Proceedings of the 36th
Conference on Decision & Control
San Diego, California USA • December 1997

Intelligent Fuzzy Controller with a Sequential Learning Mechanism

Feng-Yih Hsu[1] and Li-Chen Fu[1,2]
Dept. of Electrical Engineering[1]
Dept. of Computer Science & Information Engineering[2]
National Taiwan University, Taipei, Taiwan, R.O.C.

Abstract

This paper presents a linguistic learning-based fuzzy controller of which learning is accomplished in a sequential manner via an embedded learning mechanism. The systems handled here are characterized by a set of linguistic rules from human operator used to describe the underlying system model vaguely. The proposed control algorithm incarnates the design procedure adopted by human being through linguistic formulation of the controlled system and learning action during actual execution. Simulations on control of an inverted pendulum are conducted and results verify that the appealing performance of the proposed control architecture.

1 Introduction

When one needs to control an unfamiliar system to meet some desired performance, he is often required to observe the system first and then to describe it with sufficient information for subsequent design of suitable control methodology. The form of the information can be either some mathematical model or some set of linguistic rules, dependent upon the way of describing the underlying system. It is well known that the former mathematical model is generally considered as a basis for classic control theories, whereas the latter is often associated with intelligent control schemes, such as AI-based control, expert control and fuzzy control. For instance, for the speed control of an automotive, an engineer may first model the system by a set of dynamic equations where the accelerator, and the automotive speed are treated as the system input and the system output, respectively, after analyzing the automotive kinematics and its surrounding environment. In contrast, for a driver who is lacking physical knowledge of the automotive system, although unfamiliar with the precise way of controlling the acceleration to reach the desired speed, he can easily extract some linguistic properties about the speed control system of the automotive, such as increasing the magnitude of the control input will automatically provide higher speed than not doing so or performing its reverse. However, these linguistic rules are very fuzzy so that users may need some experiences or several trials during the procedure of designing a suitable controller. In fact, a human being can apply his well-given learning capability to update or to refine his existing knowledge.

On the other hand, when the underlying system is too complex to be easily described, a human being can decompose the system into several sub-systems, each of which can be more easily handled, and then establish the control by sequentially learning the control strategies of the individual sub-systems. In this view point, we present a linguistic learning-based fuzzy control (LLBFC) embedded into a sequential learning mechanism to imitate the procedure of controller design generally adopted by human being.

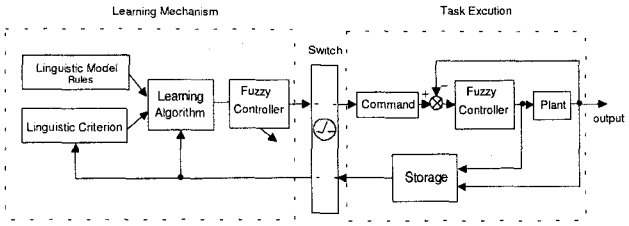

Figure 1: The architecture of LLBFC

There are two important features in LLBFC: one is to take linguistic terms to describe a target system, and the other is to use a learning algorithm to improve the previously acquired control rules. The former results in descriptive linguistic statements whereas the latter facilitates one to continuously refine the control rule base till the system response meets the desired goal. In order to validate the proposed control, in this paper, we apply the developed LLBFC to an inverted pendulum, which is often taken for a test for some learning control schemes in the literature [1]-[7].

2 Linguistic Learning-Based Fuzzy Control

The architecture of LLBFC mainly consists of the following five parts (see Fig. 1):

- fuzzy controller: which consists of *If-then* rules and drives the system to meet the specified goal,
- linguistic criterion: which declares the specification of the system performance,
- linguistic model rules: which consists of *If-then* rules that describes the behavior of the controlled system,
- storage: which saves some measurable system states and the control input during task running,
- learning algorithm: which updates the fuzzy control rules.

When the controlled plant (with control input u and state vector x) is very complex, LLBFC is incorporated a sequential learning mechanism as depicted in Fig.2. First of all, the overall system is decomposed into several sub-systems, each of which sub-system is given a

0-7803-3970-8/97 $10.00 © 1997 IEEE

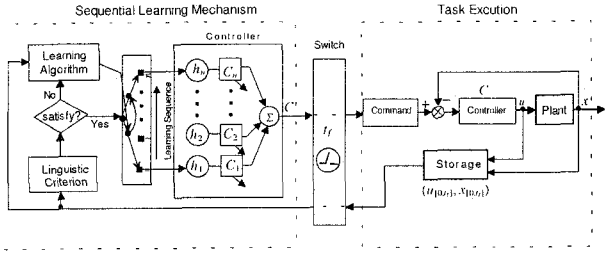

Figure 2: The sequential learning mechanism for LLBFC

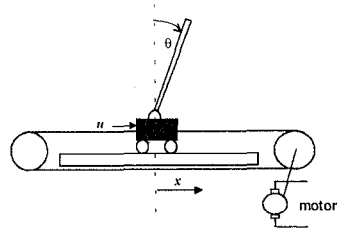

Figure 3: The diagram of an inverted pendulum system

sub-controller C_i, $i = 1, \cdots, n$. These sub-controllers will be respectively trained with some suitable learning condition functions h_i, $i = 1, \cdots, n$, in a sequential manner every time after the task is expected once. That is, the learning algorithm is to improve existing fuzzy control rules obtained from the previous system responses, which means collection of the pair of control inputs and state vectors during the task running period, t_f, denoted it $(u_{[0,t_f]}, x_{[0,t_f]}) = \{(u(kT), x(kT))|k = 0, \cdots, m\}$, where T is the sampling time to save the signals and $t_f = mT$ for some integer $m > 0$. If the response of any sub-system does not satisfy the associated linguistic criterion, the corresponding learning algorithm mechanism for the sub-controller will be activated till that criterion is met. Thus, the overall controller C is simply the sum of the control inputs from all sub-controllers. In the following section, we will apply the LLBFC to an inverted pendulum system as depicted in Fig. 3 as an example to illustrate the overall design procedure for LLBFC.

3 LLBFC to an Inverted Pendulum System

The quality of the skill of controlling an inverted pendulum system is often viewed as the level of ability for an intelligent control scheme [1]-[7]. The set-up of an inverted pendulum system consists of a DC motor and a cart carrying a pole, where the motor is to drive the cart so that the pole will not fall down. In Fig. 3, θ denotes the angle displacement of the pole, x denotes the position of the cart, and u is the control input. Besides, we denote $\dot{\theta}$ and $\dot{x}$ as the angular velocity of pole and the position velocity of the cart, respectively. First, we set a sequential learning procedure as in Fig. 4. Then the overall system is decomposed into two sub-systems, are for the angular displacement of the pole and the another for the position of the cart. The sequence is arranged such that the sub-controller (with output u_θ) for the angular position of the pole starts the learning process first till the response of the pole sub-system meets the linguistic criterion, and then the learning process of the other sub-controller (with output u_x) for the position of the cart is conducted via the help of some learning condition function $h_x(\theta, \dot{\theta}, x, \dot{x})$. Finally, the overall controller after the learning can be expressed as follows:

$$u = u_\theta + h_x(\theta, \dot{\theta}, x, \dot{x})u_x \quad (1)$$

The control objective here is to regulate both the angular displacement of the pole and the position of the cart at the zero value subject to some performance criteria. First, we let $u_x = 0$ and train u_θ by LLBFC. According to what has been derived, it is then apparent that we will first train $u = u_\theta$ by letting $u_x = 0$. In the following, we will detail the design of each sub-controller with an emphasis on individual learning mechanism.

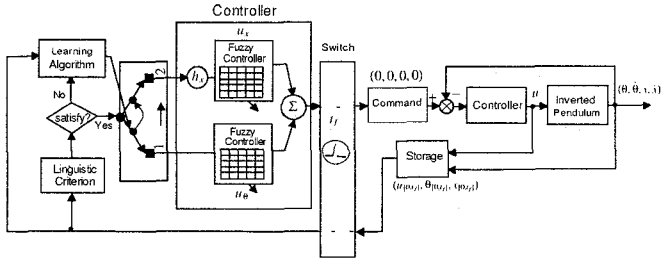

Figure 4: The learning mechanism for an inverted pendulum

A. Linguistic Criterion

If the controlled system is a linear system, the linear quadratic regulator (LQR) design is a popular approach which can achieve the control objective by choosing a suitable cost function. However, such an approach can hardly be applied to the systems whose mathematical models are not available. In contrast, a human being can often handle such system by exploiting human capabilities like observing, learning or teaching, especially when the experiences of the experts can be useful. With this in mind, we consider a linguistic criterion given as follows:

" Given the admissible overshoot, undershoot and system constraints, the controlled system should have the shortest risetime."

In the case where the angular displacement of the pole with initial conditions $(\theta(0), \dot{\theta}(0))$ needs to be regulated to zero, we denote the overshoot or undershoot and the risetime are denoted as $O_\theta|\theta_{[0,t_f]}$ and $\tau_\theta|\theta_{[0,t_f]}$, where $\theta_{[0,t_f]} = \{\theta(0), \cdots, \theta(t_f)\}$ represents the locus of θ during the period of task running, $[0, t_f]$, which is set to avoid the cart driving out of the allowed position range. Let $\theta_d = 0$ denote the angular configuration at which the pole stands vertically. If the system response has either an overshoot or an undershoot in the case where the initial condition $\theta(0) < 0$, O_θ is defined as the maximum value of θ during the period of the task running, i.e., $O_\theta = max\{\theta(t), 0 \leq t \leq t_f\}$. But, in the case where the initial condition $\theta(0) > 0$, O_θ is the minimum value of θ during the period of the task running, i.e., $O_\theta = min\{\theta(t), 0 \leq t \leq t_f\}$. Let $O_{\theta d}$ be defined as an admissible value of $|O_\theta|$, and let the risetime, denoted as

τ_θ, represent the time elapse from $t = 0$ to the time when θ first enters the interval of $O_{\theta d}$. On the other hand, let the system constraint be placed on the magnitude of the control input, say, u_{max}, i.e., $|u(t)|_\infty \leq u_{max}$.

B. Fuzzy Controller

A fuzzy controller consists of a collection of fuzzy *If-then* rules and the i-th fuzzy rule of the fuzzy rule-base R_θ for input fuzzy variables θ and $\dot{\theta}$ and the output fuzzy variable u_θ is expressed as follows:

$$R_\theta[i]: \text{If } \theta \text{ is } F_\theta^{\alpha(i)} \text{ and } \dot{\theta} \text{ is } F_{\dot{\theta}}^{\beta(i)} \text{ then } u \text{ is } F_u^{\gamma(i)},$$

where $F_\theta^{\alpha(i)} \in F_\theta$ and $F_{\dot{\theta}}^{\beta(i)} \in F_{\dot{\theta}}$ are input fuzzy sets in the families F_θ, $F_{\dot{\theta}}$ and $F_u^{\gamma(i)} \in F_u$ is the output fuzzy set in the family F_u; $\alpha_{(i)}$, $\beta_{(i)}$, and $\gamma_{(i)}$ are integer indices associated with the i-th rule, and $\gamma_{(i)}$ can be mapped from $\alpha_{(i)}$ and $\beta_{(i)}$, for $i = 1, \cdots, \Upsilon$ (total number of rules). Let (α, β, γ) be a triple of integer indices from $\alpha_{(i)}$'s, $\beta_{(i)}$'s, and $\gamma_{(i)}$'s, then F_θ^α, $F_{\dot{\theta}}^\beta$ and F_u^γ are fuzzy sets associated with the membership functions $\mu_{F_\theta^\alpha}(\theta)$, $\mu_{F_{\dot{\theta}}^\beta}(\dot{\theta})$, and $\mu_{F_u^\gamma}(u)$, respectively, lying in the range [0 1] and satisfying $\mu_{F_\theta^\alpha}(\bar{\theta}^\alpha) = \mu_{F_{\dot{\theta}}^\beta}(\bar{\dot{\theta}}^\beta) = \mu_{F_u^\gamma}(\bar{u}^\gamma) = 1$ for the centers $\bar{\theta}^\alpha$, $\bar{\dot{\theta}}^\beta$, and $\bar{u}^\gamma$ of the fuzzy sets F_θ^α, $F_{\dot{\theta}}^\beta$ and F_u^γ, as shown in Fig. 5 Let the centers of the input fuzzy sets be set as $\bar{\theta}^{i+1} = \delta_{\theta_i} + \bar{\theta}^i$, $i = 1, \cdots, r_\theta - 1$, and $\bar{\dot{\theta}}^{j+1} = \delta_{\dot{\theta}_j} + \bar{\dot{\theta}}^j$, $j = 1, \cdots, r_{\dot{\theta}} - 1$, where δ_{θ_i} and $\delta_{\dot{\theta}_j}$ are elements of the positive constant vectors δ_θ and $\delta_{\dot{\theta}}$, respectively, which represent the distances between the two contiguous centers of the input fuzzy sets, $\bar{\theta}^i$, $\bar{\theta}^{i+1}$ and $\bar{\dot{\theta}}^j$, $\bar{\dot{\theta}}^{j+1}$, respectively. Note that r_θ and $r_{\dot{\theta}}$ are the numbers of centers of the input fuzzy sets for θ and $\dot{\theta}$, respectively, and satisfy $r_\theta \cdot r_{\dot{\theta}} = \Upsilon$.

Finally, through a defuzzification process, the output variable u can be expressed as follows:

$$u = \sum_{i=1}^{\Upsilon} \bar{u}^{\gamma(i)} \xi_i(\theta, \dot{\theta}) = \Phi^T \xi(\theta, \dot{\theta}), \tag{2}$$

where $\Phi = [\bar{u}^{\gamma(1)}, \cdots, \bar{u}^{\gamma(\Upsilon)}]^T = [\Phi_1, \cdots, \Phi_\Upsilon]^T$ is regarded as a parameter vector of output fuzzy sets and $\xi = [\xi_1, \cdots, \xi_\Upsilon]^T$ is a nonlinear vector composing fuzzy basis functions. To strengthen the effectiveness of localizing the necessary fuzzy rules and to reduce the computation time, the fuzzy rules of R_θ should be classified into the set of fired rules and the set of unfired rules. Define the domain set of the total involved fuzzy rules as W, a product set of the universal discourse sets $[\bar{\theta}^1\ \bar{\theta}^{r_\theta}]$ and $[\bar{\dot{\theta}}^1\ \bar{\dot{\theta}}^{r_{\dot{\theta}}}]$, i.e, $W = [\bar{\theta}^1\ \bar{\theta}^{r_\theta}] \times [\bar{\dot{\theta}}^1\ \bar{\dot{\theta}}^{r_{\dot{\theta}}}]$. Then, we partition this domain set into a finite domain cells, each of which is defined as $W_{\alpha\beta} = [\bar{\theta}^\alpha\ \bar{\theta}^{\alpha+1}] \times [\bar{\dot{\theta}}^\beta\ \bar{\dot{\theta}}^{\beta+1}]$, for $\alpha = 1, \cdots, r_\theta - 1$ and $\beta = 1, \cdots, r_{\dot{\theta}} - 1$. Then, the following proposition is valid.

Proposition 1 *If the pair of* $(\theta, \dot{\theta}) \in W$ *and the membership functions are given as depicted in Fig.* 5, *then there exists a domain cell* $W_{\alpha\beta}$ *such that the domain set of the fired fuzzy rules is included in* $W_{\alpha\beta}$.

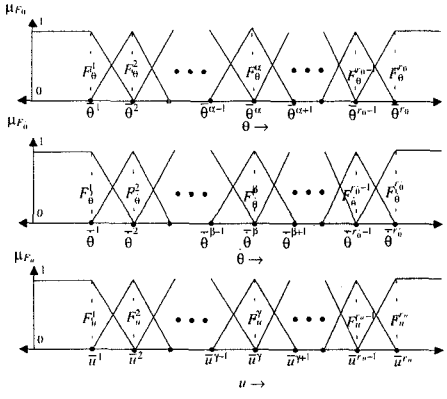

Figure 5: The families of the fuzzy sets F_θ, $F_{\dot{\theta}}$ and F_u

C. Linguistic Model Rules

After determining the structure of the fuzzy controller, we have to construct a learning algorithm which can improve those fuzzy control rules till the system response satisfies the previously mentioned linguistic criterion. To construct such a learning algorithm, we have to solve the following problems:

- How to find the fuzzy control rules which can be possibly fired in the next execution of the task?
- How to assure that the updated fuzzy control rules can really improve the performance of the controlled system?

To look for the solutions of the above-mentioned problems, we first make some observations of the behavior of the controlled inverted pendulum system. From Fig. 6, $\theta_{[0,t_f]}$ represents a locus of the angular displacement of the pole from the initial time $t = 0$ to the terminal time $t = t_f$. Apparently, an overshoot will occur if $\theta_{[0,t_f]}$ crosses the vertical axis $\theta = 0$ rather than just touches it. According to Proposition 1, only the fuzzy rules with some specified domain sets can be fired. Hence, the domain sets of all the fired rules as in Fig. 6 will fall into in a collection of domain cells, each of which encloses some part of the locus $\theta_{[0,t_f]}$. Furthermore, to update the fuzzy rules will very likely change the subsequent response of the system. This fact results in that rules selected for firing in the current task execution may not be necessarily all the same as those in the last time of task execution. Hence, to assure that the updated fuzzy rules will be still fired in every execution of the task, the domain sets of the subsequently chosen fuzzy rules to be updated must be within the domain sets of the already fired fuzzy rules.

Now, since the controlled inverted pendulum system is casual, updating those fuzzy rules whose domain sets are within the overshoot area will not help much reduce the overshoot or shorten the risetime. To eliminate the part of locus $\theta_{[0,t_f]}$ in the overshoot area, we denote $\theta_{[0,t_0]}$ to represent a portion of $\theta_{[0,t_f]}$, whose time support is from the initial time $t = 0$ to the time t_0 satisfying $\theta(t_0) = 0$ (as shown by the bold line $\theta_{[0,t_0]}$ in

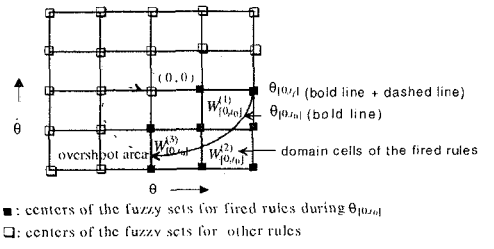

Figure 6: The domain cells of the locus $\theta_{[0,t_0]}$

Fig. 6). Corresponding to $\theta_{[0,t_0]}$, a collection of ordered domain cells are selected as $W^{(1)}_{[0,t_0]}, W^{(2)}_{[0,t_0]}, \cdots, W^{(n)}_{[0,t_0]}$ ($n = 3$ in Fig. 6). After that, we observe the behavior of the controlled inverted pendulum and choose one of $W^{(1)}_{[0,t_0]}, W^{(2)}_{[0,t_0]}, \cdots, W^{(n)}_{[0,t_0]}$ to update the fuzzy rules by the following linguistic model rules:

1. If we want to reduce the overshoot and to avoid increasing the risetime, then we need to reduce the parameters of the output fuzzy sets, namely, Φ, corresponding to the cell $W^{(n)}_{[0,t_0]}$.

2. If we want to eliminate the undershoot and to shorten the risetime, then we need to enlarge the parameters of the output fuzzy sets, Φ, corresponding to the cell $W^{(1)}_{[0,t_0]}$.

Remark:
In the former example (Fig. 6), $W^{(1)}_{[0,t_0]}$ and $W^{(3)}_{[0,t_0]}$ are domain sets of the updated fuzzy rules, respectively, for the undershoot and the overshoot case. After the input constraints u_{max} and $-u_{max}$ are reached for both iterative update, $W^{(2)}_{[0,t_0]}$ will be the common domain set of the fuzzy rules to be updated for the overshoot case and those for the undershoot case. The key spirit of the design procedure is similar to the bang-bang controller in the problem with time optimization.

D. Learning Mechanism

Based on the above linguistic model rules, we propose the learning algorithm as follows.

Learning Algorithm of LLBFC

Step. 1 (List Fired Fuzzy Rules in the Learning Test) Run the task and list the fired fuzzy control rules during the task running. If the locus $(\theta_{[0,t_0]}, \dot{\theta}_{[0,t_0]}, u_{[0,t_0]})$ is updated for $(\theta(t), \dot{\theta}(t), u(t))$ from the initial time $t = 0$ to the time t_0 with $\theta(t_0) = 0$, then the pair set of the domain cells of the fired fuzzy rules and the parameters of the output fuzzy set with respect to this locus are defined as follows: $(W_{[0,t_0]}, \Phi_{[0.t_0]}) = \{(W^{(1)}_{[0,t_0]}, \Phi^{(1)}_{[0,t_0]}), \cdots, (W^{(j)}_{[0,t_0]}, \Phi^{(j)}_{[0,t_0]}), \cdots, (W^{(n)}_{[0,t_0]}, \Phi^{(n)}_{[0,t_0]})\}$ where $\Phi^{(j)}_{[0,t_0]} = \{\Phi_i{}^{(j)}_{[0,t_0]}, i = 1, \ldots, 4\}$ is the set consisting of parameters of the output fuzzy sets with the domain cell $W^{(j)}_{[0,t_0]}$, and n is denoted as its final number ($n = 3$ in Fig. 6).

Step. 2 (Decide Those Rules to Be Updated) Generate Φ^c, the set of parameters of the output fuzzy set , Φ, to be updated.

Step. 2. 1 (Overshoot Case)

(a) First, let p be set to n (associated with the final element of $\Phi_{[0,t_0]}$).

(b) If any element of $\Phi^{(p)}_{[0,t_0]}$ is larger than $-u_{max}$, $\Phi^c \equiv \Phi^{(p)}_{[0,t_0]}$,

(c) Otherwise, let p be updated to $p - 1$. If p is equal to 0, then the system fails since, even when the control input is always set as $u = -u_{max}$, the system still has an overshoot; otherwise go to (b).

Step. 2. 2 (Undershoot Case)

(a) First, let q be set to 1 (associated with the first element of $\Phi_{[0,t_0]}$).

(b) If any element of $\Phi^{(q)}_{[0,t_0]}$ is smaller than u_{max}, $\Phi^c \equiv \Phi^{(q)}_{[0,t_0]}$,

(c) Otherwise, let q is updated to $q + 1$. If q is equal to $n + 1$, then the system fails since, even when the control input is always set as $u = u_{max}$, the system still has an undershoot; otherwise go to (b).

Step. 3 (Update Fuzzy Rules) Update Φ^c by Δ.

Step. 3. 1 Let $\Delta(k)$ be set to $-r sgn(O_\theta(k))$, where r is a positive constant and the integer index k represents the k-th learning iteration.

Step. 3. 2 $\Phi^c = \{\Phi^c_i, i = 1, \cdots, 4\}$, where Φ^c_i is the i-th element of Φ^c

(a) $\Phi^c_i(k+1) = \Phi^c_i(k) + \Delta(k)$;

(b) if $\Phi^c_i(k+1) > u_{max}$, $\Phi^c_i(k+1) = u_{max}$;

(c) if $\Phi^c_i(k+1) < -u_{max}$, $\Phi^c_i(k+1) = -u_{max}$.

Step. 4 (Set Heuristic Rules Based on the Equilibrium Point) If θ is *zero* and $\dot{\theta}$ is *zero* , then u is *zero*. Here, the centers of fuzzy sets corresponds to the fuzzy set *zero* are set to 0.

Step. 5 (Convergence in Learning Procedure)

Step. 5. 1 If Φ^c obtained in Step 2.1 is not equal to the one in Step 2.2 (i.e., $p \neq q$), go to Step 1.

Step. 5. 2 If $|O_\theta(k)| \leq O_{\theta d}$, then the learning procedure is completed.

Step. 5. 3 Otherwise, use $\Delta(k) = -rO_\theta(k)$ to replace $\Delta(k) = -r sgn(O_\theta)$in Step 3.1, and the others as in Step 3.2. Then, go to Step 1.

Step 5 is to assure that the proposed learning algorithm can converge and the learning procedure can provide satisfactory performance. If we skip Step 5 and directly go to Step 1, then when the number of iteration of the above learning procedure exceeds some pre-specified values, the algorithm Step 2.1 and Step 2.2 will generate the same fuzzy control rules. Thus, the response of the system will oscillate with alternate overshoots and undershoots. As a result, Step 5 can actually lead to the convergence of the learning algorithm.

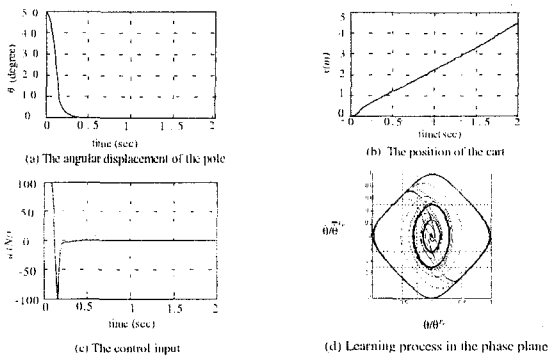

Figure 7: The simulation results for sub-controller u_θ

3.1 Simulation for the Pole Angle

Let the dynamic equations of an inverted pendulum system be expressed as follows:

$$\begin{aligned} \ddot{\theta} &= \frac{gsin(\theta) + cos(\theta)[-u - M_p l\dot{\theta}^2 + \mu_c \dot{x}][M_c + M_p]^{-1} - \mu_p \dot{\theta}(M_p l)^{-1}}{4/3l - M_p l cos(\theta)^2 (M_c + M_p)^{-1}} \\ \ddot{x} &= \frac{u + M_p l[\dot{\theta}^2 sin(\theta) - \ddot{\theta} cos(\theta)] - \mu_c \dot{x}}{M_c + M_p} \end{aligned} \quad (3)$$

where θ is denoted as angle displacement of the pole from the vertical, x is denoted as the cart position, $M_c = 2.0(Kg)$ is the cart mass, $M_p = 0.88(Kg)$ is the pole mass, $2l = 0.62(m)$ is the length of the pole, $\mu_p = 0$ and $\mu_c = 1.0$ are the damping rates of the pole and the cart, respectively, $u = u_\theta$ is the control input subjected to a constraint $u_{max} = 100$. The task is run for two extreme initial conditions, namely, $\theta(0) = 50°$ and, $\theta(0) = -50°$, both with $\dot{\theta}(0) = 0$, $x(0) = 0$ and $\dot{x}(0) = 0$. The period of task running for each initial condition is $t_f = 2$ sec. The fuzzy membership functions are set as the triangular form and the compositional operator is a max-min operator and the defuzzification strategy is the center-of-area type. Initial fuzzy rules are designed as to mimic a bang-bang controller. That is, corresponding to the domain set $\theta > 0$, the parameters of the output fuzzy set, Φ, are set to u_{max}. In contrast, corresponding to the domain set $\theta < 0$, the parameters of the output fuzzy set, Φ, are set to $-u_{max}$. After the first running, we set the centers of the input fuzzy sets as $\bar{\theta}^{r_{\dot{\theta}}} = max\{\dot{\theta}(t), t \in [0, t_f]\}$, $\bar{\theta}^{1} = min\{\dot{\theta}(t), t \in [0, t_f]\}$, $\bar{\theta}^{r_\theta} = 50°$ and $\bar{\theta}^{1} = -50°$, and r_θ and $r_{\dot{\theta}}$ are set to 9 so that the number of total fuzzy rules R_θ is set as $\Upsilon = 9 \times 9 = 81$. Let $r = 0.3u_{max}$ be as the updated increment needed in Step 3.1 of learning algorithm. The control objective is to force the overshoot or the undershoot to the allowed value $O_{\theta d} = 0.01$. Simulations are run using the structure with two sampling rates. The sampling rate with the outer loop is $1k$ Hz which mimics the controller running and the one with the inner loop is $10k$ Hz which approximates the dynamic behavior of the real plant. Fig. 7 shows the simulation results for the fuzzy controller u_θ after 16 times of running. The response of the angular displacement of the pole is shown in Fig. 7(a) with a satisfactory result and the result of the control input is similar to an optimum strategy by a well trained human operator. Fig. 7(d) shows the overall learning process in the phase plane θ-$\dot{\theta}$ for two extreme initial conditions. At the beginning, the system response has a large oscillation and gradually converges to an optimum response when the number of iteration of learning iteration increases.

3.2 Sub-Controller for Cart Position

Apparently, applying LLBFC to learn a designated sub-controller for the cart sub-system, u_x is more difficult due to the nonminimum phase characteristics of the inverted pendulum system. Here a learning condition function h_x is introduced to train u_x subjected to this condition function, subsequently referred to or explained as a hitting condition. Initially, the pole is set in the initial condition as shown in Fig. 8(a), and then the previous well trained sub-controller u_θ is activated to regulate the angular displacement of the pole to zero. Since the sub-controller u_θ is dedicated to balancing the pole, it will naturally require the cart to move for some distance in the inclining direction of the pole. While the pole is in a hitting condition, where the angular displacement of the pole is close to zero as shown in Fig. 8 (c), a hitting force is exerted to force the pole to incline to the direction in which the cart can return the origin (i.e, the position with mark +). As a result, an nonzero angular displacement of the pole will arise from again so that the cart will move back toward the origin by the force of balancing the pole. In fact, the control of hitting force at different hitting conditions will determine whether or not the cart can move back to the origin and to stay within its neighborhood. Hence, how to determine the hitting force and the hitting condition is the major problem of designing u_x. Here, we use a fuzzy rule-base, R_x, to implement different hitting forces according to different hitting condition functions h_x. In fact, in actual implementation, a hitting force will be replaced by that of driving the cart. Hence, we can design fuzzy rule-base, R_x, similar to R_θ in eq. (2) and its i-th fuzzy rule of the fuzzy rule-base, R_x, for input fuzzy variables x and $\dot{x}$ and the output fuzzy variable, u_x, is expressed as follows:

$$R_x[i]: \text{If } x \text{ is } F_x^{\alpha(i)} \text{ and } \dot{x} \text{ is } F_{\dot{x}}^{\beta(i)}, \text{ then } u_x \text{ is } F_{u_x}^{\gamma(i)},$$

where $F_x^{\alpha(i)}$, $F_{\dot{x}}^{\beta(i)}$ and $F_{u_x}^{\gamma(i)}$ are the fuzzy sets as similarly defined in R_θ.

Furthermore, the hitting condition function, h_x, is implemented via follows:

$$h_x(\theta, s_\theta, s_x) = \begin{cases} -1, & \text{if } |\theta| \le \epsilon, |s_\theta| \le \epsilon_\theta \text{ and } |s_x| \ge \epsilon_x \\ 0, & \text{otherwise,} \end{cases} \quad (4)$$

where $s_\theta = \dot{\theta} + \lambda_\theta \theta$ and $s_x = \dot{x} + \lambda_x x$ are two augmented variables with constants $\lambda_\theta > 0$ and $\lambda_x > 0$; ϵ, ϵ_θ and ϵ_x are some small positive constants. Note that $s_\theta \le \epsilon_\theta$ is utilized to shape the pole dynamics are in describable manner, whereas $s_x \ge \epsilon_x$ is to determine whether the hitting force is necessary (because the cart can naturally move back to the neighborhood of the origin, if $|s_x| < \epsilon_x$). To apply the learning algorithm of LLBFC in the last section to produce u_x, we must modify some definitions relevant to the cart sub-system to make LLBFC plausible in designing u_x. Since the initial position of the cart is at the origin, we define the following linguistic criterion as follows:

" *Given a measure of the position error, the cart can move back to the origin within the shortest time.*"

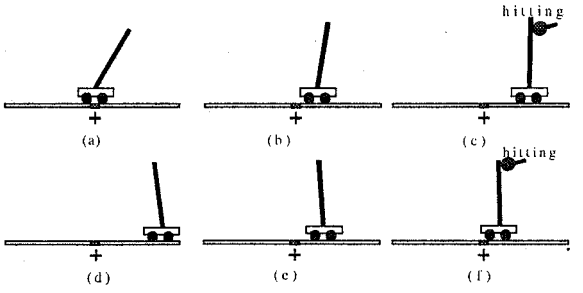

Figure 8: The inverted pendulum with various hitting

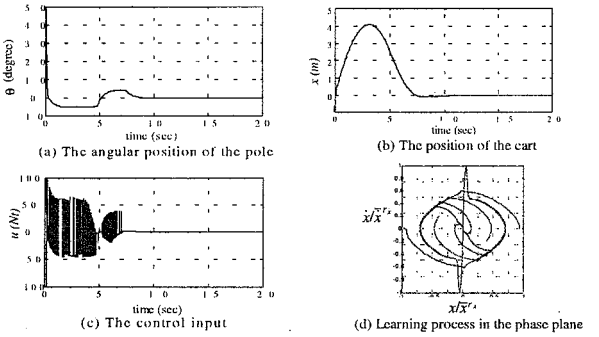

Figure 9: The simulation results for overall controller

We let $O_x = \frac{1}{t_f}\int_0^{t_f} |x|\, dt$ to be the measure as mention above and O_{xd} is given as the permissible bound on O_x. Here, O_x and O_{xd} play the same roles as O_θ and $O_{\theta d}$ in the learning algorithm of previous section. Furthermore, we save the states $(x, \dot{x})$ when the system satisfies the hitting conditions as a set $x_{[0,t_0]}$ which is similar to $\theta_{[0,t_0]}$ in the learning algorithm. With these modifications, we apply LLBFC to train u_x under previous well trained u_θ.

3.3 Simulation for Overall Controller

In this subsection, we apply the overall controller to the inverted pendulum system. With the previous well trained sub-controller u_θ in Subsection 3.2, we will train u_x at different hitting conditions. Now, let the function of hitting conditions be given as follows:

$$s_\theta = \frac{10\theta}{\overline{\theta}^{r_\theta}} + \frac{\dot{\theta}}{\overline{\dot{\theta}}^{r_{\dot{\theta}}}} \text{ and } s_x = \frac{x}{\overline{x}^{r_x}} + \frac{\dot{\theta}}{\overline{\dot{x}}^{r_{\dot{x}}}}$$

$$h_x(\theta, s_\theta, s_x) = \begin{cases} -1, & \text{if } |\theta| \le 0.1,\ |s_\theta| \le 1 \text{ and } |s_x| \ge 0.01 \\ 0, & \text{otherwise,} \end{cases} \quad (5)$$

where r_x and r_{xc} are the numbers of centers of fuzzy sets corresponding to x and $\dot{x}$; $\overline{x}^{r_x}$ and $\overline{\dot{x}}^{r_{\dot{x}}}$ are the maximum values of the centers. The number of the total rules for u_x is equal to $9 \times 9 = 81$. We now set the period of the task, $t_f = 20$ sec. Initial fuzzy rules are also set as to mimic a bang-bang controller. After the first running, we set the centers of input fuzzy sets as $\overline{\dot{x}}^1 = min\{\dot{x}(t), t \in [0, t_f]\}$, $\overline{\dot{x}}^{r_{\dot{x}}} = max\{\dot{x}(t), t \in [0, t_f]\}$, $\overline{x}^1 = min\{x(t), t \in [0, t_f]\}$, and $\overline{x}^{r_x} = max\{x(t), t \in [0, t_f]\}$. Fig. 9 shows the simulation results for the overall fuzzy controller after the task runs 18 times. Fig. 9(b) shows the response of the position of the cart with satisfactory performance. In fact , from Fig. 9(a), the response of the angular displacement for the overall controller, we find it similar to Fig. 7(c), the response of the control input u_θ. This fact illustrates that u_x is similar to an optimal strategy for a well trained human operator. Fig. 9(d) shows the overall learning process in the phase plane (x-$\dot{x}$) for two extreme initial conditions $\theta(0) = 50°$ and $\theta(0) = -50°$.

4 Conclusions

We proposed an intelligent fuzzy controller which incorporates a sequential learning mechanism. The proposed learning algorithm was based on some linguistic model rules and some linguistic criteria. The overall architecture imitated the procedure of controller design generally taken by human being. An example of controlling the inverted pendulum is given to demonstrate the effectiveness of the proposed learning controller. Simulation results showed that the proposed controller not only has the fast convergence in learning algorithm but also has satisfactory performance after sound training of the controller. When being compared with other researches in literature [1]-[7], the proposed learning algorithm only needs to take a few tens of times (16 +18 =34) to complete the process of learning a designated controller and to achieve appealing system performance.

References

[1] Jinwoo Kim and Bernard P. Zeigler, " Design Fuzzy Logic Controllers Using a Multiresolution Search Paradigm", *IEEE Transactions on Fuzzy Systems, vol. 4, no. 3, August 1996*

[2] C. C. Lee, "A Self-Learning Rule-Based Controller Employing Approximate Reasoning and Neural Net Concepts," *International Journal of Intelligent Systems, pp. 71-93, 1991*

[3] A. Pat. and John Prov. " Control of Dynamic Systems Using Fuzzy Logic and Neural Networks," *International Journal of Intelligent Systems, pp.727-748, 1993*

[4] Hamid R. B. and P. Kh. " Learning and Tuning Fuzzy Logic Controllers Through Reinforcements," *IEEE Transaction of Neural Network, pp.724-739, 1992*

[5] Jeffery A. Clouse and P. E. Utgoff, " A Teaching Method for Reinforcement Learning " *Machine Learning Conference, 1992*

[6] D. Whitley, S. Dominic, R. Das, and C. W. Anderson, " Genetic Reinforcement Learning for Neurocontrol Problems," *Machine Learning, pp.259-284, 1993*

[7] A. Barto, R. Sutton aand C. W. Anderson, " Neuronlike Adaptive Elements That Can Solve Difficult Learning Control Problems," *IEEE SMC pp.834-846, 1983*