Chapter 5

Real time Applications of Evolutionary Computational Techniques

5. Real Time Applications of Evolutionary Computation Techniques

In previous chapter, we discussed about various evolutionary computation method for the design and development of control systems design. Genetic Algorithms are generally not suitable for the design of real time controllers due to their uncontrolled recurrence computations. But, as discussed, GA along with fuzzy and/or neural network is better choice to have improved performance of the real time control applications.

5.1 GA based Fuzzy Logic controller – GAFLC for nonlinear systems

Design of the GAFLC has been in divided in two stages: Design of the fuzzy logic controller and optimization of the fuzzy controller using genetic algorithms. The most important stage for designing fuzzy logic controller for specified application of nonlinear plant is to choose appropriate dynamic operating parameters for generating appropriate membership functions. Two main dynamic characteristics of the nonlinear pendulum are the angular position and angular velocity state vectors. The characteristics of the pendulum are summarized in figure 5.1.

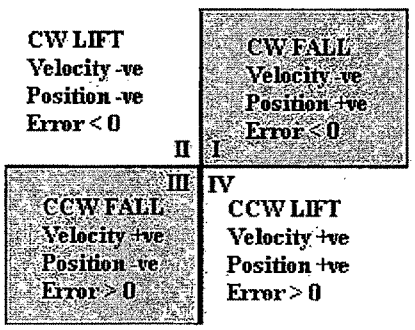

Figure 5.1: Characteristics of Pendulum on the state plane

To elaborate the operation, consider the case when the demanded value is less than the actual angular position and the pendulum is on the positive side of the state plane. The signs of the two state vectors indicate the direction and the location of the pendulum. Here, the pendulum is in the first quadrant of the Figure 5.1 and the sign of the velocity is negative then this indicates

the pendulum is moving clockwise i.e. error is negative. Therefore, if we want to maintain the required set value of the predefined rise time and overshoot then two variables will not be adequate to track the movement of the pendulum and is required to be tracked using three variables namely angular position, angular velocity and position error. Because of this two control signals are generated using three membership functions [171]. The first control signal moves the system to its set point value with desired rise time whilst the second control signal improves the steady state response. Two FAM based on the pendulum's dynamic characteristics are defined, one for position-velocity and other for error-position. Gain Scheduler is also required to be incorporated, which acts as a supervisory control so that problem of changing tuning factor for every different set point can be overcome.

Second stage of the design to obtain optimization using Genetic algorithm, conventional genetic algorithm as discussed in the previous chapter is used with uniform crossover, bit mutation and roulette wheel selection. Figure 5.2 show the flow of the design procedure for GAFLC.

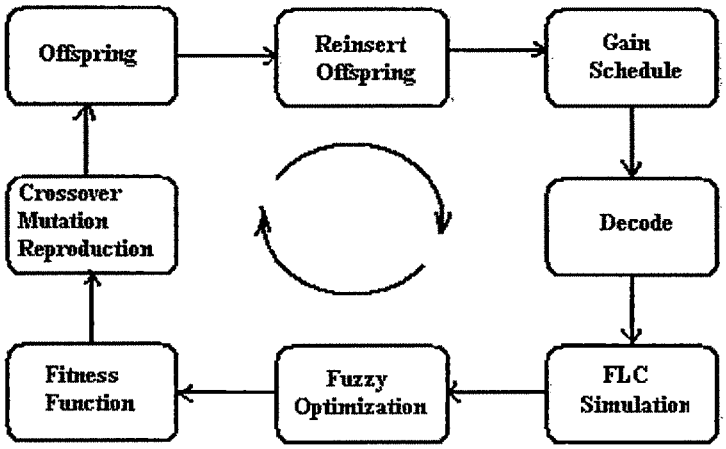

Figure 5.2: Design Flow of GAFLC

The simulation of the same [171] shows that the GAFLC for a specified demand in position gives the required performance of an angular servomechanism. The results shows that the without some form of supervisory control, the FLC's performance is very limited. By incorporating a Gain Scheduler and Genetic algorithm to optimize the scaling factors for the base rule, results have improved and yields robust controller.

5.2 Genetic Reinforcement Fuzzy Logic Controller - GRFLC

For the design of fuzzy logic system, the straight forward approach is the define rules and membership functions subjectively by studying the human operated or controlled system of an existing controller. For the cases where expert's knowledge is not available, design of the fuzzy logic controller becomes difficult. For such cases, Lin and Lee [152] proposed a fuzzy logic control / decision network, which is constructed automatically and tuned by the training examples. For training the network, supervised or unsupervised learning methods can be adopted. The learning task may include the tuning of the fuzzy membership functions used in the control rules and the identification of the main control rules. The learning task, which will be followed and developed by an architecture that can learn the fuzzy control rules automatically.

Unsupervised learning, as connectionist learning methods, do not rely on a network teacher that guides the learning process; like the supervised learning class, the teacher associates the learning system with desired outputs for each given input. Learning involves memorizing the desired outputs by minimizing the discrepancy between the actual outputs of the system and the desired output. In reinforcement learning class, the teacher provides the learning system with a scalar evaluation of the system's performance of the task according to a given index. The objective of the learning system is then to improve its performance by generating appropriate outputs.

When the input/output training data sets are available, the supervised learning is more efficient than the reinforcement learning. This has the consequences emerge over uncertain periods for which input/output training data sets are not readily available. In those cases, the reinforcement learning systems is to be employed to provide the unknown desired outputs through system with a suitable evaluation of system performances. For the cart-pole balancing problem, the reinforcement learning is more appropriate than the supervised learning.

Chiang et al [153] presented the fuzzy logic controller, which has the capability to learn its own control rules without expert knowledge. Figure 5.3 shows the architecture of the general reinforcement fuzzy logic controller (GRFLC), which uses GA's to design fuzzy logic controller in reinforcement learning where the main elements are the action evaluation network (AEN), which acts as a critic and provides advice to the main controller—the action generation network (AGN)—which includes a fuzzy controller, the stochastic action modifier (SAM), and using both F and V, produces the action F applied to the plant. The accumulator of AGN is used as a

relative measure of fitness required for GA's. The AGN is a multilayer network logic controller that consists of four components elements: GA rule base, a fuzzification, the decision making logic, and the defuzzification. The GA rule base is a collection of fuzzy *IF-THEN* statements, self-generated by a GA. The membership functions of fuzzy variables in antecedent parts, however, have to be built in advance, save the ones in consequent parts also generated by a GA. Using a GA, the optimal fuzzy singletons of the consequent parts can be obtained.

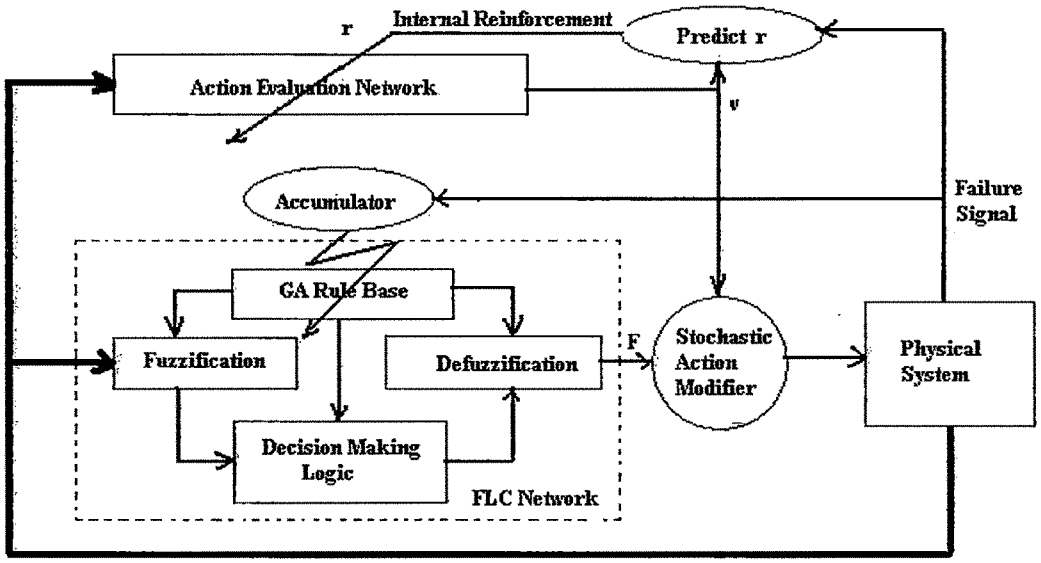

Figure 5.3: Architecture of GRFLC

The cart-pole task involves a pole hinged to the top of a wheeled cart that travels along a track. The cart and pole are constrained to move within the vertical plane. The system state is specified by four real-valued variables: x– the horizontal position of the cart; $\dot{x}$-the velocity of the cart; θ - the angle of the pole with respect to the vertical line; $\dot{\theta}$ - the angular velocity of pole; and f - the force [-10, 10] newtons, applied to the cart. The dynamics of the cart-pole system are modeled by the following nonlinear differential equations…

$$\ddot{\theta} = \frac{g\sin\theta + \cos\theta\left[\dfrac{-f - ml\dot{\theta}^2\sin\theta + \mu_c\,\mathrm{sgn}(\dot{x})}{m_c + m}\right] - \dfrac{\mu_p\dot{\theta}}{ml}}{1\cdot\left[\dfrac{4}{3} - \dfrac{m\cos^2\theta}{m_c + m}\right]} \qquad \ldots(5.1)$$

$$\ddot{x} = \frac{f + ml\left[\dot{\theta}^2\sin\theta - \ddot{\theta}\cos\theta\right] - \mu_c\,\mathrm{sgn}(\dot{x})}{m_c + m} \qquad \ldots(5.2)$$

Where g is the acceleration due to gravity, m_c is the mass of the cart, m is the mass of the pole, l is the half-pole length, μ_c is the coefficient of friction of the cart on track, and μ_p is the coefficient of friction of the pole on cart.

The goal of the cart-pole task is to apply the forces of unknown magnitude to the cart such that the pole is balanced and the cart does not hit the edge of the track. Bounds on the angle and on the cart's horizontal position specify the states for which a failure signal occurs. There is no unique problem solution—any trajectory through the state space that does not result in a failure signal is acceptable. The only information regarding the goal of the task is provided by a failure signal, which signals either the pole falling past ± 12° or the cart hitting the bounds of the track at ±2.4 m. These two kinds of failures are not distinguishable in the case considered herein. The goal, as just stated, makes this task very difficult; the failure signal is a delayed and rare performance measure.

The GRFLC architecture as shown in figure 5.3, applied to the cart–pole task. The simulation results [153] shows that because GA can seek the optimal membership functions in the consequences of control rules, the GRFLC can compensate for inappropriate definition of the membership functions in the antecedent of control rules. It is applicable to control problems for which the analytical models of the process are known since GRFLC learns to predict the behavior of the physical systems through the action evaluation network.

5.3 Multi-objective Fuzzy Genetic Algorithm for control system design

The theory of feedback linearization has provided the necessary tools for synthesis of nonlinear systems and solution of the MIMO decoupling problem. However, current industrial practice does not generally include the use of modern nonlinear robust controllers. This may be attributed to several factors, one of which is the lack of a systematic and intuitive approach in selecting the large number of free control parameters to obtain an optimal controller.

In general, the design of a control system for a nonlinear multivariable system with many degrees of freedom involves a number of constraints and competing objectives. To obtain an optimal solution (i.e. a tradeoff between the stated objectives) a formal mathematical method for decision-making is required. Multiobjective optimization techniques [173,174,175] have been proposed as a possible solution. These provide a platform for incorporating the relative

importance of each objective, a fundamental requirement when dealing with competing objectives. The only difficulty with this technique is the lack of a unique optimal solution, although the concept of a Pareto-optimum solution can be used, which will establish a tradeoff pattern such that no improvement may be made in one objective without adversely affecting another.

However, the emphasis on mathematical rigueur conflicts with the imprecision that arises in the description of constraints, objective functions and the evaluation of the relative importance of objectives. In a typical control system design the constraints are usually mixed, with equality, inequality and fuzzy constraints such as settling time and thruster limit. In order to generate flexible and robust solutions for multiobjective optimization, the concept of fuzzy sets is required to be employed to represent the vast amount of vagueness that exists in both the objective and constraint functions.

Oilennu et al [170] suggested a new framework for selecting free control parameters of an input output linearized controller with sliding mode control (SMC) for the depth control system of a remotely operated underwater vehicle (ROV). This uses the concept of multiobjective fuzzy genetic algorithm (GA) optimization and a new membership weighting strategy. An ROV used for mine countermeasure operations and remote mine-hunting missions. In this role, the ROV has sidescan sonar for mine hunting and clearance purposes. The operational requirements dictate precise control of depth and pitch in water depths of between 20m and 300m, to ensure a constant sonar footprint over an undulating seabed. Changes in depth should involve minimal pitching motion to avoid gaps in the sonar coverage.

Due to the highly nonlinear nature of the vehicle, it is natural for nonlinear control techniques such as input output linearization with SMC to be proposed as a possible solution to the depth control system. The concept of input output linearization guarantees the required decoupling between heave (depth) and pitch, while SMC provides performance and stability robustness to modeling errors. The major problem is the lack of a systematic approach for selecting the free control parameters to ensure that the actuators did not saturate.

The general multiobjective fuzzy optimization problem can be stated as:

Find X which Minimizes $f(X)$ such that $g_j \in b \cdots \sim j$,

Where, $f(X) = [f_1(X), f_2(X), \ldots, f_k(X)]$ is a vector objective function and $g_j(X)$ are constraints, with the tilde symbol indicating that the constraints fuzzy information.

The first stage is to fuzzify the objective functions and the fuzzy constraints. The membership function for the fuzzy objective functions is

$$\mu_{f_i}(X)=\begin{cases}0 & \text{if } f_i(X)>f_i^{\max}\\ -\dfrac{f_i(X)+f_i^{\max}(X)}{f_i^{\max}-f_i^{\min}} & \text{if } f_i^{\min}<f_i(X)\le f_i^{\max}\\ 1 & \text{if } f_i(X)\le f_i^{\min}\end{cases} \qquad \ldots(5.3)$$

Where, $\mu_{f_i}(X):\Re^n\rightarrow[0,1]$ is the measure of the degree of satisfaction for any $X\in\Re^n$ in the i^{th} fuzzy objective function. $f_i^{\min}$ and $f_i^{\max}$ represent the minimum and maximum values for the objective function, respectively and are defined as …:

$$f_i^{\min}=\min_i f_i(X^*) \text{ and } f_i^{\max}=\max_i f_i(X^*) \qquad \ldots(5.4)$$

Where, X* is the solution for each of the objective functions in the crisp domain. The fuzzy constraint membership function is defined as

$$\mu_{g_j}(X)=\begin{cases}0 & \text{if } g_j(X)>b_j+d_j\\ 1-\dfrac{g_j(X)-b_j}{d_j} & \text{if } b_j\le g_j(X)\le b_j+d_j\\ 1 & \text{if } g_j(X)\le b_j\end{cases} \qquad \ldots(5.5)$$

Where, $\mu_{g_j}(X):\Re^n\rightarrow[0,1]$ is the indication of the degree of satisfaction for any $X\in\Re^n$ in the j^{th} fuzzy constraint. $\mu_{g_j}(X)=1$ represents the complete satisfaction, $\mu_{g_j}(X)=0$ is not satisfied and values between 0 & 1 represents the degree of satisfaction of the j^{th} constraint. The allowable tolerances for each fuzzy constraint are given as d_j.

The optimal decisions are made by selecting the best alternatives from the fuzzy decision space D characterized by the membership function μ_D, other way round it is to find the optimum X* which maximizes μ_D.The fuzzy decisions can be made by using any one of the three generalized fuzzy decisions: intersection decision, convex decision and product decision.

Genetic algorithms are used as the preferred optimization technique to solve this problem. The choice of GAs is based on the fact that they have been shown to produce optimum solutions for parameterized nonlinear controllers in multimodal search spaces. Because GAs evaluate several points simultaneously they also have the potential to accomplish highly adaptable parallel processors, since each point is evaluated iteration by iteration. This is an important requirement in multiobjective optimization problems. There is no need to assume that

the search space is differentiable or continuous and GAs only require a knowledge of the quality of the solution produced by each point thus making them very flexible.

The multiobjective fuzzy GA optimization successfully applied to an ROV depth control problem to eliminate the actuator saturation [170], while meeting the control specification.

5.4 Optimal Fuzzy control using Genetic Algorithm for Spindle Motor

Carrying out a high speed to increase data rate and reduce access time is required in the design of compact disc servo systems. The speed of a spindle motor at the inner circle could reach 6000 rpm [176]. Hence, it is very important to control the spindle motor in a CD-ROM drive. However, the external disturbances are occurred due to the vibration of high speed. These will influence the performance of a spindle motor. To achieve the robust control against the external disturbances and the model uncertainty of the spindle motor, the fuzzy logic is implemented. Yu et al [169] proposed a novel strategy to design the optimal fuzzy controller. Genetic algorithms are applied to search the globally optimal parameters of fuzzy logic. The best ranges of membership functions; the best shapes of membership functions and the best fuzzy inference rules are dug out at the same time.

The servo systems of a CD-ROM drive are integrated optic, mechanism and electricity. They are composed of focusing servo system, seeking servo system, tracking servo system and spindle motor servo system [177].

The spindle motor servo system bears disks and rotates with an optical pick-up head to read data. The design objective is to rapidly reach the desired angular velocity and make sure the disk is stable at high speed. Therefore, the specifications of the spindle motor require high torque to start and low vibration to avoid slant. Earlier spindle motor in a CD-ROM drive was rotated by means of Constant Linear Velocity (CLV) to read data. Since the rotational speed of the spindle motor is faster at present, the angular velocity will be too high in the interior of the circle if the data is read using CLV. To reduce the noise and avoid the overheated problem, the data is read by mixing CLV and Constant Angular Velocity (CAV). The optical pick-up-head reads data in the interior of the circle using CAV, and reads data in the exterior of the circle using CLV.

The spindle motor is a kind of DC motor. The traditional DC motor needs the brush and the commutator to produce the continuously rotational torque. The drawbacks of the brush and the commutator at high rotational speed include the wastage due to friction, the difficulty in

raising the rotational speed and the loud noise. Nowadays the spindle motor in a CD-ROM drive adopts the brushless DC motor. The brushless DC motor owns high torque to start, low slant, low vibration, and without mechanical loss. The magnetic field of the brushless DC is generated by means of the driving circuit and the magnetic pole detection. The Hall sensor is applied to detect the magnetic pole of the magnet. Therefore, the advantages of the brushless DC motor include miniaturization, long life, without mechanical noise and high rotational speed.

To design the optimal fuzzy controller, the genetic algorithms are applied to search the globally optimal parameters of the fuzzy logic. The structure of the fuzzy logic controller with genetic algorithms is shown in figure 5.4.

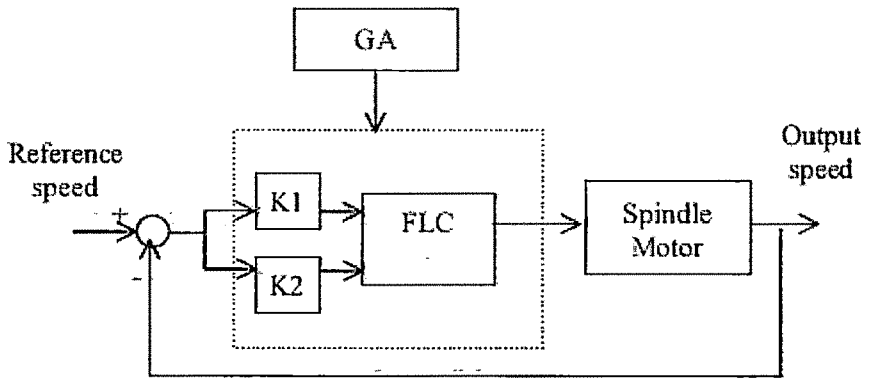

Figure 5.4: FLC with GA

Yu et al [169] formed the chromosomes of the genetic algorithms consisting of three parts: the range of the membership functions (K1 and K2), the shape of the membership functions (e1~e3,de1~de3 and u1~u3) and the fuzzy inference rules (r1~r9). It makes the output voltage to be optimal such that the steady-state error of the response is zero. The genes in the chromosomes are defined as [K1, K2, e1, e2, e3, de1, de2, de3, u1, u2, u3, r1, r2, r3, r4, r5, r6, r7, r8, r9].

In implementing fuzzy controller, the linguistic variables are defined as $\{N, Z, P\}$, where N means negative, Z means zero and P means positive. The triangular membership functions are defined; where as the Mamdani type fuzzy inference engine is used. GA is used with arithmetic crossover and uniform mutation. Yu et al [169] simulated for control of the speed of a spindle motor in a CDROM drive by means of three different fuzzy controllers. The optimal fuzzy logic is designed using genetic algorithms is found to be the best controller and possesses good

robustness, no matter how the spindle motor is subjected to disturbances or has model uncertainty.

5.5 Evolutionary Design for Robust Flight Control

The basic structure of the controllers is predetermined [178] and the free parameters are then optimized by a genetic algorithm, so that the simulated flight responses for a variety of initial conditions display desirable properties, such as long term stability, fast settling, disturbance rejection and broad range performance. The genetic algorithm is a zero-order search procedure, where the only information used to direct the search process is a performance measure, referred to as the objective function, computed from a set of simulations. Though the design procedure is essentially a brute force approach, it has been configured, in terms of the controller structure, the search algorithm, and the adaptive performance measure, to moderate the computation time required.

There are a number of advantages to designing the controller with an optimization tool and a performance metric abstracted from the randomly perturbed flight responses. Firstly, it relieves a common issue faced by many control design approaches, namely representing the vehicle mathematically in an appropriate form. The accuracy of the model is a function of available computing power and the knowledge of the vehicle physical properties and the processes governing the performance, rather than being bound by the structure of the control design procedure. In conventional design theories the system is typically assumed to be LTI and, in the case of robust control theory, uncertainty added to the system to account for system nonlinearities and variations with time. Representation of performance uncertainty is critical for the development of a robust control law. Much work in robust control theory is directed towards the development of compatible structured and unstructured uncertainty models. When the simulated flight responses are used, the inclusion of parametric uncertainty can describe the physical process leading to the variations in the vehicle performance, through the inclusion of appropriate simulation models.

Another advantage of the design approach is that the control law development is linked directly to the time history responses, allowing stability and performance measures to be easily quantified. The genetic algorithm does not need the components of the objective function to be the same throughout the design. They too can evolve with the controller design so that as the

controlled flight responses improve, greater demands can be placed on the performance of the controller. The only constraint on the objective function is that it be reduced to a scalar performance measure. Further supporting the evolutionary design procedure is the inherent robustness to noisy objective functions. The source of the noise is typically due to randomly sourced uncertainty models and sensor noise, but may also be generated by variation in the initial condition set used to assess the controller performance. Since the search is probabilistic, a significant level of noise in the objective function can be tolerated, while still allowing the genetic algorithm to successfully configure the control law.

Though the genetic algorithm is noted for its global search capabilities, it is also extremely opportunistic. Considerable care is therefore needed when defining objective functions, and when combining multiple and possibly conflicting design objectives. However, this is a feature which must be addressed in all optimal control theories. In problems where non-commensurate objectives are unavoidable, evolutionary algorithms are considered to be particularly suited since a set of solutions are processed in parallel. One means of dealing with such problems is to use a multi-objective genetic algorithm [179,180] to obtain Pareto-optimal solutions.

One potential problem in an iterative design approach is the "curse of dimensionality". As the number of design parameters increases there may be an exponential increase in the effort required to arrive at the solution. Though this can be mitigated by providing some structure to the design, it is important that a large number of design parameters can be dealt with. Evolutionary based search procedures are readily applied to problems of high dimension, and are able to rapidly extract useful designs in spite of the size of the problem. If the absolute global minimum or maximum of a complex multi-modal search space is required, then the computing effort remains considerable. However there are few algorithms capable of performing well on such functions and the notion of an efficient search procedure is still being established. The focus of this effort is the design of an inner-loop attitude controller which would offer closed-loop vehicle stability, subject to system uncertainties, broad range performance variations, disturbances, sensor noise, and severe operational constraints.

5.6 Self Tuning Neuro-Fuzzy Controller by GA (NFCGA) for Coupled Tank level control

As we know, fuzzy logic control involves three main stages: fuzzification, inference, and defuzzification. The first and last stages are needed to convert and re-convert real world crisp signals into fuzzy values and vice-versa. The inference or reasoning mechanism can be described as follows, the first is to determine the matching degree of the current fuzzy input (class) with respect to each rule, i.e. the IF part. The mechanism then decides which rules are to be fired according to the input field. Finally, the fired rules are combined to form the control actions, i.e., the THEN part or consequents of the fuzzy control rules. The above procedure can be further simplified to as only pattern matching and weights averaging, thereby, eliminating the procedure of fuzzification and defuzzification [181, 182]. The first operation of the algorithm deals with the *IF* part of the fuzzy control rules; it determines the matching degree of the current input to the condition of each of the fuzzy control rules. The matching degree process is simply an operation that returns the matching level, $h_i \in [0,1]$ between the inputs and the rule pattern for the i^{th} rule. The matching formula can be written as follows:

$$h_i = \exp\left(-\left\| \frac{c^i_{x,n} - x_n}{D^i_{x,n}} \right\|^2 \right) \quad for\ i = 1\ to\ T \qquad \ldots(5.6)$$

Where T is the total number of fuzzy rules, and $\|\bullet\|$ is the norm operator given by

$$\|\bullet\| = \sqrt{\sum_{n=1}^{N} (\bullet_n)^2} \qquad \ldots(5.7)$$

Then averaging of the weights is applied to obtain the control action of each output variable, control action y_m, which corresponds to the input vector x, is given by

$$y_m = \frac{\sum_{i=1}^{T} \left(h_i \cdot c^i_{y,m} \right)}{\sum_{i=1}^{T} h_i} \qquad \ldots(5.8)$$

This weight averaging method uses only the center of the *THEN* part of the rule $c^i_{y,m}$, which is defined as a singleton output variable. The algorithm therefore can be understood as a modification of the maximum membership decision scheme, where the global center is calculated by the center of gravity algorithm.

The computational procedure of the above fuzzy algorithm is closely similar to the implementation of a normalized version of radial basis function neural network (RBF). However, the center and the width of the radial basis nodes in the RBF are determined differently, i.e., by clustering algorithms and 'root mean square' method respectively [183]. The radial function of the RBF can be treated as the fuzzy membership functions, which are characterize by two parameters, i.e., the center and the width. By appropriately choosing the center and width of the radial basis units, which forms the fuzzy membership functions, the network can be used to represent the rule-base fuzzy knowledge. Each of the fuzzy rule inference mechanism is being process in a radial basis node, thus, the i^{th} rule in (5.6) can be written as:

$$IF\left(c_{x,1}^{i}, D_{x,1}^{i}\right) and \ldots\ldots \left(c_{x,n}^{i}, D_{x,n}^{i}\right) \ldots\ldots and \left(c_{x,N}^{i}, D_{x,N}^{i}\right) THEN\left(w_{i1}\right) and \ldots\ldots \left(w_{im}\right) \ldots\ldots and \left(w_{iM}\right) \quad \ldots(5.9)$$

Where, the control action is determined by the connection weights w_{im}, a singleton real number value.

A general block diagram of a multi-input and multi-output RBF based FLC is shown as in Fig. 5.5. The input layer accepts the system state feedback (x_1, x_2, ..., x_n ,.....,x_N), and the fuzzy inferencing is processed at the hidden layer. The output layer implements the normalization operation to produce the control signals (y_1, y_2, ..., y_m,.....,y_M).

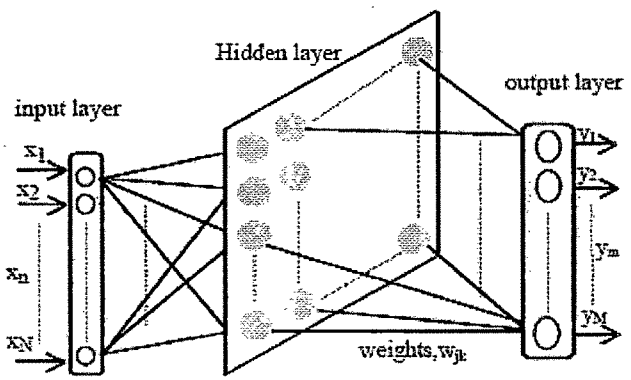

Figure 5.5 MIMO RBF based FLC Network Architecture

In order to further visualize this concept, consider a fuzzy logic control system where the FLC has two input variables, namely, the error (e) and the change of the error (De). Each of these variables takes five Gaussian types of fuzzy membership functions that are labeled as PB, PS, Z, NS and NB. Each of the membership functions has two parameters, i.e., the center and

width of the Gaussian functions. The multi-variants Gaussian can also be viewed as the product of a single-variants Gaussian function. It performs conjunctive operation in the 'premise' part of the fuzzy rules in the hidden layer. Figure 5.6 shows the rule base matrix of the corresponding fuzzy basis units at the hidden layer of the controller. Each of the kernel squares represents one control rule. Thus, the number of the hidden nodes for this network is exactly equal to the number of fuzzy control rules. The output of these units is the matching degree or inferred result of the particular fuzzy control rules.

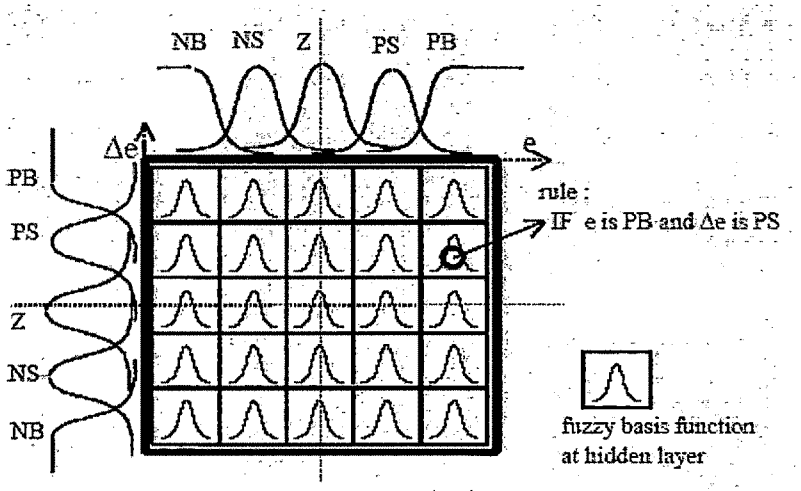

Figure 5.6 Fuzzy Basis functions at hidden layer.

The strength of the controller output depends on the interconnected weights between the hidden layer and the output. The output is computed by normalizing the weights as follows:

$$y_m = \frac{\sum_{i=1}^{P}(h_i \cdot w_{im})}{\sum_{i=1}^{P} h_i} \qquad \ldots(5.10)$$

Where P is the number of hidden units (rules), and $h_i \in [0,1] \in \Re$ is the output of the fuzzy basis function of each rule, w_{im} is the weight that connects the i^{th} local unit to the m^{th} output node. Figure 5.7 shows graphical view of this implementation.

The controller output y_k is a crisp value that can be readily applied to the system. GAs are then implemented as an optimization algorithm to tune all the parameters of this RBF-based FLC.

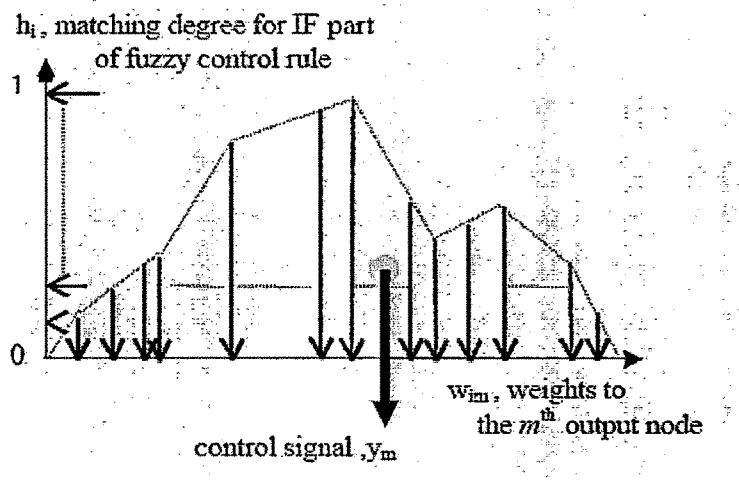

Figure 5.7 Computation of y_m for RBF based FLC

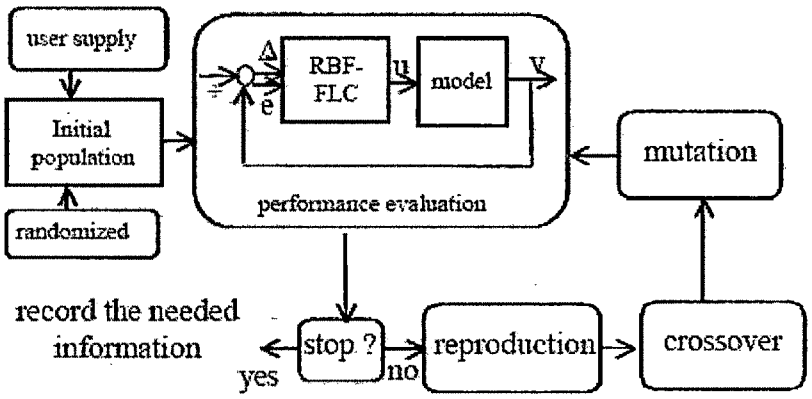

Figure 5.8: Functional block diagram of GA optimization process

Figure 5.8 shows the functional block diagram of GA optimization process. At the beginning of the process, the initial populations comprise a set of chromosomes scattered all over the search space. The use of heuristic knowledge of the controller is minimized, by way of random initialization of the initial population.

After each of the chromosomes is evaluated and associated with its fitness value, the current population undergoes reproduction process to create the next generation of population using the 'Roulette wheel' selection scheme [144]. After the new group of population is built, the mating pool is formed and the crossover is carried out. This is then followed by the mutation

operation. Generally, after these three operations, the overall fitness of the population improves. Each of the population generated then goes through a series of evaluation, reproduction, crossover and mutation, the procedure is repeated until the termination condition is reached. After the evolution process, the final generation of population consists of highly fit strings that provide optimal or near optimal solutions.

Seng et al [184] implemented a self-tuning FLC based on the RBF neural network (NFCGA), where all of its parameters are simultaneously tuned by GA. The NFCGA is then applied to a noisy coupled-tank fluid level control with non-linear dynamics in real-time. The performance of the NFCGA has been compared with a conventional FLC and PID controller in terms of transient response, load disturbance and changes in plant dynamics. It was observed that the NFCGA is more superior to the other two controllers [184].

5.7 Genetic Programming based Design of Fuzzy logic controller for Mobile Robot Path Tracking

A variety of evolutionary algorithms, operating according to Darwinian concepts, exists to approximately solve problems of common engineering applications. Increasingly common applications involve automatic learning of nonlinear mappings that govern the behavior of control systems. In many cases where robot control is of primary concern, the systems used to demonstrate the effectiveness of evolutionary algorithms often do not represent practical robotic systems. Genetic programming (GP) is the evolutionary strategy, which can be applied to learn fuzzy control rules for a practical autonomous vehicle steering control problem, namely, path tracking. GP handles the simultaneous evolution of membership functions and rule bases for the fuzzy path tracker.

In the GP paradigm, a population is comprised of computer programs or procedures (individuals) that are candidate solutions to a particular problem. These individuals participate in a simulated evolution process wherein the population evolves over time in response to selective pressure induced by the relative fitness of individuals in the problem domain. In [185] approach, each program executes condition-action statements, which collectively serve as a rule base to be embedded in a fuzzy controller. To preserve diversity among populations and vital genetic information among individuals, genetic operators are applied to create new individuals for

succeeding generations. When the algorithm finally converges or satisfies its termination criteria, it is anticipated that the best or most fit individual will be representative of an optimum or near optimum solution.

Let us first discuss about Mobile Robot path tracking problem, which was formulated by Hemami et al [186] for a class of low speed (less than 2 m/s) tricycle-model vehicles. Essentially, the control objective is to successfully navigate a mobile robot or Automated Guided Vehicle (AGV) along a desired path in a two-dimensional environment. We have to automatically design a fuzzy controller that will achieve this objective. The inputs consist of a measurable position error, ε_d, and a measurable orientation error, ε_θ associated with path following in the plane, refer figure 5.9. The output is the steering angle, δ, which is the corrective control action that would cause the errors to approach zero and, thus, force the robot to follow the desired path.

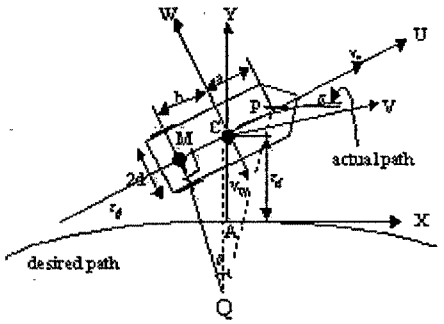

Figure 5.9: Mobile Robot – tracking control and error

The position error is taken as the deviation of the center of gravity, *C*, or any other desired point of the robot from the nearest point on the path. The orientation error is the angular deviation of the robot from the tangent of the desired path. Hemami et al [186] derived a state-space kinematics model for this robot where the state vector was comprised of the pose errors described.

$$\begin{bmatrix} \dot{\varepsilon}_d \\ \dot{\varepsilon}_\theta \end{bmatrix} = \begin{bmatrix} 0 & V_u \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \varepsilon_d \\ \varepsilon_\theta \end{bmatrix} + \begin{bmatrix} MC/MP \\ 1/MP \end{bmatrix} V_u \tan\delta \pm \begin{bmatrix} \dot{\eta}_d \\ \dot{\eta}_\theta \end{bmatrix} \qquad \ldots(5.11)$$

Where V_u is forward linear velocity of the robot, and $\dot{\eta}_d$ and $\dot{\eta}_\theta$ are rates of change of the effects of path curvature. We know that for a small steering angle, $\delta(\tan\delta \cong \delta)$, Equation 5.11 approximates the slow dynamics of the vehicle when its forward velocity is low. For simulations, robot kinematics model have been simplified by assuming small steering angle approximation into account. Furthermore, the controller is applied to straight-line path following and, therefore, neglect the model effects of path curvature. Such a simplification does not preclude autonomous tracking of reasonably complicated paths since multi-segment paths can be defined to be piecewise linear.

The path tracker to be learned by GP is a two-input, single-output fuzzy controller that will map the error states into a proper steering angle at each time step. A population of candidate solutions is created from which a solution will emerge. The allowance for rule bases of various sizes enhances the diversity of the population. That is, the GP system creates individuals in the initial population that each have possibly different numbers of rules within a range (15-30) specified before a run. In the process of learning fuzzy control rules and membership functions, GP manipulates the linguistic variables directly associated with the controller. Given a desired motion behavior, the search space is contained in the set of all possible rule bases that can be composed recursively from a set of functions and a set of terminals. The function set consists of membership function definitions (describing controller inputs), components of the generic fuzzy *if-then* rule, and common fuzzy logic connectives. More specifically, these include functions for fuzzy sets, rule antecedents and consequents, fuzzy set intersection and union, and fuzzy inference. The terminal set is made up of the input and output linguistic variables and the corresponding membership functions associated with the problem.

Selection of appropriate t-norms is automated, thereby, giving the GP system greater control of the evolutionary design. That is, the influence of GP is extended to include selection of the type of t-norm employed to compute the conjunction of fuzzy propositions in the antecedent of a rule. The two most commonly used t-norms for fuzzy control are Mamdani's *min* and Larsen's *product* [187]. T-norms for each conjunctive rule are selected at random by GP for rule bases in the initial population, and are carried along based on fitness in successive generations.

To achieve the goal of evolving FLCs, the GP system must conform to strong syntactic constraints when breeding individuals. Special rules of construction were introduced in [154]. In order to accommodate evolution of input membership functions, in addition to the rule base, the allowable syntax was extended in [188] and implemented using a representation proposed in [160].

GP was successfully applied to discover fuzzy controllers capable of navigating a mobile robot to track straight-line paths in the plane. The performance of the best-evolved FLC was comparable to that of a manually derived FLC, which required a considerably longer design cycle. GP simultaneously evolved membership functions and rules for an FLC that demonstrated satisfactory responsiveness to various initial conditions while utilizing minimal human interface. The speed of evolution alone serves as a strong basis for practical application of GP in the controller design process. The approach enables expeditious design of FLCs that can be directly applied to a physical system. Alternatively, human experts can use the rapidly evolved FLCs as design starting points for further manual refinement. Finally, the evolved FLC is robust to perturbations of sensor noise and an increase in nominal robot speed.