ADAPTIVE BACKSTEPPING AND PID OPTIMIZED BY GENETIC ALGORITHM IN CONTROL OF CHAOTIC

LAAREM GUESSAS\textsuperscript{1,2} AND KHIER BENMAHAMMED\textsuperscript{3}

\textsuperscript{1}Department of Electronic
Faculty of Technology
\textsuperscript{2}Intelligent Systems Laboratory
Faculty of Sciences of Engineer
Ferhat Abbes University
Setif, Algeria
guessaslar@yahoo.fr

\textsuperscript{3}Electrical Department
Oumm El-Kora University
Mecca, Saudi Arabia
khierben@ieee.org

Received January 2010; revised August 2010

Abstract. In this paper, the robust adaptive control scheme based on adaptive backstepping design is used to control the autonomous second order strict feedback form such as the Lorenz Chaotic system. The design procedure is recursive; at the $i$\textsuperscript{th} step, the $i$\textsuperscript{th} subsystem is stabilized with respect to a Lyapunov function $V_i$ by the design of a stabilizing function $\alpha_i$, tuning function $\tau_i$, and the update law $\hat{\theta}_i$ for the unknowns' adaptive parameters estimates $\hat{\theta}_i$, the feedback control $u$ is designed at the final step. This procedure possesses strong properties of global stability and tracking which are built into the nonlinear system in a number of steps, which is never higher than the system order, without any growth restrictions on nonlinearities. The results show the guarantee of all the parameter estimates to converge to their true values. Some simulation results for the chaotic system cited above are shown to illustrate the parameter convergence with adaptive backstepping design.

Keywords: Adaptive backstepping design, Adaptive parameters estimates, Lyapunov, Stabilizing and tuning functions, Update and control laws

1. Introduction. In the first years after the penetration of the concept of deterministic chaos into the scientific literature \cite{1,2}, chaotic behavior was regarded as an exotic phenomenon which might be of interest only as a mathematical speculation and would never be encountered in practice, yet further development highlighted a number of applications where chaotic modes may appear sometimes as harmful, sometimes as useful. Moreover, entire classes of problems that are of practical importance arose where one has to control a nonlinear system by reducing or, on the contrary, increasing the degree of its chaoticity \cite{3}.

Yet in the last few years, adaptive control of nonlinear systems has emerged as an exciting research area. Early efforts focused on the state-feedback problem and resulted in a systematic design procedure called adaptive backstepping. Backstepping is a systematic method for nonlinear control design, which can be applied to a broad class of nonlinear systems in the so-called non or autonomous “strict-feedback” form; the name backstepping refers to the recursive nature of the design procedure. First, only a small subsystem is considered, for which a virtual control law is constructed. Then, the design
is extended in several steps until a control law for the full system has been constructed. Along with the control law, a Lyapunov function for the controlled system is successively built [4-6]. An important feature of backstepping is that nonlinearities can be dealt with in several ways: useful nonlinearities, which act stabilizing, can be retained, and sector bounded nonlinearities may be dealt with using linear control; retaining nonlinearities instead of canceling them requires less precise models and may also require less control effort. Further, the resulting control laws can sometimes be shown to be optimal with respect to a meaningful performance index, which guarantees certain robustness properties. In particular, many adaptive backstepping control schemes have been successfully applied to the control and the synchronization of chaotic systems, in [7-12], had shown in their papers that many chaotic systems as paradigms in the research of chaos can be transformed into a class of nonlinear systems in the so-called non or autonomous “strict-feedback” form, and that they extended the adaptive backstepping design method to the non or autonomous “strict-feedback” system, which may be naturally applied to control this class of chaotic systems. Some of them used modified backstepping design schemes by using added gains to relax the design, and many fuzzy adaptive control schemes have been reported to combine the backstepping technique with adaptive fuzzy logic systems for unknown nonlinear systems. Fuzzy adaptive backstepping control schemes can provide a systematic framework for the design of tracking or regulation strategies, in which the fuzzy logic systems are used to approximate the unknown nonlinear functions, and an adaptive fuzzy controller is constructed recursively. These approaches are suitable for some uncertain nonlinear systems which do not satisfy the requirement of matching conditions, or require the linearity in the parameter assumption of nonlinear systems.

To highlight the robustness properties of adaptive Backstepping design, a PID controller in which parameters are optimized by genetic algorithm is used to control the same system. The remainder of the paper is organized as follows: in Section 2, we present the formulation design procedure, and Section 3 is concerned to the simulation main results for the chaotic system cited above. Section 4 is concerned to the control by PID controller based on Genetic algorithm. Finally, the note is concluded in Section 5.

2. Adaptive Backstepping Design. Control engineers are familial with linear or non-linear feedback control, adaptive control, optimal control, sliding mode control, self-tuning control, robust control, digital control, stochastic control, intelligent control distributed control and many others system regulation and stabilization methodologies. However, in the past decade, backstepping has become one of the most popular design methods for adaptive nonlinear control because it allows one to design adaptive controllers for a class of nonlinear systems in the so-called non or autonomous “strict-feedback” form, the synthesis of the stabilizing, the Lyapunov functions and the control, the update laws are carried out at the same time in the design procedure, it can guarantee global stabilities and transient performance.

It has been shown that many well-known chaotic systems as paradigms in the research of chaos can be transformed into a class of nonlinear systems in the strict-feedback form, and the adaptive backstepping has been employed and extended to the control of these chaotic systems. Let’s consider the general systems transformable into the parametric-strict-feedback form:

\[
\begin{align*}
\dot{x}_i(t) &= g_i(\bar{x}_i, t) x_{i+1} + f_i(\bar{x}_i, t) + \theta_i^T F_i(\bar{x}_i, t), \quad (i = 1, \ldots, n - 1) \\
\dot{x}_n(t) &= g_n(\bar{x}_n, t) u + f_n(\bar{x}_n, t) + \theta_n^T F_n(\bar{x}_i, t) \\
y &= x_1
\end{align*}
\]
where \( x_i = [x_1, x_2, \ldots, x_i]^T \in \mathbb{R}^i, i = 1, \ldots, n, u \in \mathbb{R} \) and \( y \in \mathbb{R} \) are the states, input, and output respectively. \( g_i(\cdot) \neq 0, f_i(\cdot), i = 1, \ldots, n \) are known, smooth nonlinear functions. \( g_n(\cdot) \neq 0, f_n(\cdot) \) are known continuous nonlinear functions. \( \theta \) is the vector of unknown constant parameters.

Our control objective is to regulate \( x_1 \) to \( x_{\text{equilibrium}}_1 = x_{eq} = 0 \) and to stabilize the corresponding equilibrium \( x_{eq} \). The design procedure is recursive, at the \( i^{th} \) step the \( i^{th} \) subsystem is stabilized with respect to a Lyapunov function \( V_i \) by the design of a stabilizing function \( \alpha_i \) and a tuning function \( \tau_i \), the update law \( \dot{\theta}_i \), for the unknowns adaptive parameters estimates \( \hat{\theta}_i \), and the feedback control \( u \) is designed at the final step.

The main idea is:

1) To put: \( z_1 = x_1 \): For the control,
\[ z_1 = x_1 - y_r \]: For the track, \( y_r \) arbitrary reference signal.

2) Then, introducing: \( z_i = x_i - \alpha_{i-1} \) or \( z_i = x_i - y_{r_i} - \alpha_{i-1} \) for each subsystem, where \( z_1, z_i \) are virtual variables and \( \alpha_{i-1} \) is used as a control to stabilize the \((i-1)\), equation of the system, and to be defined later, \( i = 2, \ldots, n-1 \).

3) We rewrite
\[ \dot{x}_i(t) = g_i(\bar{x}_i, t) x_{i+1} + f_i(\bar{x}_i, t) + \theta_i^T F_i(\bar{x}_i, t) \tag{2} \]
as
\[ \dot{z}_i = g_i(\bar{x}_i, t) z_{i+1} + g_i(\bar{x}_i, t) \alpha_{i-1} + f_i(\bar{x}_i, t) + \theta_i^T F_i(\bar{x}_i, t) \tag{3} \]

4) With respect to Lyapunov function
\[ V_i = V_{i-1} + \frac{1}{2} z_i^2 + \frac{1}{2} \theta^T \Gamma^{-1} \theta \tag{4} \]
where \( \theta^T = \hat{\theta}^T - \theta^T \), \( \theta^T \) is the adaptive current parameter. \( \hat{\theta}^T \) is the adaptive estimate parameter. \( \hat{\theta}^T \) is the adaptive error parameter.

The derivative of \( V_i \) in time is
\[ \dot{V}_i = \dot{V}_{i-1} + z_i \dot{z}_i + \hat{\theta}^T \Gamma^{-1} \hat{\theta} \tag{5} \]
First, we extract the term \( z_i z_{i+1} \) from the bracketed term multiplying \( z_i \) which will be cancelled at the next step.

5) To make \( \dot{V}_i \) negative definite, we would design \( \alpha_{i-1} \) such that the bracketed term multiplying \( z_i \) equals \(-c_i z_i \).

If \( x_{i+1} \) were our actual control, we would let \( z_{i+1} \equiv 0 \), and eliminate \( \hat{\theta} \) from \( \dot{V}_i \) with the update law \( \dot{\hat{\theta}} = \tau_i \), where \( \tau_i \) is the tuning function in the form \( \tau_i = \Gamma z_i F_i(x_i, t) \), where \( \Gamma^T > 0 \) is the adaptive gain matrix.

This term is tolerated until the \((n-1)\) step, and the decision about \( \dot{\hat{\theta}} \) is also postponed.

6) The closed-loop form in the \((z_{i-1}, z_i)\) coordinates is written as
\[ \dot{z}_i = g_i(\bar{x}_i, t) z_{i+1} + g_i(\bar{x}_i, t) \alpha_i + f_i(\bar{x}_i, t) + \theta_i^T F_i(\bar{x}_i, t) - \frac{\partial \alpha_{i-1}}{x_{i-1}} \dot{x}_{i-1} - \frac{\partial \alpha_{i-1}}{\theta} \hat{\theta} \tag{6} \]

7) At the last step with \( z_n = x_n - \alpha_{n-1} \), we rewrite
\[ \dot{x}_n(t) = g_n(\bar{x}_n, t) u + f_n(\bar{x}_n, t) + \theta_n^T F_n(\bar{x}_n, t) \tag{7} \]
as
\[ \dot{z}_i = g_n(\bar{x}_n, t) u + f_n(\bar{x}_n, t) + \theta_n^T F_n(\bar{x}_n, t) - \frac{\partial \alpha_{n-1}}{x_{n-1}} \dot{x}_{n-1} - \frac{\partial \alpha_{n-1}}{\theta} \hat{\theta} \tag{8} \]
We now design our actual update law $\dot{\theta} = \tau_n$ and feedback control $u$ to stabilize
the full $z$ system and reached our goal $\dot{V}_n$ negative definite as

$$\dot{V}_n = - \sum_{k=1}^{n} c_k z_k^2$$

(9)

c_k is a positive constant. In the $z$ coordinates, the global stability of the equilibrium
point $z = 0$, $\theta = \dot{\theta}$. From LaSalle’s invariance theorem for the proof see [5,6,14]. All
these steps are used to control the Lorenz Chaotic system.

3. Adaptive Control Laws for Lorenz Chaotic System. Chaos control refers to
manipulating the dynamical behavior of a chaotic system, in which the goal is to suppress
the chaos when it is harmful or to enhance or create chaos when it is beneficial [3].

Several lower-dimensional systems are frequently used as benchmark examples for ver-
ification and validation of a proposed theory, method and algorithm. These examples
include Lorenz system [7,14]. It is interesting to note that this chaotic system mentioned
above can be rewritten into the non-autonomous strict feedback form.

Lorenz system is an approximation of a partial deferential equation for fluid convection,
where a flat fluid layer is heated from below and cooled from above. It has become one of
paradigms in the research of chaos, and is described with three key parameters unknown.
In 1963, Lorenz found the first chaotic attractor in a three-dimensional autonomous system
which can be rewritten in to the autonomous third order strict feedback form

$$\begin{align*}
\dot{x}_1 &= \theta_1^T (x_2 - x_1) \\
\dot{x}_2 &= -x_3 x_1 + \theta_2^T x_1 - x_2 \\
\dot{x}_3 &= x_1 x_2 - \theta_3^T x_3 + u
\end{align*}$$

(10)

Originally, $u = 0$, in the chaotic state with initial parameters $\theta_1 (0) = 10$, $\theta_2 (0) = 28$,
$\theta_3 (0) = 3$.

![Chaotic behavior in space wave](image)

**Figure 1.** Chaotic trajectory in portrait phase of $(x_3, x_2, x_1)$ for Lorenz system

We develop a design procedure via adaptive backstepping which can overcome the
singularity caused by $-x_3 x_1$ in the second equation of (10).

Main Results.
The backstepping design procedure contains three steps, the stabilizing, Lyapunov functions, up date laws and the control action are

\[
\begin{align*}
\alpha_1 &= -c_{11}z_1 + x_1 \\
\alpha_2 &= \frac{z_2 (c_2 - (1 - c_{11} \dot{\theta}_1^T))}{x_1} + \dot{\theta}_2^T + \left( \dot{\theta}_1^T - \left( 1 - c_{11} \left( 1 - c_{11} \dot{\theta}_1^T \right) \right) \right) \\
\dot{\theta}_1 &= \Gamma z_1 (z_2 - z_1 c_{11}) \\
\dot{\theta}_2 &= \Gamma z_2 (x_1 + c_{11} (z_2 - z_1 c_{11})) \\
\dot{\theta}_3 &= \Gamma z_3 x_3 \\
u &= -c_3 z_3 + z_2 x_1 - x_1 x_2 + \dot{\theta}_3^T x_3 - (1 - c_{11}^2) \dot{\theta}_1^T - \dot{\theta}_2^T \\
\dot{z}_1 &= \dot{\theta}_1^T z_2 - c_1 z_1 - \dot{\theta}_1^T (z_2 - z_1 c_{11}) \\
\dot{z}_2 &= -x_1 z_3 - c_2 z_2 - x_1 \dot{\theta}_1^T - \dot{\theta}_1^T c_{11} (z_2 - z_1 c_{11}) - \dot{\theta}_2^T x_1 \\
\dot{z}_3 &= -c_3 z_3 + z_2 x_1 + \dot{\theta}_3^T x_3
\end{align*}
\]

where \( c_1 = \dot{\theta}_1^T c_{11} \) and \( \dot{\theta}_1^T > 0 \).

In this section, the nonlinear Equations (13) and (10) with ((11) and (14)) are integrated numerically by using the fourth order Runge-Kutta integration algorithm in Matlab version 5. The parameter values were fixed at \( c_1 = 10, c_2 = 10 \), the adaptation gain matrix \( \Gamma = [0.1; 0.1; 0.1; 0.1] \). The initial conditions are chosen as \( x_1 (0) = x_2 (0) = x_3 (0) = 0 \).

**Figure 2.** Time waveform of components of controlled autonomous Lorenz system via adaptive backstepping.

Figure 2 shows the convergence of different components to the equilibrium and the efficiency of the laws to drive the circuits to the zero point in a few seconds, to achieving good control results \( c_i \) and \( \Gamma \) are sometimes chosen properly most common way is to choose possibly smaller \( m \) here the optimization algorithm is recurred.
4. **PID Controller Based on Genetic Algorithms.** Control of nonlinear systems is difficult in the absence of a systematic procedure as available for linear systems. Many techniques are limited in their application to special class of systems. Here again, more commonly available methods are heuristic in nature and the genetic algorithms technique can reduce the arbitrariness in the design of a controller to a great extent. However, even if a model of the nonlinear system is available, no systematic and generally applicable control theory is available for the design of controllers for nonlinear systems. The best-known controllers used in industrial control processes are proportional-integral derivative (PID) controllers because of their simple structure and robust performance in a wide range of operating conditions.

4.1. **Introduction.** In the past decades, control theory has gone through major developments. Advanced and intelligent control algorithms have been developed. However, the PID-type controller remains the most popular in industry, studies even indicate that
approximately 90% of all industrial controllers is of the PID-type. Reasons for this are the simplicity of this control law and the few tuning parameters. Hundreds of tools, methods and theories are available for this purpose. However, finding appropriate parameters for the PID controller is still a difficult task, so in practice control engineers still often use trial and error for the tuning process. PID control consists of three types of control, Proportional, Integral and Derivative control. PID controller algorithm can be implemented in many forms, however, they are mostly used in feedback loops.

It is interesting to note that more than half of the industrial controllers in use today utilize PID or modified PID control schemes. Below is a simple diagram illustrating the schematic of the PID controller. Such set up is known as no interacting form or parallel form [17].

![Figure 5. Schematic of the PID controller](image)

\[
u(t) = K_p e(t) + K_I \int_0^t e(\tau) d\tau - K_D \frac{de(t)}{dt}
\]

where \( u(t) \) is the output PID controller, \( e(t) \) is the error signal, \( K_p, K_I \) and \( K_D \) are respectively the proportional, integral and, the derivative gains.

The proportional controller output uses a ‘proportion’ of the system error to control the system. However, this introduces an offset (static) error into the system. The integral controller output is proportional to the amount of time there is an error present in the system. The integral action removes the offset introduced by the proportional control but introduces a phase lag into the system. The derivative controller output is proportional to the rate of change of the error. Derivative control is used to reduce or eliminate overshoot and introduces a phase lead action that removes the phase lag introduced by the integral action.

This work will look at how effective PID controller’s parameters are optimized using genetic algorithms. Genetic Algorithms (GA’s) [15-17] are a stochastic global search method that mimics the process of natural evolution. The genetic algorithm starts with no knowledge of the correct solution and depends entirely on responses from its environment and evolution operators (i.e., reproduction, crossover and mutation) to arrive at the best solution. By starting at several independent points and searching in parallel, the algorithm avoids local minima and converging to sub optimal solutions. Presently GA has been receiving a lot of attention and more research has been done to study its applications. Application in the area of Control Engineering has also developed tremendously. Even though in control system design, issues such as performance, system stability, static and dynamic index and system robustness have to be taken into account. However, each of these issues strongly depends on the controller structure and parameters. This dependence usually cannot be expressed in a mathematical formula, however, often a trade-off has to be made among conflicting performance issues. This way, GAs have been shown
to be capable of locating high performance areas in complex domains without experiencing the difficulties associated with high dimensionality, as may occur with gradient decent techniques or methods that rely on derivative information using genetic algorithms to perform the controller, PID control consists of three types of control, Proportional, Integral and Derivative control.

A genetic algorithm is typically initialized with a random population consisting of between 20-100 individuals. This population is usually represented by a real-valued number or a binary string called a chromosome. How well an individual performs a task is measured by the objective function. The objective function assigns each individual a corresponding number called its fitness. The fitness of each chromosome is assessed and a survival of the fittest strategy is applied. The magnitude of the error is usually used to assess the fitness of each chromosome.

Writing an objective function is the most difficult part of creating a genetic algorithm. An objective function could be created to find a PID controller that gives the smallest overshoot, fastest rise time or quickest settling time, however, in order to combine all of these objectives it was decided to design an objective function that will minimize the error of the controlled system. Each chromosome in the population is passed into the objective function one at a time. The chromosome is then evaluated and assigned a number to represent its fitness, the bigger its number the better its fitness. The genetic algorithm uses the chromosome’s fitness value to create a new population consisting of the fittest members.

4.2. Genetic algorithm stages. There are three main stages of a genetic algorithm, these are known as reproduction, crossover and mutation.

4.2.1. Reproduction. Just like in natural evolution, during the reproduction phase the fitness value of each chromosome is assessed. This value is used in the selection process to provide bias towards fitter individuals; a fit chromosome has a higher probability of being selected for reproduction.

An example of a common selection technique is the ‘Roulette Wheel’ selection method. Each individual in the population is allocated a section of a roulette wheel; the size of the section is proportional to the fitness of the individual. A pointer is spun and the individual to whom it points is selected. This continues until the selection criterion has been met. The probability of an individual being selected is thus related to its fitness, ensuring that fitter individuals are more likely to leave offspring. Multiple copies of the same string may be selected for reproduction and the fitter strings should begin to dominate.

4.2.2. Crossover. Once the selection process is complete, the crossover algorithm is initiated. The crossover operation swaps certain parts of the two selected strings in a bid to capture the good parts of old chromosomes and create better new ones. Genetic operators manipulate the characters of a chromosome directly, using the assumption that certain individual’s gene codes, on average, produce fitter individuals. The crossover probability indicates how often crossover is performed. A probability of 0% means that the ‘offspring’ will be exact replicas of their ‘parents’ and a probability of 100% means that each generation will be composed of entirely new offspring. The simplest crossover technique is the Single Point Crossover.

Using selection and crossover on their own will generate a large amount of different strings. However, there are two main problems with this depending on the initial population chosen, there may not be enough diversity in the initial strings to ensure the GA searches the entire problem space. The GA may converge on sub-optimum strings due to a bad choice of initial population.
These problems may be overcome by the introduction of a mutation operator into the GA.

4.2.3. Mutation. Mutation is the occasional random alteration of a value of a string position. It is considered a background operator in the genetic algorithm.

The probability of mutation is normally low because a high mutation rate would destroy fit strings and degenerate the genetic algorithm into a random search.

Mutations probability values of around 0.1% or 0.01% are common, these values represent the probability that a certain string will be selected for mutation, i.e., for a probability of 0.1%; one string in one thousand will be selected for mutation. Once a string is selected for mutation, a randomly chosen element of the string is changed or ‘mutated’.

![Figure 6. Graphical illustration of the genetic algorithm](image)

4.3. Genetic algorithm process. The steps involved in creating and implementing a genetic algorithm are as follows:

1) Generate an initial, random population of individuals for a fixed size.
2) Evaluate their fitness.
3) Select the fittest members of the population.
4) Reproduce using a probabilistic method (e.g., roulette wheel).
5) Implement crossover operation on the reproduced chromosomes (choosing probabilistically both the crossover site and the ‘mates’).
6) Execute mutation operation with low probability.
7) Repeat Step 2) until a predefined convergence criterion is met.

The convergence criterion of a genetic algorithm is a user-specified condition, i.e., the maximum number of generations or when the string fitness value exceeds a certain threshold.

Since this work is using genetic algorithms to optimize the gains of a PID controller there are going to be three strings assigned to each member of the population, these members will be comprised of a P, I and a D string that will be evaluated throughout the course of the GA, real (floating point) numbers will be used to encode the population.

The three terms are entered into the genetic algorithm via the declaration of a three-row. Between −100 and 100, the controlled system is given a step input and the error is assessed using an appropriate error performance criterion, i.e., \( \text{error} = -\text{output} \). The
chromosome is assigned an overall fitness value according to the magnitude of the error, the smaller the error the larger the fitness value.

4.4. A genetic algorithm options. Variable bounds matrix = $[-100, 100]$; Each member of the population is initialized with values randomly selected between $-100$ and $100$. Population Size = 80; GA. Generally, the bigger the population size the better is the final approximation. Number of generations = 100; Selection function = stochastic uniform; Crossover fraction = 0.8; Mutation function = Gaussian; Stopping criteria = error performance criterion.

In order to evaluate the PID values chosen by the genetic algorithm, the objective function is written based on error performance criterion \( \text{sum absolute error} \) which gets the absolute value of the error to remove negative error component this kind of error performance criteria is good for simulation studies.

The best population may be plotted to give an insight into how the genetic algorithm converged to its final values.

Table 1 shows the PID gain values the best solution tracked over generations, for the Lorenz chaotic system.

<table>
<thead>
<tr>
<th>Controller</th>
<th>( K_P )</th>
<th>( K_I )</th>
<th>( K_D )</th>
</tr>
</thead>
<tbody>
<tr>
<td>PID</td>
<td>0.9501</td>
<td>0.0579</td>
<td>0.8381</td>
</tr>
</tbody>
</table>

From Figure 6, the response of system looks reasonable stable.

![Figure 7](image)

**Figure 7.** Time waveform of components of controlled autonomous Lorenz system via genetic algorithm

4.5. Genetic algorithms advantages from traditional methods. Genetic algorithms are substantially different to the more traditional search and optimization techniques. The five main differences are:

1) Genetic algorithms search a population of points in parallel, not from a single point.
Figure 8. Time waveform of Proportional PID gain of the controller for the Lorenz chaotic system

Figure 9. Time waveform of integral PID gain of the controller for the Lorenz chaotic system

2) Genetic algorithms do not require derivative information or other auxiliary knowledge; only the objective function and corresponding fitness levels influence the direction of the search.

3) Genetic algorithms use probabilistic transition rules, not deterministic rules.

4) Genetic algorithms work on an encoding of a parameter set not the parameter set itself (except where real-valued individuals are used).

5) Genetic algorithms may provide a number of potential solutions to a given problem and the choice of the final is left up to the user.

However, such drawbacks:

1) Uses more memory spaces and hence takes up more time to reach the best solution.
2) The range of each parameters must be known by others methods, a large range of possible combinations for the P, I and D gains between the bounds of –100 and 100.

3) Local optimality is the most drawback in the GAs.

4) The genetic algorithm proved difficult to test in simulation due to the fact that simulations do not run in real time.

5) It was discovered that the genetic algorithm would sometimes create a controller that would make the overall system unstable, additional functionality was added so that the last stable controller created by the genetic algorithm would be implemented if the current controller were unstable.
5. Conclusion. For over a decade, dynamic chaos theory and control has been deeply studied and applied to many fields extensively, such as optical system, biology, secure communication and plasma, amongst others. The interest in the dynamic chaos theory is hinged on the well demonstrated applications of chaos including the explanation of many physical processes such as transition from laminar to turbulent fluid, multi-photon infrared absorption, microwave ex-citation, ionization of Rydberg atoms and more specifically dust-charge fluctuation in dusty plasma. At the same time in many situations, chaos is an undesirable phenomenon which often leads to violent vibrations, irregular operations in mechanical systems, breaking materials and so on.

Thus, from a practical point of view, it is often desired to convert and control the system dynamics with minimal effort suitably so that whenever chaotic motion is physically harmful it can be changed to a desired study stat. In this paper, the adaptive schemes applied is the backstepping technique, it is used to control the autonomous third order strict feedback form such as the Lorenz Chaotic system, the synthesis of the stabilizing, the Lyapunov functions and the control, the update laws are carried out at the same time in the design procedure, it can guarantee global stabilities and transient performance. The different figures shown the efficiency of the laws to drive the circuits to the zero point and in few seconds with less effort, to achieving good control results $c_i$ and $\Gamma$ are sometimes chosen properly most common way is to choose possibly smaller the optimization algorithm is recurred. Versus to the genetic algorithm uses more memory spaces and hence takes up more time to reach the best solution, optimality is the most drawback in the GA.

The simulation results indicated the feasibility of the design procedure.

REFERENCES