Novel Fuzzy Logic Controllers with Self-Tuning Capability

F. C. TENG[1]
Department of Electrical & Computer Engineering
Polytechnic University of Puerto Rico, San Juan, Puerto Rico, USA
Email: fchwee@pupr.edu

A. LOTFI
School of Computing & Informatics
The Nottingham Trent University Nottingham, UK
Email: ahmad.lotfi@ntu.ac.uk

A. C. TSOI
Hong Kong Baptist University, Kowloon, Hong Kong
Email: vpra@hkbu.edu.hk

Abstract—**Two controllers which extend the PD+I fuzzy logic controller to deal with the plant having time varying nonlinear dynamics are proposed. The adaptation ability of the first self tuning PD+I fuzzy logic controller (STPD+I_31) is achieved by adjusting the output scaling factor automatically thereby contributing to significant improvement in performance. Second controller (STPD+I_9) is the simplified version of STPD+I_31 which is designed under the imposed constraint that allows only minimum number of rules in the rule bases. The proposed controllers are compared with two classical nonlinear controllers: the pole placement self tuning PID controller and sliding mode controller. All the controllers are applied to the two-links revolute robot for the tracking control. The tracking performance of STPD+I_31 and STPD+I_9 are much better than the pole placement self tuning PID controller during high speed motions while the performance are comparable at low and medium speed. In addition, STPD+I_31 and STPD+I_9 outperform sliding mode controller using same method of comparison study.**

Index Terms— **Self-tuning PD+I fuzzy logic control, output scaling factor, self tuning pole placement PID control, sliding mode control, two-links revolute robot, minimum number of rules.**

I. INTRODUCTION

Many PID-like fuzzy logic controllers have been proposed with various degree of success in the past years [3, 4]. Among them, the PD+I fuzzy logic controller reported in [6] has an interesting design. Their design comprises of a conventional fuzzy PD controller in parallel with a one dimension integral fuzzy controller. The parallel structure has an important advantage because it allows substantial reduction of rule base size. This attractive feature will be further exploited in this paper. In addition, we also aim to extend the capability of the PD+I fuzzy logic controller by including the self tuning features so that plants with time varying nonlinear dynamics can be handled. First proposed self tuning fuzzy logic controller (STPD+I_31) uses rule base structure similar to the original PD+I fuzzy logic controller. Second proposed controller STPD+I_9 is much more efficient than STPD+I_31 because additional constraint is imposed on the size of rule bases. This constraint allows only minimum number of rules in the rule bases. As a result, there are only 3 fuzzy labels for the main PD controller and 2 fuzzy labels for the two auxiliary controllers. Consequently, STPD+I_9 works on *9 rules in total.* To the best of our knowledge, STPD+I_9 is the first working 9 rules adaptive fuzzy logic controller that can be applied to the highly complex robot tracking control problems. How good can this seemingly simple fuzzy logic controller perform? To this end, two classical nonlinear controllers are used as the benchmark for comparison: one is based on parameter based adaptive control and another on sliding mode control theory. They are compared on their tracking performance in controlling a two-links revolute robot with different speed settings. It is well known that the two-links revolute robot is a highly coupled nonlinear system [12] because the inertia loading, the coupling between joints and the gravity effects are highly sensitive to position and velocity conditions. During high speed motions, the inertia loading terms can change drastically. As such, it serves as a good test bed for the comparison of the newly proposed controllers and the benchmark controllers. Extensive simulation results are included to support the performance analysis.

[1] Corresponding author

II. Robot model

Based on Lagrangian-Euler equation, the dynamics of a robot with *n* links is given by

$$T = M(\theta)\ddot{\theta} + V(\theta,\dot{\theta}) + G(\theta) \qquad (2.1)$$

where $M(\theta)$: $n \times n$ mass matrix

$V(\theta)$: $n \times 1$ vector of centrifugal and coriolis terms

$G(\theta)$: $n \times 1$ vector of gravity term

T : External force or torque applied.

The elements in the matrix $M(\theta), V(\theta,\dot{\theta}), G(\theta)$ are highly coupled nonlinear function of the joint configurations, velocity, frictional torque and payload.

The nonlinear robot model is the controllable form of the general nonlinear differential equations expressed as follows:

$$\begin{aligned} \dot{x} &= \tilde{f}(x) + Bu + Z \qquad (2.2) \\ y &= Cx + Du \\ u &= h(w, y) \end{aligned}$$

$\tilde{f}(x)$ is a vector whose elements are nonlinear functions of x, while B, C and D are constant matrices. x is the state vector of dimension (1 x *l*). u is the input vector of (1 x *k*). y is the output vector of (1 x *m*). w is the desired output vector of (1 x *m*). Z is the disturbance vector of (1 x *k*). Although they are function of time, for brevity sake *t* is dropped as the argument of states or functions of states.

In this work, two-links revolute robot is chosen as the target plant to test the controllers. When the robot moves from low to high speed, many parameter values in the linearized discrete time model can go up to 10 times for each link [12]. Moreover, strong interactions between the two links can cause severe additional problems to the motion of each individual joint controller. Refer to [7] for the derivation of the complete model of the two-links revolute robot. Each part of the complete model for the two-link revolute robot is given as follows:

a) $T = [\tau_1 \quad \tau_2]^T$

τ_1 and τ_2 are the torques applied to link 1 and 2 respectively.

b) The mass matrix is given by the following expression:

$$\begin{bmatrix} (m_1+m_2)l_1^2 + m_2l_2^2 + 2m_2l_1l_2C2 & (m_2l_2^2 + m_2l_1l_2C2) \\ (m_2l_2^2 + m_2l_1l_2C2) & m_2l_2^2 \end{bmatrix} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix}$$

c) Expressions for centrifugal and coriolis terms are

$$\begin{bmatrix} 0 & -m_2l_1l_2S2 \\ m_2l_1l_2S_2 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1^2 \\ \dot{\theta}_1^2 \end{bmatrix} + \begin{bmatrix} -m_2l_1l_2S2 & -m_2l_1l_2S2 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1\dot{\theta}_2 \\ \dot{\theta}_2\dot{\theta}_1 \end{bmatrix}$$

d) Expressions of the gravity terms are

$$\begin{bmatrix} (m_1+m_2)gl_1S1 + m_2gl_2S12 \\ m_2gl_2S12 \end{bmatrix}$$

where m_1, l_1 : Mass and length of Link 1

m_2, l_2 : Mass and length of Link 2

III. PID-like fuzzy logic controller

The plant model to be controlled by a PID-like fuzzy logic controller [2] is a second order system with the differential equations

$$\begin{aligned} \frac{d^2x}{dt^2} &= f(\underline{x}, \underline{\dot{x}}, t, u) \\ y &= g(x) \end{aligned}$$

where $\underline{x} = [x_1 \quad x_2]$ is the state vector

u, y are plant input, output

f, g are assumed to be nonlinear functions

The PID-like fuzzy logic controller is designed using a structure similar to the PID controller, thus

$$u = K_p e + K_I \cdot \delta + K_D \cdot \dot{e}$$

where K_p, K_I, K_D are proportional, integral and derivative gain constant.

The fuzzy rules are given as follows:

R^i_{PID} : if (e is LE^i) and ($\dot{e}$ is $L\dot{E}^i$) and (δ is $L\delta^i$) then u is LU^i

where

e : Error defined on the non-normalized domain ε

$\dot{e}$: Change of error defined on the non-normalized domain $\dot{\varepsilon}$

δ : Integral of error defined on the non-normalized domain Δ

u : Controller output defined on the non-normalized domain Ω

LE^i, $L\dot{E}^i$, $L\delta^i$ and LU^i are the fuzzy labels of e, $\dot{e}$, δ and u in the *i-th* rule.

After normalization, the domains of e, $\dot{e}$, δ and u are all normalized onto the same universe of discourse.

$$e = N_e \cdot e$$

$$\dot{e} = N_{\dot{e}}\, \dot{e}$$

$$\delta_N = N_\delta \cdot \delta$$

$$u = N_u \cdot u_N$$

N_e, $N_{\dot{e}}$, N_δ and N_u are scaling factors for error, change of error, integral of error and output signals. Hence, inference and defuzzification functions are performed on this common universe of discourse. Notice that

$$K_p = N_e \cdot N_u \qquad (3.1)$$

$$K_I = N_\delta \cdot N_u \qquad (3.2)$$

$$K_D = N_{\dot{e}} \cdot N_u \qquad (3.3)$$

As reported in [15], the performance of the proposed PID-like fuzzy logic controller is found better than the experienced operators in the start-up operations of the catalytic reactor.

PD-like fuzzy logic controller

Neglecting the integral term, PD-like fuzzy logic controller can be formed. The rule form is simpler as shown below:

R^i_{PD} : if (e is LE^i) and ($\dot{e}$ is $L\dot{E}^i$) then u is LU^i

PI-like fuzzy logic controller

Changing the controller output variable u of R^i_{PD} rules into its derivative $\dot{u}$, we have PI-like fuzzy logic controller with the rule form given as follows:

R^i_{PI} : if (e is LE^i) and ($\dot{e}$ is $L\dot{E}^i$) then $\dot{u}$ is LU^i

PD+I fuzzy logic controller

By having I mode controller in parallel with a PD mode controller, a pseudo PID controller is formed [6]. I mode controller can be implemented by dropping the proportional action in PI controller and generate one dimension rules as shown below:

R^i_{PI} : if ($\dot{e}$ is $L\dot{E}^i$) then $\dot{u}$ is LU^i

Compared to the original PID-like fuzzy logic controller, the use of parallel structure for the PD+I fuzzy logic controller helps to reduce the size of rule base considerably when number of fuzzy labels is high. Notice that this particular structure has an important advantage which allows further substantial reduction of rule base. The details will be described in Section VI.

IV. Tuning of PD+I fuzzy logic controller

In this work, our original version of the PD+I fuzzy logic controller consists of 29 rules: 25 in PD mode rule base and 4 in I mode rule base. It is observed that two fuzzy labels "positive zero" and "negative zero" are used in [6] which in our opinion are redundant. As such, only the merged "zero" fuzzy label is used in our design. The rules are acquired using the verbalization approach [3] in which classical second order system output response is the desired output response.

Similar to the tuning of conventional linear PID controller, the value of N_u for link 1 and link 2 are slowly increased from unity until 1000 and 100. The tracking error decreases significantly with these settings of N_u. Further small reduction of tracking error is possible by making suitable adjustments on the values of N_e, $N_{\dot{e}}$ and N_δ which are set to 3, 0.1 and 1.0 respectively for link 1 controller and 5, 0.01 and 1.8 for link 2.

We conducted a few system identification experiments using recursive least square and found that variation of the static gains for link 1 and link 2 are about 3 and 5 times when the desired reference positions are set at 0.1 and 1.5 radians. The effect of time varying gain was observed when link 2 control loop become unstable after we set the reference trajectories to *(1.2sin5t, 1.2cos5t).* Using smaller value of the gain factor of N_u cannot rectify the system's instability problem. This is because the problem is due to insufficient gain of the main fuzzy controller when the robot is required to operate with bigger range of motion. The stability is recovered after adjusting N_u to the higher values. However, the same set of chosen N_u values causes severe deterioration to the tracking performance at low speed motions due to excessive loop gains. As such, we conclude that the controllers with fixed gain constants are not sufficient for the robot control task. To achieve good dynamic performance, at least two modes of control are needed: a low gain as well as high gain control. When the robot links are moving at high speed with bigger range of motion, high gain controllers should put to work in order to generate sufficient strength of control output signal. On the other hand, the low gain controller should be used when the robot moves slowly near the reference positions. If we are able to expand the two modes control to multiple modes, the control actions will be smoother.

In the limit, a controller that generates infinite number of control modes with gentle transitions should provide the smoothest control actions. Of course, good gain matching between the controller and the robot remains critical in order to maintain satisfactory tracking performance at different motion speed. A controller with such a capability will be described in the next section.

V. Self-tuning PD+I fuzzy logic controller (STPD+I_31)

We propose a self-tuning fuzzy logic controller (STPD+I_31) which enables correct values for N_u to be generated automatically at each sampling instant. This is achieved by sending the controller output signal through a multiplier block as shown in Fig. 1. It is noted that different tuning schemes for the PI or PD fuzzy controllers using non-parallel structure have also been reported [8][16]. Here, we choose tracking error as the auxiliary variable which correlates well with the magnitude of the robot motion range. Based on the analysis described in Section IV, low gain control action should be generated when tracking error is low (at low speed of motions near reference position) and high gain control action when the tracking error is high (at high speed with bigger range of motion). Smooth control actions in between can be generated by the good approximation ability of the fuzzy logic controller [9]. As such, output signal to the multiplier block is determined by the output value of the fuzzy logic controller which uses only two rules (see Table 1).

Table 1
Fuzzy rules of multiplier block

Ave. Abs. Error	**PB**	PB
	NB	NB

Gaussian type membership function is selected for the input variable and triangular type for the output variable. The input to the fuzzy logic controller is the average absolute value of the tracking errors which is computed by finding the mean of the absolute errors taken at previous three sampling instants. The resulting fuzzy logic controller block shown in Fig. 1 can generate a suitable gain value which multiplies the output signal from main PD+I fuzzy logic controller at each sampling instant.

The initial selection of the universe of discourse for the output variable is decided based on the engineering judgment. Since we aim to expand the possible range of N_u, 2 to 3 times the value of non-self-tuning version can be a good choice. Magnitude of N_u depends very much on the magnitude of the controlled variable and static gain value of the plant. In our design, the universe of discourse for link 1 is chosen as (0~1500) and (0~300) for link 2 initially. The selection for the universe of discourse of the input variable is not obvious. So they are all set to unity initially and we found that simple settings chosen so far are good enough to enable the controller to control the robot. To further improve the performance of the controller, the universe of discourse for the input variable is reduced due to small average absolute errors discovered in the simulations. The universe of discourse are set at (0~0.02) for link 1 and (0~0.05) for link 2. These small values help to maintain the crisp output values stay within mid zone of the universe of discourse thereby avoiding frequent extreme values generated at either end. Lastly, in order to track the anticipated large reference trajectory signals, the universe of discourse of the output variable for link 1 is increased to (1000~3000) and (0~800) for link 2.

There are three features that distinguish our work from the automatic tuning mechanism reported in [15]. First, choice of performance index in their work is the value of average squared errors while we use average absolute error. We found that the value of average squared errors is not suitable for the tracking control application as the control objective is to minimize the deviation between the reference and actual trajectory. Good choice of performance index is important since it directs the tuning mechanism of the output scaling factor. Second, the use of the look-up table which relates the performance index to the output to the multiplier block is avoided. The ability of self adaptation that can be derived by the look-up table approach is rather limited because the input/output mapping essentially is a set of coarse linear approximation functions and it is deterministic by nature. Instead, we exploit the fuzzy inference mechanism which follows the inexact but powerful way of human reasoning process. As described earlier in this section, a simple proportional type of fuzzy logic controller with only two rules is proposed for this function. Third, instead of trying to adjust the scaling factors for error and error change signals simultaneously, we choose to adjust only the output scaling factor which has the strongest influence on closed loop dynamics among the three scaling factors. From (3.1), (3.2) and (3.3), it is clear that the adjustment of N_u lead to the alteration of the proportional, integral and derivative gain constant simultaneously. Some preliminary results of this design have been reported in [14].

In summary, STPD+I_31 consist of three separate fuzzy logic controllers: one for the PD mode and one for the I mode and lastly the controller for the automatic tuning of the output scaling factor. The proposed design improves the closed loop performance significantly in terms of tracking accuracy and robustness against the time varying plant parameters. Section VIII describes the detailed evaluation.

VI. Self-tuning PD+I fuzzy logic controller with 9 rules (STPD+I_9)

It is noted that many PID-like fuzzy logic controllers reported earlier tend to use the complete rule base with large number of rules. This may not be the best approach because certain regions of input domain (combinations of error and error change) are of no significance. Besides, it is highly desirable that number of rules in a working controller to be as small as possible. Simple controller structure enables shorter controller execution time thereby allowing the use of small sampling interval. This issue is especially important in real-time implementation using microcontrollers [10] or similar devices which are equipped with slow CPU speed. Generally, microcontroller is also equipped with small memory size and a real-time controller with smaller size of object code can release more computing resources for other essential tasks. From our experiences in implementing the fuzzy logic controller in real-time [13], it was found that it is not necessary to use large number of rules. As a matter of fact, we used a 9 rules fuzzy logic controller to stabilize the double link inverted pendulum which exhibits severe nonlinearity and strong mutual interactions. It is our opinion that that number of rules used in the proposed fuzzy logic controllers reported in [15] and [6] may be excessive. This motivates us to investigate further the possibility of using a more efficient rule base for the PID-like fuzzy logic controllers. In our new design, only 3 fuzzy labels are used consequently 9 rules must be generated in order to cover the entire state space for both e and $\dot{e}$. However, we aim to impose the design constraint which allows only *minimum number of rules* for each individual fuzzy logic controller. Therefore, as shown in Table 2, only five rules are used for PD mode and 2 rules are used for I mode as shown in Table 3.

Table 2
Fuzzy rules of PD controller

	Error change ($\dot{e}$)		
Error (e)	**NS**	**AZ**	**PS**
NS		PS	
AZ	PS	AZ	NS
PS		NS	

Table 3
Fuzzy rules of I mode controller

Error (e)	Output
NB	NB
PB	PB

The automatic tuning of the output scaling factor is achieved by the same fuzzy logic controller adopted by STPD+I_31 . The corresponding rule base uses only two rules, hence the *total number of fuzzy rules for STPD+I_9 is 9* which is substantially smaller than those of STPD+I_31 (31 rules). Notice that no additional tuning work is required because the choice of all controller parameters for STPD+I_9 is same as STPD+I_31.

VII. Design of the pole placement self-tuning PID controller (PP_STC) and sliding mode controller (SMC)

Refer to [11] for the detailed design of the pole placement self-tuning PID controller. Similar to the classical pole placement design, the controller tuning parameters damping factor ξ and natural oscillation frequency ω_n affect the tracking performance. We adjust these two factors so that the robot performs well from low to high speed motion. The values of (ξ, ω_n) are set at (0.7, 10) for link 1 and (1.0, 10) for link 2. For sliding mode controller, refer to [5] for the detailed analysis. Tuning of each controller is carried out by adjusting the values of the lower bound of B, i.e. B_0 and the upper bound of $\tilde{f}(x)$, i.e. $\tilde{f}_0(x)$ in (2.2) so that good tracking performance is achieved from low to high speed motion. To avoid chattering problems, the ideal relay is replaced by the saturation limiter. The boundary zone is chosen as +/- 0.5 for link 1 and +/- 0.8 for link 2. Other parameter values are chosen as follows, $B_0 = 0.5$ (Link1), $B_0 = 0.25$ (Link2), $\tilde{f}_0(x) = 100$ for both links. The sliding variable s is in the form of $\dot{\varepsilon}(t) + 5\varepsilon(t)$.

VIII. Simulation results

All simulations are conducted using MATLAB Simulink. The Fuzzy Logic Toolbox provided by MATLAB is a useful tool for the implementation of the fuzzy logic controllers proposed in this work. The Self-Tuning Toolbox included in [1] is used in the implementation of PP_STC. In the case of sliding mode controller, it is constructed by means of the basic Simulink library blocks. The simulation step size is fixed at 0.01(sec) and the numerical integration algorithm used in all simulations is "ODE4 (Runge-Kutta)". Initial positions of link 1 and link 2 are set at starting values of the reference trajectory in all simulations. Notice that decentralized control strategy is adopted in all the simulations. Physical parameters of the two-links revolute robot model used in the simulations are given as follows: Mass and length for both links are 2 kg and 0.5 m.

The time varying static gains and the disturbances due to coupled centrifugal and coriolis forces in (2.1) create unfavorable working conditions for the controllers. In fact, the original PD+I fuzzy logic controller fails to control the robot when the maximum amplitude of the reference trajectories is set higher than 1.1 radians. Fig. 2

Table 4
Tracking performance of STPD+I_31 and the original version

	Low speed (Ref.:*0.1sin0.1t*)		**Medium speed** (Ref.: *0.5sin0.5t*)		**High speed** (Ref. : *0.5sin5t*)	
	Link 1	*Link 2*	*Link 1*	*Link 2*	*Link 1*	*Link 2*
C1	69	510	13200	13420	46200	117200
C2	43	150	8200	5310	32300	26300

Notes: C1: Original version; C2: STPD+I_31; The multiplier of all numerical values is 10^{-6}

Table 5
Tracking performance of STPD+I_31, STPD+I_9, PP_STC, SMC under low & medium speed motion

	Low speed (Link 1 : *0.1sin0.1t*) (Link 2 : *0.1cos0.1t*)		**Medium speed** (Link 1: *0.5sin0.5t*) (Link 2: *0.5cos0.5t*)	
	Link 1	*Link 2*	*Link 1*	*Link 2*
STPD+I_31	0.000034	0.000137	0.0053	0.0164
STPD+I_9	0.000132	0.000568	0.0202	0.0511
PP_STC	0.0000069	0.0000335	0.00114	0.00878
SMC	0.0310	0.0567	0.4968	3.50
Performance ratio (PP_STC)	0.20	0.24	0.24	0.53
Performance ratio (SMC)	911	413	93	213

shows that link 2 is out of control when a reference trajectory of *1.2sin5t* is applied. It occurs not long after the simulation begins.

We compare the original fuzzy PD+I controller with our self tuning version STPD+I_31. Table 4 shows the results of tracking performance. It is clear that STPD+I_31 can achieve more than the original version whether at low, medium or high speed motion. Notice that when strong disturbances are experienced on link 2 at high speed motions, tracking error is not more than 17% of the original version.

In contrast to the rest of the controllers, the pole placement self-tuning controller requires a start-up period to generate correct plant parameter estimates. As such, meaningful comparison can only made by not using the readings of tracking errors during the initial period of operation. Therefore, all the readings shown are the integral of square errors in the final 20 seconds of simulation. Due to high nonlinearity and strong interactions experienced by both links of the robot, the tracking position errors become larger as the robot moves faster. Tables 5 and 6 summarize the results when reference trajectories of $A\sin\omega t$ and $A\cos\omega t$ are applied to link 1 and link 2 respectively.

Table 6
Tracking performance of STPD+I_31 STPD+I_9 PP_STC SM under high speed motion

	High speed (Link 1: *0.5sin5t*) (Link 2: *0.5cos5t*)		**High speed with higher peak value** (Link 1: *1.5sin5t*) (Link 2: *1.5cos5t*)	
	Link 1	*Link 2*	*Link 1*	*Link 2*
STPD+I_31	0.1370	0.003358	0.6635	0.1253
STPD+I_9	0.0651	0.0648	0.5670	0.4541
PP_STC	0.1410	6.50	3.38	51.02
SMC	9.88	42.3	81.76	104.35
Performance ratio(PP_STC)	1.02	1935	5.12	407
Performance ratio (SMC)	72	12596	123	832

After each round of simulation, integral of square errors for the closed loop system with STPD+I_31, PP_STC and the SMC are computed. The ratio between them is also computed and listed as "Performance ratio" in each table. Performance ratio having a numerical value less than unity indicates that PP_STC or SMC performs better than STPD+I_31. As shown in Table 6, the performance ratios for PP_STC versus STPD+I_31 are 0.2 for link 1 and 0.24 for link 2 at low speed motion hence PP_STC performs slightly better than STPD+I_31. At medium speed, readings recorded are 0.24 for link 1 and 0.53 for link 2 which indicates similar level of performance. The situation is drastically different at high speed motions. The readings are 1.02 for link 1 and 1935 for link 2 (Table 7). It is clear that tracking performance of PP_STC is much inferior than STPD+I_31. When the reference trajectory maximum amplitude is further increased to 1.5 radians, the readings are 5.12 and 407 respectively. Therefore, it can be concluded that in comparison with STPD+I_31, PP_STC is not a good choice for tracking control of two-links revolute robot as such robots often operate at high speed motions in practice. Results from Tables 6 and 7 also indicate STPD+I_9 are better than PP_STC at high speed motion though the margin of improvement is not as high as STPD+I_31. Results from Tables 6 and 7 indicate that the tracking performance is unsatisfactory in all cases for SMC in comparison with STPD+I_31, STPD+I_9 as well as PP_STC. The tracking performance cannot improve further in spite of the long and tedious tuning work.

The tracking performance of the four controllers can be compared vividly by examining the output trajectories in the form of X-Y graph. Figures 3, 4, 5 and 6 shows the X-Y graph of STPD+I_31, PP_STC, SMC and

STPD+I_9 for the final 10 seconds of simulation with reference trajectories set to *1.5sin5t* and *1.5cos5t***.** It is observed that the output trajectories of STPD+I_31 and STPD+I_9 can follow the reference trajectories closely and consistently whereas the other two controllers display multiple traces with different loci during each cycle of motion. Figs 4 and 5 clearly show that there are wide and irregular deviations between the reference trajectories and output trajectories for PP_STC and SMC.

Referring to Tables 5 and 6, it is observed that the tracking performance of STPD+I_9 in general is below that of STPD+I_31 though they are not far off. This outcome is not surprising since STPD+I_9 supposed to be inferior in function approximation ability due to smaller rule base size. However, referring to Table 7, we are caught by surprises that the performance of STPD+I_9 is actually better than STPD+I_31 for link 1 when reference trajectories are (*1.5sin5t*, *1.5cos5t).* This is unexpected since under identical working conditions and with much smaller number of rules, STPD+I_9 is not supposed to perform better than STPD+I_31.

As expected, STPD+I_31 and STPD+I_9 should have better stability margin than the original version which fails to keep the stability when reference trajectories are set higher than *1.1sin5t* (see also Fig.2). Additional simulations show that the current configuration of STPD+I_31 is capable of handling reference trajectories up to (3*sin5t*, *3cos5t)* or (*1.1sin13t*, *1.1cos13t)* with no loss of stability. The results for STPD+I_9 are (3.8*sin5t*, *3.8cos5t)* and (*1.1sin13t*, *1.1cos13t).* Again, notice that the maximum working range of STPD+I_9 is wider than STPD+I_31 (3.8 versus 3.0).

Although both set of data are obtained during the high speed motions, whether they are caused by the same mechanism remain unclear. Finding the explanation to these two interesting observations shall be the work of future investigations.

IX. Conclusions

This paper describes the detailed design of two self-tuning PD+I fuzzy logic controllers. The second proposed self-tuning PD+I fuzzy logic controller uses only 9 rules. The controllers can work on the plant with time varying nonlinear dynamics. The extended self-tuning PD+I fuzzy logic controllers are applied to the highly coupled two-links revolute robot. It is found that the tracking performances of both controllers outperform the pole placement self-tuning PID controller by a big margin at high speed motions and the performance are comparable at low and medium speed motion. From the simulation study, it is also found that classical sliding mode controller causes large tracking errors compared to the two self-tuning PD+I fuzzy logic controllers.

REFERENCES

[1] V. Bobal, J. Bohm, J. Fessl & J. Machacek, *Digital Self-Tuning Controllers,* New York: Springer-Verlag, 2005.
[2] D. Driankov, H Hellendoorn, & M Reinfrank, *An Introduction to Fuzzy Control,* New York: Springer-Verlag, 1993.
[3] C. J. Harris, C. G. Moore & B. Brown, *Intelligent Control*, Singapore: World Scientific, 1993.
[4] N. A. Johnson & M. H. Moradi, *PID Control: New Identification and Design Method*, New York: Springer-Verlag, 2005.
[5] H. K. Khalil, *Nonlinear Systems*, New York: Prentice Hall, 2000.
[6] D. P. Kwok, P. Tam, C. K. Li, & P. Wang "Linguistic PID Controllers,", *Proc of IFAC Triennial World Congress*, Tallinn, Estonia, pp 205-210, 1990.
[7] Zhilong Man, *Robotics*, Singapore: Prentice Hall, 2005.
[8] R. K. Mudi & N. R. Pal "A Robust Self Tuning Scheme for PI and PD Type Fuzzy Controller," *IEEE Trans on Fuzzy Systems*, Vol 7, No 1, pp 2-16, 1999.
[9] K.M. Passino & S. Yurkovich *Fuzzy Control,* USA: Addison-Wesley, 1998.
[10] F. C. Teng, W. K. Ow-Yong, P. Y. Chan & C. K. Chan "An 8052-Based Digital Controller with Real Time Programming," *J of Microcomputer Applications*, Vol 13, pp 291-304. 1990.
[11] F. C. Teng, "Investigation of the Enlarged Least Square Estimator Combined with a Classical Controller," *Trans. Inst. of Measurement and Control*, Vol 12, No 4, pp 224-228. 1990.
[12] F. C. Teng, G. F. Ledwich & G. Shannon. "Adaptive Control Scheme for Robot Manipulator: Direct Decoupler and PID Controller," *Int J of Systems Science*, Vol 24, No 2, pp 315-327, 1994.
[13] F. C. Teng, H. F. Fang, W. Y. Tan & A. C. Tsoi "Real-Time Fuzzy Logic Controller of Double-Link Inverted Pendulum," *Proc 5th International Conference on Control, Automation, Robotics and Vision* (ICARCV'98), pp 1127-1131 (Invited paper), 1998.
[14] F. C. Teng, A. Lotfi & A. C. Tsoi "Self-Tuning PD+I Fuzzy Logic Controller With Minimum Number of Rules," *Proc IEEE Systems, Man & Cybernetics*, Montreal, Canada, 2007, pp 865-870, 2007.
[15] Y. Yamashita, M. Matsumoto & M. Suzuki 1988 "Start-Up of a Catalytic Reactor by Fuzzy Controller," *J of Chemical Engineering of Japan*, pp 277-282, 1988.
[16] Y. Zhao & E. G. Collins "Fuzzy PI Control Design for an Industry Weigh Belt Feeder," *IEEE Trans on Fuzzy Systems*, Vol 11, No 3, pp 311-319. 2003.

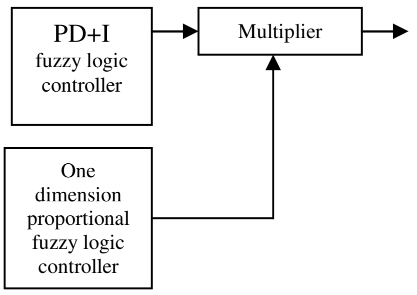

Figure 1. Self tuning mechanism

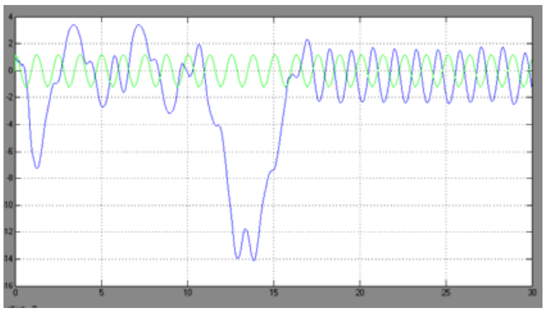
Figure 2. Link 2 is out of control when reference trajectories (*1.2sin5t*, *1.2cos5t)* are applied to the original version of fuzzy logic controller

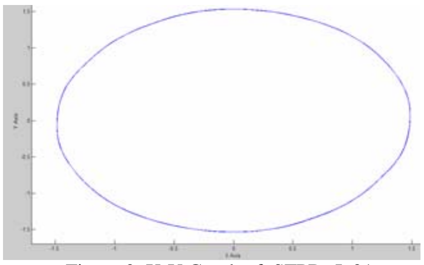
Figure 3. X-Y Graph of STPD+I_31

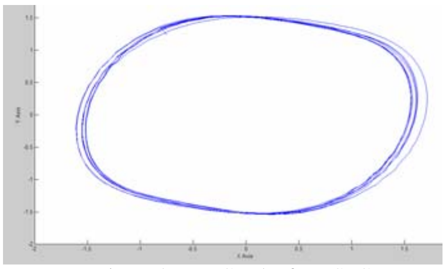
Figure 4. X-Y Graph of PP_STC

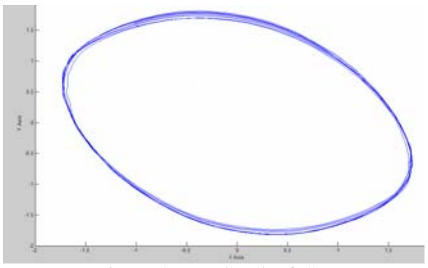
Figure 5. X-Y Graph of SMD

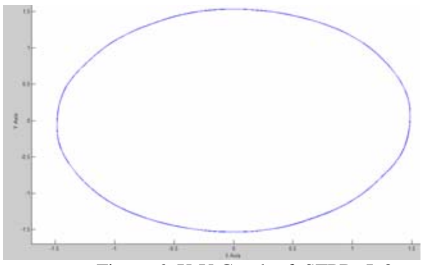
Figure 6. X-Y Graph of STPD+I_9

F. C. TENG was born in Singapore and received his PhD (1993) in the field of Electrical & Computer Engineering from the University of Queensland, Australia. Prior to that, he received BSc(Hons) (1978) from the Sunderland Polytechnic (Now University of Sunderland), UK and MEng (1984) from the University of Canterbury, New Zealand. In 2006, he joined the Department of Electrical & Computing Engineering, Polytechnic University of Puerto Rico, San Juan, Puerto Rico, USA where he is now an Associate Research Professor in Control & Robotics. He teaches master courses related to advanced control including “Intelligent Control”, “Multivariable control”, “Adaptive Control”, “Advanced Control of Robotic Manipulator” and “Nonlinear Control”. He is the author of 26 peer-reviewed international journal and conference papers; several of which attracted 70 citations. His main research activities focus on the applications of advanced controller design such as parameter adaptive control, fuzzy control, neural network control, multivariable control and real-time computer control. Dr TENG has served as a keynote speaker, journal reviewer and member of program committees in several international conferences.

A. LOTFI received his BSc and MTech. in control systems from Isfahan University of Technology, Iran and Indian Institute of Technology, India respectively. He received his PhD degree in Learning Fuzzy Systems from University of Queensland, Australia in 1995. He is currently a senior lecturer in School of Science and Technology, Nottingham Trent University, UK. He is the group leader for Ambient and Computational Intelligent research group. Dr LOTFI is the author of over 60 scientific papers in the area of computational intelligent and control. His main scientific research interest includes, intelligent control, computation intelligence, fuzzy logic and systems and intelligent data analysis.

A. C. TSOI studied Electronic Engineering at the Hong Kong Technical College (graduated in 1969); Electronic Control Engineering (graduated in 1970) and Control Engineering (graduated in 1972) at University of Salford, England. Since graduation, he worked as a post doctoral fellow at the Inter-University Institute of Engineering Control at University College of North Wales, Bangor, North Wales, a lecturer at Paisley College of Technology, Paisley, Scotland, before emigrating to New Zealand. He worked as a Senior Lecturer in Electrical Engineering in the Department of Electrical Engineering, University of Auckland, before emigrating to Australia. He worked as a Senior Lecturer in Electrical Engineering, University College University of New South Wales for 5 years. He then served as Professor of Electrical Engineering at University of Queensland, Australia; Dean, and simultaneously Director of Information Technology Services, and then foundation Pro-Vice Chancellor (Information Technology and Communications) at University of Wollongong; before joining the Australian Research Council as an Executive Director, Mathematics, Information and Communications Sciences. He was Director, Monash e-Research Centre, Monash University in Melbourne, Australia. In April 2007, he took up the position of Vice President (Research and Institutional Advancement), Hong Kong Baptist University, Hong Kong. He works in the area of artificial intelligence in particular neural networks and fuzzy systems in recent years. He has published in neural network literature. In more recent years, Professor TSOI works in the application of neural networks to graph domains, with applications to the world wide web searching, and ranking problems, sub-graph matching problem.