Development of Fuzzy-logic-based Self Tuning PI Controller for Servomotor

Oyas Wahyunggoro and Nordin Saad
Department of Electrical and Electronics Engineering,
Universiti Teknologi
PETRONAS
Malaysia

1. Introduction

Servomotors are used in a variety of applications in industrial electronics and robotics that includes precision positioning as well as speed control. Basically, any motor can be used in a servo system (Kissell, 2002). There are two types of motors: (a) DC motors, and (b) AC motors. DC motors have better starting torque than AC motors although they are more expensive than AC motors. Servomotors use feedback controller to control the speed or the position, or both. The basic continuous feedback control is PID controller. The PID controller posses good performance but is not adaptive enough. This is appealing when the load is changed, where the original controller generally cannot maintain the design performance and thus should be re-designed for the new system conditions (Shieh & Li, 1998).

The pioneering work dealing with expert knowledge that can be well applied to the control of systems with uncertained, nonlinear dynamics is credited to Zadeh (Zadeh, 1968) who proposed fuzzy control theory to overcome the weakness of conventional controllers, and investigated by which owns good robustness (Yu *et. al.*, 2004). Fuzzy systems are capable of handling complex, non-linear and sometimes mathematically intangible dynamic systems using simple solutions. Fuzzy logic uses human-like but systematic properties of converting linguistic control rules based on expert knowledge into automatic control strategies. But, the response of a fuzzy logic controller is slower than a PID controller. It has been reported in a number of papers that hybrid of PID or PI, with fuzzy logic in control system can overcome the set-back of fuzzy logic controller, see (Yeh & Tsao, 1994), (He *et. al.*, 1993), (Ga & Feng, 2005) and (Jee & Koren, 2004). In fuzzy systems, the numerical input values should be first converted into the corresponding fuzzy representations by using 'fuzzifiers'. The fuzzy output are then provided by a fuzzy model, which could be a set of fuzzy logic rules, fuzzy relations or even a simple fuzzy table, with or without deep fuzzy reasoning. Finally, the fuzzy output can be converted back into their relevant numerical (crisp) output through 'defuzzifiers' (Lu, 1996). Basic configuration of fuzzy systems with fuzzifier and defuzzifier is shown in Figure 1.

Singleton fuzzifier maps a real-valued point $x^* \in U$ into a fuzzy singleton A' in U, which has membership value 1 at x* and 0 at all other points in U (Wang, 1997). If $\bar{y}^l$ is the center of the $l'th$ fuzzy set and w_l is its height, then the center of average defuzzifier determines y* as (Wang, 1997).
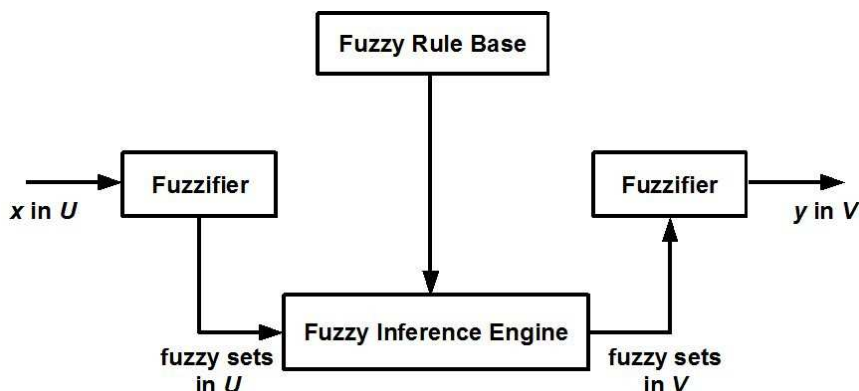
Fig. 1. Basic configuration of fuzzy systems with fuzzifier and defuzzifier (Wang, 1997)

$$y^* = \frac{\sum_{l=1}^{M} \overline{y}^l w_l}{\sum_{l=1}^{M} w_l} \tag{1}$$

A study on induction machine speed control proved that PID controller's parameters can be tuned on-line by an adaptive mechanism based on a fuzzy logic (Bousserhanel *et. al.*, 2006). It is expected that hybrid of PID and fuzzy logic in control system can overcome the problem of fuzzy logic controller.

Servomotor controllers need optimization to give a good performance as desired. Sometimes, it requires time and has big risk in the optimization process. To investigate this issue, a detail study on servomotor control is conducted. A real plant, constituting of a DC motor and its controller has been built and this system is modeled and simulated to allow detail analysis about its control system. The system identification problem is to estimate a model of a system based on the observed input-output data (Takami & Mahmoudi, 2007). The expected output of identification is an s-domain model of a real system. System identification technique is used to get a transfer function of the plant and is used to build a virtual controller, which is basically the software equivalent of the real controller at the abstraction level. The virtual controller is optimized, and then the optimized parameters are applied to the real controller in the real control system.

Therefore, this work discusses the modeling, simulation and hardware implementation of a DC servomotor controller built using MATLAB/Simulink, and the analysis of controller's performance, namely a PID controller, PI controller, fuzzy logic controller, fuzzy-logic-based self tuning PI controller, and a fuzzy-logic-based self tuning PID controller on the system. In this work the aim is to improve the controller's performance using hybrid fuzzy and PID. Figure 2 outlines the background and purpose of this work.

2. The experimental DC servomotor system

The experimental rig constituiting the servo speed and position controller system consists of the servomotor and load, measuring and controlling devices. The servo system contains a DC motor driven by an IGBT chopper inverter. The measuring device is the speed sensor (tachogenerator), ADC and a digital filter i.e., finite input response (FIR), while the controlling

devices are DAC, differential amplifier, and the IGBT inverter circuit. The measuring devices provide status of the output responses of the speed and position where the information about the speed and position is fed through signal conditioning circuit and anti-aliasing filter for analysis and calculation of the control signal. The speed and position requirements proportional to the manipulated variable of the controller's output are fed to a computer.
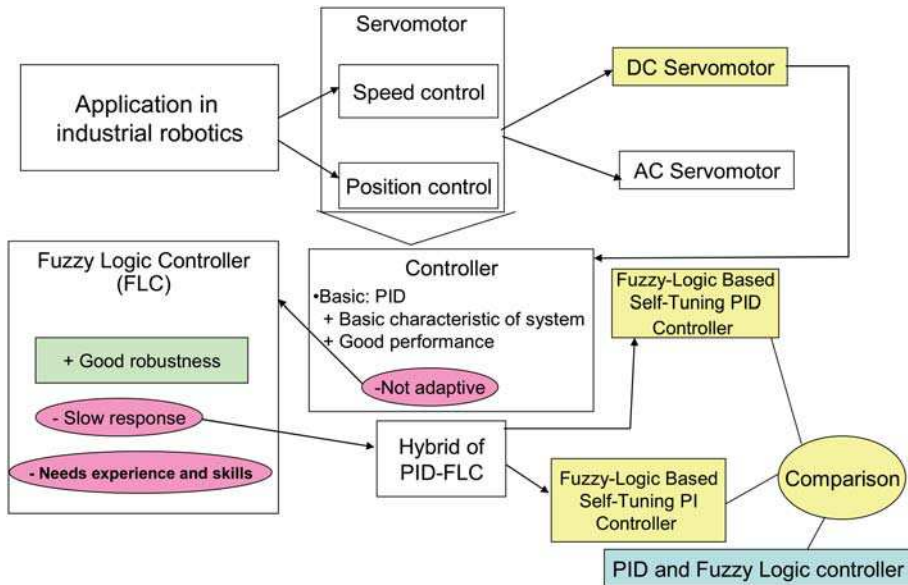

Fig. 2. Overview of the study background

The modeling and analysis of the DC servomotor for the speed and position was conducted using Simulink, with the block diagram is as shown in Figure 3. In this arrangement, the speed control loop is in the position control loop (Lacevic *et. al.*, 2007). Basically, the position control loop work until reaching the position set point while the speed is limited to the speed set point. Position control systems with conventional position sensor have well-known drawbacks. Notably, friction, stiction, backlash, and hysteresis limit the positioning accuracy. This would lead to mechanical wear that introduces impurities in the form of dust particles into the manufacturing environment. The other consequence would be mechanical coupling transmits to the microstepper vibrations from the surrounding environment, (Fulford *et. al.*, 2009). Notably, in this experimental set-up the position control is sensorless. Practically, position control is preferred to be sensorless to reduce cost and size and increase the reliability of overall system (Montanari *et. al.*, 2007).

The input, feedback, and output elements for position and speed are implemented in the Simulink diagram with specifications as follows:

A_P=1
A_v=0.002
K_v=9.5455
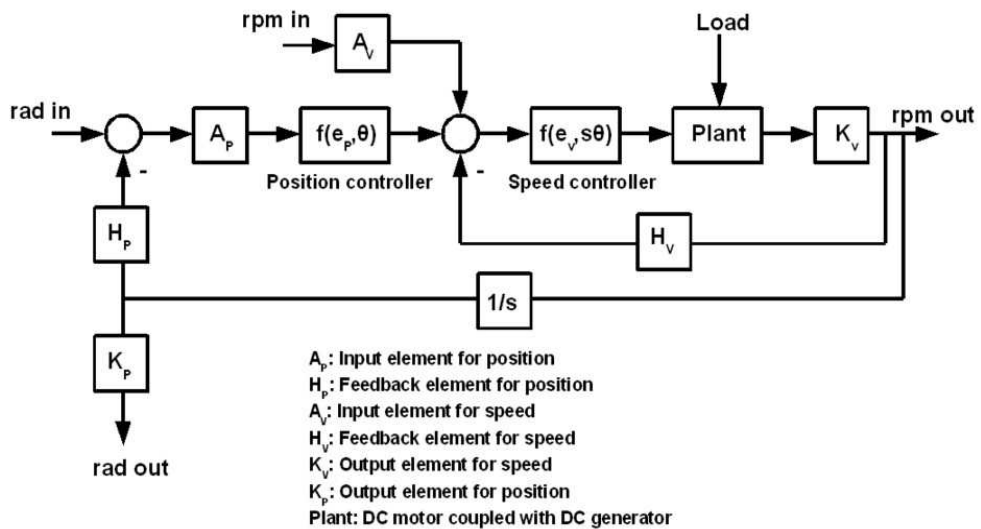H_v=0.002
H_P=0.005
K_P=0.005

Fig. 3. Structure of feedback controller (Lacevic *et. al.*, 2007)

These specifications have been based on realistic assumptions.
The hardware implementation for the block diagram is as follows:

> DC motor: 175 W, 1500 rpm, 240 V, 1.1 A
> Load: Dynamometer load controller
>
> Input elements:
> Tacho-generator with 500 rpm/volt
> ADC: 1 channel 0 to 10 volts
> Filtering: FIR with 30 points
> Output elements:
> DAC: 2 channels 0 to 4 volts
> Differential amp: HA-17741
> Power amp: Chopper/Inverter and IGBT
>
> Controller elements:
> Computer with Intel Pentium Dual Core T2080 processor, Windows XP SP3,
> MATLAB/SIMULINK software.

The block diagram of hardware design is shown in Figure 4, and the power amplifier and differential amplifier are presented more detail in Figures 5 and 6.
As an illustration to substantiate the applicability of this approach, the following test parameters have been chosen. The set point for position is 5 rad, and the set point for speed equals 250 rpm. The load is initially at 0 Nm and then abruptly increases to 1 Nm after about 15 sec. The total duration of experiment is 90 sec. The speed of DC motor is detected by a tacho-generator, sampled every 0.01 sec, and filtered by 30-point FIR.
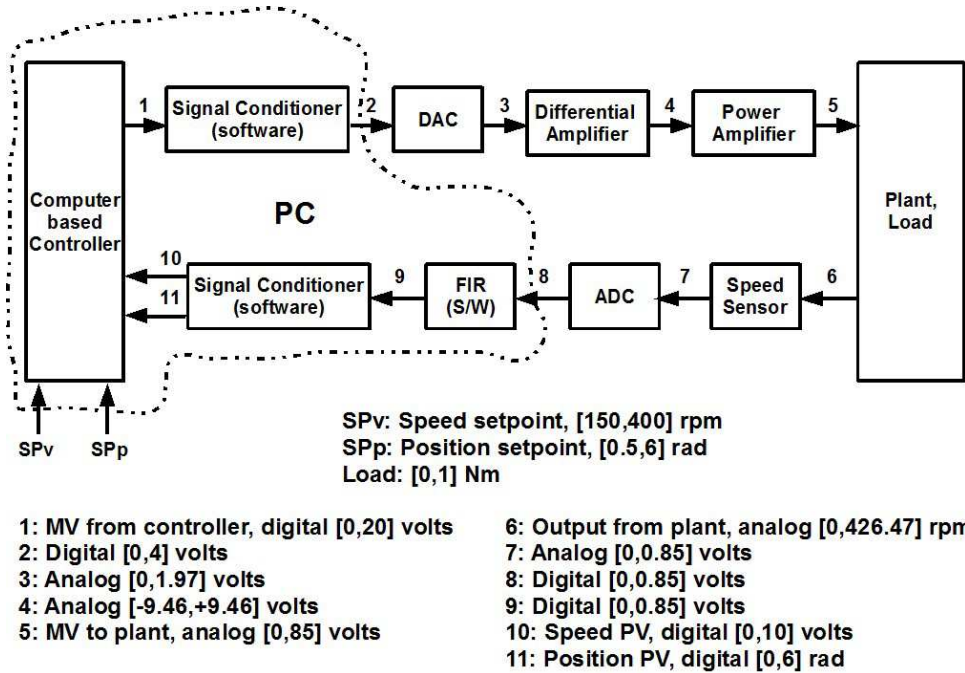
SPv: Speed setpoint, [150,400] rpm
SPp: Position setpoint, [0.5,6] rad
Load: [0,1] Nm

1: MV from controller, digital [0,20] volts
2: Digital [0,4] volts
3: Analog [0,1.97] volts
4: Analog [-9.46,+9.46] volts
5: MV to plant, analog [0,85] volts

6: Output from plant, analog [0,426.47] rpm
7: Analog [0,0.85] volts
8: Digital [0,0.85] volts
9: Digital [0,0.85] volts
10: Speed PV, digital [0,10] volts
11: Position PV, digital [0,6] rad

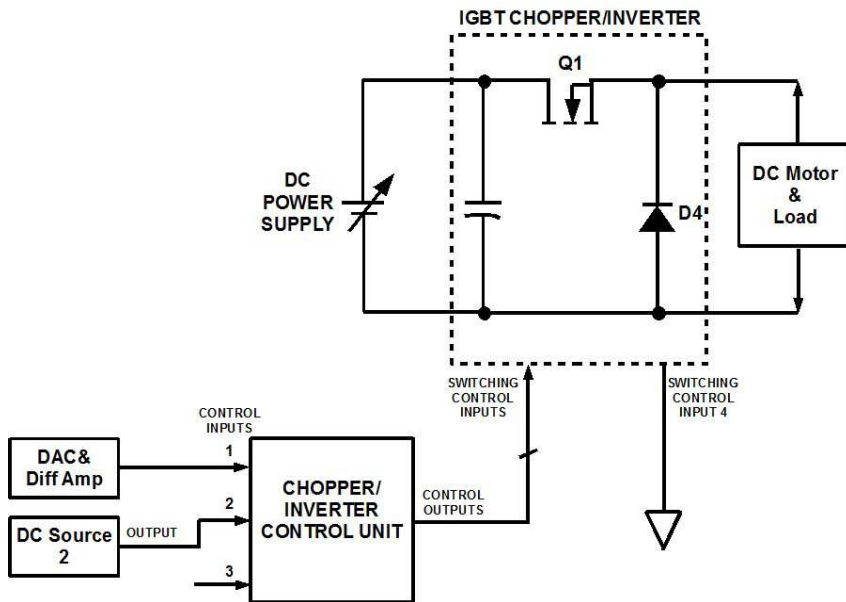Fig. 4. Block diagram of hardware design

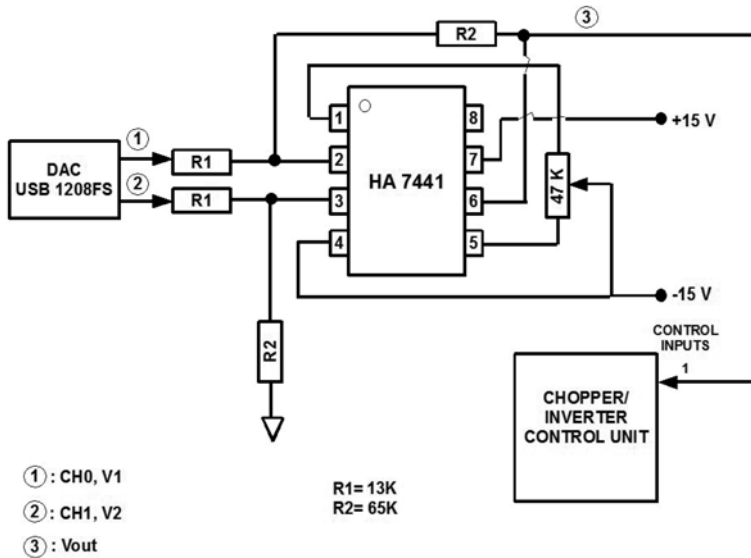Fig. 5. Power amplifier circuit diagram

Fig. 6. Differential amplifier circuit diagram

3. Experimental procedures

3.1 The identification process

The typical identification process consists of stages where the model structure is iteratively selected, computed and updated for the best model in the structure, and finally the evaluation of the model's properties (Takami & Mahmoudi, 2007). The steps can be itemized as follows:

a. Design an experiment and collect input-output data from the process to be identified.
b. Examine the data. Polish the data by removing trends and outliers, and select useful portions of the original data. Apply filters to the data to enhance important frequency ranges.
c. Select and define a model structure (a set of candidate system descriptions), within which a model is to be found.
d. Compute the best model in the model structure according to the input-output data and a given criterion for goodness of fit.
e. Examine the properties of the model obtained. If the model is good enough, then stop; otherwise go back to step c to try another model structure. Attempt other estimation methods (step 4), or work further on the input-output data (steps a and b).

3.1.1 Process I: Open loop analysis

Open loop characteristic of the gray box is tested using Simulink, which is applied to the plant through a data acquisition (DAQ), 30-point FIR, chopper/inverter control unit, and IGBT. The input voltage is varied randomly in the range of 0 to 20 volts for 100 seconds. The output of the gray box model is in rad/sec. There are five variations of input, and the characteristic of one of the input-output is shown in Figure 7.

3.1.2 Process II: Best model selection

There are six process models that can represent the system. To select the most suitable model, these models will be analysed and compared:

a. Process model using 1st order transfer function without zero,
b. Process model using 1st order transfer function with zero,
c. Process model using 2nd order transfer function without zero,
d. Process model using 2nd order transfer function with zero,
e. Process model using 3rd order transfer function without zero, and
f. Process model using 3rd order transfer function with zero.

The DC motor is operated for 150 sec with variations of input voltages as shown in Figure 7. The sampling time is 0.01 sec. The result (output voltage) and the input are then saved in workspace. The data for identification is obtained from the MATLAB using the command as below:

```
data=iddata(workspace_output,workspace_input,0.01);
```

Using MATLAB command, the general command to get the process model is as follows:

```
model=pem(data, 'PnDZU');
```

where:

P : (required) for process model
n : 0,1,2, or 3 (required) for the number of poles
D : (optional) to include a time-delay term
Z : (optional) to include a process zero (numerator term)
U : (optional) to indicate possible complex-valued (underdamped) poles

Fig. 7. Graphical process of identification

To select the best process model, the fitness value has to be obtained. If i_FIT is the fitness value for identification process, Y is the real (measured) output, $\hat{Y}$ is the estimated output,

and $\bar{Y}$ is the mean value of real output, then the fitness value can be obtained from the formula below, see (Montanari *et. al.,* 2007):

$$i\_FIT = \left(\frac{1 - NORM\left(Y - \hat{Y}\right)}{NORM\left(Y - \bar{Y}\right)} \right) * 100 \tag{2}$$

3.2 The controllers
3.2.1 Conventional controllers
In this work the following two conventional controllers, namely the PI and PID are used. Each controller is optimized using Ziegler-Nichols (ultimate cycle) method. If K_{Pu} is the minimum value of K_P resulting undamped oscillation and T_u is the oscillation period, then

$$K_P = 0.45 * K_{Pu}$$
$$K_I = \frac{1}{0.83 * T_u} \tag{2a}$$

for the PI controller, and

$$K_P = 0.6 * K_{Pu}$$
$$K_D = \frac{T_u}{8} \tag{2b}$$
$$K_I = \frac{2}{T_u}$$

for the PID controller.

3.2.2 Fuzzy logic controllers
In this work, a fuzzifier is a singleton mode with two inputs consisting of error and change of error, each with seven uniform triangular membership functions, and one output with seven triangular membership functions. As an inference engine, the Mamdani product is used and as a defuzzifier, the center of average is used. The controllers use the rules as shown in Table I.

E, error	D, change of error						
	NB	NM	NS	Z	PS	PM	PB
NB	NB	NM	NM	NM	NM	NM	NM
NM	NM	NM	NM	NM	NM	NM	NM
NS	NS	NS	NS	NS	NS	NS	NS
Z	Z	Z	Z	Z	PS	PM	PB
PS	Z	PS	PM	PB	PB	PB	PB
PM	Z	PS	PM	PB	PB	PB	PB
PB	Z	PS	PM	PB	PB	PB	PB

Table I. The rules for FLC

NB: Negative Big; NM: Negative Medium; NS: Negative Small Z: Zero; PS: Positive Small; PM: Positive Medium; PB: Positive Big

The fuzzy sets and their corresponding membership functions for input (error, and change of error) are shown in Figure 8, and for output are shown in Figure 9.
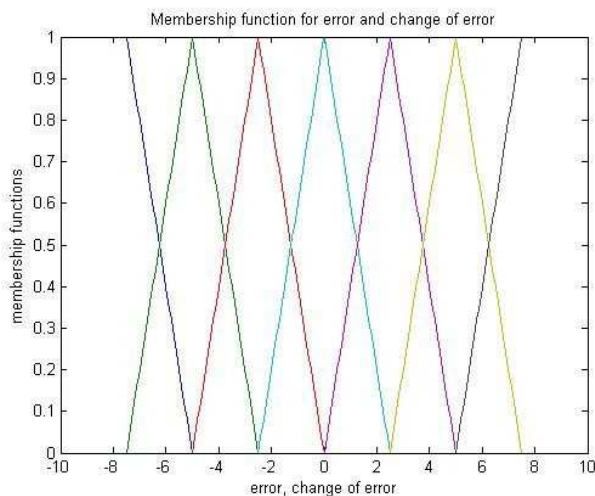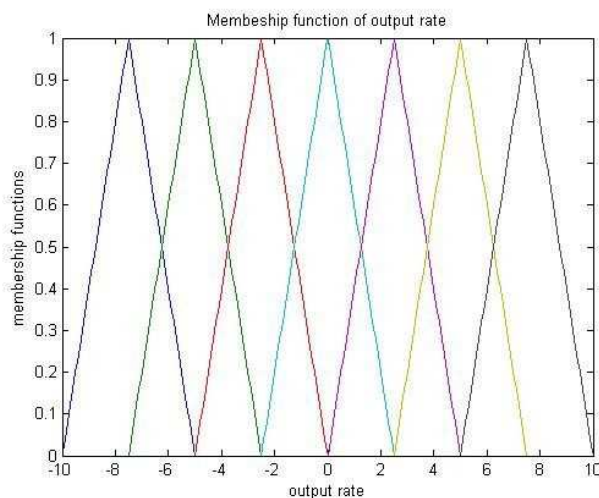

Fig. 8. Fuzzy sets for input

Fig. 9. Fuzzy sets for output rate

3.2.3 Hybrid controllers
There are two kinds of hybrid controllers that be the focus of this work: FLBPI and FLBPID. FLBPID uses the basis of FLBPI with additional of fixed value of K_D. The block diagram of fuzzy-logic-based self tuning PI for speed controller is shown in Figure 10. In this figure, $\omega_m(k)$ is the process value of speed and $\omega_m^*(k)$ is the setpoint of speed.
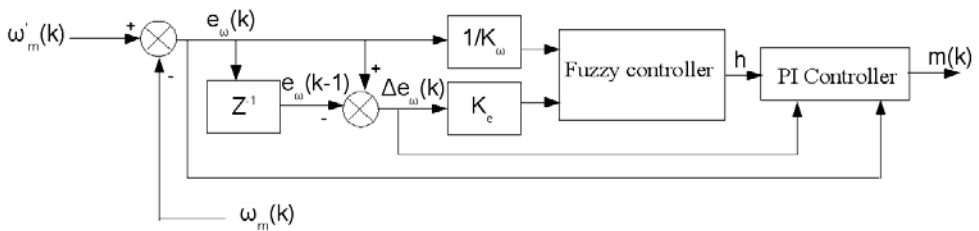
Fig. 10. Block diagram of fuzzy-logic-based self-tuning PI for the speed controller [11]

The fuzzy sets and their corresponding membership functions for input (error, $e_\omega(k)$ and change of error, $\Delta e_\omega(k)$) and output (h) are shown in Figure 11. The rules for FLBPI and FLBPID (K_P and K_I) are shown in Table II. The value of K_D is constant based on Ziegler-Nichols method in PID tuning.
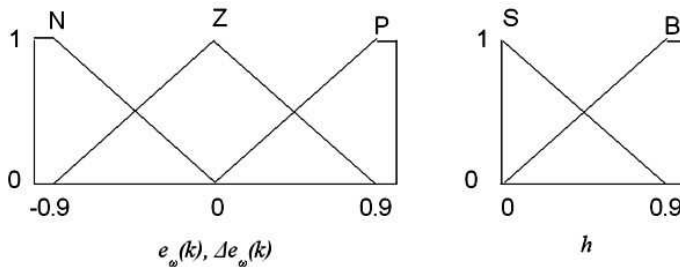
Fig. 11. Fuzzy sets and their corresponding membership functions, (Mannan *et. al.,* 2004).

Change of Error	Error		
	N	Z	P
N	S	B	S
Z	S	B	S
P	S	B	S

N: Negative, Z: Zero, P: Positive, S: Small, B: Big

Table II. Fuzzy Rules Base for K_P and K_I in Hybrid (Mannan *et. al.,* 2004).

The values of K_P, K_I, and K_D for hybrid controllers are obtained from the following formula:

$$K_p = h.K_{Pm} ; K_I = h^2.K_{Im} ; K_D = K_{Dm} \tag{3}$$

Proportional based controller is used for the position control since the requirement is such that the motor would rotate in one direction. The proportional controller for position (K_{PP}) is tuned using experimental method with no overshoot criteria.

Integral of Absolute value of Error (IAE) is used as a performance index for overall speed control (IAE_v) in which the formula is as follows, see (Marlin, 2000):

$$IAE_v = \int_0^{90} |SP(t) - PV(t)| dt \tag{4}$$

Integral of Time Absolute value of Error (ITAE) is used as a performance index for position control ($ITAE_p$), the first 8-second of starting speed control ($ITAE_{vp}$), and the 9-second of loading speed control ($ITAE_{vpl}$) which the formulas are as follows, see (Marlin, 2000):

$$ITAE_{vp} = t\int_0^8 |SP(t) - PV(t)| \, dt \tag{5a}$$

$$ITAE_{vpl} = t\int_{14}^{23} |SP(t) - PV(t)| \, dt \tag{5b}$$

$$ITAE_p = t\int_0^{90} |SP(t) - PV(t)| \, dt \tag{5c}$$

and the fitness value is obtained from the basic formula as follows, see (Xiu & Ren, 2004):

$$f(i) = \frac{ITAE - ITAE\,(i)}{\sum_i ITAE_{max} - ITAE\,(i)} \tag{6}$$

Based on Eq (6), the fitness values for overall speed (f_v), first 8-second starting speed (f_{vp}), 9-second loading speed (f_{vpl}) and position (f_P) are as follows:

$$f_v(i) = \frac{10(IAE_{v,max} - IAE_v(i))}{\sum_{i=1}^5 \left(IAE_{v,max} - IAE_v(i)\right)} \tag{7a}$$

$$f_{vp}(i) = \frac{10(ITAE_{vp,max} - ITAE_{vp}(i))}{\sum_{i=1}^5 \left(ITAE_{vp,max} - ITAE_{vp}(i)\right)} \tag{7b}$$

$$f_{vpl}(i) = \frac{10(ITAE_{vpl,max} - ITAE_{vpl}(i))}{\sum_{i=1}^5 \left(ITAE_{vpl,max} - ITAE_{vpl}(i)\right)} \tag{7c}$$

$$f_p(i) = \frac{10(ITAE_{p,max} - ITAE_p(i))}{\sum_{i=1}^5 \left(ITAE_{p,max} - ITAE_p(i)\right)} \tag{7d}$$

4. Experimental results

The comparison on the effectiveness of modeling the DC motor based on the fitness value is presented in Table III.
Selecting the best process model in Table III, the transfer function (s-domain model) of the plant based on the best model is as follows:

$$\frac{Speed_{OUT}(s)}{Voltage_{IN}(s)} = G(s) = \left[\frac{456.3713}{s^3 + 9.5040s^2 + 80.7000s + 204.5000} \right] e^{-0.1682s} \qquad (8)$$

Process Model	Fitness	Remark
1	84.1416	
2	84.4121	
3	84.8531	
4	-34.0071	
5	86.5877	Best
6	75.5157	

Table III. Fitness value modelling the DC motor

Graphical comparison of actual (real time) response and s-model (estimated) response in open loop analysis is shown in Figure 12.

The PI, PID, FLBPI and FLBPID have parameters as follows:

K_P=10.58 and K_I=1.47 for PI controller

K_P=14.10, K_D=0.1, and K_I=2.44 for PID controller

K_{Pm}=6 and K_{Im}=7.9 for FLBPI controller (experiment)

K_{Pm}=10.4 (experiment), K_{Dm}=0.1 (Ziegler-Nichols), and K_{Im}=7.2 (experiment) for FLBPID controller
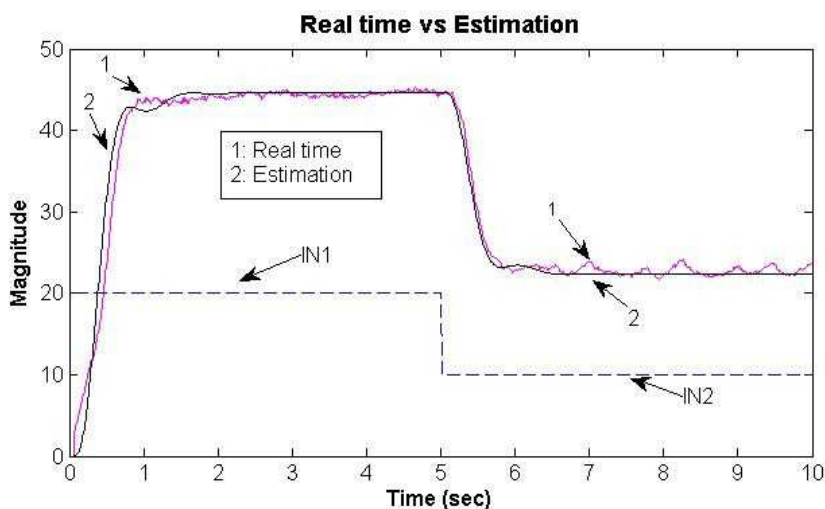
Fig. 12. Graphical comparison of actual (real time) and estimation (s-model)

The relevant parameters were obtained during the simulation (offline optimization) using the transfer function as shown in Eq (8). The comparison on the effectiveness of implementing FLBPI as compared to FLBPID, FLC, PID, and PI controller based on the performance metrics for real time experiment is presented in Table IV.

The ability to understand the influence of the different controllers on the servomotor speed and achieving its target position is one of the important aspects of this study. It is noted that the speed response with the FLBPID-type controller has a relatively good settling time, but

the fitness values for the motor during starting and loading is not good enough due to the fact that the noise is a bit larger as compared to FLBPI, albeit the fitness value of the overall speed for the FLBPI and the FLBPID are similar. A closer observation at the starting and loading performances during the real time experimentations, FLBPI performances is relatively better as compared to the PI, PID, FLBPID or FLC for speed control. A similar performance is also observed in the fitness value of position for real time experiment, where the FLBPI is relatively better as compared to the PI, PID, FLBPI or FLC for position control.

NO.	PERFORMANCE INDEX	PI	PID	FLC	FLBPI	FLBPID
1	Overshoot (%)	0.0000	0.0000	11.4299	9.5200	0.0000
2	Settling Time (sec)	3.4900	3.6200	8.9400	2.1700	1.1100
3	#SSEP (%)	0.3914	0.0334	0.5290	0.0896	0.0083
4	Undershoot (%)	38.4279	36.5108	41.5249	41.2515	40.7199
5	IAE v all	1.15E+03	1.22E+03	1.34E+03	7.12E+02	7.13E+02
6	ITAE v start	2.54E+02	1.84E+02	5.17E+02	1.45E+02	1.49E+02
7	ITAE v load	2.28E+03	1.59E+03	2.42E+03	1.54E+03	1.62E+03
8	ITAE p	1.40E+03	1.30E+03	1.49E+03	1.29E+03	1.30E+03
9	fv all	1.2137	0.7666	0.0000	4.0140	4.0057
10	fv start	1.9715	2.4915	0.0000	2.7820	2.7550
11	fv load	0.0528	0.3132	0.0000	0.3321	0.3019
12	fp	1.3433	2.8358	0.0000	2.9851	2.8358

#SSEP: Steady State Error of Position

Table IV. Performance Index Comparison for Real time experiment

Comparing the conventional and hybrid controllers based on fitness values in real time experiments, the PID performs better than the PI for speed and position control in the case for conventional controllers. Interestingly, for hybrid controllers, the FLBPI is better than the FLBPID for speed and position control, whilst the FLBPI is better than the PID for speed and position control. This shows that the performance has improved when having the hybrid fuzzy and PI as a controller. In the standalone condition and using default parameters, the FLC is not as good as its competitors for both speed and position controls.

Plots for the speed control in response to a set-point specified in both simulation and real time experiment for FLBPI controller are presented in Figure 13 and for the position control are presented in Figure 14. Comparing the simulation and real time experiment results as shown in Figures 13 and 14, the performances are not exactly the same but the output patterns are similar and the results in real time experiments are validly representing the real system's performance. This shows that identification (s-modeling) is an estimation of real hardware plant.

Plots for the speed control in response to a set-point specified in the real time experiment for FLBPI compared to PID controller are presented in Figure 15 and for the position control are as presented in Figure 16. A sudden change in load requirements that happens at t=15 seconds causes the speed to fall and then rises and stabilized at the original level within a reasonable range of time. This demonstrates the action of the controllers to regulate the speed.
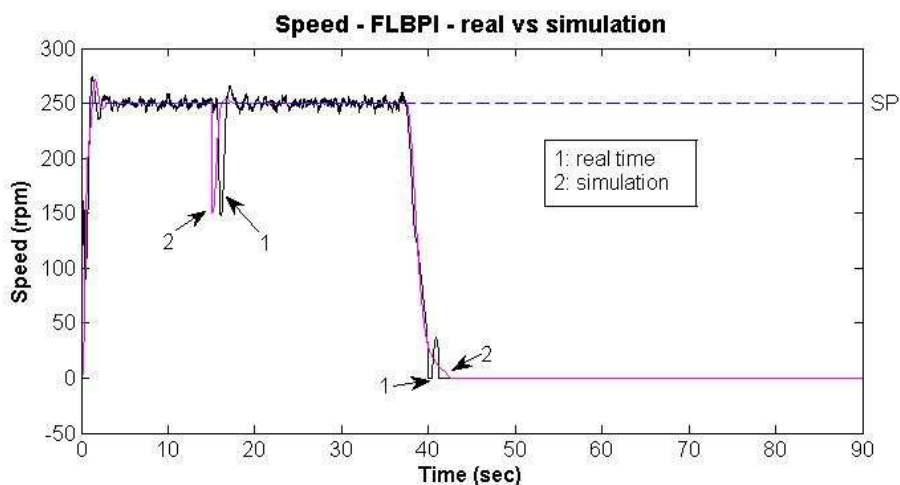
Fig. 13. Speed control of DC servomotor using FLBPI in real time experiment vs simulation
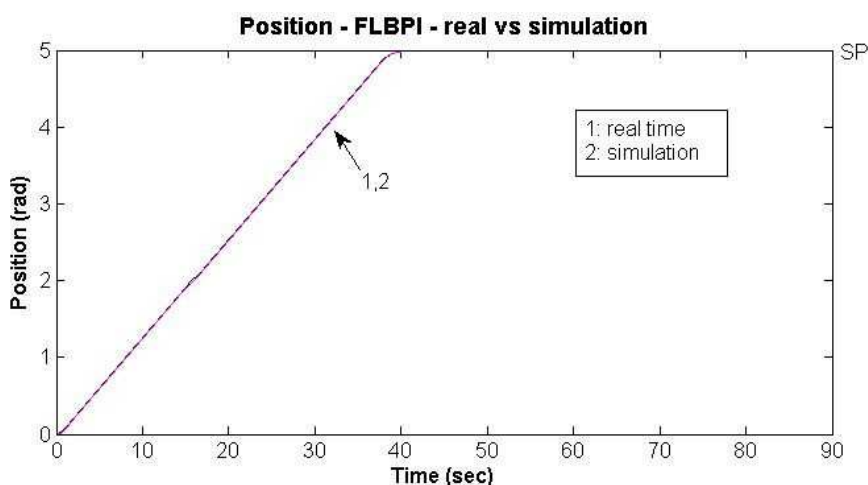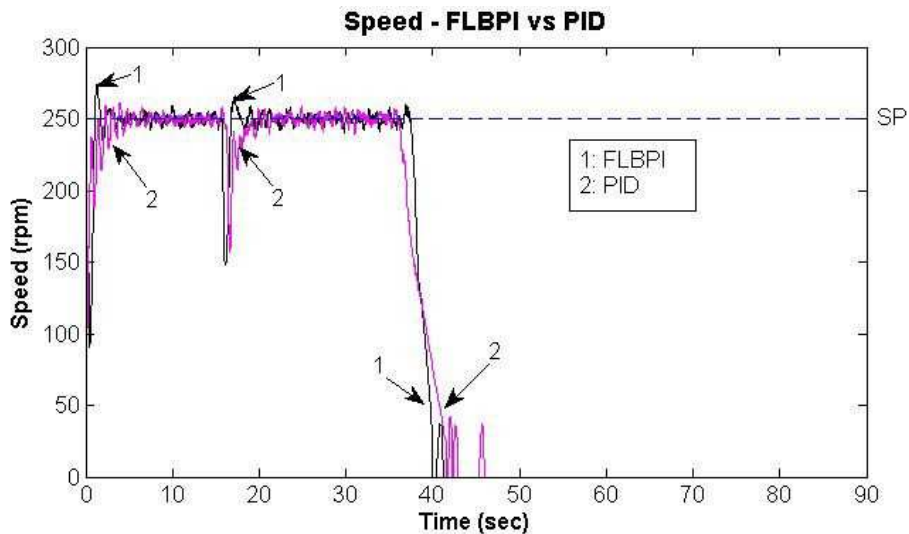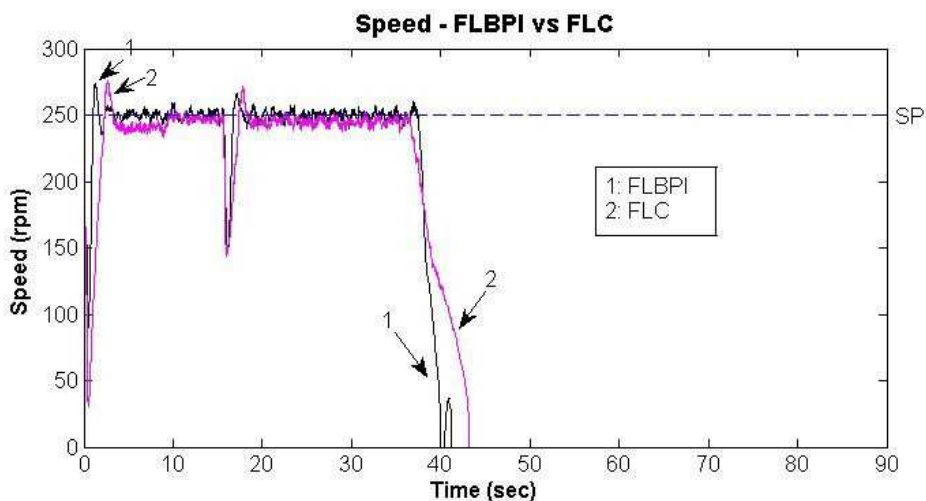

Fig. 14. Position control of DC servomotor using FLBPI in real time vs simulation

Plots for the speed control in response to a set-point specified in the real time experiment for FLBPI compared to FLC are presented in Figure 17 and for the position control are presented in Figure 18. A sudden change in load requirements that happens at t=15 seconds causes the speed to fall and then rises and stabilized at the original level within a reasonable range of time. Interestingly, it can be seen the delay in speed and position when using FLC alone as the controller. This demonstrates the action of the controllers to regulate the speed, and the effect of the PI in improving the response time of the hybrid controller.

Fig. 15. Speed control of DC servomotor using FLBPI vs PID in real time experiment
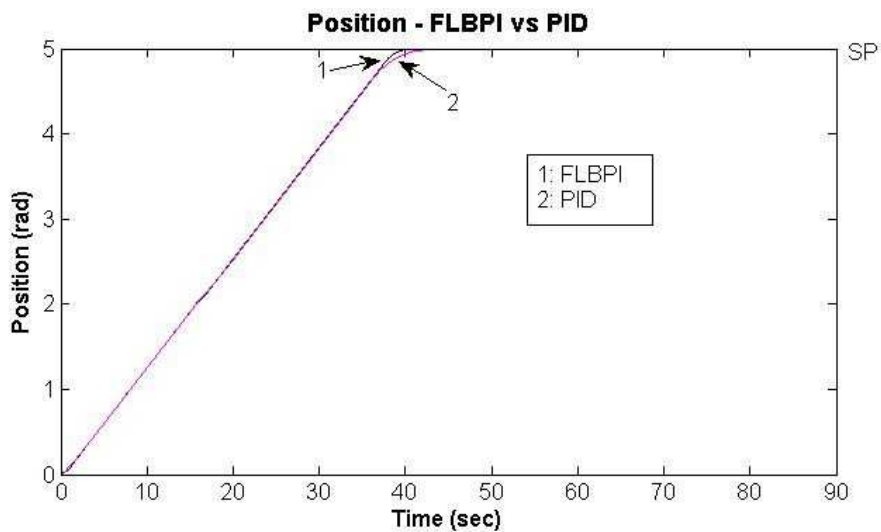
Fig. 16. Position control of DC servomotor using FLBPI vs PID in real time experiment

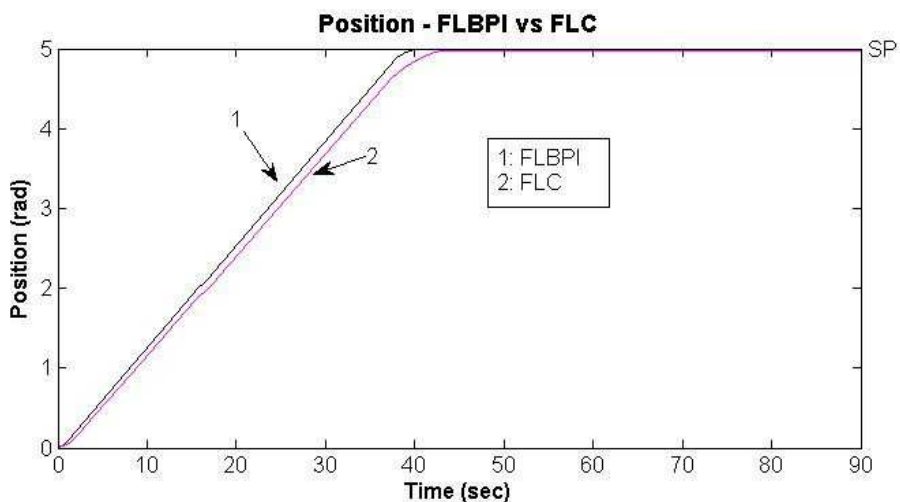Fig. 17. Speed control of DC servomotor using FLBPI vs FLC in real time experiment

Fig. 18. Position control of DC servomotor using FLBPI vs FLC in real time experiment

5. Conclusions

This work discusses the modeling of a DC servomotor from gray box identification and performance evaluations of real time experiment using a fuzzy-logic-based self tuning PI controller as compared to fuzzy-logic-based self tuning PID controller, fuzzy logic controller, PID controller and PI controller on the DC servomotor system. Here, the s-model transfer function of a DC servomotor is identified as a third order transfer function without

zero. This identification is useful in offline optimization of a DC servomotor control, and the performance of s-modeled and real DC servomotor are similar.

Two control modes are applied in sequential to the plant: speed control in the position control loop. The open loop characteristic of a DC servomotor is sampled at 0.01 sec interval by a DAQ based on Simulink platform. Some controllers are applied to both s-modeled and real DC servomotor. It has been demonstrated that defining the fuzzy rules for the fuzzy logic-based self-tuning PI/PID controller is a much simpler task than for fuzzy logic controller. Based on the real time experiment, hybrid controllers are better than conventional controllers and fuzzy logic-based self-tuning PI controller tends to be the better choice for implementation in the hybrid controller.

6. References

Bousserhanel, I.K., Hazzabl, A., Rahli, M., Kamli, M. and Mazari, B. (2006). Adaptive PI Controller using Fuzzy System Optimized by Genetic Algorithm for Induction Motor Control, presented at CIEP-IEEE, Puebla, Mexico.

Fulford, C., Maggiore, M. and Apkarian, J. (2009). Control of a 5DOF Magnetically Levitated Positioning Stage, IEEE Transaction on Control System Technology, vol. 17 no. 4, pp. 844-852.

Ga, X. and Feng, Z.J. (2005). Design study of an adaptive fuzzy-PD controller for pneumatic servo system, Control Engineering Practice, vol. 13, issue 1, pp 55-65.

He, S.Z., Tan S., and Xu, F.L (1993). Fuzzy self-tuning of PID controllers, Fuzzy Sets and Systems, vol. 56, pp. 37-46.

Jee, S. and Koren, Y. (2004). Adapative fuzzy-logic controller for feed drive of a CNC machine tool, Mechatronics, vol: 14, pp. 299-326.

Kissell, T. (2002). Motor Control Technology for Industrial Maintenance. New Jersey 07458: Prentice-Hall, Inc., Upper Saddle River.

Lacevic, B., Velagic, J. and Osmic, N. (2007). Design of Fuzzy Logic Based Mobile Robot Position Controller Using Genetic Algorithm," presented at International Conference on Advanced Intelligent Mechatronics, IEEE/ASME 2007, vol., no., pp.1-6.

Lu, Y.Z. (1996). Industrial Intelligent Control, Fundamentals and Applications. West Sussex PO19 1UD, England: John Wiley&Sons Ltd.

Mannan, M.A., et al. (2004). Fuzzy-Logic-Based Self-Tuning PI Controller for Speed Control of Indirect Field-Oriented Induction Motor Drive, in SICE Annual Conference. Sapporo: Hokkaido Institute of Technology.

Marlin, T.E. (2000). Process Control: Designing Processes and Control Systems for Dynamic Performance, 2nd International ed. Singapore: McGraw-Hill Book Companies, Inc.

Montanari, M., Peresada, S.M., Rossi, C. and Tilli, A. (2007). Speed Sensorless Control of Induction Motors Based on a Reduced-Order Adaptive Observer," IEEE Transaction on Control System Technology, vol. 15 no. 6, pp. 1049-1064.

Shieh, M. Y. and Li, T.H.S. (1998). Design and implementation of integrated fuzzy logic controller for servomotor system, Mechatronics, vol. 8, pp. 217-240.

Takami, K.M. and Mahmoudi, J. (2007). Identification of a Best Thermal Formula and Model for Oil and Winding of Power Transformers Using Prediction Methods, presented at The 48th Scandinavian Conference on Simulation and Modeling (SIMS 2007), pp. 182-188.

Wang, L.X. (1997). A Course in Fuzzy Systems and Control. New Jersey 07458: Prentice-Hall, Inc, A Division of Simon and Schuster Upper Saddle River, New Jersey 07458.

Xiu, Z. and Ren, G. (2004). Optimization Design of TS-PID Fuzzy Controllers Based on Genetic Algorithms," presented at 5th World Congress on Intelligent Control and Automation, Hangzhou, P.R. China.

Yeh, E.C., and Tsao, Y.J. (1994). A fuzzy preview control scheme of active suspension for rough road, Inc. Journal of Vehicle Design, vol. 15, pp. 166-180.

Yu, G.R., Hwang, R.C. and Lin, C.P. (2004). Optimal Fuzzy Control of the Spindle Motor in a CD-ROM Drive Using Genetic Algorithms, presented at Asian Control Conference, vol. 5, pp. 51-57.

Zadeh, L.A. (1988). Fuzzy Logic, Computer, vol. 21 no. 24, pp. 83-93.
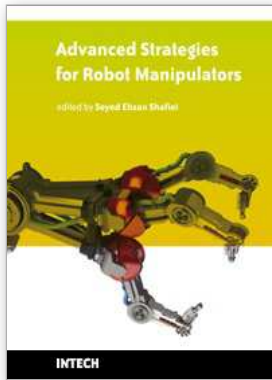
Advanced Strategies for Robot Manipulators

Edited by S. Ehsan Shafiei

Amongst the robotic systems, robot manipulators have proven themselves to be of increasing importance and are widely adopted to substitute for human in repetitive and/or hazardous tasks. Modern manipulators are designed complicatedly and need to do more precise, crucial and critical tasks. So, the simple traditional control methods cannot be efficient, and advanced control strategies with considering special constraints are needed to establish. In spite of the fact that groundbreaking researches have been carried out in this realm until now, there are still many novel aspects which have to be explored.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Nordin Saad and Oyas Wahyunggoro (2010). Development of Fuzzy-Logic-Based Self Tuning PI Controller for Servomotor, Advanced Strategies for Robot Manipulators, S. Ehsan Shafiei (Ed.), ISBN: 978-953-307-099-5, InTech, Available from: http://www.intechopen.com/books/advanced-strategies-for-robot-manipulators/development-of-fuzzy-logic-based-self-tuning-pi-controller-for-servomotor

INTECH
open science | open minds