σ_U	σ_L	m	σ_U	σ_L	M	σ_U	σ_L	m	σ_U	σ_L				
10.31	34.1	21.1	30.7	31.8	26.8	51.2	20.1	10.2	61.69	41.8	29.3	94.2	57.3	64.1				
MFs OF ϕ																		
NL			NM			NS			ZE			PS			PM			PL		
m	σ_U	σ_L	m	σ_U	σ_L	m	σ_U	σ_L	m	σ_U	σ_L	m	σ_U	σ_L	m	σ_U	σ_L	m	σ_U	σ_L
-90.81	94.17	84.21	2.16	189.5	171.3	75.5	135.9	92.60	90.8	122.1	98.12	145.1	74.28	51.2	182.3	83.00	44.28	252.4	135.4	96.4
MFs of ϑ																		
NB			NM			NS			ZE			PS			PM			PB		
m	σ_U	σ_L	m	σ_U	σ_L	m	σ_U	σ_L	m	σ_U	σ_L	m	σ_U	σ_L	m	σ_U	σ_L	m	σ_U	σ_L
-38.16	22.1	16.1	-25.22	22.8	11.1	-8.5	25.1	20.7	2.1	17.1	13.1	12.28	16.15	8.16	24.16	27.14	16.39	36.17	19.13	8.16

6.3.2.2 Mutation

Gaussian Mutation operator is applied to update population at each generation. The i^{th} allele is randomly selected from p_j^t, t[th] generation j[th] population, and the centers (c) of the input-output variables in the rule of the selected allele are changed as follows:

$$C^{t+1} = C^t + (Max - Min)\, N(0, (\sigma_{j,i}^t)^2) \tag{6.1}$$

where Max and Min are respectively upper and lower bound of the regions in which C^t lies. C^{t+1} may not be within the limit so it is clipped to keep it within the region of C^t. The center of θ is considered always the whole range of θ. $N(0, (\sigma_{j,i}^t)^2)$ denotes the Gaussian distribution of mean 0 and variance $(\sigma_{j,i}^t)^2$. The value of variance $(\sigma_{j,i}^t)^2$ is either $|\alpha_{j,i}^t|^2$ or $|\beta_{j,i}^t|^2$ /5 based on "Fine Search" or "Coarse Search" to be implemented [32]. The width (W^t) of each center is updated as follows:

$$W^t = \begin{cases} r * (Max - C^{t+1}) & if\ C^{t+1} > (Max + Min)/2 \\ r * (C^{t+1} - Min) & Otherwise \end{cases}$$

where r is the uniformly distributed random number in the range [0, 1]. The pair probability amplitudes of the i^{th} allele is updated by the quantum rotation gate (QRG) as follows

$$\begin{pmatrix} \alpha_{j,i}^{t+1} \\ \beta_{j,i}^{t+1} \end{pmatrix} = \begin{pmatrix} \cos(\Delta\Theta_{j,i}^t) & -\sin(\Delta\Theta_{j,i}^t) \\ \sin(\Delta\Theta_{j,i}^t) & \cos(\Delta\Theta_{j,i}^t) \end{pmatrix} \begin{pmatrix} \alpha_{j,i}^t \\ \beta_{j,i}^t \end{pmatrix} \tag{6.2}$$

Here $\Delta\Theta_{j,i}^t$ is rotation angle of Q-bit and it is calculated as follows,

$$\Delta\Theta_{j,i}^t = \boldsymbol{sgn}\,(\alpha_{j,i}^t\, \beta_{j,i}^t)\, \Theta_0\, \exp\left(\frac{|\beta_{j,i}^t|}{|\alpha_{j,i}^t| + \gamma}\right) \tag{6.3}$$

Where Θ_0 is the initial rotation angle, γ is the scale parameter. These control the rotation angle and increase the speed of convergence, the sign *sgn* (.) determines the direction of the rotation angle.

Table 6.2. CONTROL RULE MATRIX

θ		ϕ						
		NL	*NM*	*NS*	*ZE*	*PS*	*PM*	*PL*
x	*LE*	[1]*NL*	[2]*ZE*	[3]*PB*	[4]*NM*	[5]*ZE*	[6]*NB*	[7]*PS*
	LC	[8]*PM*	[9]*NL*	[10]*NS*	[11]*NM*	[12]*ZE*	[13]*PB*	[14]*NB*
	CE	[15]*NM*	[16]*PM*	[17]*NL*	[18]*ZE*	[19]*PS*	[20]*NM*	[21]*NS*
	RC	[22]*ZE*	[23]*PB*	[24]*ZE*	[25]*PM*	[26]*ZE*	[27]*NS*	[28]*NM*
	RI	[29]*NS*	[30]*PB*	[31]*NM*	[32]*PL*	[33]*PS*	[34]*ZE*	[35]*ZE*

6.3.2.3 Discrete Crossover (DC) and Elitism

DC is performed repeatedly after a fixed number of generations and it expands the search space to find the suitable steering angle θ with respect to input variables with minimized trajectory error (fitness value). The elitism technique is used to ensure that the rule base with best fitness value will not be lost.

6.3.2.4 RCQEA procedure

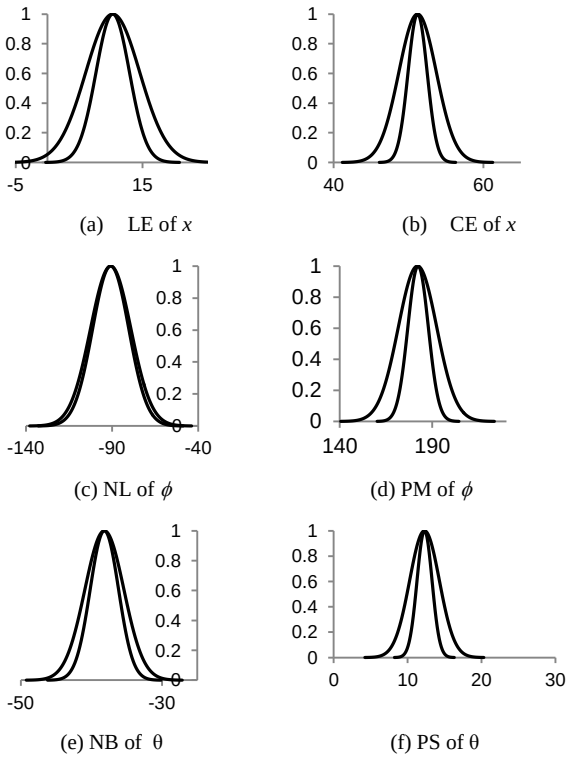
Step1: **Initialization:** A Population of N individuals P^t={p_1^t,⋯,p_j^t,⋯,p_N^t} is initialized by randomly chosen real numbers, where p_j^t is an individual.

Step2: **Decode and Evaluation**: At each generation *t* RCQGA maintains a population of real-coded triploid chromosome. A rule $R_{j,i}^{t}$ in p_j^t contains nine values, center (m), lower MFs and upper MFs for each of three variables (x, φ and θ). Decode the every chromosome into RB and MFs for the construction of interval type-2 FLC and the constructed FLC is executed on the robot until it reaches the goal position or near to the goal position. Each potential solution (FLC) is evaluated and assigned a fitness value according to its performance to the problem. The fitness value for each chromosome is defined as the trajectory error which is defined as follows:

$$Trajectory / Fitness = \frac{Lengh\ of\ Robot\ Trajectory}{\text{Distance(initial position, final position)}}$$

Step3: **Recombination:** Apply mutation and discrete crossover operator to chromosomes and generate new chromosomes as well as new generation. Check the termination condition and go to step 2 if the termination condition is true otherwise go to step 4.

Step4: **Stop:** The best fitted chromosome is kept and solution has been achieved.



(a) LE of *x* (b) CE of *x*

(c) NL of ϕ (d) PM of ϕ

(e) NB of θ (f) PS of θ

Fig. 6.4. Tuned Gaussian Type-2 Fuzzy sets

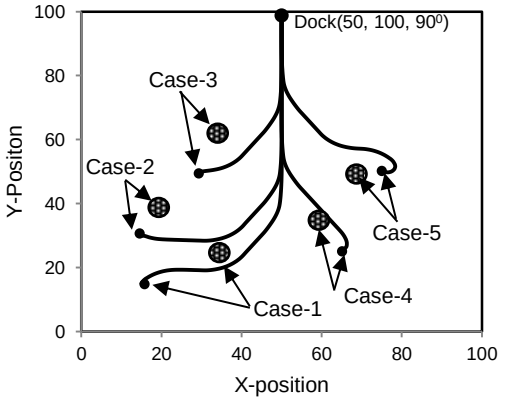
6.4 Simulation Results and Evaluation

To evaluate the accuracy of the proposed system, we have carried out a series of experiments which the controller were evolved in our simulated arena. The average optimal means and standard deviations of MFs for x, ϕ, and θ are shown in table 6.1. The generated optimal control rule base (after the conversion from optimal parameter to linguistic form) also shown in table 6.2.

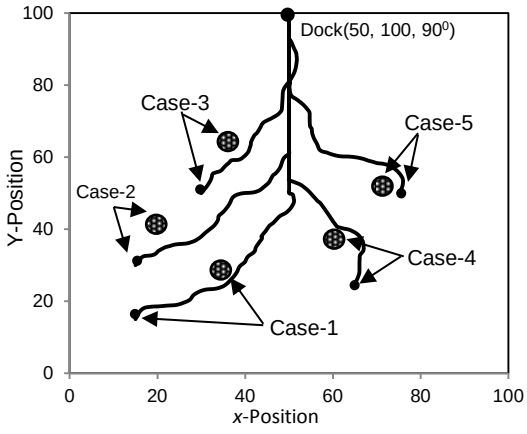
Table 6.3. FIVE INITIAL POSITIONS (x, y, ϕ) AND THEIR STEPS OF RESULT

Case	1		2		3		4		5	
x y	15	15	15	30	30	50	65	25	75	50
ϕ	60^0		-30^0		15^0		25^0		-45^0	
T1FLC	40		56		34		31		45	
T2FLC	28		40		27		22		38	

We have obtained the fuzzy control rules from the best chromosomes of QGA after 100 generations. Figures 6.4(a)-6.4(f) shows an example of tuned type-2 fuzzy sets for the efficient design of interval type-2 FLC. Figure 6.6 shows the times for the mobile robot to reach the goal position in 5 different initial conditions and their trajectories are plotted in Fig.6.5. Table 6.3 shows the five initial conditions for (x, y, ϕ) with their steps.



(a)



(b)

Fig. 6.5. (a) Show robot trajectories avoiding stationary obstacle (cross-hatched circle) via T1QFLC, and (b) Show via interval T2QFLC, all from 5 different initial conditions

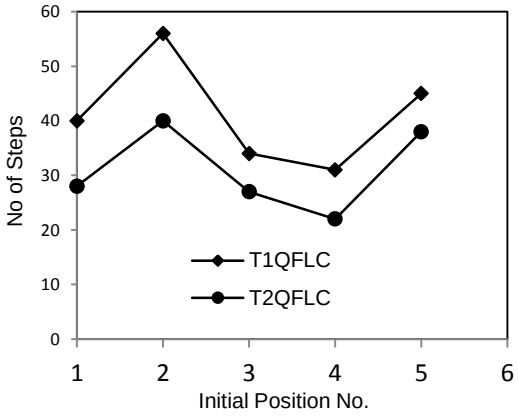


Fig. 6.6. Show the total steps to reach goal position by 5 different initial cases

6.4.1 Two goal Seeking Technique

This study presents an evolutionary strategy based fuzzy controller for autonomous robot navigation in which a robot find its path and avoid any collision with obstacles. The evolutionary strategy based controller is represented by the coordination between two tasks: the one for reaching the target point and the other for fixed obstacle avoidance. To tackle these two tasks, we divide the interval of definition of *y* into just two regions - below and above the obstacle - with overlapping membership functions. In below the obstacle, the fuzzy controller works as an obstacle avoidance controller that decides for the robot how to avoid the fixed obstacle. In above the obstacle, the controller works as goal seeking controller. Each bit-string will now consist of 70 rules; 35 rules are used for obstacle avoidance controller and another 35 rules is used for goal seeking controller. Given the fixed obstacle position, the obstacle avoidance controller first consider two optional goal position, one is left and another one is right of the obstacle. When the robot is reach to the one the optimal goal which is the close to the robot, the goal seeking controller is activated and finally the robot reach to the goal position.

6.4.2 Evaluating the work

The mobile robot/truck is a classical control problem that is generally used as well-known benchmark problem for the evaluation of new control algorithms and as such it has been well analyzed [75][3]. The graph of in Figures 6.5, 6.6, 6.7 and 6.8 shows the performance comparison with QGA evolved type-1 FLC (T1QFLC), neural coded type-1 FLCs, traditional type-1 FLC and T2QFLC. From Figure 6.5, 6.6, 6.7, 6.8 and table 6.3 and table 6.4, it is obvious that the performances of T2GFLC are better than those in T1QFLC, neural type-1 and traditional type-1 FLCs. It not only takes less steps to arrive the goal position using interval T2QFLC, but also it shows the smoother trajectories (shown in Fig. 6.5 and Fig. 6.7).

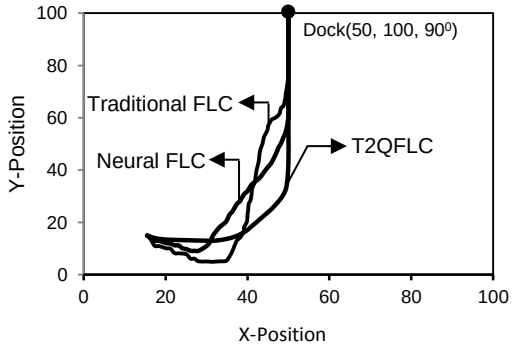


Fig. 6.7. Evaluation T2QFLC with other methods

Recently Li and Li in [3] have presented the fuzzy control system based on a hybrid clustering method and neural network. In their work, a two-stage technique, structure identification step and parameter identification step, are used for the constructing fuzzy logic control system. The clustering method is proposed to construct an initial fuzzy model to determine the number of fuzzy rules from the intuitionistic-desired trajectories. The clusters are automatically generated and the data are appropriately classified through clustering method. Then neural network is applied to obtain a more precise fuzzy model in the parameter identification for the truck/mobile robot control. The clustering method (off-line approach) is one of the most promising techniques when input/output samples are available for the system. It is not possible however to generate initial fuzzy model without such type of input output data.

Table 6.4. TRAJECTORY STEPS FOR DIFFERENT VALUES OF ϕ WITH SAME (x, y) *COORDINATE*

Case		1		2		3	
x	y	15.5	15	15.5	15	15.5	15
ϕ		0^0		-30^0		-90^0	
Traditional FLC		68		84		91	
Neural FLC		52		70		79	
T2QFLC		29		32		34	

The number of steps of trajectories depends on the physical orientation of the robot. We also analyse and compare the robot trajectories with other existing methods for different values of ϕ. The number of steps of trajectories of the robot known as truck from given positions to back up to the loading zone controlled by the neural fuzzy system [3], traditional fuzzy system [3] and T2QFLC are given in IV.

It has been found that the QGA based system evolves to optimal type-2 MFs and RB after some generations. An example of the best fitness quantum genetic progress is presented in Fig. 6.8 which demonstrations the performance of the best chromosome found so far against the number of generations.

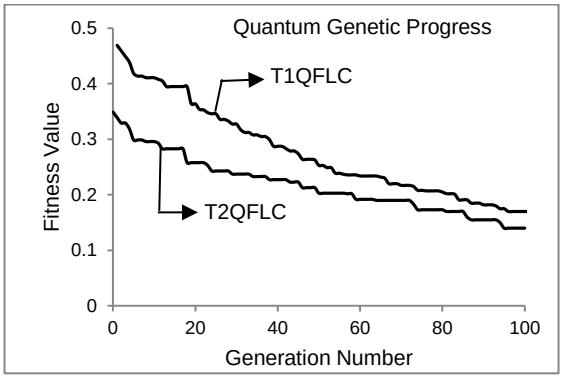


Fig. 6.8. Shows the results of Best fitness trajectory errors in T1QFLC and T2QFLC

6.5 Conclusions

This paper has revealed the possibility of using QGA based adaptive architecture to evolve the type-2 MFs and their corresponding rule set parameters of interval type-2 FLCs. We have shown that T2QFLC is an adaptive model that is able to tune MFs parameters and establish a reliable fuzzy control rules aimed at mobile robot control in real world unstructured environments. The type-2 based control architecture could cope with the unstructured noisy environments and achieved a superior control performance that outperformed the QGA evolved type-1 FLCs, traditional type-1 FLCs and neural coded FLCs.

CHAPTER 7

CONCLUSIONS AND FUTURE WORKS

7.1 Summary

In this thesis, we have developed several optimization, learning, tuning and rule base size reduction algorithms for both type-1 and type-2 fuzzy logic controllers (FLCs), which are an emerging research area. In order to show the usefulness of the developed methods several benchmark control applications, including backing up a truck, trailer-and-cab, nonlinear motion control of nonholonomic robots and two degrees of freedom inverted pendulum have been illustrated.

In chapter 1, we have demonstrated some motivation and detail contribution of this thesis. In this thesis, we have focused four main contributions, firstly, to automatically generate fuzzy knowledge base components through the newly developed hybrid genetic algorithms. Secondly, a rule base reduction and tuning algorithm is proposed as a design tool for the knowledge-based fuzzy controller. Thirdly, an efficient tool is proposed for automatically build the hierarchical fuzzy systems to deal with the rule explosion problem. Fourthly, we proposed a bio inspired optimization methods used in the design of type-2 fuzzy systems which are relatively novel models of imprecision. In the designing phase, the use of bio-inspired optimization methods has helped in the complex task of finding the appropriate parameter values and structure of the fuzzy systems.

Chapter 3 illustrated the difficulties of finding or optimizing the fuzzy knowledge base components in the FLCs. The topic is one of the major concerns in the development of FLCs. This chapter presents new hybrid (binary and real coded coupled) GAs to automatically generate the FKB for building the optimal FLCs with different tradeoffs between complexity and accuracy. The main novelty of the algorithm is that rules and their corresponding MFs type are learning and selecting concurrently. Selection of optimal MFs type for each rule individually is another novelty for designing adaptive FLCs also. For this concurrent learning process, HGAs is employed in two simultaneous phases; the first one, referred in this paper as real phase, used for the selection and definition of fuzzy control rules as well as adjustment of MFs shape while the second one, binary phase, used for the selection of optimum MFs type of each rule. The suggested method can deal with mixed MFs type (i.e., fuzzy sets) for fuzzy rule base. It is shown that the proposed HGAs can generate comprehensible and reliable fuzzy rules and at the same time select the optimal MFs (fuzzy sets) for each rule separately. The HGAs allows one to improve the FKB performance in terms of accuracy and simplicity by learning both parts concurrently.

Chapter 4 illustrated the Rule base size reduction technique that can be important for fuzzy systems in those cases where the fuzzy system needs to be implemented in real time. In this study, Firstly, an

adaptive learning method is developed for tuning the MFs parameters. Secondly, develop an automatic rule generating and reducing mechanism, which is capable of generating a rule base and reduce the rule base size i.e., finding the optimal number of rules for building the fuzzy controller to maximize the FLCs performance. The main advantages of the proposed method is that redundant, irrelevant and erroneous rules are removed by setting their all consequent weight factors to zero, merging the conflicting rules that have the same antecedents value during the learning process. The final rule base usually contains the much fewer rules than the initial one and thus improving the computational efficiency and interpretability of the fuzzy models. In this way we simply obtained fuzzy controllers to enhance the model interpretability while maintaining satisfactory accuracy.

Chapter 5 illustrated a method is described for the automatic design of an HFLC using an evolutionary algorithm. In our proposed method, evolutionary mechanism is used in two stages simultaneously: first stage involves identify the optimal hierarchical structure using the genetic programming (GP), while the second stage performs an optimal tuning of antecedent and consequent parameters of fuzzy control rules and their corresponding MFs. The fine tuning of the antecedents and consequents rule's parameters and their corresponding MF's parameters encoded in the structure is accomplished using genetic algorithm (GA).The main idea of this paper is in the usage of evolutionary mechanism for selecting the fuzzy sets and for constructing a hierarchical architecture of fuzzy model automatically. This study differences between the previous approaches lie mainly adaptive selection of hierarchical architecture through genetic programming (GP).

Chapter 6 illustrated bio inspired optimization method for designing type-2 fuzzy systems. Quantum genetic algorithm is employed as an optimization tool to tune type-2 fuzzy sets and rule sets simultaneously for effective design of interval type-2 FLCs. The type-2 FLCs exhibit better performance for compensating the large amount of uncertainties with severe nonlinearities.

7.2 Future Works

Although we have contributed only a little in the progressing research of fuzzy logic, there are several interesting scopes of further development which we list below:

- The research work is to be extended by embedding rule base size reduction technique in the proposed hierarchical architecture. In this case orthogonal transformation is used to determine the importance order of rules and cut the undesired and conflicting rules.
- This study also extended through the parallel implementation of the proposed algorithm that may be used to reduce the computation time of the proposed method. In order to validate the proposed control algorithms, we will compare it against the well-known traditional controllers, proportional derivative (PD) and proportional integral derivative (PID) controllers. Finally, apply the proposed method applied to more complex real world applications including intelligent control of a robotic arm in the presence of moving obstacle, the path planning problem for multiple mobile robots with more than one obstacles either moving or fixed in the workspace.
- This study work is to be extended to use of evolutionary strategies to develop an adaptive fuzzy logic controller and more thorough comparisons of adaptive FLCs with a standard PID controller. This study work is also to be extended to intelligent control of a mobile robot, truck-trailer controller and control of a robotic arm in the presence of moving or fixed single or multiple obstacles.

References

References

[1] F. J. Cabrerizo, J. Lopez-gijon, and A. A. Ruizo, "A model based on fuzzy linguistic information to evaluate the quality of digital libraries," *International Journal of Information Technology & Decision Making*, vol. 9, no. 3, pp. 455–472, 2010.

[2] R. J. Stonier, Evolutionary learning of fuzzy logic controllers over a region of initial states, in *Proc. congress on Evolutionary Computation*, 1999.

[3] L. Ying and L. Yuanchun, "Neural-fuzzy Control of Truck Backer-Upper System using a Clustering Method," *Neurocomputing*, vol. 70, no.4-6, pp. 680-688, 2007.

[4] F. Herrera and M. Lozano, "Adaptive Genetic Operators Based on Coevolution with Fuzzy Behaviors," *IEEE Trans. Evolutionary Computation*, vol. 5, no. 2, pp. 149-165, 2001.

[5] D. G. Stavrakoudis and J. B. Heocharis, Handing Highly Dimensional Classification Tasks with Hierarchical Genetic-Fuzzy Rule Based Classifiers, *Int. J. of Uncertainty, Fuzziness and Knowledge-Based Systems*, Vol. 20, Suppl. 2, pp. 73-104, October 2012.

[6] B. Kosko, *Neural Networks and Fuzzy Systems* (Englewood Cliffs: Prentice Hall Inc., 1992).

[7] F. J. Cabrerizo, S. Alonso, and E. Herrera Viedma, "A consensus model for group decision making problems with unbalanced fuzzy linguistic information," *International Journal of Information Technology & Decision Making*, vol. 8, no. 1, pp.109–131, 2009.

[8] Z. S. Xu and H. Hu, "Projection models for intuitionistic fuzzy multiple attribute decision making," *International Journal of Information Technology and Decision Making* vol. 9, no. 2, pp.267–280, 2010.

[9] Y. Shi, "The research trend of information technology and decision making in 2009," *International Journal of Information Technology and Decision Making*, vol. 9, no. 1, pp. 1–8, 2010.

[10] Z. W. Gong, L. S. Li *et al.*, "On additive consistent properties of the intuitionistic fuzzy preference relation," *International Journal of Information Technology and Decision Making*, vol. 9, no. 6, pp. 1009–1025, 2010.

[11] O. Cordón, F. Herrera "A proposal for improving the accuracy of linguistic modeling," *IEEE Transactions on Fuzzy Systems*, vol. 8, no. 3, pp. 335–344, 2000.

[12] A. González and R. Pérez, "SLAVE: A genetic learning system based on the iterative approach," *IEEE Transactions on Fuzzy Systems*, vol. 7, no. 2, pp. 176–191, 1999.

[13] F. Hoffmann and G. Pfister, "Evolutionary design of a fuzzy knowledge base for a mobile robot," *International Journal of Approximate Reasoning*, vol. 17, no. 4, pp. 447–469, 1997.

[14] O. Cordón and F. Herrera, "A three-stage evolutionary process for learning descriptive and approximative fuzzy logic controller knowledge bases from examples," *International Journal of Approximate Reasoning*, vol. 17, no. 4, pp. 369–407, 1997.

[15] R. Rovatti, R. Guerrieri, and G. Baccarani, "An enhanced two-level boolean synthesis methodology for fuzzy rules minimization." *IEEE Transactions on Fuzzy Systems,*vol. 3, no. 3, pp. 288–299, 1995.

[16] T. Heckenthaler, and S. Engell, "Approximately time-optimal fuzzy control of a two-tank system," *IEEE Control Systems*, vol. 4, pp. 24–30, 1994.

[17] H. Bezine, N. Derbel, and M. A. Alimi, "On the reduction of large scale fuzzy controller," *in Proc. of the Int. Conference on ACIDCA'2000*. Monastir, Tunisia, pp. 135–140.

REFERENCES

[18] E. Ruspini, "A new approach to clustering," *Information and Control*, vol. 15, pp.22–32, 1969.

[19] Q. Liang and J. M. Mendel, "MPEG VBR video traffic modelling and classification using fuzzy technique," *IEEE Trans. Fuzzy Syst*., vol.9, pp.183-193, Feb.2001.

[20] Q. Liang and J. Mendel, "Overcoming time-varying co-channel interference using type-2 fuzzy adaptive filter," *IEEE Trans. circuits Syst.-II: Analog and Digital Signal Processing*, vol. 47, pp. 1419-1428, Dec. 2000.

[21] Q. Liang, N. N. Karnik and J. M. Mendel, "Connection admission control in ATM networks using survey-based type-2 fuzzy logic systems," *IEEE Trans. Syst., Man, cybern. Part C: Appl. Reviews*, vol. 30, pp.329-339, 2000.

[22] K. C. Wu, "Fuzzy interval control of mobile robots," *Comput. Elect. Eng.*, vol.22, no.3, pp.211-229, 1996.

[23] R. R. Yager, "Fuzzy subsets of type II in decisions," *J. Cybern.*, vol.10, pp.137-159, 1980.

[24] J. L. Chaneau, M. Gunaratne, and A. G. Altschaeffl, "An application of type-2 sets to decision making in engineering," in Analysis of Fuzzy Information, vol. II: Artifical Intelligence and Decision Systems, J . Bezdek, Ed. Boca Raton, FL: CRC Press, 1987.

[25] N.N. Karnik, J.M. Mendel and Q. Liang, "type-2 fuzzy logic systems," *IEEE Trans. on Fuzzy Syst.*, vol.7 no.6, pp.643-658, Dec. 1999.

[26] Q. Liang and J. M. Mendel, "Equalization of nonlinear time varying channels type-2 fuzzy adaptive filters," *IEEE Trans. on Fuzzy Syst.*, vol. 8, no.5, pp.551-563, Oct. 2000.

[27] J. M. Mendel, "Uncertainty, fuzzy logic, and signal processing," Signal Processing, vol.80, pp.913-933, 2000.

[28] Q. Liang and J. M. Mendel, "Decision feedback equalizer for nonlinear time-varying channels using type-2 fuzzy adaptive filters," *in Proc. FUZZ-IEEE, may 2000*.

[29] R. I. John and C. C. zarnecki, "An adaptive type-2 system for learning linguistic membership grades," *in Proc. Int. conf. Fuzzy Systems*, Aug. 1999, pp. 1552-1556.

[30] R. I. John, P. R. Innocent, and M. R. Barnes, "Neuro-fuzzy clustering of radiographic tibia images using type-2 fuzzy sets," *Inform Sci.*, vol.125, pp.65-82, 2000.

[31] P. R. Innocent and R. I. John, "Type-2 fuzzy diagnosis," *in Proc. FUZZ-IEEE Init. Conf.*, May 2002, pp. 1326-1330.

[32] P. R. Innocent, R. I. John, and J. M. Garibaldi, "The fuzzy medical group in the centre for computational Intelligence," *Artificial Intell. Medicine*, vol. 21, pp. 163-170, 2001.

[33] P. Melin and O. Castillo, "A new approach for quality control of sound speakers combining type-2 fuzzy logic and fractal theory," *in Proc. FUZZ-IEEE Init. Conf.*, May 2002, pp. 825-830.

[34] D. A. Chiang, L. R. Chow, and N. C. Hsien, "Fuzzy information in extended fuzzy relational databases" *Fuzzy Sets and Systems*, vol. 92, pp. 1-20, Nov. 1997.

[35] N.N. Karnik and J. M. Mendel, "Application of type-2 fuzzy logic systems: handing the uncertainty associated with surveys," *in Proc. IEEE FUZZ Conf.*, 1999, Seoul, Korea.

[36] J. M. Mendel, *Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions*, Prentice Hall, NJ. 1999.

[37] S. Auephanwiriyakul, A. Adrian and J. M. Keller, "Type-2 fuzzy set analysis in management surveys, " *in Proc. FUZZ-IEEE Init conf.*, May 2002, pp. 1321-1325.

[38] N. N. Karnik and J.M. Mendel, "Application of type-2 fuzzy logic systems to forecasting of time series," *Inform. Sci.*, vol. 120, pp. 89-111, 1999.

[39] R. I. John, "Type-2 inferencing and commumity transport scheduling ," *in Proc. Fourth European congress on Intelligent Techniques and soft Computing*, EUFIT'96, Aachen, Germany, Sept. 1996, pp. 1369-1372.

[40] S. Park and H. Lee-Kwang, "A designing method for type-2 fuzzy logic systems using genetic algorithms," *Proc. Of the Joint 9th IFSA World Congress and 20th NAFIPS International conference*, Vancouver, Canada, July.2001, pp. 2567-2572.

[41] L. X. Wang, *Adaptive Fuzzy Systems and Control – Design and Stability Analysis* (Prentice Hall, 1994).

[42] A. Daftaribesheli, M. Ataei, and F. Sereshki, "Assessment of rock slope stability using the fuzzy slope mass rating (FSMR) system," *Applied Soft Computing*, vol. 11, pp. 4465–4473, 2011.

[43] M. Iphar and R.M. Goktan, An application of fuzzy sets to the diggability index rating method for surface mine equipment selection, *Int. J. of Rock Mechanics and Mining Sciences*, vol. 43, pp.253–266, 2006.

[44] L. X. Wang, Adaptive Fuzzy Systems and Control: Design and Stability Analysis, PTR Prentice-Hall, Englewood Cliffs, NJ. 1994.

[45] L. X. Wang, A course in Fuzzy Systems and Control, Prentice Hall, Upper Saddle River, NJ. 1997.

[46] D. Driankov, H. Hallendoorn and M.Reinfrank, An introduction to Fuzzy control (2nd Ed.), Springer-Verlag, 1996.

[47] G. C. Mouzouris and J. M. Mendel, "Nonsingleton Fuzzy Logic Systems: Theory and Application," IEEE Trans. On Fuzzy Syst., vol. 5, no. 1, Feb. 1997.

[48] N. N. Karnik, J. M. Mendel, and Q. Liang, "Type-2 fuzzy logic systems," IEEE Trans. On Fuzzy Syst., vol. 7, no. 6, pp. 643-658, Dec. 1999.

[49] L. A. Zadeh, "The concept of a linguistic variable and its application to approximate reasoning-I," *Inform. Sci*., vol. 8, pp.199-249, 1975.

[50] J. Mendel and R. John, "Type-2 fuzzy sets made simple," *IEEE Trans. Fuzzy Syst*., vol. 10, pp. 117–127, Apr. 2002.

[51] Q. Liang, N. Karnik, and J. Mendel, "Connection admission control in ATM networks using survey-based type-2 fuzzy logic systems," *IEEE Trans. Syst., Man, Cybern. C*, vol. 30, pp. 329–339, Aug. 2000.

[52] Qilian Liang and Jerry M. Mendel, "Interval Type-2 Fuzzy Logic Systems: Theory and Design," IEEE Trans. On Fuzzy Systems, vol. 8, no. 5, pp. 535-549, oct. 2000.

[53] J. M. Jou, P. Y. Chen, and S. F.Yang, "An Adaptive Fuzzy Logic Controller: Its VLSI Architecture and Applications," *IEEE Trans. Very Large Scale Integration (VLSI) Systems*, vol.8, no.1, pp. 52-60, Feb. 2000.

[54] N. N. Karnik and J. M. Mendel, "Operations on type-2 fuzzy sets," *Fuzzy Sets and Systems*, vol. 122, pp. 327–348, 2001.

[55] M. Mizumoto and K. Tanaka, "Some Properties of fuzzy sets of type-2," *Infom.Control*, vol. 31, pp.312-340, 1976.

[56] S. M. Bai and S. M. Chen, "Automatically Constructing Grade Membership Functions of Fuzzy Rules for Students' Evaluation," *Expert Systems with Applications*, vol. 35, no. 3, pp.1408–1414, 2008.

[57] A. Bagis, "Determining Fuzzy Membership Functions with Tabu Search - an Application to Control," *Fuzzy Sets and Systems*, vol.139, no.1, pp.209–225, 2003.

[58] F. Askari, M.R. Rahimpour, A. Jahanmiri, and A. Khosravanipour Mostafazadeh, "Dynamic simulation and optimization of dual-type methanol reactor using genetic algorithms," *Chemical Engineering & Technology*, vol. 31, p. (4), 2008.

[59] O. Cordón and F. Herrera, "A proposal for improving the accuracy of linguistic modeling, *IEEE Trans. on Fuzzy Systems*, 8 (3) (2000), pp. 335–344.

[60] A. González, R. Pérez SLAVE, A genetic learning system based on the iterative approach," *IEEE Trans. on Fuzzy Systems*, vol. 7, no. 2, pp.176–191, 1999.

[61] F. Hoffmann and G. Pfister, "Evolutionary design of a fuzzy knowledge base for a mobile robot," *Int. Journal of Approximate Reasoning*, vol. 17, no. 4, pp.447–469, 1997.

[62] A. Arslan and M. Kaya, "Determination of Fuzzy Logic Membership Functions using Genetic Algorithms," *Fuzzy Sets and Systems*, vol. 118, no. 2, pp.297–306, 2001.

[63] S. Oha, W. Pedryczb, and K. Parka, "Identification of Fuzzy Systems by Means of Genetic Optimization and Data Granulation," *Journal of Intelligent & Fuzzy Syst.*, vol. 18, pp. 31–41, 2007.

[64] C. C. Wong, H. Y. Wang and S.A. Li, "PSO-based Motion Fuzzy Controller Design for Mobile Robots," *International Journal of Fuzzy Syst.*, vol. 10, no. 1, pp. 24–32, 2008.

[65] P. Ganesh Kumar, C. Rani, and S. N. Deepa, "Formation of Fuzzy If-Then rules and Membership function using enhanced particle swarm optimization," *Int, J. of Uncertainty, Fuzziness and Knowledge-Based Systems*, Vol. 21, No. 1, pp. 103-126, 2013.

REFERENCES

[66] C. C. Yang and N. K. Bose, "Generating Fuzzy Membership Function with Self-Organizing Feature Map," *Pattern Recognition Letters*, vol. 27**,** pp.356–365, 2013.

[67] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning* (Addison-Wesley, New York, 1989).

[68] C. H. Chen, T. P. Hong, and Y.C. Lee, "Genetic fuzzy mining with taxonomy," *Int. J. Uncertainty, Fuzziness and Knowledge-Based Systems*, Vol. 20, Suppl. 2, pp.187−205, Oct. 2012.

[69] J. Sanz, H. Bustince, A. Fernandez, and F. Herrera, "IIVFDT: Ignorance Functions Based Interval-Valued Fuzzy Decision Tree with Genetic Tuning," *Int. J. of Uncertainty, Fuzziness and Knowledge-Based Systems*, Vol. 20, No. supp02, pp. 1-30.

[70] D. L. Meredith, C. L. Karr, and K. K. Kumar, "The use of genetic algorithms in the design of fuzzy logic controllers," in *proc. third Workshop on Neural Networks: Academic/Industrial/Defence (WNN'92),* 1992, pp. 549-555.

[71] F. Herrera, M. Lozano and J. L. Verdegay, "Tuning Fuzzy Controllers by Genetic Algorithms," *Internat. J. Approx. Reason.*, vol. 12, pp.299–315, 1995.

[72] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs* (Springer-Verlag, New York, 1996).

[73] T. Koga, K. Horio and T. Yamakawa, "Self-organizing relationship (SOR) network with fuzzy inference based evaluation and its application to trailer-truck back-up control," in *Proc International Conf. of Neural Information processing*, **11** (2004), pp.368-374.

[74] M. Mohammadian and R.J. Stonier, "Generating Fuzzy Rules by Genetic Algorithms," in *Proc. of 3rd International Workshop of Robot and Human Communication*, 1994, pp.362-367.

[75] M. Mohammadian and R. J. Stonier, "Fuzzy logic and genetic algorithms for intelligent control and obstacle avoidance," *in Complex Systems: Mechanism of Adaptation, R. J. Stonier and X. H. Yu, Eds. Amsterdam: IOS Press*, 1994, pp. 149-156.

[76] G. H. Park and Y. H. Pao, "Training neural-net controllers with the help of trajectories generated with fuzzy rules (demonstrated with the truck backup task)," *Neurocomputing*, vol. 18**,** pp.51-105, 1998.

[77] E. Vandewalle, "Constructing fuzzy models with linguistic integrity from numerical data AFRELI algorithm," *IEEE Trans. Fuzzy Systems*, vol. 8, no.5, pp. 591–600, 2000.

[78] C. T. Chao, Y. J. Chen and C. C. Teng, "Simplification of fuzzy neural systems using similarity analysis," *IEEE Trans. on Syst. Man and Cyber.* Part B, vol. 26, no. 2, pp. 344–354, 1996.

[79] M. Setnes, R. Babuska, U. Kaymak and H. R. N. Lemke, "Similarity measures in fuzzy rule base simplification," *IEEE Trans. on Syst. Man and Cyber.* Part B, vol. 28, no. 3, pp. 376 – 386, 1998.

[80] J. Yen and L. Wang, "Simplifying fuzzy rule-based models using orthogonal transformation methods*," IEEE Trans. on Syst. Man and Cyber. Part B*, vol. 29, no.1, pp.13–24, 1999.

[81] Y. Yam, P. Baranyi and C. T. Yang, "Reduction of fuzzy rule base via singular value decomposition," *IEEE Trans. Fuzzy Systems*, vol. 7, no. 2, pp. 120–132, 1999.

[82] Y. Jin, "Fuzzy modeling of high-dimensional systems: complexity reduction and interpretability improvement," *IEEE Trans. Fuzzy Systems*, vol. 8, no. 2, pp. 212–221, 2000.

[83] H.Ishibuchi, K. Nozaki, N.Yamamoto and H.Tanaka, "Selecting fuzzy if-then rules for classification problems using genetic algorithms," *IEEE Trans. on Fuzzy Syst*., vol. 3, pp.260 – 270, 1995.

[84] C. L. Karr, "Applying genetics to fuzzy logic," *AI Expert*, vol. 6, pp. 38 – 43, 1991.

[85] M. A. Lee and H. Takagi, "Integrating design stages of fuzzy systems using genetic algorithms," in *Proc. 2nd IEEE Conf. Fuzzy Systems*, 1993, pp. 612 – 617.

[86] H. Roubos and M. Setnes, "Compact and transparent fuzzy models and classifiers through iterative complexity reduction," *IEEE Trans. Fuzzy Systems*, vol. 9, no. 4, 516–524, 2001.

[87] B. Kosko and J.A. Dickerson, Function approximation with additive fuzzy systems, *Theoretical Aspects of Fuzzy Control*, eds. H.T. Nguyen, M. Sugeno, R. Tong, R. R. Yager (Wiley, NewYork, 1995).

[88] M. Setnes, "Supervised fuzzy clustering for rule extraction," *IEEE Trans. Fuzzy Systems*, vol.8, no. 4, pp.416–424, 2000.

[89] T. Sudkamp, A. Knapp and J. Knapp, "Model generation by domain refinement and rule reduction," *IEEE Trans. Systems Man Cybernet. B*, vol**.** 33, no. 1, pp. 45–55, 2003.

[90] L. T. Koczy, "Fuzzy if…then rule models and their transformation into one another," *IEEE Trans. on Syst. Man and Cyber.* Part A,vol. 26, no. 5, pp.621 – 637, 1996.

[91] S. Guillaume, B. Charnomordic, A new method for inducing a set of interpretable fuzzy partitions and fuzzy inference systems from data, *Interpretability Issues in Fuzzy Modeling*, eds. J. Casillas, O. Cordón, F. Herrera, L. Magdalena, (Springer-Verlag, 2003), pp. 148–175.

[92] S. Guillaume and B. Charnomordic, "Generating an interpretable family of fuzzy partitions from data," *IEEE Transactions on Fuzzy Systems*, vol. 12, pp. 324–335, 2004.

[93] F. Liu, C. Quek, and G. S. Ng, "A novel generic hebbian ordering-based fuzzy rule base reduction approach to Mamdani neuro-fuzzy system," *Neural Computation*, vol. 19, pp. 1656–1680, 2007.

[94] P. Pulkkinen, J. Hytönen, and H. Koivisto, "Developing a bioaerosol detector using hybrid genetic fuzzy systems," *Engineering Applications of Artificial Intelligence*, vol. 21, pp. 1330–1346, 2008.

[95] L. J. Eshelman, The CHC adaptive search algorithm: how to have safe search when engaging in nontraditional genetic recombination, *Foundations of Genetic Algorithms,* eds. G. Rawlin (Morgan Kaufman,1991), vol. 1, pp. 265–283.

[96] U. Rudolph, "Convergence analysis of canonical genetic algorithms," *IEEE Trans. Neural Networks*, vol. 5, pp.96–101, 1994.

[97] L .J. Eshelman, Real-coded genetic algorithms and interval schemata, *Foundations of Genetic Algorithms,* eds. L.D. Whitley(Morgan Kaufmann, San Mateo, 1993), vol. 2, pp. 187–202.

[98] D. E. Moriarty and R. Miikkulainen, "Efficient reinforcement learning through symbiotic evolution," *Machine Learning*, vol. 22, pp. 11–32, 1996.

[99] R. M. Kandadai, J. M. Tien, "On a fuzzy-neural hierarchical controller with a self-generating knowledge base," *in Proc. IEEE Int. Conference on Systems*, Man, and Cybernetics, vol.4, pp. 2625-2630, 1996.

[100] S. Kawaji, and T. Maeda, "Fuzzy servo control system for an inverted pendulum," i*n Proc. IFES'91*, vol. 2, 1991. pp. 812–823.

[101] K H. Kyung and B. H. Lee, "Fuzzy rule base derivation using neural network-based fuzzy logic controller by self-learning," *Proc. IECON'93*, vol. 1, 1993. pp. 435–40.

[102] Y . Pan and K. Furuta, "VSS controller design for discrete-time systems," in *Proc. IECON'93*, vol. 3, 1993. p. 1950–5.

[103] S. Sakai, T. Takahama, "Learning fuzzy control rules for inverted pendulum by simplex method," in *Proc.13th Fuzzy System Symposium*, 1997. pp. 61–64.

[104] S. I. Horikawa, T. Furuhashi and Y. Uchikawa, "On fuzzy modeling using neural networks with the back propagation algorithm" *IEEE Trans. Neural Networks*, **3**(1992.), pp.801-806.

[105] C.T. Lin, and C.S.G. Lee, "Neural network based fuzzy logic control and decision systems," *IEEE Trans. Computers*, vol. 40, pp. 1320-1336, 1992.

[106] A. Patricar and J. Provence, "A self–organizing controller for dynamic processes using neural networks," in *Proc Intl. Joint Conf. Neural Networks*, 1990, vol. 39, pp.359-364.

[107] P. T. Chan, A. B. Rad, and K. M. Tsang, "Optimization of Fused Fuzzy Systems via Genetic Algorithms," *IEEE Trans. Industrial Electronics,*vol. 49, no. 3, June 2002.

[108] J. Yen and L. Wang, "Application of statistical information criteria for optimal fuzzy model construction" *IEEE Trans. on fuzzy system*, pp. 362-372, Vol. 6, No. 3, Aug. 1998.

[109] R. R. Yager, "An alternative procedure for the calculation of fuzzy logic controller values," *J. Japanese Soc. Fuzzy Technol.*, vol. SOFT 3, pp. 736–746, 1991.

[110] G. V. S. Raju, J. Zhou, and R. A. Kisner, "Hierarchical fuzzy control," *Int. J. Contr.*, vol. 54, no. 5, 1991, pp. 1201-1216.

[111] G. V. S. Raju, and J. Zhou, "Adaptive Hierarchical Fuzzy Controller," *IEEE Trans. on systems, man and cybernetics*, vol. 23, no. 4, pp. 973-980, Jul/Aug.1993.

[112] R. R. Yager, "On a Hierarchical Structure for Fuzzy Modeling and Control," *IEEE Trans. Syst. Man Cybernet*, vol. 23, pp. 1189–1197, July/August 1993.

[113] Chung, F. Lai and J. C. Duan, "On multistage fuzzy neural network modeling," *IEEE trans. on fuzzy systems*, vol. 8, no. 2, 2000, pp. 125-142.

REFERENCES

[114] P. C. Shill, M. F. Amin, M. A. H. Akhand and K. Murase, "Design of a Self-Tuning Hierarchical Fuzzy Logic Controller for Nonlinear Swing Up and Stabilizing Control of Inverted Pendulum," *in Proc. IEEE Int. Conf. on Fuzzy Systems*, 2012, pp.1035-1042.

[115] H. Yo-Ping et al., "Designing a fuzzy model by adaptive macroevolution genetic algorithms," *Fuzzy Sets and Systems*, Vol.113, pp.367-379, 2000.

[116] W. Baolin et al., "Fuzzy modeling and identification with genetic algorithm based learning," *Fuzzy Sets and Systems*, Vol.113, pp.352-365, 2000.

[117] D. Maurizio et al., "Learning fuzzy rules with tabu search an application to control," *IEEE Trans. on Fuzzy Systems*, Vol.7, No.2, pp.295-318, 1999.

[118] C. L. Karr, L. Freeman, and D. Meredith, "Improved fuzzy process control of spacecraft autonomous rendezvous using a genetic algorithm," *In: SPIE Conf on Intelligent Control and Adaptive Systems*, 1989. pp 274 -283.

[119] K. Shimojima, T. Fukuda and Y. Hasegawa, "Self-tuning fuzzy modeling with adaptive membership function, rules, and hierarchical structure based on genetic algorithm," *Fuzzy Sets Syst*, vol. 71, no. 3, pp. 295–309, 1995.

[120] S. Matsushita, A. Kuromiya, and M. Yamaoka, T. Furuhashi , and Y. Uchikawa "Determination of antecedent structure of fuzzy modeling using genetic algorithm," *in: Proc IEEE Intl Conf on Evolutionary Computation (ICEC'96),* 1996. pp 235–238.

[121] T. Furuhashi, S. Matsushita, and H. Tsutsui, "Evolutionary fuzzy modeling using fuzzy neural networks and genetic algorithm." *In: Proc 1997 IEEE Intl Conf on Evolutionary Computation*, ICEC'97, 1997, pp. 623–627.

[122] H. A. Hagras,"A hierarchical type-2 fuzzy logic control architecture for autonomous mobile robots," *IEEE Trans. Fuzzy Systems*, vol.12, no.4, pp.524–539, 2004.

[123] R. Martínez-Soto, O. Castillo, and L.T. Aguilar, "Optimization of interval type-2 fuzzy logic controllers for a perturbed autonomous wheeled mobile robot using genetic algorithms," *Information Sciences, vol.* 179, pp. 2158–2174, 2009.

[124] P. Melin and O. Castillo,"Intelligent control of nonlinear dynamic plants using type-2 fuzzy logic and neural networks," *in Proc. of NAFIPS*, 2002, pp. 22–27.

[125] D. W. W. W. Tan, "A simplified type-2 fuzzy logic controller for real-time control," *ISA Trans.*, vol. 45, pp.503-516, 2006.

[126] P. C. Shill, M.A.H. Akhand, M. F. Amin, K. Murase "Optimization of Fuzzy Logic Controller for Trajectory Tracking using Genetic Algorithm," *J. Advanced Computational Intelligence and Intelligent Informatics (JACIII)*, Vol. 15, No. 6, pp.639-651, 2011.

[127] R. Alcalá, J. Alcalá-Fdez, M. J. Gacto, and Francisco Herrera, "Rule base reduction and genetic tuning of fuzzy systems based on the linguistic 3-tuples representation," *Soft Comput*, vol. **11**, pp. 401–419, 2007.

[128] M. Kemal Ciliz, "Rule base reduction for knowledge-based fuzzy controllers with application to a vacuum cleaner," *Expert Systems with Applications*, vol. **28,** pp.175–184, 2005.

[129] S. Hameed, B. Das, and V. Pant, "Reduced rule base self-tuning fuzzy PI controller for TCSC," *Int. J. Electrical Power & Energy Systems*, vol. **32**, no.9, pp. 1005–1013, 2010.

[130] F. Herrera, "Genetic Fuzzy Systems: Taxonomy, Current Research Trends and Prospects," *Evol. Intell.*, vol. **1,** 27–46, 2008.

[131] D. H. Kim, K. B. Kim and E.Y. Cha, "Fuzzy Truck Control Scheme for Obstacle Avoidance," *Journal of Neural Computing and Applications*, vol. 18, no. 7, pp.801-811, 2009.

[132] A. K. Nandi, and D. K. Pratihar, "Automatic Design of Fuzzy Logic Controller using a Genetic Algorithm—to Predict Power Requirement and Surface Finish in Grinding," *Journal of Materials Processing Technology*, vol. 148, no.3, pp.288–300, 2004.

[133] R. Arora, R. Tulshyan, R. Deb, Parallelization of binary and real coded genetic algorithms on GPU using CUDA, in *Proc of 2010 IEEE congress on Evolutionary Computation, CEC 2010*, pp. 1-8.

[134] M. G. Joo and J. S. Lee, "A class of hierarchical fuzzy systems with constraints on the fuzzy rules," *IEEE Trans. Fuzzy Syst.*, vol. 13, no. 2, pp. 194–203, Apr. 2005.

[135] J. Lin, "Hierarchical fuzzy logic controller for a flexible link robot arm performing constrained motion tasks*," in proc. Control theory and applications*, vol. 150, no. 4, pp. 355 364, Jul. 2003.

[136] S. Paulo, "Clustering and hierarchization of fuzzy systems," *J. Soft Comput.*, vol. 9, no. 10, pp. 715–731, 2005.

[137] R. Zhang and H. Gao, "Real -coded quantum evolutionary algorithm for complex function with high dimension," *in Proc. IEEE Int. Conf. Mechatronics and Automation*, Aug. 2007, pp. 2974-2979.

[138] R. I. John, " Embeded interval valued type-2 fuzzy sets," *in Proc. FUZZ-IEEE Int. Conf.*, May 2002, pp. 1316-1319.

[139] R.I. John and C. Czarnecki, "A type 2 adaptive fuzzy inferencing system," *in Proc. IEEE Systems, Man and Cybernetics*, pp.2068-2073, 1998.