NEURAL CONTROLLER UTILIZING GENETIC ALGORITHM TECHNIQUE FOR DYNAMIC SYSTEMS

MARWAN A ALI

UNIVERSITI SAINS MALAYSIA
2009

NEURAL CONTROLLER UTILIZING GENETIC ALGORITHM TECHNIQUE FOR DYNAMIC SYSTEMS

by

MARWAN A. ALI

Thesis submitted in fulfillment of the
Requirements for the degree of
Master of Science

October 2009

Acknowledgements

I would like to express my heartfelt gratitude to my supervisor, Dr. Harsa Amylia Mat Sakim for her support and motivation. Her guidance and advice has inspired me to generate fruitful approaches in achieving the objective of this research. Without her effort, I would not be able to bring this research to a completion. Also, my gratitude goes to my co-supervisor; Dr Rosmiwati Mohd Mokhtar for her support and advice to complete this thesis.

My gratitude is extended to my beloved family members that support and encourage me all along without hesitation. I treasure dearly their encouragement and moral support.

Last but not least, I would like to express my sincere gratitude to Malaysia my second country for unlimited support and hospitality.

TABLE OF CONTENTS

CHAPTER 5 – DISCUSSION AND CONCLUSION

List of Tables

List of Figures

List of Abbreviations and Symbols

AI	Artificial Intelligent
ANNs	Artificial Neural Networks
BIBO	Bounded Input-Bounded Output
BP	Back Propagation
CMAC	Cerebeller Model Articulation Controller
EA	Evolutionary Algorithms
EC	Evolutionary Computation
EP	Evolutionary Programming
ES	Evolutionary Strategies
ESN	Echo State Network
FLC	Fuzzy Logic Controller
FNN	Feedforward Neural Network
GAs	Genetic Algorithms
GDR	Generalized Delta Rule
MIMO	Multi Input Multi Output
MLP	Multilayer Perceptron
MRAC	Model Reference Adaptive Controller
MRAFC	Model Reference Adaptive Fuzzy Controller
MSE	Mean Square Error
NNs	Neural Networks
PI	Proportional Integral
PID	Proportional, Integral, Derivative
RLS	Recursive Least Square
SAE	Sum Absolute of Error
SISO	Single Input Single Output
STC	Self-Tuning Controller
TDL	Tapped Delay Line
TSS	Total Sum Squared
$a_{m0,}\ a_{m1}\ldots.$	Parameters of the Denominator Polynomial of the Model Reference
A-Z	Non-Terminal Symbols
a-p	Terminals Symbols
$b_{m0,}\ b_{m1}\ldots.$	Parameters of the Nominator Polynomial of the Model Reference

C_{max}	Constant Value to Map the Objective Function to Fitness Function
$e(k)$	Output Error Between $y_p(k)$ and $y_m(k)$
$e_f(k)$	Feedback Error Between $r(k)$ and $y_p(k)$
$e_i(k)$	Error between $y_p(k)$ and $\hat{y}_p(k)$
F	Cord Tension
g	Gravity Constant
$G_P(s)$	Plant Transfer Function in s-domain
$G_m(s)$	Model Reference Transfer Function in s-domain
h_s	Simulation Step Size
H	Hyperbolic Tangent Activation Function
k	Discrete Time Instant
K	Gain
k_P	Proportional Gain
k_i	Integral Gain
k_d	Derivative Gain
I	Actual Cord Length
L	Linear Activation Function of the Output Node
m	Mass Load
n_i	Number of Input Nodes
n_h	Number of the Hidden Nodes
n_o	Number of Output Nodes
n_l	Number of Hidden Layers
N_p	Number of the Training Patterns
Net_j	Weighted Sum of the Inputs of the Output Node j in the Hidden Layer
net_o	Weighted Sum of the Inputs of the Output Nodes
Pc	Crossover Probability or Crossover Rate
Pm	Mutation Probability or Mutation Rate
$r(k)$	Reference Input
S	Start Symbol and Non-Terminal
T_{ob}	Observation Time
T_i	Integer Time
T_d	Derivative Time
T_s	Sampling Time

u	Control Signal
$u\,(k)$	Control Signal at Sample k
$W,\ V$	Weight Matrices
$x_{\,t}$	Trolley Position
$x_{\,l}$	Actual Horizontal Load Position
$y_p\,(k)$	Output of the Plant at Sample k
$y_m\,(k)$	Output of the linear Model-Reference at Sample k
$y_{\,l}$	Actual Vertical Load Position
$\hat{y}_{\,p}\,(k)$	Estimated Plant Output
$\hat{p}$	Identification Model
μ	Constant Value to Avoid Division by Zero
ε	Empty-String Terminal
τ	Time Constant
ω	Swing Frequency
α	Angle Between the Cord and the Vertical Axis

PENGAWAL NEURAL MENGGUNAKAN TEKNIK ALGORITMA GENETIK UNTUK SISTEM-SISTEM DINAMIK

ABSTRAK

Kajian ini mengemukakan kaedah pembelajaran rangkaian neural (NN) pelbilang lapisan dengan menggunakan teknik algoritma genetik (GA). Teknik evolusionari berasakan GA ini dikaji dan digunakan untuk skima model rujukan kawalan suai (MRAC) bagi loji-loji yang berbeza. GA digunakan untuk memilih bilangan nodus-nodus tersembunyi yang optima bagi pengawal neural, di samping melatih rangkaian berkenaan bertujuan untuk meminimakan ralat antara keluaran loji dengan keluaran model rujukan. Contoh-contoh simulasi mempamerkan bagaimana nodus-nodus tersembunyi tersebut beradaptasi melalui penjanaan sehinggalah bilangan integer optima tercapai, yang mana ia bergantung kepada kerencaman loji kawalan. Operator pengkodan sebenar GA digunakan dalam kajian ini memandangkan pengkodan binari tradisional adalah terhad. Seterusnya, kaedah pemilihan hibrid serta strategi elitisme digunakan untuk proses pengeluaran semula GA. Perbandingan juga dibuat antara pengawal neural yang dicadangkan ini dengan pengawal perkadaran, pembezaan, pengkamiran (PID) klasik yang telah dilaras secara genetik. Hasil akhirnya digunakan sebagai pengawal suap belakang berasaskan model rujukan. Analisis perbandingan juga dibuat berasaskan kadar penumpuan, keupayaan pengesanan model rujukan yang diingini dan ketegapan pengawal-pengawal terhadap gangguan keluaran. Berdasarkan keputusan simulasi, dapat disimpulkan bahawa pengawal neural yang dicadangkan ini adalah lebih baik berbanding pengawal PID dalam kedua-dua aspek: ketegapan dan keupayaan pengesanan.

NEURAL CONTROLLER UTILIZING GENETIC ALGORITHM TECHNIQUE FOR DYNAMIC SYSTEMS

ABSTRACT

This research presents a method of learning multilayer Neural Network (NN) using Genetic Algorithms (GAs) techniques. The evolutionary techniques based on GAs are studied and employed for the Model Reference Adaptive Control (MRAC) scheme of different plants. The GAs are used for selecting an optimal number of hidden nodes for the neural controller, as well as training the network to minimize the error between the output of the plant and the output of the model reference. The simulation examples demonstrate how the hidden nodes are adapted through the generation until they reach their optimal integer number, which depends on the complexity of the controlled plant. The real-coding operators of GAs have been used in this work because of the limitations of traditional binary coding. Moreover, a hybrid selection method plus elitism strategies are used for reproduction process of the GAs. A comparison between the proposed neural controllers with a classical genetically tuned, Proportional Derivative Integral (PID) controller is made, in which the final outcome is used as a feedback controller based on model reference. A comparative analysis is also made based on the speed of convergences, the tracking ability of the desired model reference and robustness of these controllers to output disturbances. Based on simulation results, it is concluded that the proposed neural controller is better than PID controller in both: robustness and tracking ability.

CHAPTER 1

GENERAL INTRODUCTION

1.1 Background of Research

In general, the theory of dynamical systems is the paradigm for modeling and studying phenomena that undergo spatial and temporal evolution. These phenomena range from simple pendulum to complex atomic lattices, from planetary motion to the weather system, from population dynamics to complex biological organisms. The application of dynamical systems has nowadays spread to a wide spectrum of disciplines including physics, chemistry, biochemistry, biology, economy and even sociology. In the past, modeling was mainly restricted to linear or almost linear systems for which an analytical treatment is tractable.

In recent years, with the advancement of powerful computers and the theory of dynamical systems, it is now possible to tackle, at some extent into nonlinear systems. After all, nonlinearity is at the heart of most of the interesting dynamics. The application of the intelligent control schemes has attracted the attention of the researchers in the field of control dynamic systems (Yurkovich and Passino, 1999). In the areas of control and system engineering, the feed forward networks, such as the Multilayer Preceptron (MLP), Radial Basis Function networks (RBF) and the Cerebeller Model Articulation Controller (CMAC), seem to have attracted the most attention in the area of adaptive control (Brown *et al*., 1997; Lightbody and Irwin, 1995).

Neural networks have broad applicability to real world problems, such as in pattern recognition, diagnostic, optimization, system identification control, robotics and others. They have already been successfully applied in many industries, as they are well suited in prediction or forecasting due to abilities in identifying patterns or trend in data (Lightbody and Irwin 1995). According to their structures and learning algorithms, neural networks can be classified into two main types that are supervised such as Back Propagation (BP) and unsupervised learning algorithms such as Kohonen. There exists a third type that is called "reinforcement learning", which can be regarded as a special form of supervised learning. An example of reinforcement-learning algorithm is the Genetic Algorithm (GAs).

1.2 Neural Networks and Genetic Algorithms in Control Engineering: A Survey

Many control engineering problems that require some optimal performance have been solved in recent years using NNs and GA. For instance, Karunanithi, *et al,* (1992) applied techniques of GAs to NNs problems. These problems include a process of optimizing the weighted connections in feedforward NNs using both binary and real value representations. They discovered novel architectures in the form of connectivity patterns for neural networks by using error propagation.

Park *et al,* (1995) introduced a new method for training multilayer NN that used GA techniques. They have tested that a method, called “Genlearn” is significantly faster than methods that used the Generalized Delta Rule (GDR). Unlike the GDR, GA has the advantage, in which, it can escape local minima in its search of weighting space. The inventions of GAs have been used over the past twenty-five years in the control field. Porter and Jones (1992) have explicitly adopted GAs to solve a three-term

controller design problem. They offered new guideline in control and automation. Many researchers in control engineering field have used GAs to solve many complex problems that are difficult or impossible to solve by traditional methods.

Canelon *et al.* (2005) proposed a new approach to control nonlinear discrete dynamic systems, which relies on the identification of a discrete model of the system by a neural network. A locally equivalent optimal linear model is obtained from the neural network model at every operating point that the system goes through during the control task. Based on the linear model, a linear state-space control design technique can be used to design a local control action and is applied at that particular operating point.

Jones and Oliveira (1995) proposed the techniques of GAs to identify a model of the process using on-line data. Then, the identified model, the GA and simulation methods, are used to off-line tune a PID controller, so as to minimize a time domain based cost function. Finally, the genetically tuned controller is implemented on-line on the real process. In same year, Zitar and Hassoun (1995) introduced a machine learning system for the rule extraction of temporal control problems. The system used a hybrid of reinforcement learning and GA search to train Feedforward Neural Networks (FNNs) that can successfully control highly nonlinear and noisy systems.

Ajlouni *et al.* (1996) proposed the techniques of GAs as a mean of designing fuzzy gain-scheduled PI control schemes for a class of nonlinear plants, where the nonlinearity is a function of plant output. They showed that the use of GAs for this purpose results in highly effective fuzzy gain-scheduled control systems. Other than

that, Porter and Mohamed (1993) discussed the use of GAs to solve the dynamic plant inversion problem for trajectory control.

Park *et al.* (1995) investigated a neural-controller for non-minimum phase system, which trained off-line with GA. Jonse *et. al.* (1997) adopted the techniques of GAs to identify a model of process using experimental data. The identified model, the GA and simulation method, are used to off-line tune the dual mode controllers-based cost function. Bonissone *et al*. (1996) used GAs to tune the fuzzy controller's performance by adjusting its parameters (the scaling factors and the membership functions) in a sequential order of significance. They demonstrated this approach to the manually designed one and with only modest computational effort. Finally, the genetically tuned controller is implemented on-line on real process.

Krohling *et al.* (1997) proposed an alternative procedure based on real-coded GAs with appropriate operators to determine PID controller parameters by minimization of an integral performance criterion in frequency domain. As well as, Krohling (1998) proposed the techniques of GAs to design crane controllers. The crane is controlled by parameters of control law without linearizing the model of system. The proposed method offers an alternative to more conventional method such as pole placement design.

Soft computing is proposed by Zadeh to construct new Artificial Intelligent (AI) and to solve nonlinear and mathematically unmodeled systems problems (tractability). It is the fusion or combination of fuzzy, neural and evolutionary GA computing (Dote,

1999). Krishnapura and Jutan (2000) proposed an adaptive neural network controller for the control of nonlinear dynamical systems.

1.3 Evolutionary Computation

Several computer scientists in 1950s and 1960s studied evolutionary systems with the idea that evolution could be used as an optimization tool for engineering problems. The idea in these systems was to evolve a population of candidate solutions to a given problem using operators inspired by natural genetic variations and natural selection. Evolutionary Strategies (ES) were introduced in the 1960s by Rechanberg. He used it to optimize real-valued parameters for devices such as airfoils. The field of ES has remained an active area of research, mostly developing independently from the field of GAs, which will be explained later in more details.

Evolutionary Programming (EP) developed by Fogel, Owen and Walsh (Mitchell, 1998) was a technique in which candidate solution to a given tasks are represented as finite-state machines. They are developed by randomly mutating their state-transition diagrams and selecting the fittest. Together, ES, EP and GAs form the backbone of the field of Evolutionary Computation (EC). GAs were invented by John Holland and were developed by him and his colleagues at the University of Michigan in the 1960s and 1970s (Mitchell, 1998).

In contrast with ES and EP, Holland‘s original goal was not to design algorithms to solve specific problems, but rather to formally study the phenomenon of adaptation as it occurs in nature and to develop ways, in which the mechanism of natural adaptation might be imported into computer systems. Holland‘s GA is a method for

moving from one population of "chromosomes" (e.g., string of ones and zeros, or "bit") to a new population by using a kind of "natural selection" together with genetic inspired operators of crossover, mutation and inversion. Each chromosome consists of "genes" (e.g, "bit") and each gene being an instance of a particular "allele" (e.g., 0 or 1). The selection operators chose those chromosomes in the population that will be allowed to reproduce and on an average the fitter chromosomes produce more offspring's than the less fit ones.

Crossover exchange subparts of the two chromosomes roughly mimicking biological recombination between the two single-chromosome organisms. Mutation randomly changes the allele values of some sections of the chromosome. Holland's introduction of a population-based algorithm with crossover, inversion and mutation was major innovation. Until recently, the theoretical foundation of schema theorem of Holland was the basis of almost all-subsequent theoretical work on GAs and computational evolution (Mitchell, 1998).

1.3.1 How are GAs Different from Traditional Methods

EC including GAs are found to be very good at optimization in applications for engineers and scientists. GA is a search algorithm based on the mechanics of natural selection and natural genetic (Goldberg, 1989). GAs work with a coding of the parameter set not the parameters themselves. Instead of searching a single point, GAs search from a population of points in the decision space to determine the next point. Therefore, this point-to-point method is dangerous because it is a perfect prescription for locating false peaks in multi-modal search spaces. In this way, GAs work from rich database of point simultaneously (a population of strings). The probability of finding a

false peak is reduced. GAs use payoff (objective function) information, instead of derivatives or other auxiliary knowledge. The transition rules of GAs are stochastic while many other methods have deterministic transition rules. A distinction exists, however, between the randomized operators of GAs and other methods that are simple random walks. GAs use random chance to guide a highly exploitative search. This may seem unusual, using chance to achieve the directed results (the best points), but nature is full of precedent. The power of GAs is clearly beneficial in the case of searching among complex search space with many discontinuities or local maxima (or minima). GAs manipulates decision on control variable representations at the string level to exploit similarities among high performance strings.

Other methods usually deal with functions and their control variables directly. Because GAs operate at the coding level, they are difficult to fail when the function may be difficult for traditional schemes. Huge search space does not affect the complexity of GAs computation but affect the time of computation. Other methods rely heavily on such information and on problems where the necessary information is not available or difficult to obtain. GAs remain general by exploiting information available for any search problem. GAs processes similarities in the underlying coding together with information ranking the structure according to their survival capability in the current environment.

1.3.2 The Simple GA

Given a clearly defined problem to be solved and bit string representation for candidate solutions, a simple GA works as follows:

1- Start with a randomly generated population of (n_1-bit) chromosome (candidate solutions to a problem).

2- Calculate the fitness of each chromosome in the population.

3- Repeat the following steps until *n* offspring's have been created.

   a) Select a pair of parent chromosomes from the current population, the probability of selection being an increasing function of fitness. Selection is done "with replacement", meaning that the same chromosome can be selected more than once to become a parent.

   b) With the probability, denoted as *Pc* (the "crossover probability" or "crossover rate"), crossover the pair at a randomly chosen point (chosen with uniform probability) exact copies of their respective parent.

   c) Mutate the two offspring's at each locus with probability *Pm* (the mutation probability or mutation rate), and place the resulting chromosomes in the new population.

4- Replace the current population with new population.

5- Go to step 2.

Each iteration is called a "generation". The entire process of generation is called a "run". At the end of a run there are often one or more highly fit chromosomes in the population. Since randomness plays a large role in each run, two runs with different random-number seeds will generally produce different detailed behaviors. The schematic diagram of GA is represented in Figure 1.1.

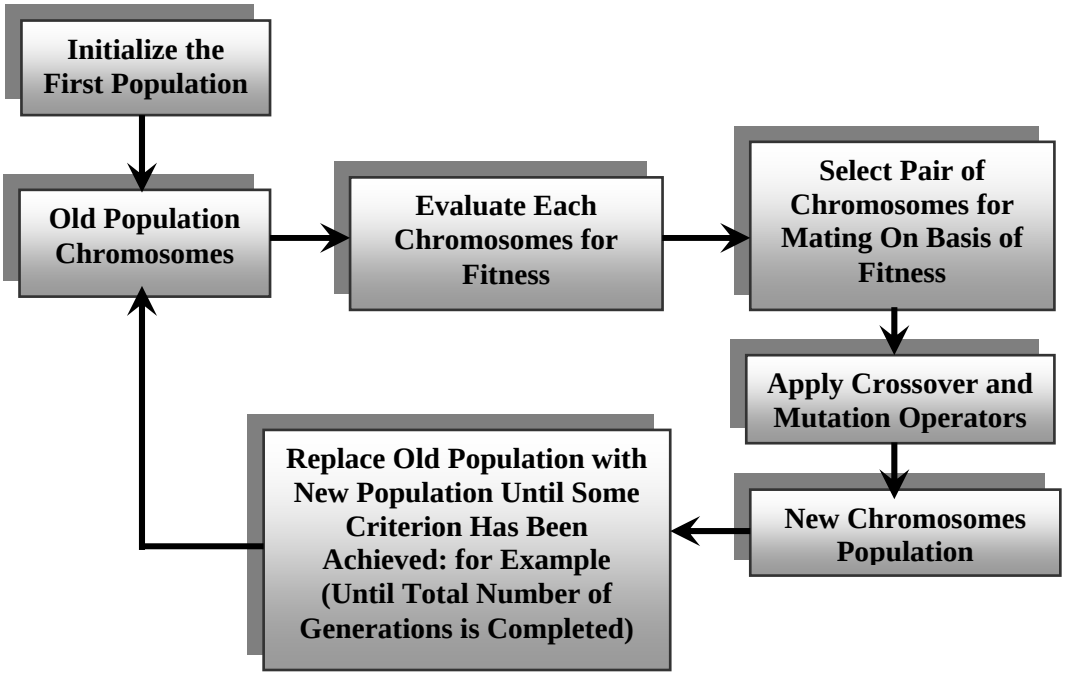


Figure 1.1 − Schematic Diagram of Genetic Algorithm

The simple procedure described as above, is the basis for most application of GAs. There are number of details to fill in, such as the size of the population, the probabilities of crossover and probability mutation. The success of the algorithm often depends greatly on these details. However, the parameters of the simple GA are not always the most effective or appropriate ones (Mitchell, 1998; Goldberg, 1989).

1.3.3 Parameters of GA

When choosing the values for the various parameters, such as population size, crossover rate, and mutation, the following details may be important:

a - Population Size (Npop)

The population size affects both the ultimate performance and the effectiveness of GAs. GAs generally does poorly with very small population because the population provides an insufficient sample size for most hyper planes. On the other hand, a large population requires more evolutions per generation, possibly resulting in an unacceptable slow rate of convergence. Typically the best population size is selected from 50-100 individuals (Mitchell, 1998).

b - Crossover Rate (Pc)

The crossover rate controls the frequency with which the crossover operator is applied. If this rate is too high, it results in the wastage of a lot of computation time in exploring promising regions of the solution space. If the crossover ratio is too low, the searching stagnates due to the lower exploration rate. Typically the best crossover probability is selected such as ($0.5 \leq Pc \leq 0.8$) (Mitchell, 1998).

c - The Mutation Rate (Pm)

The mutation rate controls the rate at which new genes are introduced into the population for trial. Usually, the mutation probability is chosen to be quite small, since this will help guarantee that all the individuals in the mating pool are not mutated so that any search progress that was made is lost (i.e., we keep it relatively low to avoid degradation to exhaustive search via a random walk in the search space). Typically the best mutation probability is selected from 0.005-0.01 (Mitchell, 1998).

1.4 Problem Statement

Most systems of interest to human are complex systems but the nonlinear dynamic system is one of the most interesting types of complex system. Increased in applications of uncertainty complex dynamical systems and also the limitations of traditional methods have forced system designers to turn away from the conventional control methods. The large dimensionality and interactions between variables of dynamical systems are the major obstacles for successful attempts to extend the classical techniques to design controllers for multivariable plants.

In other words, it is difficult to control different complex dynamic models which have a huge number of parameters under unexpected environment change with a very good accuracy. Thus, this led to a more general concept of control. The controller ability is necessary to control uncertainty system accurately and with acceptable performance characteristics over a very wide range. For this purpose, the Neural Network technique (NNs) co-operating with Genetic algorithm technique (GAs) are used in this work to control highly nonlinear and noisy systems.

1.5 Research Objectives

The main aim of the project is to investigate the use of Feedforward Neural Network (FNN) as an adaptive controller for different linear and nonlinear dynamical plants including Single Input Single Output (SISO) and Multi Input Multi Output (MIMO) models utilizing the genetic algorithm techniques. The specific research objectives are as follows:

(a) To design a structure of neural genetic controller employing the parallel model reference then, to compare the neural genetic controller with a conventional controller. The comparison of simulation results is based on the speed of convergences, the tracking ability of the desired model reference and the robustness of these controllers to output disturbances.

(b) To design neural genetic controller for an overhead crane and gas-turbine as nonlinear dynamic applications.

(c) To optimize the genetic algorithm with real-coding and hybrid selection method, and use it as learning procedure to find the parameters and the required architecture of the NN controllers when they are used as ordinary feedback controllers.

1.6 Research Scope and Approach

This research focuses on complex dynamic systems which have huge number of parameters. As explained in the problem statement, the large dimensionality of many processes and the significant interaction between variables of dynamical systems are the major obstacles for the successful attempts of extending the classical techniques for the design of controllers to multivariable plants. The use of NNs in control systems can be seen as a natural step in the evolution of control methodology to meet the needs of

rapidly advancing technology and a competitive market. Basically, there are three major needs:

1- The need to deal with increasingly complex systems.
2- The need to accomplish increasingly demanding design requirements.
3- The need to attain these requirements with less precise advanced knowledge of plants and their environment.

As well as, many techniques have been used for structure and optimizing the design of control system. Genetic algorithm technique is attractive for a number of reasons.

1- It can handle problem constraints by simply embedding them into the chromosome coding.
2- GA can solve multi-objective problem.
3- Since it is a technique independent of the error surface, it is ready to solve multi-modal, non-differentiable, and non-continuous problems.
4- It is very easy to understand the technique with very few (or even none) mathematics.

The genetic algorithm technique and simulation of neural networks system is developed in MATLAB 7.5.a. As for any search and learning method, the way in which candidate solutions are encoded, is a factor in the success of GA. In this research, the direct parallel MRAC incorporating neural controller and FNNs will be used as feedback controller to make the plant track the model reference trajectory. The evolutionary approach such as GAs can serve as a very powerful tool in helping to find the network (controller) parameters and the architecture for a specific problem. The powerful abilities of the proposed neural genetic controller mode are tested through a discrete model representation for speed of convergence, the tracking ability and

robustness by simulation. Then, comparative study will be done between this proposed controller and the classical PID by applying both of them to control different plants.

1.7 Thesis Outline

The thesis is organized as follows:

Chapter one introduces a general background about NNs and GAs. Their terminology and some of the GAs applications are surveyed with the most attention given to those in control systems engineering.

Chapter two introduces the principle concept of adaptive control, concentrating on model reference and self-tuning control scheme.

Chapter three presents the genetic parameters optimization of fixed neural controller structure. Also, a comparison with PID controller is introduced based on the model reference configuration for controlling different linear and non-linear plants.

Chapter four discusses the use of GAs to find the optimal number of hidden nodes for the proposed neural controller in chapter three, to control different plants. In other words, this chapter presents the utilization of genetic optimization of variable neural controller structure (i.e. with variable chromosome length). This chapter provides with a discussion and survey on the previous work on the problem concerned.

Chapter five summarizes the conclusions and suggestions for future work.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

Recently, many attempts in field of intelligent control using different approaches have been proposed to control complex dynamic models with unexpected environment changes. Dongming Xu *et. al,*(2005) presented a direct adaptive approach to design controllers for nonlinear dynamical systems, where system identification of the unknown dynamical system is not required. That approach used both Echo State Network (ESN) and Genetic Algorithm (GA). ESN is a simple modeling of the controller which only a linear readout needs to be trained. The GA is used to optimize ESN linear readout directly so that system identification is not required. Other than that, Dracopoulos and Jones, (1997) also applied an adaptive control architecture using neural networks and genetic algorithms to a complex, highly nonlinear and chaotic dynamic system which uses small control adjustments to stabilize a chaotic system (for a satellite) . In an otherwise unstable but natural periodic orbit of the system, the neural genetic controller may uses large control adjustments and proves capable of actively attaining any specified system state, with no prior knowledge of the dynamics even in the presence of significant noise.

Meanwhile, in this research the model reference adaptive control architecture was used as an alternative for using classical optimization objectives that are represented in terms of performance indices. Therefore, the problem is to tune a fixed controller for linear and nonlinear time invariant plants so that the response of control system is almost similar with the response of the model reference adaptive control. In other words, all required performance specifications of control system are given in terms of

the response of the model. The use of model reference has the following advantages (Ioannou and Fidan, 2008):

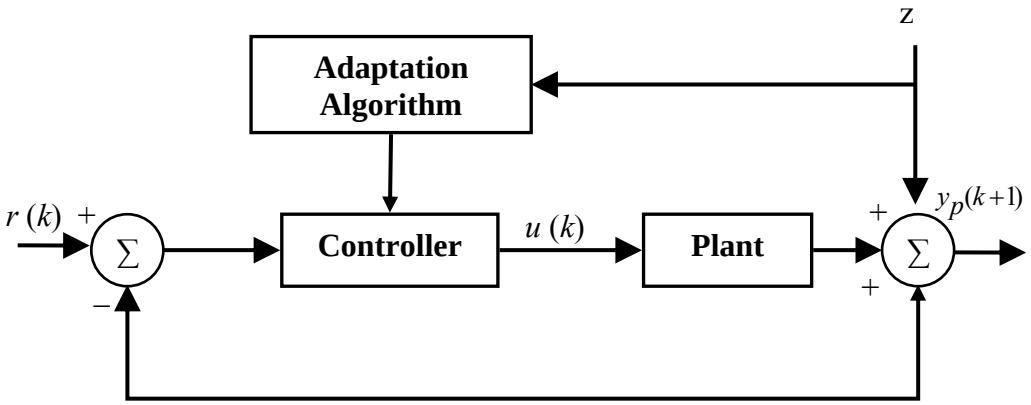
i. The capability of achieving a high performance taking into consideration all terms that describe transient response behavior.
ii. Avoiding the use of multi-objective optimization techniques, which will become difficult when they are used for large dimension of objectives.
iii. All optimization requirements are simply specified by one statement (the model-reference).

2.2 Adaptive Control

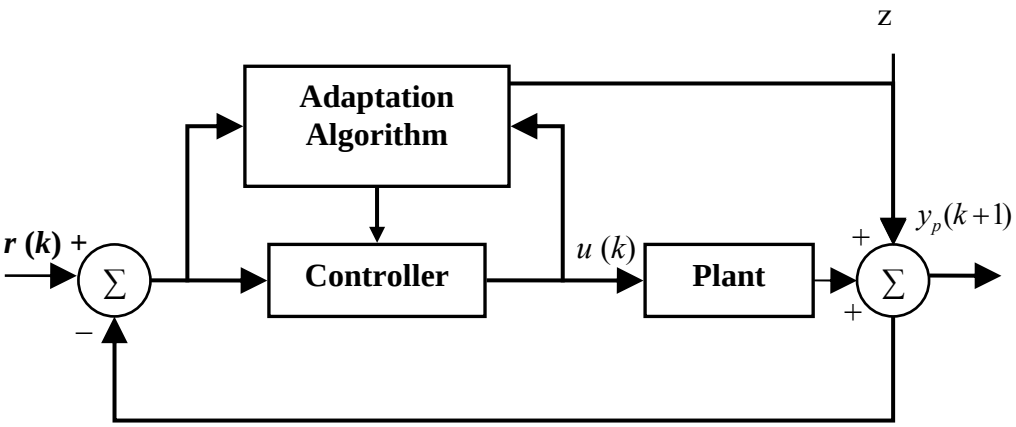
The requirement for more complex and higher performance control system has been the stimulus for the development of a systematic control theory. Even with the development of such a systematic control theory, there is still something else missing. The adaptive control concept seems old and has gained a significant interest since the early 1950‘s (Ioannou and Fidan, 2008; Lecchinia *et. al.*, 2006). Unlike fixed control systems, the adaptive control systems adapt or adjust their behavior to the changing properties of the controlled plants and their environment.

The two basic schemes of controller‘s adaptation can be distinguished, as shown in Figure 2.1. The behavior changes are indicated by measurable signals (where z is external disturbance), and if it is known in advance, then the controller has to be adapted in dependencies of these signals. Feed forward adaptation (open loop adaptation) can be applied as shown in Figure 2.1(a). Here no feedback for the adaptation exists from internal closed loop signals to the controller. The other scheme can be called the feedback adaptation (closed loop adaptation) as shown in Figure

2.1(b). In the first step of closed loop adaptation, information on the plant behavior (structure and parameters) is gained by measuring plant input $u(k)$ and output signals $y_p(k+1)$. This can be performed by plant identification. Based on this information, the controller can be calculated and adapted (Ioannou and Fidan, 2008; Tin and Poon, 2005; Lecchinia *et. al.*, 2006).



(a)



(b)

Figure 2.1 − Basic Adaptation Schemes (a) Feed forward Adaptation. (b) Feed back Adaptation.

Two types of adaptive control are considered in the literature. They are: Self-Tuning Controller (STC) and Model Reference Adaptive Controller (MRAC).

2.2.1 Self –Tuning Controller

Self-tuning control is an approach to the automatic tuning problem. It can be viewed either as a tuning aid for control laws which has fixed parameters (like the widely used three-term controller in industry), or as means of which a time varying plant can be controlled in a consistent way. The self-tuner has three main elements as shown in Figure 2.2. A recursive parameter estimator monitors the plant‘s input and output and computes an estimate of the plant dynamic in terms of a set of parameters in a prescribed structural model. The parameter estimates are then fed into a control algorithm, which then provides a new set of coefficient for feedback law. The control algorithm simply accepts the current estimates and ignores their uncertainties; such a procedure is called certainty-equivalent. The rationale for this approach is that, although there may be a poor control during the tuning phase, taking suitable precautions can minimize this, but the overall algorithm is considerably simplified (Krstic *et al*, 1997).

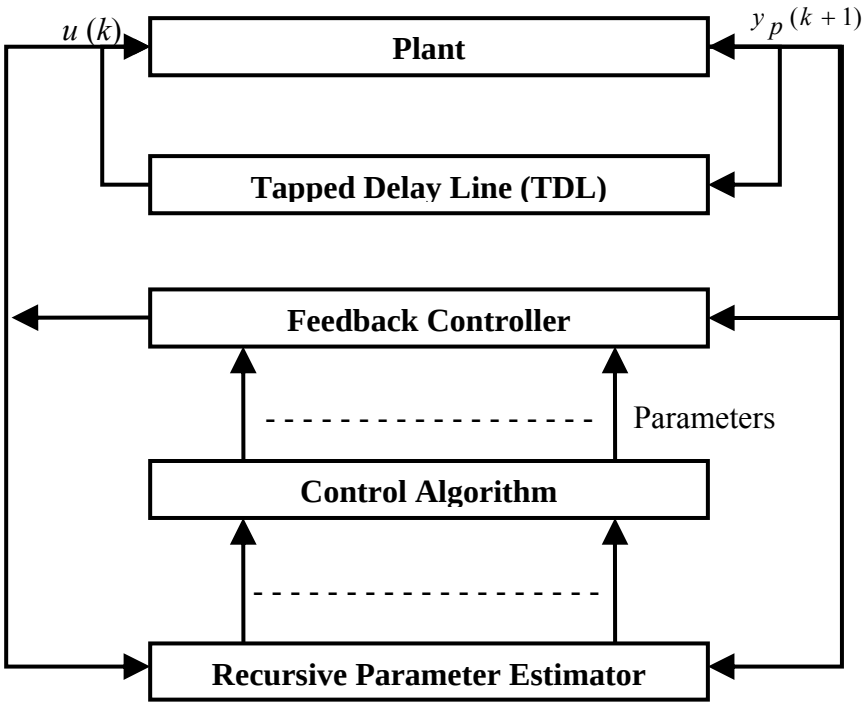


Figure 2.2 − Structure of An Explicit Self-Tuner

2.2.2 Model Reference Adaptive Control Schemes

The basic principle of a model reference control scheme is shown in Figure 2.3.

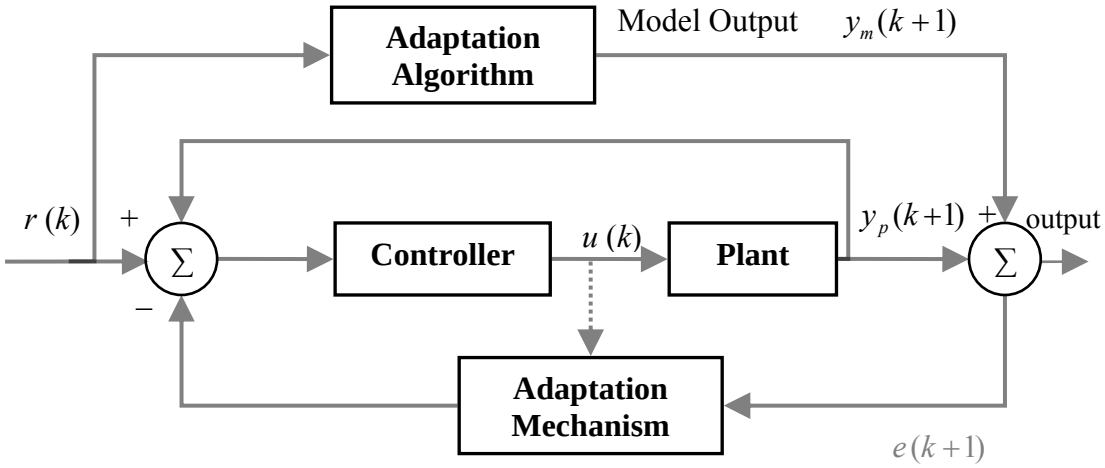


Figure 2.3 − Model Reference Adaptive Control

As mentioned before, the desired performance is expressed in terms of a model reference, which gives the desired specification (response) to a reference input. The system also has an ordinary feedback loop which is composed of plant and the controller. The error is the difference between the output of plant and the model reference. The controller has parameters that changes based on this error (called modeling error to distinguish it from the feedback error). There are thus two feedback loops in Figure 2.3. An inner loop, which provides the ordinary control feedback, and an outer loop adjusts the parameters in the inner loop. The inner loop is assumed to be faster than the outer loop (Alexander *et al*, 2001; Patiño and Liu, 2000). The idea of MRAC is compelling to the closed loop system behavior following a desired model reference for all operating conditions (i.e. the error will be minimum). So the control law is derived from the plant's parameters and the model's parameters. If the plant's parameters are not known, then the estimated parameters will be used instead. The

MRAC can be classified according to the position of model reference with respect to the controlled plant (Alexander *et al.*, 2001).

a. Parallel MRAC

The most popular form of MRAC scheme is shown in Figure 2.4.

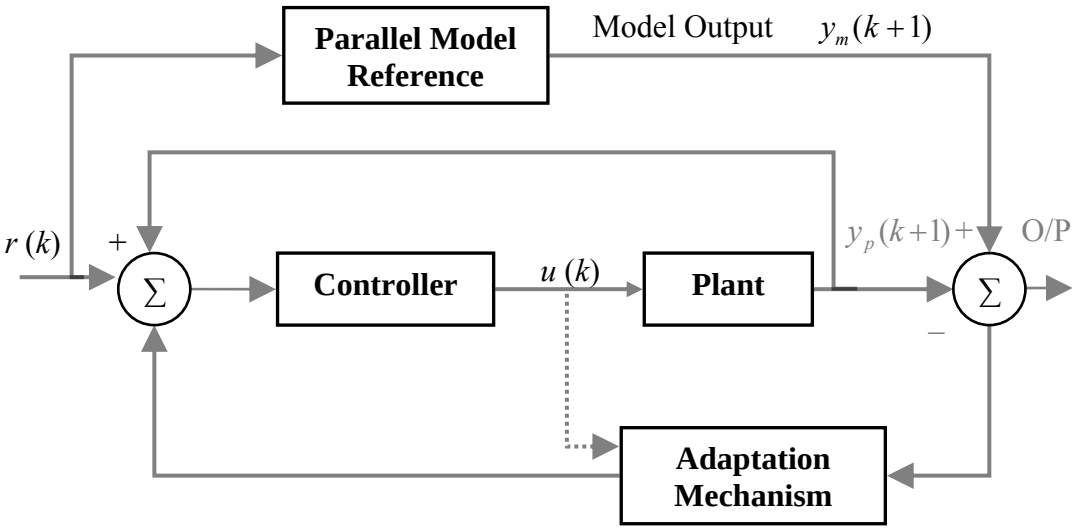


Figure 2.4 − Parallel MRAC

Figure 2.4 shows that a model reference is placed in parallel with the plant. The plant input or the reference input may be used by the adaptation mechanism. A characteristic of most (but not all) MRAC scheme is the direct adaptation, without an explicit parameter estimation part.

b. Series MRAC

In this scheme, the model reference is placed in cascade with the plant and can be regarded as a trajectory generator. The series type is illustrated in Figure 2.5.

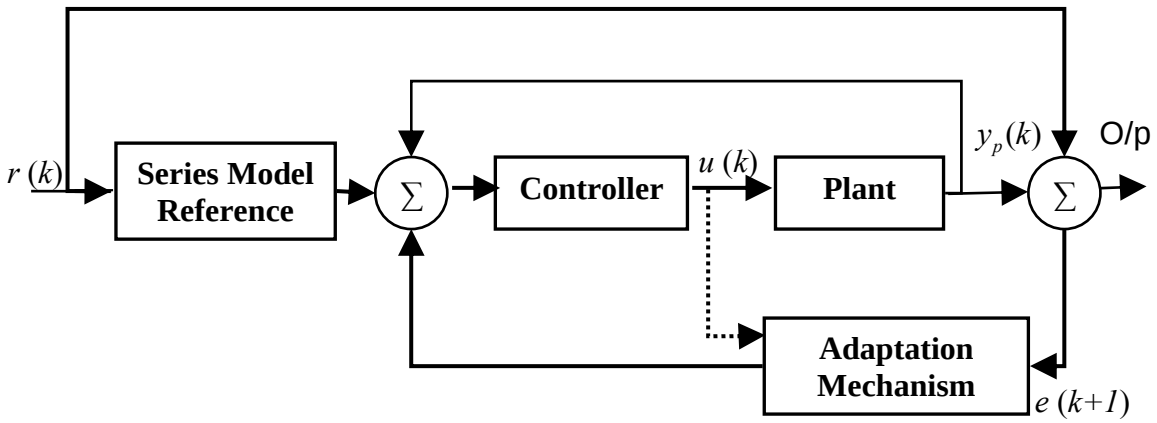


Figure 2.5 − Series MRAC

c. Series-Parallel MRAC

This is a combination of a parallel and series configuration.

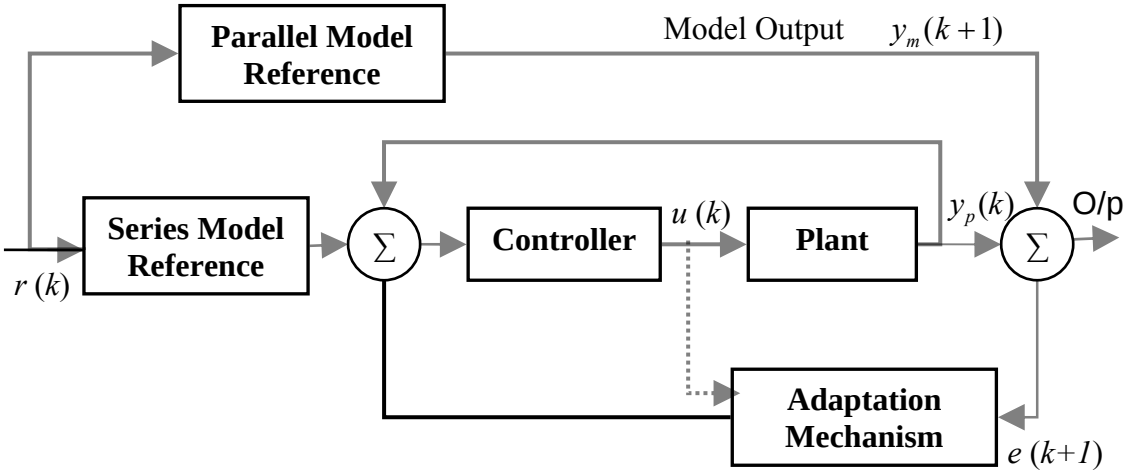


Figure 2.6 − Series Parallel MRAC

Moreover, MRAC systems can be classified into direct and indirect control (Alexander *et al.*, 2001).

- Direct Control - Comparison is between model reference and a control loop with an adaptive controller as shown in Figure 2.7.

- Indirect Control - Comparison is between the system and its model directly, (explicit identification and implicit control) as shown in Figure 2.8.

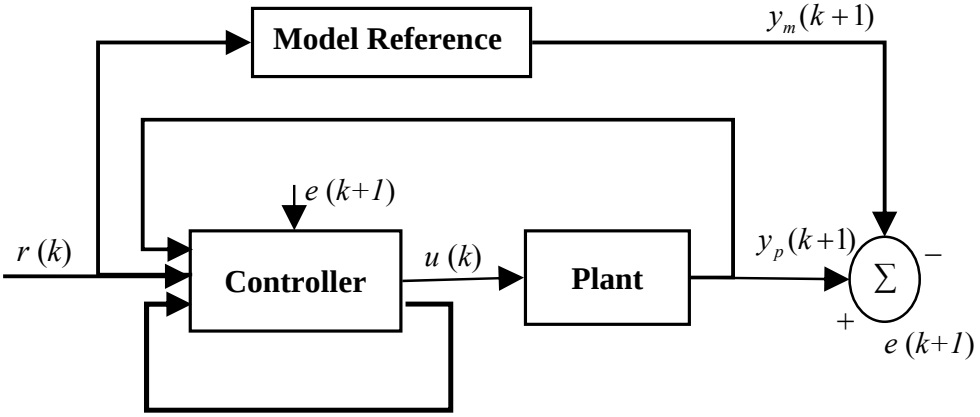


Figure 2.7 − Direct MRAC

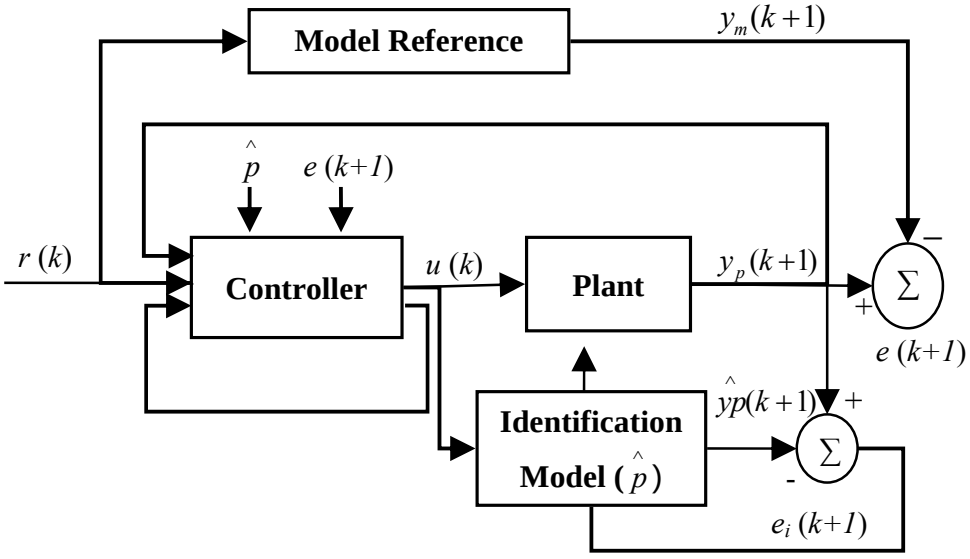


Figure 2.8 − Indirect MRAC

2.3 Basic Functions to Implement Adaptive Control

Adaptive control is usually based on a simulation model identification and control. To implement adaptive control techniques on real system, the following procedures must be considered (Alexander *et al.*, 2001; Filip *et al*., 2009).

2.3.1 Identification

Identification is a fundamental problem of the system theory. The control problem so closely related that the control system design largely depends on the knowledge of the plant. Even though there are several control methods dealing with the uncertainties of the plant, the more knowledge of the plant, the more accurate and easier to design a control system. Therefore it is important to get as much as possible information on the plant (Jagannathan, 2001). The activity of identification that can be carried out includes:

- On–line identification in which the identification with computers is conducted on–line with the plant (i.e. the additional of a new data point or data set is employed to update the model parameters). This is also called real–time processing or sequential identification.
- Off–line identification which means the identification when the measured signals are first stored in a block or array (i.e. all data are analyzed once). This is called batch processing or non-sequential identification (Jagannathan, 2001; Sun and Zheng, 1998).

There are two types of identification which can be classified as parametric and nonparametric. When the model is described by numbers, which represent the parameters of the plant, the determination of these numbers is called parametric identification. The input-output description of the plant and constructing a model, which behaves similarly, is called nonparametric identification (Jagannathan, 2001). Algorithms that are suited to a real-time usage and are based on the successive updating of the model parameters are called recursive.

Least squares and related methods are the most widely used for plant identification. These methods are attractive because of their simplicity but only if the model identified is being linear in the parameters. Other methods such as maximum likelihood estimate and stochastic least squares are also used. The most popular parameter estimation algorithm is the recursive least square method (RLS), which is simple to implement and it requires only small computational inputs.

2.3.2 Modification

The control signal modification depends on the decision. It initiates the required modification so as to drive the system towards an optimum performance. The decision function acts upon the information about the present system performance and the desired or optimum performance. This function is to decide what corrections (if any) to take and to determine signal (Alexander *et al.*, 2001; Kalkkuhl *et al*, 1997).

2.4 The Selection of Model Reference

The concept of model reference was originally born as a particular class of control systems. The desired performance of control system is expressed in terms of a model reference which gives the desired response to a command signal. The adaptive control techniques are essentially evolved to implement high performance control systems when the plant's dynamic characteristics are poorly known or when unpredictable variations occurred (Ioannou and Fidan, 2008). In these techniques, the parameters of controllers are adjusted on-line based on the error between the outputs of model reference and control system through an adaptation mechanism.