TELKOMNIKA Indonesian Journal of Electrical Engineering
Vol. 16, No. 1, October 2015, pp. 65 ~ 74
DOI: 10.11591/telkomnika.v16i1.8499

Compared with PI, Fuzzy-PI and PSO-PI Controllers of Robotic Grinding Force Servo System

Adnan Jabbar Attiya*[1,2]. Yang Wenyu[1], Salam Waley Shneen[3]
[1]School of Mechanical Science and Engineering,
Huazhong University of Science and Technology (HUST), Luoyu Road 1037, Wuhan, China
[2]Alkwarizmi Engineering college/Baghdad University, Baghdad, Iraq
[3]Huazhong University of Science and Technology (HUST) / University of Technology,
Baghdad, Iraq
*Corresponding author, e-mail: rainman3009@yahoo.com, mewyang@mail.hust.edu.cn,
Salam_waley73@yahoo.com

Abstract

By grinding process, when an industrial robot is used to finish a curved surface, both feed movement and contact force must controlled at the similar time in order that the grinding tool would machine the work-piece at the right position in right posture with required force. A passive wrist system is advanced, in this paper, to conform the shape of the machining propeller by altering its posture along with the surface. The proportional-integral (PI) controller, due to its simplicity, robustness, and affordable price, is extremely often used in practical applications, but it is effective for linear systems, as well as, the challenging task is to find its optimal gains. If the processes involved higher order and time delay systems, many intelligent controllers were appeared. In this paper, to cope with nonlinearities, improve the controller parameters and at the same time modeling uncertainties of grinding marine propeller surface, a PI torque controller is proposed such that its optimal gains are derived via a modern systems based on fuzzy logic theory and particle swarm optimization algorithm which are used to solve various engineering problems. Grinding force is controlled under Fuzzy-PI controller which is being assembled and compared with a PSO-PI controller to obtain which controller that provides grinding with higher quality. The compared controllers have been optimized together with the parameters of the Two-Phase Hybrid Stepping Motor. The suggested fuzzy rule function and PSO algorithm improve the response of the controlled system and searches a high-quality solution impressively. Simulation and comparison results are presented and that the proposed control systems are coping well with nonlinearities and uncertainties while find PI control parameter set effectively, the PSO-PI controller has a better control performance with improved step response for robotic grinding force servo system. These control methods was simulated using MATLAB/SIMULINK.

Keywords: *PI controller, Fuzzy-PI controller, particle swarm optimization (PSO), PSO-PI controller, force control, grinding robot*

1. Introduction

Grinding is a mechanical process intended to remove a very thin and even layer of materials on the outer edge of the workpiece. Field practice previously demonstrated the difficulty in grinding complex shaped workpiece like a marine propeller or turbine blade which demands the end-effector of the manipulator to keep a stable contact force with the environment while the grinding tool moves along the profile of the workpiece. To perform such a task, concurrently, both force and position of the robot should be controlled. These control strategies have been classified into two main systems, impedance control and hybrid position/force control. These methods demand an accurate dynamic model of the manipulator and of the contact force interaction. In a hybrid position/force control scheme only path normal direction is subject to a constant pressure or force control. There are several advantages of this solution. Firstly, the dimensional variation due to the tool wear is compensated automatically, and secondly, the necessity of accuracy on the programmed path is relaxed since the assured contact from the force control loop will recompense the program error [1].

In the PI controller there are two parameters: proportional coefficient and integral coefficient. the PI controller can present individualized control necessities by tuning these two

Received June 30, 2015; Revised August 7, 2015; Accepted August 30, 2015

parameters. Many intelligence algorithms are suggested to tuning the PI parameters. the plurality part of control systems use regular PI control algorithms with fixed constraint values set through the empowering. This is guide to the eases of design and low cost but it remains uncertain guide to more complexity that is mathematical. The combination of P-I controller parameters depend on type of methods and well-known design methods ordinarily demands a mathematical model that can describe the control object dynamical performance precisely [2]. In addition, there are numerous linear methods used in the design of PI stables designated as conservative controller anyway, research is a distinguished deals with carried out in the design of uncommon controllers using new computational techniques such as neural networks and fuzzy logic [3]. Depending on the change of control objects parameters, the performance of the system will change when this controller is applied to a nonlinear control system [4]. Fuzzy logic is widely used in processes where system dynamics is either very complex or demonstrate a highly nonlinear character. In order to get expectant control effect, fuzzy control rules design should take full statement of the control specialty which is different from conventional control [5]. Hsieh et al. present an optimal predicted fuzzy-PI gain scheduling controller to control the constant turning force process with a fixed metal removal rate under various cutting conditions [6]. On the other hand, for the same magnitude of negative and positive reference input changes, the execution of a linear control law will cause different responses of a nonlinear system. Many design strategies will developed to defeat the disadvantages of linear P-I controllers. Such methods created for obtaining a goal transform a linear P-I controller into unconventional PI controllers [7]. To defeat these difficulties, several types of modified PI controllers such as adaptive PI controllers and auto tuning were advanced lately [8]. Also, a class of nonconventional type of PI controller applying fuzzy logic has been designed and simulated for this purpose [9]. The Particle Swarm Optimization, is an additional popular optimal algorithm proposed by Kennedy and Eberhart [10] in 1995. It is an algorithm for swarm intelligence based on population-based adaptive optimization, swarm theory and stochastic based on the simulation of animal social behaviors like fish swarms and bird flocks. Compared with other methods such as genetic computation, machine learning, and neural network learning, it furnishes better performance in computing accuracy, computing speed, and memory size in spite of the fact that the original PSO is very simple with only a few parameters to modify. Each parameter in PSO extremely affects the performance of PSO. Although, it is yet to be found how to determine suitable values of parameters in PSO that can be considered as level optimization [11,12]. The PSO technique has a steady meeting characteristic as compared to other stochastic methods and it can achieve a high quality solution during a shorter computation time [13]. It has various implementations in engineering fields. In the PI controller design, the PSO algorithm is applied to search a best PI control parameters [14].

In this paper, the PI controller has been proposed first, then tuning it by fuzzy logic and a particle swarm optimization (PSO) algorithm to improve the controller parameters. The Force Control Algorithms are described in Section 2. The proposed fuzzy logic and PSO algorithm are delineated in Section 3. MATLAB simulation results and some compare results are shown in Section 4. Finally, conclusions are made in Section 5.

2. Force Control Algorithms

As shown in Figure 1, the robot end-effector or tool of an industrial manipulator in contact with a workpiece.

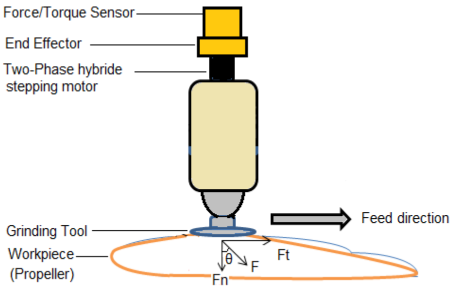


Figure 1. Main parameters used in grinding policies

The objective is to make the tool softly into contact with the workpiece, apply a contact force, which is well kept while the tip of the tool follows the profile of the part to be grinded. The ordinary method to satisfy the above objective is to make the tool always normal to profile of the workpiece and to control the orientation of the end-effector. At the same time this demands only the composite force in Fn and Ft direction to be controlled. The force control system structure can be described as a position-based controller with an outer fuzzy PI or PSO PI force control loop. The profile data for the marine propeller can be produced off-line and stored in microprocessor memory as a reference path for the robot to follow. The grinding tool cannot keep the specified contact force by moving basically along the propeller profile path on account of the geometric and displacement errors in the propeller location and other fine errors, so needs to be an external force control loop to accomplish a practical path.

3. The Model of the Two-phase Hybrid Stepping Motor

The transfer function G(s) of the open-loop system of the two-phase Hybrid Stepping Motor is as follows [15]:

$$\mathrm{G(s)} = \frac{\mathrm{A(s)}}{\mathrm{B(s)}} \tag{1}$$

Where,

$$\mathrm{A(s)} = (\mathrm{KPis} + \mathrm{KIi})\mathrm{KH} \tag{2}$$

$$\begin{aligned}\mathrm{B(s)} &= \mathrm{s(THs+1)(Ls+R)} + (\mathrm{KPis} + \mathrm{KIi})\mathrm{KH} \\ &= \mathrm{LTHs}^3 + (\mathrm{RTH} + \mathrm{L})\mathrm{s}^2 + \mathrm{Rs} + (\mathrm{KPis} + \mathrm{KIi})\mathrm{KH}\end{aligned} \tag{3}$$

The subdivided driving is assumed for the Hybrid Stepping Motor in order to reach to the actual system performance parameter and to decrease the intricacy of the system transfer function. In simulation, the parameters of the two-phase Hybrid Stepping Motor selected are as follows:

Inertia Constant J = 250 kg · m, Inductance L= 0.33 H, Resistance R = 8 Ω, , β = 1, Coefficient of Viscous Friction B = 0 N · m · s/rad, Kpv = 500, KIv = 0, KPi = 5, KIi = 500, ke = 0.25 N · m/A, N = 180, KH = 15 and TH = 0. The Transfer function will be:

$$\mathrm{G(s)} = \frac{75\mathrm{s}+7500}{0.33\mathrm{s}^2+83\mathrm{s}+7500} \tag{4}$$

4.1. Conventional PI Strategy for Grinding Force Servo Unit

The most conventional PI controller or linear PI controller is described as follows:

$$\mathrm{Y(t)} = \mathrm{e(t)Kp} + \mathrm{Ki}\int_0^t \mathrm{e(t)dt} \tag{5}$$

Where KP is the proportional constant gain and KI is the integral constant gain according to manual expertise. The signal e(t) is the error signal between the reference and the process output c(t) it is explained as: e(t) = r(t) − c(t).

Table 1. The effect of Kp and Ki to the controlled system

Parameter	Rise time	Overshoot	Turning time	Error
Kp	decrease	increase	Small change	decrease
Ki	decrease	increase	increase	eliminte

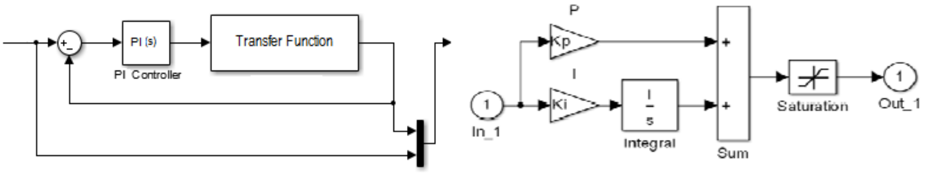


Figure 4. Simulink and Block diagram of PI controller

And the PI controller internal structure is explained in Figure 5, the input parameters for the PI controller are Kp, Ki,and the output for the controller is u.

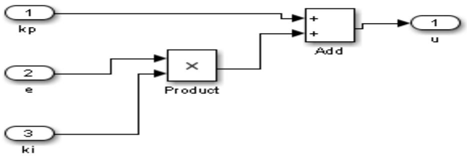
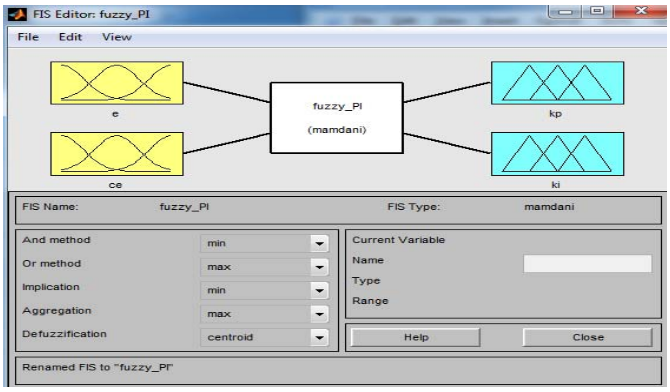


Figure 5. The structure for PI controller

4.2. Fuzzy-PI controller for Grinding Force Servo Unit

The Rule base is formed based on the following concepts. The fuzzy rules and the ranges of input membership functions are entered from the obtainable data. The following steps can represent the procedure of the suggested fuzzy logic:

Step 1: Initialize (FIS editor) the input of fuzzy logic controller error (e), change of error (ec) and output (KP) and (KI), see Figure 6



FIS Name: fuzzy_PI FIS Type: mamdani

And method min

Or method max

Implication min

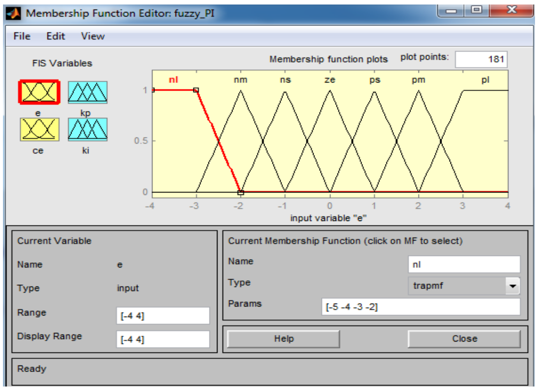
Aggregation max

Defuzzification centroid

Renamed FIS to "fuzzy_PI"

Figure 6. FIS Editor of Fuzzy-PI controller

Step 2: Set the system with tow fuzzy logic controller (KP,KI) each controller has two inputs, (e,ec) and the output are (KP,KI), e input has seven fuzzy set associated with it, which sorted as negative large (Nl), negative medium (NM),negative small (NS), Zero error (Z), positive small (PS), positive medium (PM) and positive large (Pl), see Figure 7, 8.



Name e

Type input

Range [-4 4]

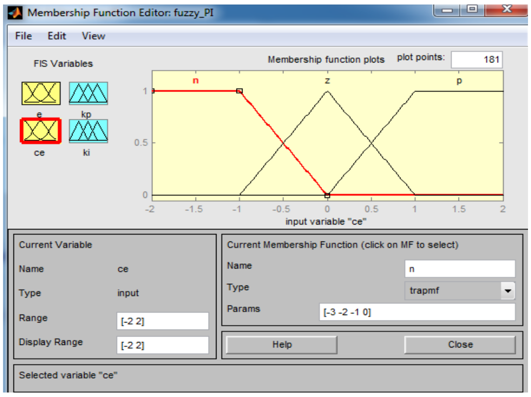
Display Range [-4 4]

Name nl

Type trapmf

Params [-5 -4 -3 -2]

Ready



Name ce

Type input

Range [-2 2]

Display Range [-2 2]

Name n

Type trapmf

Params [-3 -2 -1 0]

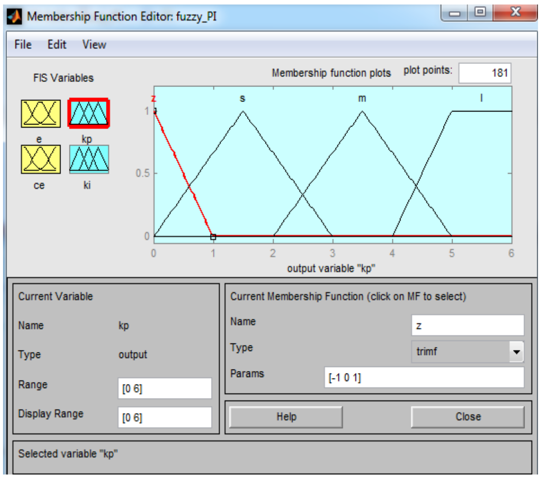
Selected variable "ce"

Figure 7. Membership error function (e)

Figure 8. membership change error function (ec)

Step 3: Set the input range from -3 to 3, where the input e and ec are shown in the following equations: e={NB(-4,-3,-2), NM(-3,-2,-1), NS(-2,-1,0),Z(-1,0,1), PS(0,1,2), PM(1,2,3], PI(2,3,4)}, ec={[N(-2,-1,0), Z(-1,0,1), PB(0,1,2)}.

Step 4: set the output range from 0 to 6. Figure 9 shows the membership function of the output variable KP.



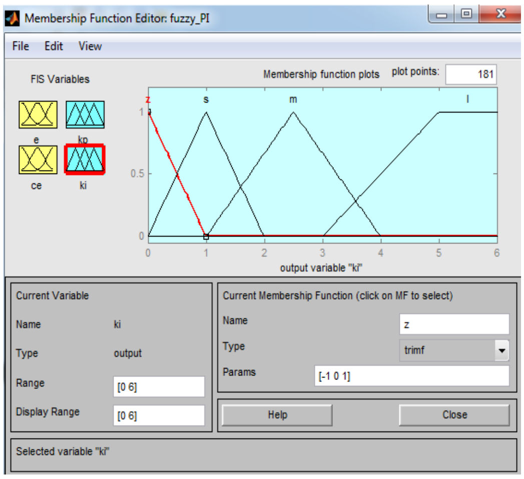
plot points: 181

Current Variable
Name: kp
Type: output
Range: [0 6]
Display Range: [0 6]

Current Membership Function (click on MF to select)
Name: z
Type: trimf
Params: [-1 0 1]

Selected variable "kp"

Figure 9. Membership Kp function



plot points: 181

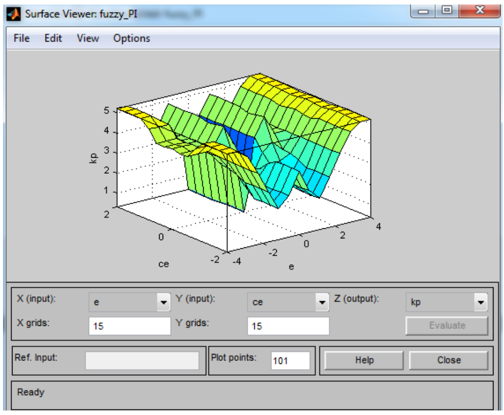
Current Variable
Name: ki
Type: output
Range: [0 6]
Display Range: [0 6]

Current Membership Function (click on MF to select)
Name: z
Type: trimf
Params: [-1 0 1]

Selected variable "ki"

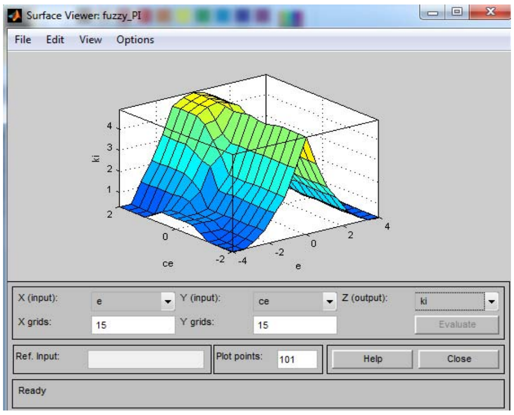
Figure 10. Membership Ki function

The surface viewer is shown in Figure 11, 12.



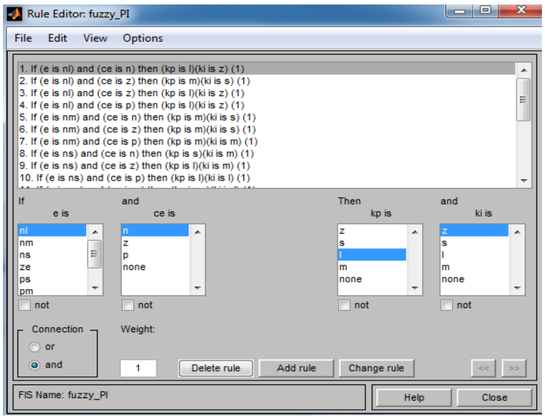
X (input): e
Y (input): ce
Z (output): kp
X grids: 15
Y grids: 15
Ref. Input:
Plot points: 101
Ready

Figure 11. Surface view of Kp of fuzzy controller



X (input): e
Y (input): ce
Z (output): ki
X grids: 15
Y grids: 15
Ref. Input:
Plot points: 101
Ready

Figure 12. Surface view of Ki of fuzzy controller

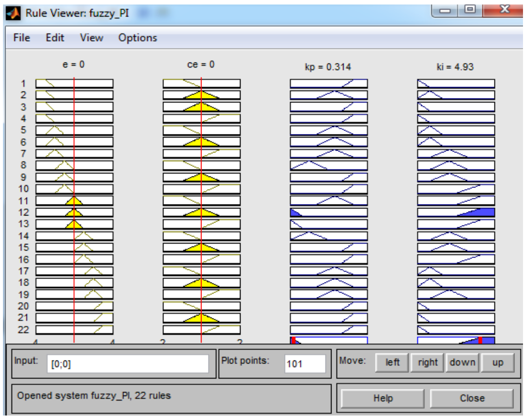


1. If (e is nl) and (ce is n) then (kp is l)(ki is z) (1)
2. If (e is nl) and (ce is z) then (kp is m)(ki is s) (1)
3. If (e is nl) and (ce is z) then (kp is l)(ki is z) (1)
4. If (e is nl) and (ce is p) then (kp is l)(ki is z) (1)
5. If (e is nm) and (ce is n) then (kp is m)(ki is s) (1)
6. If (e is nm) and (ce is z) then (kp is m)(ki is s) (1)
7. If (e is nm) and (ce is p) then (kp is m)(ki is m) (1)
8. If (e is ns) and (ce is n) then (kp is s)(ki is m) (1)
9. If (e is ns) and (ce is z) then (kp is l)(ki is m) (1)
10. If (e is ns) and (ce is p) then (kp is l)(ki is l) (1)

e is: nl, nm, ns, ze, ps, pm — □ not
ce is: n, z, p, none — □ not
kp is: z, s, l, m, none — □ not
ki is: z, s, l, m, none — □ not

Connection: ○ or ● and
Weight: 1
FIS Name: fuzzy_PI

Figure 13. Rule bases for Fuzzy control system



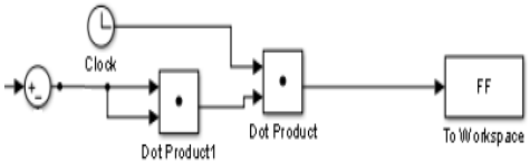
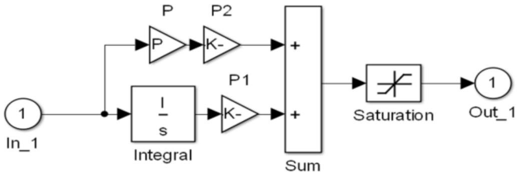
e = 0
ce = 0
kp = 0.314
ki = 4.93
Input: [0;0]
Plot points: 101
Opened system fuzzy_PI, 22 rules

Figure 14. Rule bases view for Fuzzy control system

Step 5: The estimation performance measures are Rise time (RT), peak overshoot (OV), and Amplitude (AM). Figure shows an example of a desired time response to improve the response of the system by change the controller parameters until we reach the optimal response of the system.

4.3. PSO Algorithm for Grinding Force Servo Unit

In a PSO system, particles fly approximately in a multidimensional search space with an adaptable velocity which modified dynamically according to its own and the other particles flying experience. During flight, each particle modifies its position according to its own experience giving the best previous position (the minimum fitness value) is called personal best (P best) which keeps path of its coordinates associated with the best solution that obtained so far in the problem space, P best is recorded and delineated as PI=(pi1,pi2,…,pid), while the index of the best particle among all the neighboring particles in the population is called the global best (G best), it is delineated by the symbol g and it is the best solution in the whole group [16].

	 P P2 P1 1 In_1 Integral Sum Saturation 1 Out_1
Figure 15. Particle swarm optimization (PSO) strategy	Figure 16. The structure of Particle swarm optimization (PSO) strategy

For a PI- controlled system, there are often four indices to characterize the system performance: ISE, IAE, ITAE and ITSE. In this paper we select ITSE which is defined as:

$$\mathrm{ITSE} = \int_0^\infty te^2\ (t)dt \tag{6}$$

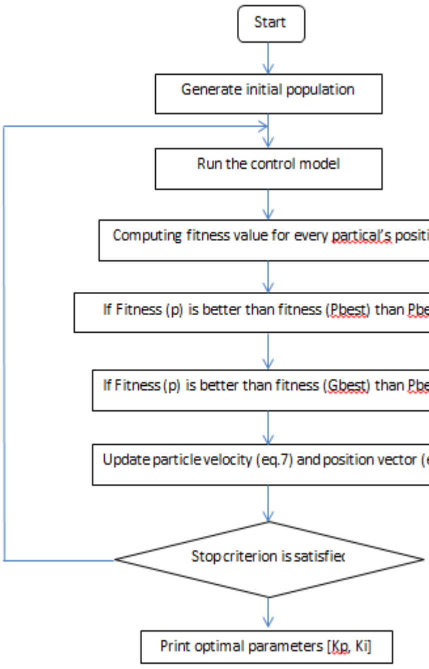

Start
Generate initial population
Run the control model
Computing fitness value for every partical's position
If Fitness (p) is better than fitness (Pbest) than Pbest=P
If Fitness (p) is better than fitness (Gbest) than Pbest=P
Update particle velocity (eq.7) and position vector (eq.8)
Stop criterion is satisfiec
Print optimal parameters [Kp, Ki]

Figure 8. The flowchart of the PSO-PI control system

The modified PSO framework is described as follows:

1. Generate the initial velocity and position for each part randomly delineated one input-output system controller in particle according to its lower and upper values of each PI controller parameters in system to form parents [17].

2. For each part in each particle in the swarm, estimate P best (the personal fitness.

3. To find the best PI controllers represented by the best parts in the swarm, evaluate G best (the global best fitness) for all varied parts from all particles in the swarm.

4. Determine G best of all particles in the population by adding all G best for all parts from each particle in swarm.

5. Update each particle to form progeny.

6. Compare P best (the personal fitness) of each part of progeny with their similar parts in parents and choose the best ones to form new offspring to the next generation.

7. Determine G best (the global fitness) of the varied parts in the population as to the new offspring and adding to be the best ones for the next generation.

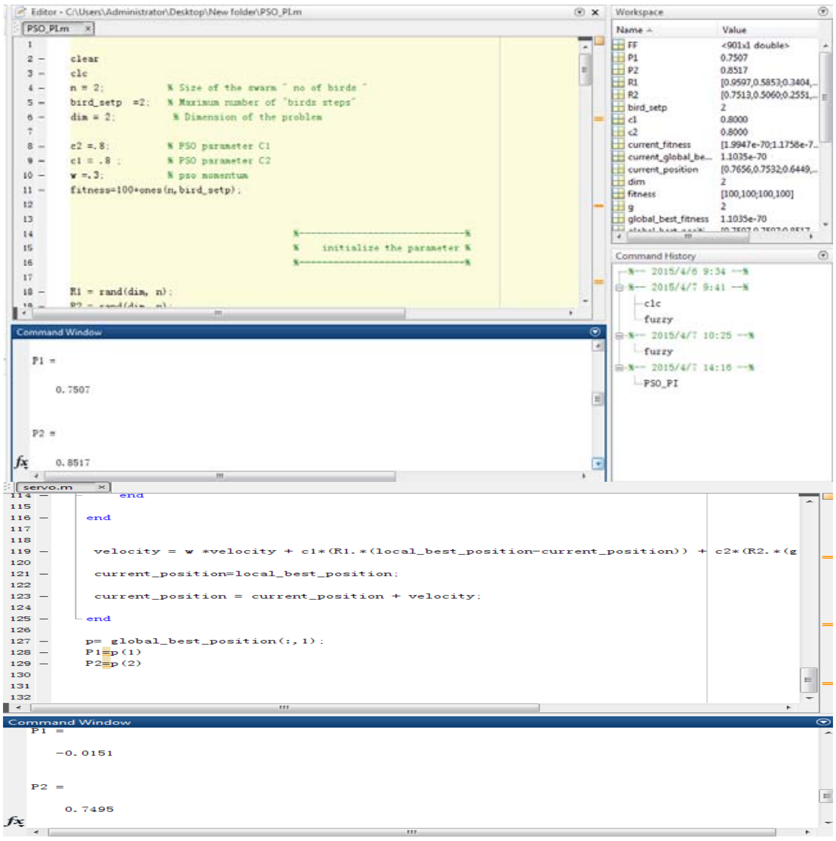
8. Stop if the stopping standard is satisfied otherwise, go to step 5. See Figure 8

Each particle is treated as a point in a D-dimensional space. The ith particle is represented as xi= (xi1,xi2,…,xid). The velocity of each agent which gradually gets close to Pbest and Gbest is represented as vi=(vi1,vi2,…,vid), it can be adjusted by the following equation:

$$v_{id}^{n+1} = w.v_{id}^{n} + c1.\mathrm{rand}\,().(p_{id}^{n} - x_{id}^{n}) + c2.\mathrm{rand}\,().\left(p_{gd}^{n} - x_{id}^{n}\right) \qquad (7)$$

Where v_{id}^{n} is current velocity, v_{id}^{n+1} is modified velocity, n represents iteration, w is an inertia weight, p_{id}^{n} = Pbest, p_{gd}^{n} = Gbest, c1 and c2 are two positive constants, rand ( ) is a random generated number with a range of [0,1], and x_{id}^{n} is the current searching point. The searching point in the solution space or the current position can be adjusted by the following equation:

$$x_{id}^{n+1} = x_{id}^{n} + v_{id}^{n+1} \qquad (8)$$



Editor - C:\Users\Administrator\Desktop\New folder\PSO_PI.m

PSO_PI.m

```

clear
clc
n = 2;              % Size of the swarm " no of birds "
bird_setp  =2;      % Maximum number of "birds steps"
dim = 2;             % Dimension of the problem

c2 =.8;             % PSO parameter C1
c1 = .8 ;           % PSO parameter C2
w =.3;              % pso momentum
fitness=100*ones(n,bird_setp);



                                        %------------------------------%
                                        %    initialize the parameter %
                                        %------------------------------%


R1 = rand(dim, n);
R2 = rand(dim, n);
```

Workspace

Name	Value
FF	<901x1 double>
P1	0.7507
P2	0.8517
R1	[0.9597,0.5853;0.3404,...
R2	[0.7513,0.5060;0.2551,...
bird_setp	2
c1	0.8000
c2	0.8000
current_fitness	[1.9947e-70;1.1758e-7...
current_global_be...	1.1035e-70
current_position	[0.7656,0.7532;0.6449,...
dim	2
fitness	[100,100;100,100]
g	2
global_best_fitness	1.1035e-70

Command History

```
%-- 2015/4/6 9:34 --%
%-- 2015/4/7 9:41 --%
clc
fuzzy
%-- 2015/4/7 10:25 --%
fuzzy
%-- 2015/4/7 14:16 --%
PSO_PI
```

Command Window

```
P1 =

    0.7507


P2 =

    0.8517
```

servo.m

```
        end

    end


     velocity = w *velocity + c1*(R1.*(local_best_position-current_position)) + c2*(R2.*(g

     current_position=local_best_position;

     current_position = current_position + velocity;

    end

    p= global_best_position(:,1);
    P1=p(1)
    P2=p(2)
```

Command Window

```
P1 =

   -0.0151


P2 =

    0.7495
```

Figure 18. Steps of PSO-PI controller

As recommended in PSO, the constants are c1=c2=0.8, Equation (7) is used to calculate new velocity of the particle according to its previous velocity and its prevalent position distances of from its own best position (experience) and the group's best experience. Then the particle flies in the direction of a new position according to Equation (8).

Inertia weight, w in the Equation (7) is to make balance between the local search and global search capability. It can be a nonlinear function of time or positive constant or even positive linear [18]. In this paper we set w=0.3.

5. Results and discussion

To verify the efficiency of the proposed fuzzy rule method and a particle swarm optimization (PSO) algorithm. With the MATLAB SIMULINK, a famous simulation software, and for a conventional PI, Fuzzy_PI, and PSO-PI controller as shown in Figure 19.

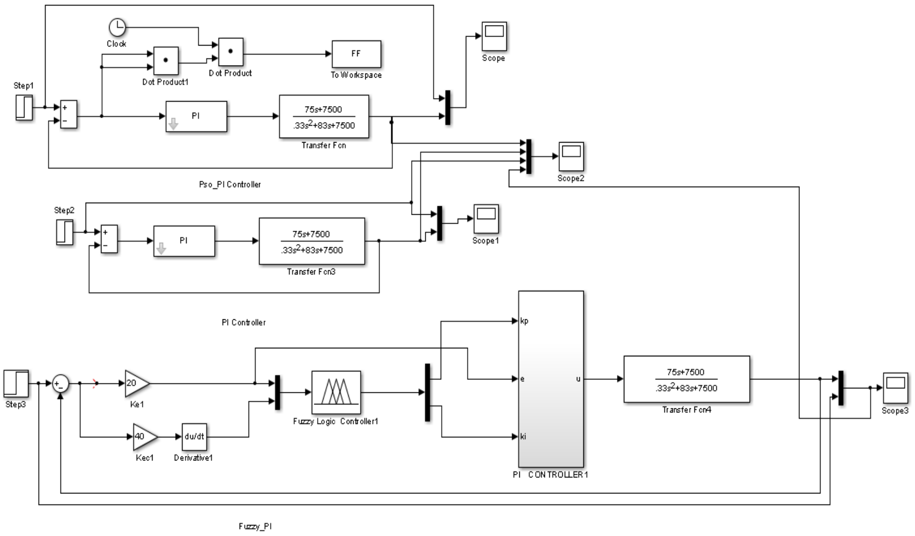


Figure 19. Simulink figure of PI, Fuzzy_PI, and PSO-PI controller

Step responses is shown in Figure 20, 21, 22, and 23 with Kp= 20 and Ki = 40

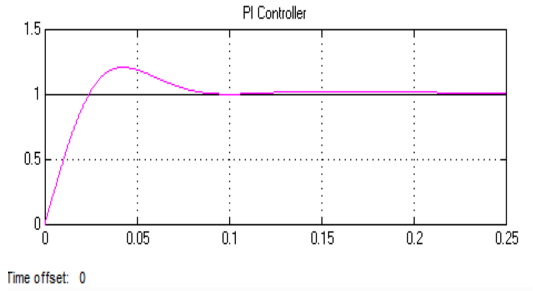


Figure 20. Step response of the system under PI controller

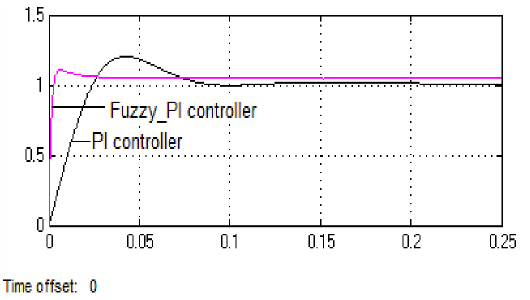


Figure 21. Step response of the system under PI, Fuzzy-PI controller

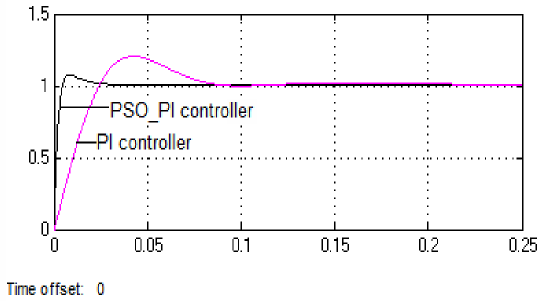


Figure 22. Step response of the system under PI, PSO-PI controller

First, for the step response of the system under PI controller shown in Figure 20, we note that the Rise time value = 0.016, the overshoot value = 20%, and the steady state error also we note that there is no an undershoot value while when we submit Fuzzy-PI strategy we remark that the rise time value (decrease) = 0.01, the overshoot value (decrease) = 10% where the system takes short time to reach the steady state as showing in Figure 21.

The system response of PI controller tuning using Particle Swarm Optimization is shown in Figure 22 show that the system will reach the stability quickly than the system under PI & Fuzzy-PI controller and the peak overshoot (decrease) = 7% while the Rise time = 0.013 so that the system got good response. See Table 2 and Figure 23.

Table 2. Step response performance for PI,Fuzzy-PI & PSO-PI controllers

Control Method	Overshoot	Rise Time	Steady State Error
	(%)	(s)	
PI	20	0.016	0.083
Fuzzy_PI	10	0.01	0.035
PSO-PI (ITAE)	7	0.013	0.027

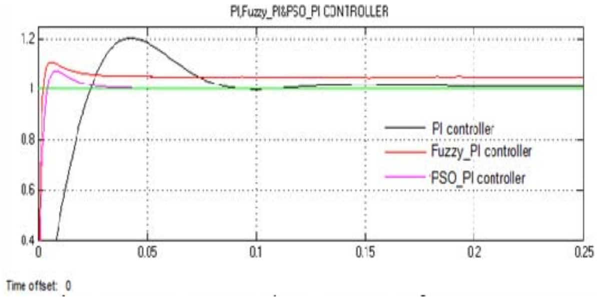


Figure 23. Step response of the system under PI, Fuzzy_PI, and PSO-PI controller

6. Conclusion

In this paper research works are taken for grinding force controlling. When a grinding wheel grinds propellers as a free-form surface by a robot, the grinding force, at the mentioned machining point, must be controlled in the normal direction in order that both grinding force and feed movement could be controlled. The modeling, control and simulation of the Two-Phase Hybrid Stepping Motor have been done using the software package MATLAB/SIMULINK. The design has been remarked that the system response is improved by setting the parameters of the transfer function until we obtain an optimal response of the system after performing the value of PI constants achieved from the fuzzy rule method and PSO. The submitted methodology gives better performance in the peak overshot, rise time, and the steady-state error. The response remarked from the PSO-PI controller has a small over shoot, small steady state error, and small rise time than PI and has small over shoot, small steady state error than

Fuzzy-PI controller. However, it is remarked from the simulation that the controller performs improved in Fuzzy-PI (the over shoot, rise time, settling and steady state better than PI controller.

Acknowledgements

This work was partially supported by the National Basic Research Program of China (2014CB046704).

References

[1] Jianjun Wang, Hui Zhang, Thomas A Fuhlbrigge. *Force Control Technologies for New Robotic Applications.* IEEE International Conference on Technologies for Practical Robot Applications. 2008.
[2] WR Hwang, WE Thompson. *Design of intelligent fuzzy logic controllers using genetic algorithms*. In Proceedings of the 3rd IEEE Conference on Fuzzy Systems, IEEE World Congress on Computational Intelligence. 1994: 1383-1388.
[3] Zulfatman, MF Rahmat. Application of self-tuning fuzzy PID controller on industrial hydraulic actuator using system identification approach. *International Journal on Smart Sensing and Intelligent Systems.* 2009; 2(2).
[4] Seema Chopra, R Mitra, Vijay Kumar. Auto tuning of fuzzy PI type controller using fuzzy logic. *International journal of computational cognition.* 2008; 6(1).
[5] Ma Qiujie, Shi Jingzhuo. Fuzzy PID Speed Control of Two Phase Ultrasonic Motor. *TELKOMNIKA Indonesian Journal of Electrical Engineering.* 2014: 6560-6565.
[6] Hsieh CH, Chou JH, Wu YJ. Optimal predicted fuzzy PI gain scheduling controller of constant turning force systems with fixed metal removal rate. *Int J Adv Manuf Technol.* 2002: 714-721.
[7] Jihong Lee. On methods for improving performance of PI-type fuzzy logic controllers. *IEEE Transactions On Fuzzy Systems.* 1993; 1(4).
[8] Rajani K Mudia, Chanchal Deyb, Tsu-Tian Lee. An improved auto-tuning scheme for PI controllers. *ISA Transactions.* 2008: 45-52.
[9] WM Tang, G Chen, RD Lu. A modified fuzzy PI controller for a flexible-joint robot arm with uncertainties. *Int. J. Fuzzy Sets Syst*. 2001: 109–119.
[10] J Kennedy, RC Eberhart. *Particle swarm optimization*. Proceedings IEEE International Conference on Neural Networks IV. 1995: 1942-1948.
[11] Juing-Shian Chiou, Shun-Hung Tsai, Ming-Tang Liu. A PSO-based adaptive fuzzy PID-controllers. *Simulation Modelling Practice and Theory.* 2012; 26: 49-59.
[12] Zhang Xin, Dang Jianwu, Liu Min. Parameter Optimization of PID Controller Based on PSO for Multi-leaf Collimator. *TELKOMNIKA Indonesian Journal of Electrical Engineering.* 2013: 6127-6134.
[13] Gaing ZL. A Particle Swarm Optimization Approach for Optimum Design of PID Controller in AVR System. *IEEE Transactions on Energy Conversion.* 2004; 19: 384-394.
[14] Liu Y, Zhang J, Wang S. *Optimization Design Based on PSO Algorithm for PID Controller*. 5th World Congress on Intelligent Control and Automation. 2004; 3: 2419-2422.
[15] XU Wenqiang, YAN Jianhong. Derivation of Transmission Function Model of Two-phase Hybrid Stepping Motor. *Space Electronic Technology.* 2011, 3: 50-53.
[16] Lalitha MP, Reddy VCV, Usha V. Optimal DG placement for minimum real power loss in radial distribution systems using PSO. *J Theor Appl Inf Technol*. 2010; 13(2): 107-116.
[17] Sawsan Gharghory, Hanan Kamal. Modified PSO for Optimal Tuning of Fuzzy PID Controller. *IJCSI International Journal of Computer Science Issues.* 2013; 10(2): 462-471.
[18] Mahmud Iwan Solihin, Lee Fook Tack, Moey Leap Kean. *Tuning of PID Controller Using Particle Swarm Optimization (PSO)*. Proceeding of the International Conference on Advanced Science, Engineering and Information Technology. 2011: 458-461.