Fuzzy logic controller based on genetic algorithms

Li RenHou*, Zhang Yi
Institute of Systems Engineering, Xi’an Jiaotong University, 28 Xianming Road, Xi’an, Shaanxi, 710049, People’s Republic of China
Received April 1995; revised July 1995

Abstract

Based on genetic algorithms (GAs), a method of designing a fuzzy logical controller for complex processes is proposed. After a brief overview of GA, the selection of GA’s control parameters is discussed for designing MIMO fuzzy control systems. We explained the optimization procedure of the fuzzy rules’ selection on a large fuzzy input search space. A double inverted pendulum device is used to test the effectiveness of the design method based on GAs. Not only simulation results are shown in the paper, but also the practical operation results are given.

Keywords: Genetic algorithm; Fuzzy set; Control theory; Engineering

1. Introduction

Fuzzy logic control has been developing rapidly in recent years, and is being used successfully and widely in an increasing number of application areas, especially in control of complex processes, such as the control of the pH of a laboratory acid–base system [7], the DC motor’s speed control [13], etc. At the same time, many scholars have been attracted by the problems of combining the fuzzy logic control with other new intelligent methods, and there are many successful examples such as: achieving fuzzy controller with neural networks [3, 4, 8], optimizing neural network configurations with fuzzy logic [15], and optimizing fuzzy controller with new optimization methods [13, 7].

Conventional control theory is well suited for applications where the process can be reasonably well characterized in advance and where the number of parameters that must be considered is small. There are many important processes, however, that are not well characterized or are subject to a large number of uncontrolled, changeable or unmeasurable parameters. Fuzzy logical controllers appear to offer new method to produce high-performance control rules without having good models of the processes being controlled. But there exist many problems in the control of complex systems using fuzzy logic controller (FLC). Firstly, in the cases where the numbers of input variable are large, the possible selective space of fuzzy rules will increase sharply. For instance, if each variable has five fuzzy subsets, then for a fuzzy controller with four inputs and one output, the number of rules can be raised...
to 1024. It is very difficult to determine and select which rules in such large search space are the most suitable for controlling the process. Secondly, the membership function plays an important role in determining the control action prescribed and the performance of the system. In multivariable complex processes, the optimization and selection of membership function will also be very difficult.

Essentially, the problem of dynamically controlling a complex system using fuzzy controllers, can be considered as a multi-parameters' optimization problem. In general, the main task for controlling a complex process with FLC is to define a performance response surface which must be explored by direct search techniques to locate high-performance control outputs. Because fuzzy control systems are highly non-linear systems which have high-dimensional, multi-model, discontinuous response surface, the choice of optimization technique may not be obvious and easy. Even when an appropriate classical optimization algorithm is available, there are usually various parameters that must be tuned, e.g., the step size in variable metric technique.

This makes attempts to determine the optimization of fuzzy control rules by using a global optimization procedure called genetic algorithms (GAs). GAs are kinds of search algorithms based on the mechanics of nature genetics which are capable of rapidly locating near-optimal solution to difficult problems [5]. They have been used to alter membership function in response to the changes of task environment to produce more efficient FLC performance [6]. Charles et al. [7] successfully used Genetic algorithm to produce adaptive FLC’s for a laboratory acid-base system. In Ref. [18] Varsek et al. proposed a genetic algorithm-based technique to design and tune the parameters of fuzzy controller, and take the control of a single inverted pendulum as an example to indicate the effectiveness of GAs.

The remainder of this paper is organized as follows: Section 2 contains a brief overview of GAs, Section 3 describes the design of fuzzy controller based on genetic algorithm which is regarded as an optimization strategy to search near-optimal fuzzy control sets. The experimental results are given in Section 4, in which a double inverted pendulum system is investigated, not only for simulation, but also for practical real-time control, to show the merits of new design method. The discussion and conclusion are summarized in the final section of this paper.

2. Brief overview of genetic algorithms

Genetic algorithms are global optimization techniques that avoid many shortcomings exhibited in conventional search techniques on a large and complicated search space. They simulate the procedures of nature evolution which ensure the proliferation of quality solution via a systematic information exchange that depends on probabilistic decision. GAs are iterative procedures in which a constant size of population P of candidate solution is maintained, the structures in the current population are evaluated, and on the basis of that evolution a new population of candidate solution is formed.

In GAs, the solution structures are determined by genes that are in code term and are consisted of a number of chromosomes, each of which represents an individual. GAs ensure the gradual increasing of the good solutions, through checking the new solutions which are generated from old population with random. As a new kind of optimization method, GAs are different from many old optimization methods in many aspects. Firstly, stochastic search processes play an important role in GAs, not deterministic processes as in many other optimization methods. Secondly, GAs consider many points in a search space simultaneously, but only one point is considered in many other methods. Therefore, GAs have lower chance to converge into a local optimization than others. Thirdly, GAs do not require the structures, parameters or other information about the problem. Finally, GA works with a chromosome space which represents the search space, not the search space itself.

In a simple genetic algorithm (SGA), the solution structure is required to be represented as binary strings. There are three most commonly used operators: reproduction, crossover, and mutation. The reproduction operator produces a new population from the old one by using the elitist selection
mechanisms or other strategies which keep the worthy individuals in the next generation. Crossover operator provides a mechanism for strings to produce new genes through information exchange in a random way (probability decision). Crossover proceeds as follows: two strings are selected from the population, then a crossing site or more than one crossing sites along the two strings are selected with some probability. Finally, all characters following the crossing site are exchanged. Below is an example that shows two strings A and B with length of 10 before and after crossover procedure (see Fig. 1).

The next commonly used operator is mutation. The mutation of a bit involves flipping it: change a 0 to 1 or vice versa. The bits of a string are independently mutated, i.e. the mutation of a bit does not affect the mutation probability of other bits. The role of mutation operator is to restore new genetic materials. For example, suppose all the strings in a population have converged to a 0 at a given position, and the optimal solution has a 1 at that position. Then crossover cannot regenerate a 1 at that position, while mutation operator can do it.

\[
\begin{align*}
A &= 1010110111 \\
B &= 1110101010 \\
A' &= 1010101101 \\
B' &= 1110111011
\end{align*}
\]

Before Crossover

After Crossover

Fig. 1. An example of crossover.

For a simple genetic algorithm, the structure can be expressed as Fig. 2. From the GA's structure, we can see that it has the following components:

- a population of binary strings,
- genetic operators (reproduction, crossover, and mutation),
- control parameters (population size \(N\), crossover rate, mutation rate, selection strategies and generation gaps),
- a fitness function, and
- a mechanism to encode the solution (genes) as binary strings.

3. Fuzzy controller based on genetic algorithms

In this section, the application of GAs to the problem of selecting membership functions and fuzzy rules for a complex process is presented. We consider a fuzzy system whose basic structure is shown in Fig. 3. There are four principle components in such a fuzzy system: fuzzifier, fuzzy rule base, defuzzifier and fuzzy inference engine.

Here we only consider multi-input, single-output systems. A multi-input system (MIMO) can always be divided into a group of single-output systems.

The fuzzifier performs a mapping from the observed crisp input space \(U \in \mathbb{R}^n\) that is formed by the output of the process to be controlled to a fuzzy set defined in \(U\), which is characterized by a linguistic term function \(\mu_F: U \rightarrow [0,1]\), and is labeled by a linguistic term such as "small", "medium", "large" or "very large". The most commonly used membership function has three types: bell-shaped, triangle-shaped and trapezoid-shaped. Now we choose

![Fig. 3. The structure of the fuzzy logic system based on GA.](image-url)
bell-shaped form as the following:
\[ \mu_{R_j}(x_j) = \exp \left\{ -\left( \frac{(x_j - c_j)}{a_j} \right)^2 \right\}, \]
(1)
where \( a_j, b_j, c_j \) are the parameters of the membership function. Changing the values of these parameters, the shape of membership functions will vary accordingly.

The fuzzy rule base consists of a set of linguistic rules in the form of "IF \( \langle \text{condition} \rangle \) THEN \( \langle \text{action} \rangle \)". In this paper, we consider the case where the rule base consists of \( N \) rules in the following form:
\[ R^i: \text{If } x_1 \text{ is } F^i_1 \text{ and } x_2 \text{ is } F^i_2 \text{ and } \ldots \text{ and } x_n \text{ is } F^i_n, \]
Then \( y \) is \( G^i, \)
(2)
where \( x_j, j = 1, 2, \ldots, n, \) and \( y \) are state variables and control variables of the process, respectively. \( F^i_j \) and \( G^i \) are the linguistic terms characterized by fuzzy membership function \( \mu_{F^i_j}(x_j) \) and \( \mu_{G^i}(y) \), respectively. Each \( R^i \) can be viewed as a fuzzy implication \( F^i_1 \times \cdots \times F^i_n \rightarrow G^i, \) which is a fuzzy set in \( U \times R \) with the form as the following:
\[ \mu_{R^i} = \mu_{F^i_1}(x_1) \land \cdots \land \mu_{F^i_n}(x_n) \land \mu_{G^i}(y), \]
(3)
where we used the operation "\( \land \)"("min"), which is the most commonly used operation in fuzzy systems.

The fuzzy inference engine is a decision-making logic which employs fuzzy rules from the fuzzy rule base, to determine a mapping from the fuzzy set in the input space \( U \) to the fuzzy sets in the output space \( R \). Let \( A^i \) be an arbitrary fuzzy set in \( U \); then each \( R^i \) determines a fuzzy set \( A^i \circ R^i \) in \( R \) based on the sup-star composition \( [9, 10] \):
\[ \mu_{A^i \circ R^i}(y) = \sup_{x \in U} \left[ \mu_{A^i}(x) \ast \mu_{R^i}(x_1) \ast \ldots \ast \mu_{R^i}(x_n) \ast \mu_{G^i}(y) \right]. \]
(4)

The defuzzifier performs a mapping from the fuzzy set \( A^i \circ R^i \) in \( R \) to a crisp point in \( y \in R \). If the defuzzifier is chosen as the following center average defuzzifier, then the output of fuzzy control system can be obtained:
\[ y = \left[ \sum_{j=1}^{n} \omega_j \mu_{A^i}(x_j) \right] / \left[ \sum_{j=1}^{n} \mu_{G^i}(y_j) \right]. \]
(5)
where \( \omega_j \) is the point in \( R \) at which the membership function \( \mu_{G^i}(y_j) \) reaches its maximum value. In the case that Mamdani's minimum operation rule is employed, Eq. (4) can be reduced as
\[ y = \left( \sum_{j=1}^{n} \omega_j \mu_{R^i} \right) / \left( \sum_{j=1}^{n} \mu_{R^i} \right). \]
(6)

In order to control a complicated process, we choose fuzzy rules that have the form as provided by Sugeno [6, 17]:
\[ R^i: \text{If } x_1 \text{ is } A^i_1 \text{ and } x_2 \text{ is } A^i_2 \ldots \text{ and } x_n \text{ is } A^i_n, \]
Then \( y^i = p^i_0 + p^i_1 x_1 + \cdots + p^i_n x_n, \)
(7)
where \( n = 1, 2, \ldots, m. \)
Thus, the final output \( y \) can be computed as the following:
\[ y^* = \sum_{i=0}^{n} \left( \min \left( \mu_{A^i_1}(x_1), \ldots, \mu_{A^i_n}(x_n) \right) \right) \cdot (p^i_0 + p^i_1 x_1 + \cdots + p^i_n x_n) \]
\[ \sum_{i=0}^{n} \min \left( \mu_{A^i_1}(x_1), \ldots, \mu_{A^i_n}(x_n) \right). \]
(8)

According to (1) and (8), the final output \( y \) is a non-linear function of parameter set \( \{a_i, b_i, c_i\} \), and \( \{p^0_0, p^i_1, \ldots, p^i_n, p^i_0, \ldots, p^i_n\} \). This is a quite complicated non-linear function. The determination of parameter set cannot be solved using conventional method. In order to employ GA to select optimal parameters of (8), we must establish an optimization problem which minimizes the following cost function:
\[ J = \sum_{i=1}^{n} \beta_i |x_i - x^d_i| \text{ or } J = \sum_{i=1}^{n} \beta_i (x_i - x^d_i)^2 \]
(9)
and subject to Eq. (8). Where \( x_i, x^d_i \) are the output and the desired output of the process, respectively. Thus the design of a fuzzy control system which satisfies a prescribed target is an optimization search procedure over a large parameter space. For a complex process, usually, there
are many fuzzy rules and a great number of parameters which should be determined to achieve optimal or near-optimal performances. Because the detailed optimization procedure of GA is problem-dependent, we take a typical example of controlling a double inverted pendulum to explain how GA can be used to select optimal parameters for the FLC.

4. Design of FLC based on GA

The double inverted pendulum system (DIPS) is a complicated nonlinear system and have been investigated by many researchers. There exist many conventional methods to control DIPS [2], but the difficulties of the design of controller parameters based on the classical theory of linear system, always make the performance of the DIPS, such as robustness, flexibility, not satisfactory than expected. The fuzzy controller based GA described in this paper can overcome these difficulties of design problems.

4.1. Description of the double inverted pendulum system

The double inverted pendulum system (DIPS), as shown in Fig. 4, is composed of a cart and two rigid poles with the length \( L_1 \) and \( L_2 \). The end of the lower pole is mounted on the cart \( W \) and the top end of the lower pole is connected with the end of the higher pole. All the conjunction is in such a way that the poles can only swing in a vertical plane parallel to the direction of the motion of the cart. The control goal is to balance the poles by pushing the cart back and forth on a track of limited length.

The state of the double inverted pendulum system can be described by six variables as shown in Fig. 4: the displacement \( x \) of the cart, its velocity \( \dot{x} \), the poles' angular rotation \( \theta_1 \) and \( \theta_2 \), and their angular velocity \( \dot{\theta}_1 \) and \( \dot{\theta}_2 \). The force \( u \) applied to the cart provides controlling actions. The behavior of the DIPS is governed by the following three second-order differential equations [2]:

\[
M(\theta_1, \theta_2) \begin{bmatrix} \ddot{x} \\ \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} + F(\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2) \begin{bmatrix} \dot{x} \\ \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} = G(u, \theta_1, \theta_2)
\]

(10)

where

\[
M(\theta_1, \theta_2) = \begin{bmatrix}
M_0 + M_1 + M_2 & (M_2 L_1 + M_1 L_1) \cos \theta_1 & M_2 L_2 \cos \theta_2 \\
(M_2 L_1 + M_1 L_1) \cos \theta_1 & J_1 + M_1 l_1^2 + M_2 L_1^2 & M_2 L_2 L_1 \cos(\theta_2 - \theta_1) \\
M_2 L_2 \cos \theta_2 & M_2 L_2 L_1 \cos(\theta_2 - \theta_1) & J_2 + M_2 l_2^2
\end{bmatrix},
\]

(11)

\[
F(\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2) = \begin{bmatrix}
F_0 & -(M_2 L_1 + M_1 L_1) \dot{\theta}_1 \sin \theta_1 & -M_2 L_2 \dot{\theta}_2 \sin \theta_2 \\
0 & F_1 + F_2 & -M_2 L_2 L_1 \dot{\theta}_2 \sin(\theta_2 - \theta_1) - F_2 \\
0 & M_2 L_2 L_1 \dot{\theta}_2 \sin(\theta_2 - \theta_1) - F_2 & F_2
\end{bmatrix}
\]

(12)

\[
G(u, \theta_1, \theta_2) = [u \quad (M_1 l_1 + M_2 L_1) g \sin \theta_1 \quad M_2 L_2 g \sin \theta_2]^T.
\]

(13)

with

- \( M_0 \): the mass of the cart
- \( F_0 \): the friction coefficient between the cart and track
- \( l_1, l_2 \): the length of poles between the gravity center to the lower end of the corresponding pole
- \( F_1, F_2 \): the friction coefficient between the lower pole and the cart and between the two poles, respectively
- \( J_1, J_2 \): the inertia of the poles respected to the end point of the lower and higher pole, respectively.
**4.2. Selecting and optimizing fuzzy rule base by GAs**

Before starting the design procedure using GA, we must reduce the number of fuzzy rules and the number of parameters of membership function to be selected as small as possible.

Now let us return to Eq. (10), and combine with Eqs. (11)–(13), we have six fuzzy state variables: $x, \dot{x}, \theta, \dot{\theta}, \omega$, and $\dot{\omega}$. Assume that each of fuzzy variable only has two subsets defined as “positive” and “negative” for the sake of simplicity. Thus the rule base at most has $2^6 = 64$ rules, i.e. $n = 6$, $m = 64$. According to Eq. (1), each of membership functions has three parameters, $a_i, b_i, c_i$. Because the membership function for “positive” and “negative” are symmetrical to the original point of the coordinates, the parameters of “positive” and “negative” membership functions must have the relationship as the following:

$$a^+ = a^-, \quad b^+ = b^- , \quad c^+ = -c^- .$$

where symbol $+$ and $-$ represent the parameters of “positive” and “negative” membership function, respectively. Thus the parameters’ number of membership functions can be reduced to 18. From the knowledge about linear system control, without losing the generality, we can assume that

$$p_1^0 = \cdots = p_6^0 = p_1,$$

$$\vdots$$

$$p_6^1 = p_6^2 = \cdots = p_6^{64} = p_6.$$

It means that in the consequent part of each fuzzy rules only one parameter is different among them. Considering again the symmetric of the movement of the cart, the number of fuzzy rules can be half less. Thus, the total number of parameters to be selected becomes 56.

As mentioned above, there is a basic decision to be made before using GAs in an optimization problem. It is how to evaluate the merits of each string (each membership function set). The task of defining a fitness function is always application specific; it always comes down to accurately describing the goal of the controller. In the case of controlling DIPS, the objective of the fuzzy controller is to drive the cart to the equilibrium position that requires the displacement and velocity of the cart and the angular rotation and velocity of the two poles as small as possible at that position.

The commonly used criterion (cost function) has been shown in (9). For the case of controlling DIPS, we establish the criterion of the form as

$$J = \sum_{k=0}^{K} (\beta_1 x^2 (k) + \beta_2 \dot{x}^2 (k) + \beta_3 \theta^2 (k) + \beta_4 \dot{\theta}^2 (k)$$

$$+ \beta_5 \omega^2 (k) + \beta_6 \dot{\omega}^2 (k)),$$

where $k$ is the time interval. $K$ is the final time instant.

$$\beta_1, \beta_3 \geq 10, \quad \beta_4 \geq 100.$$
Obviously, the bigger $\beta_i$ is, and the smaller $x_i$ or $\theta_i$ or $\theta$ will be, the less change of these variables will occur.

The unequal constraints of (17) mean that the motion of the higher pole should be constricted in more narrow scope than the lower pole and the cart.

Now we are in the position to select the parameters of GA as explained in Section 2.

(1) **Encoding the fuzzy variables**

The string length $N$ is an important parameter for GA. The selection of $N$ must consider the accuracy of fuzzy variables and the complexity of operation. There is always a trade-off between complexity and accuracy in the choice of string length $N$. Here we choose 32 bits as the length of string $N$. Given the length $N$ and the change range of fuzzy variables, the encoding of variables (genes) can be calculated as

$$B(x_i) = \left[(x_i - x_{i0})/x_{iC}\right] \cdot 2^{N-1}, \quad (18)$$

where $x_{iC}$ is the change range of variable $x_i$, and $x_{i0}$ is the central point of the range. $B$ is the mapping from the input variable into binary value.

(2) **Crossover rate $z$ and mutation rate $\beta$**

Crossover rate $z$ and mutation rate $\beta$ are not fixed during evolution period. It may be bigger in initial stages, and gradually become smaller in the evolution later. Because it can increase the appearance probability of gene patterns in initial periods of the evolution and speed up the optimization procedure. We select $z = 0.1$, $\beta = 0.05$ in the beginning, then decrease 5% in each generation until $z = 0.01$, $\beta = 0.005$.

(3) **The gap of the generation**

The goal of the choice of the generation gap is to control the number of individuals that are survived in the next generation. We use the following strategy: firstly the $g_1$ percentage of the most elitists of the population are maintained in the next generation, then selecting the $g_2$ percentage of the remainder individuals randomly into the next generation. Such that the goal of maintaining gene patterns in the population can be achieved.

(4) **The size of the population**

To mimic the evolution of nature better, the population size in the paper is not a constant, which can vary during the evolution according to the crossover rate $z$ and mutation rates $\beta$.

The population of the new generation can be determined by the following expression:

$$P_{\text{new}} = \theta_1 P_{\text{old}} + (1 - \theta_1)P_{\text{old}} \theta_2 + zP_{\text{old}} + \beta P_{\text{old}}$$

(19)

When the population decreases to a prescribed value (50, in this paper), some individuals will be added randomly in order to ensure the size of the population and new gene's patterns.

5. **Simulation and real-time operation**

5.1. **Simulation results**

The simulation is carried with the following parameter sets of DIPS (Table 1):

We establish the control goal by minimizing the following cost function:

$$J = \sum_{i=0}^{N} (x_i^2 + 500\theta_i^2 + 2500\theta_i^2)$$

(20)

and subject to

$$x^* = \max \{\mu_{\xi_1}(x_1), \ldots, \mu_{\xi_n}(x_m)\}$$

$$\cdot (p_0 + p_1 x_1 + \cdots + p_n x_n)$$

$$\sum_{i=0}^{N} \min \{\mu_{\xi_1}(x_1), \ldots, \mu_{\xi_n}(x_m)\}.$$  

According to (9), the output of fuzzy controller is determined by parameter sets $\{a_i, b_i, c_i\}$ and $\{p_0, p_1, \ldots, p_n, p_1, \ldots, p_m\}$ $n = 6$, $m = 32$, each chromosome is represented by fifty-six integers with 32 binary bits for each. Each gene corresponds to

<table>
<thead>
<tr>
<th>Table 1</th>
<th>The parameters of the practical DIPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_0$</td>
<td>1.0 kg</td>
</tr>
<tr>
<td>$F_0$</td>
<td>10.6 N/m</td>
</tr>
<tr>
<td>$\ell_1, \ell_2$</td>
<td>0.25 m</td>
</tr>
<tr>
<td>$F_1, F_2$</td>
<td>0.0015 N/s/m</td>
</tr>
<tr>
<td>$J_1, J_2$</td>
<td>0.0044 Kg m</td>
</tr>
<tr>
<td>$M_1, M_2$</td>
<td>0.1 kg</td>
</tr>
<tr>
<td>$L_1$</td>
<td>0.5 m</td>
</tr>
</tbody>
</table>
a parameter. To begin with GA, the initial population is selected at random.

The final parameter sets \([a_i, b_i, c_i]\) that are shown in Table 2 are generated after 50 generations.

Fig. 5 demonstrates the trajectories of displacement \(x\) of the cart and angular rotation \(\theta_1\) and \(\theta_2\) of the two poles under the control \(u\) of fuzzy controller with the parameters that are selected by Genetic Algorithm. In comparison with results under the linear controller which are also shown in the same figures, we can see that the fuzzy control system which are designed by GA has much better performance than conventional control.

In order to check the robustness of the fuzzy control system based on GA, we change the length of the two poles by
\[
l_1 = l_2 = 0.12, \quad L_1 = 1.0.
\]

The trajectories of \(x, \theta_1, \theta_2\) and \(u\) are shown in Fig. 6. The simulation results prove that the DIPS under the control of the fuzzy logic controller designed by GA, has perfect performances under structural perturbation.

### 5.2. Real-time operation

As we described in Section 3, the real structure of the DIPS is quite different from the model expressed in Eq. (10). Because there are many non-linear factors which have not been taken account into. Additionally, noise will be unavoidably penetrated into the system.

<table>
<thead>
<tr>
<th>Variable</th>
<th>(a)</th>
<th>(b)</th>
<th>(c)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x)</td>
<td>10.747</td>
<td>0.926</td>
<td>0.824</td>
</tr>
<tr>
<td>(\dot{x})</td>
<td>104.970</td>
<td>0.660</td>
<td>49.209</td>
</tr>
<tr>
<td>(\theta_1)</td>
<td>1.199</td>
<td>2.532</td>
<td>0.969</td>
</tr>
<tr>
<td>(\dot{\theta}_1)</td>
<td>0.544</td>
<td>3.287</td>
<td>0.876</td>
</tr>
<tr>
<td>(\theta_2)</td>
<td>1.114</td>
<td>1.193</td>
<td>0.773</td>
</tr>
<tr>
<td>(\dot{\theta}_2)</td>
<td>1.093</td>
<td>0.255</td>
<td>0.949</td>
</tr>
</tbody>
</table>

Table 2. The final parameters' values after optimization.

Fig. 5. The result comparison between FLC based on GAs and linear control: (a) the cart displacement; (b) the lower pole' rotation angular; (c) the higher pole' rotation angular; and (d) the control input.
Fig. 6. Simulation results of DIPS with FLC based on GA: (a) the cart displacement; (b) the lower pole\'s rotation angular; (c) the higher pole\'s rotation angular; and (d) the control input.

Fig. 7. Practical experiment results of DIPS with FLC based on GA: (a) the cart displacement; (b) the lower pole\'s rotation angular; (c) the higher pole\'s rotation angular; (d) the control input.
With the design method provided above, we also construct a real-time fuzzy logical controller on PC386 to control a practical DIPS. The characteristic curves of $x, \theta_1$ and $\theta_2$ under the fuzzy control after applying a disturbance are shown in Fig. 7. The sampling time for the experiment is 9.6 ms. Although they are different from the simulation results, which is caused by the difference between the model and practical DIPS, the behavior performance of the real DIPS with FLC based on GA’s are much better than the results of classical control.

6. Conclusion

In the paper we apply GA to the design of a fuzzy controller for complex processes. Essentially, it is an adaptive fuzzy controller to solve the uncertainty and non-linear phenomena in complex systems. The simulation and real-time operating results proved that the proposed method based on GAs is an effective way to design fuzzy logical controller. With the advantage of GAs, which can deal with optimization problems that have large-scale search space, we can design optimal fuzzy controllers for controlling complicated processes. The method provided in this paper is general and can be applied to a wide control fields. Genetic algorithm combined with other intelligent techniques, such as neural networks, expert systems and fuzzy logical control systems, open a new way to design and construct intelligence control systems adapted to complex processes. Surely, it will have a bright future.

References