Improve Fuzzy-PSO PID Controller by Adjusting Transfer Function Parameters

Chaman Yadav[1] Mahesh Singh[2] Kushal Tiwari[3] Richa Tiwari[4] Rahila Parveen[5]
M.Tech Scholar[1], Sr.Asst. Prof[2], Asst. Prof[3] , Asst. Prof[4] , Asst. Prof[5],
SSTC BHILAI[12],CSIT DURG[345],
INDIA

***Abstract*- In today's world, fuzzy logic and particle swarm optimization are used to answer various engineering problems. In this paper, the proportional integral derivative (PID) controller tuning by fuzzy rule method (FRMs) and a particle swarm optimization (PSO) algorithm are proposed to improve controller by adjusting transfer function parameters. The proposed fuzzy rule function and PSO algorithm searches a high-quality solution effectively and improves the response of the controlled system by adjusting the transfer function. Simulation and comparison results are presented and that the proposed algorithm finds a PID control parameter set effectively so that the PID controller has a better control performance.**

***Index Terms*- Fuzzy Rule Method (FRMs), PID Controller, Particle Swarm Optimization (PSO) Type Style and Fonts**

I. INTRODUCTION

Here there are three parameters: differential coefficient, proportional coefficient, and integral coefficient in the PID controller. By tuning these three parameters, the PID controller can offer individualized control necessities. In current years, many intelligence algorithms are proposed to tuning the PID parameters. Where, the majority part of control loops in engineering control systems use standard PID control algorithms with fixed constraint values set through the commissioning. This is lead to the low cost and eases of design but the tuning of PID controller remains unsure lead to more complexity that is mathematical. The combination of P-I-D controller parameters depend on well-known design methods and type of methods usually requires a mathematical model, which can accurately describe the dynamical performance of a control object [1].On the other hand, there are many linear methods used in the design of PID stables termed as conservative controller however, research is a great deals with carried out in the design of unusual controllers using modern computational techniques such as fuzzy logic and neural networks [2]. If this controller is applied to a nonlinear control system, so that the performance of the system will change depending on the variation of control objects parameters [3]. R.Swaroop, B. george and. P.K, Sadhu Proposed a novel design for automatic tuning of PID controller using sugeno based fuzzy logic [4]. Also, the procedure of a linear control law will cause diverse responses of a nonlinear system for the same magnitude of positive and negative reference input changes. Various design strategies will have been developed with the point to overcome the disadvantages of linear P-I-D controllers. Such methods created for achieving a goal transform a linear P-I-D controller into unconventional PID controllers [5].To overcome these difficulties, various types of modified conventional PID controllers such as auto tuning and adaptive PID controllers were developed lately [6,7], [8]. In addition, a class of nonconventional type of PID controller employing fuzzy logic has been designed and simulated for this purpose [9, 7, 10].

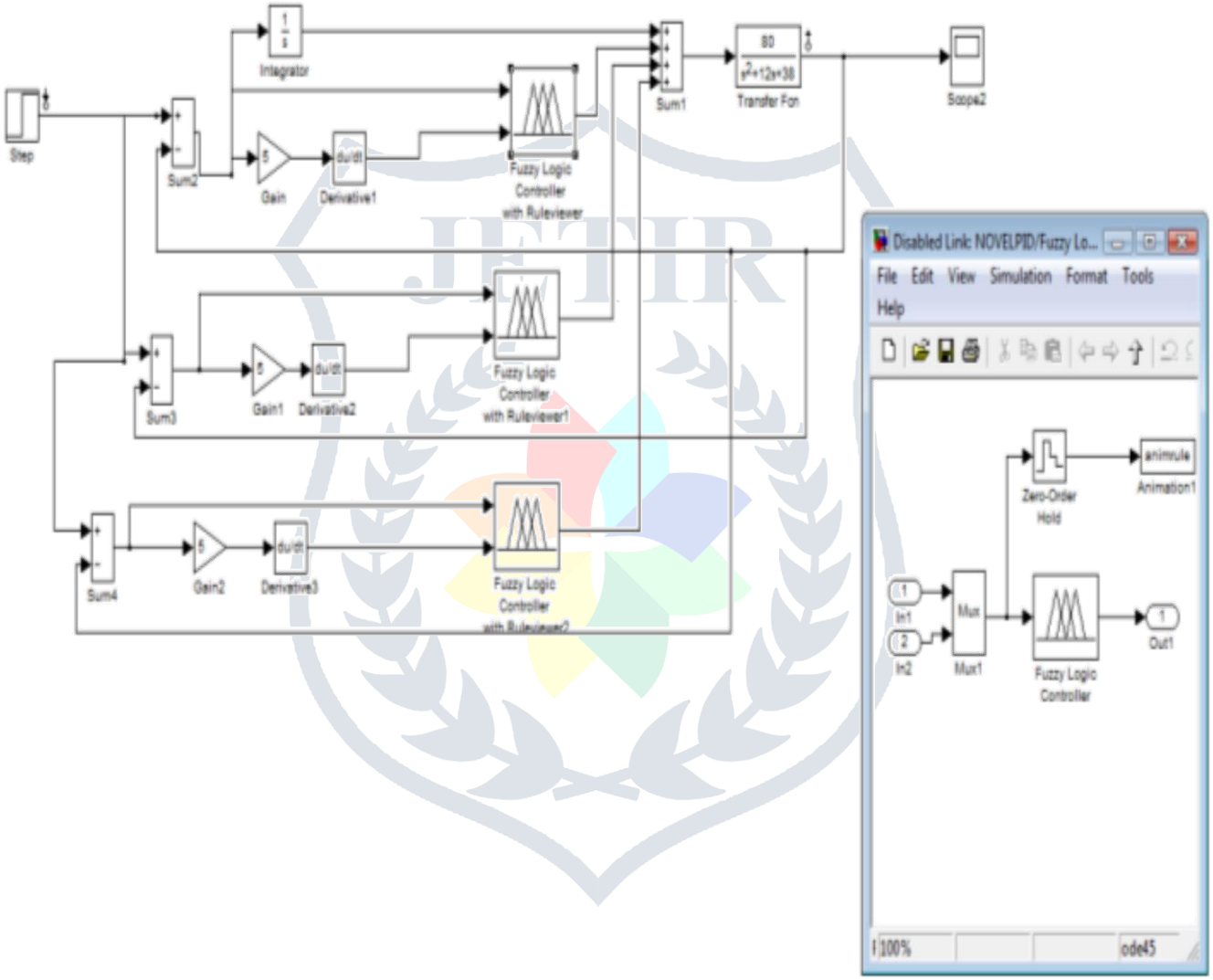
The PSO algorithm, proposed by Kennedy and Eberhart [11] in 1995, is an additional popular optimal algorithm. It was developed through a simulation of a simplified societal system where some papers were proposed to progress the PSO algorithm [12,13]. The PSO technique can produce a high quality solution within a shorter computation time and have a steady meeting characteristic than other stochastic methods [14,15]. It has many applications in engineering fields. In the PID controller design, the PSO algorithm is applied to search a finest PID control parameters [16, 17]. A lot of research papers provided many advanced methods for the particle swarm optimization algorithms [18]. In this paper, the PID controller tuning by fuzzy rule method (FRMs) and a particle swarm optimization (PSO) algorithm are proposed to improve controller by adjusting transfer function parameters. The proposed fuzzy rule method (FRM) and PSO algorithm are described in Section II. MATLAB simulation results and some comparison results are shown in Section III. Finally, conclusions are made in Section IV.

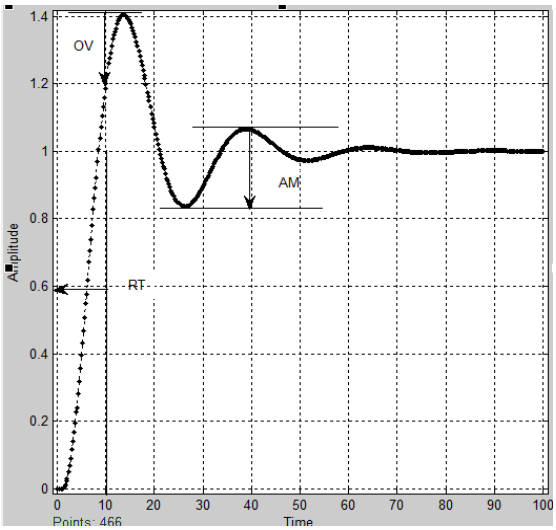
II. MYTHOLOGY

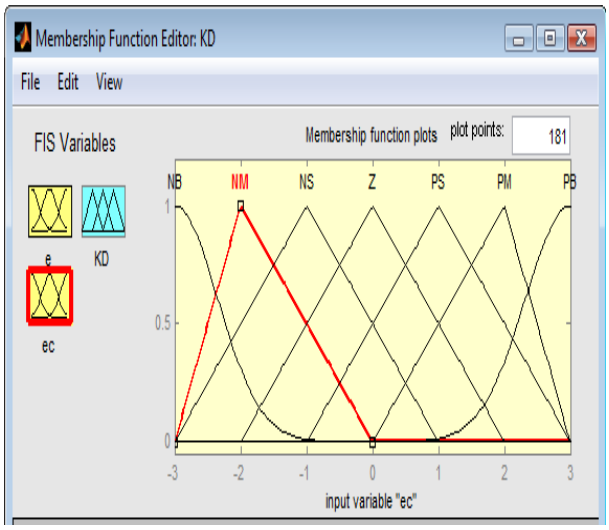
A. Design Mythology of Fuzzy system

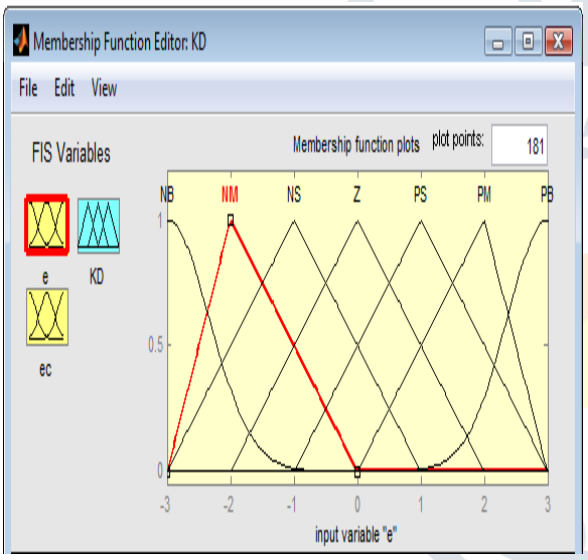
The fuzzy related matrix is used for representing the Fuzzy rules. The number of input to the rules determines the dimension of a fuzzy associated matrix [19].The Rule base is designed based on the following concepts. The ranges of input membership functions and the fuzzy rules are entered from the available data.

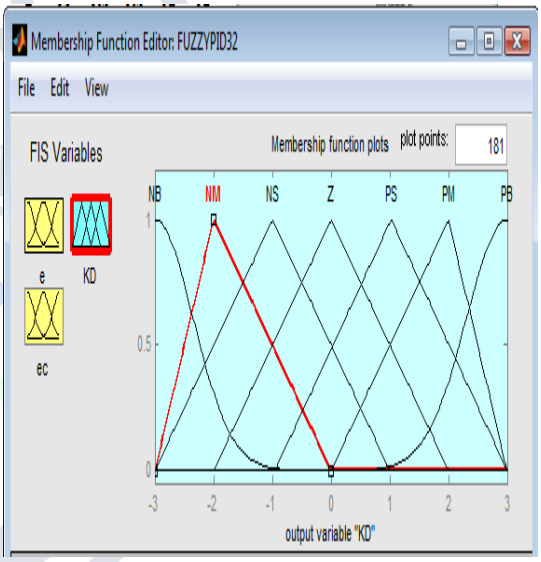
The following steps can describe the procedure of the proposed fuzzy rule method (FRM):
Step 1: Initialize (FIS editor) the input of fuzzy logic controller error (e), change of error (ec) and output (KD),(KP) and (KI) .
Step 2: Set the system with three fuzzy logic controller(KD,KP,KI) each controller has two inputs,(e,ec) and each input has seven fuzzy set associated with it, which labelled as negative big (NB), negative medium (NM),negative small (NS),Zero error (Z), positive small (PS), positive medium(PM) and positive big(PB) and the output (KD,KP,KI)depend on equation {[NB(-6,-3,-1),NM(-3,-2,0),NS(-3,-1,1),Z(-2,0,2),PS(-1,1,3),PM(0,2,3],PB(1,3,6)} Step 3: Set the input range from -3 to 3, where the input e and ec are shown in the following equations.
ec={[NB(-6,-3,-1),NM(-3,-2,0),NS(-3,-1,1),Z(-2,0,2),PS(-1,1,3),PM(0,2,3],PB(1,3,6)}
e={[NB(-6,-3,-1),NM(-3,-2,0),NS(-3,-1,1),Z(-2,0,2),PS(-1,1,3),PM(0,2,3],PB(1,3,6)}
Step 4: set the output range from -3 to 3, where the output setting depends up rule base on Table I

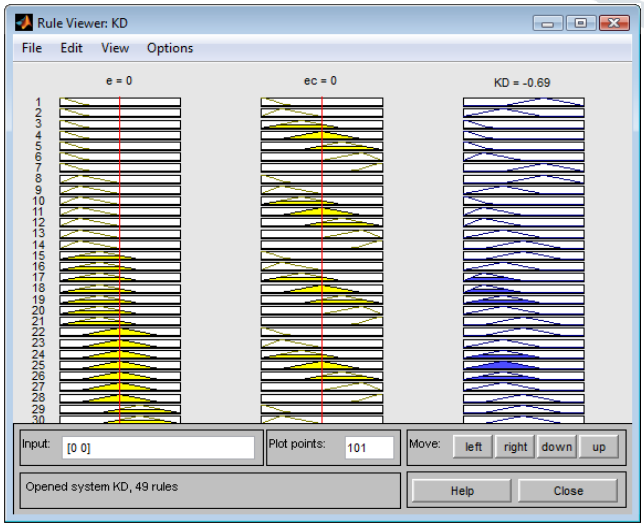



OV
AM
RT
Amplitude
Time
Points: 466


Membership Function Editor: KD
File Edit View
FIS Variables
Membership function plots
plot points:
181
e
KD
ec
NB NM NS Z PS PM PB
input variable "ec"


Membership Function Editor: KD
File Edit View
FIS Variables
Membership function plots
plot points:
181
e
KD
ec
NB NM NS Z PS PM PB
input variable "e"


Membership Function Editor: FUZZYPID32
File Edit View
FIS Variables
Membership function plots
plot points:
181
e
KD
ec
NB NM NS Z PS PM PB
output variable "KD"


Rule Viewer: KD
File Edit View Options
e = 0
ec = 0
KD = -0.69
Input: [0 0]
Plot points: 101
Move: left right down up
Opened system KD, 49 rules
Help
Close

To improved the response of the system by change transfer function parameter until we reach the optimal response of the system.

I. At starting keeps the denominator coefficients are equal to (1, 4, 4) and change a numerator from 1 to 80 with step 20, the response is not improved where the rising time is very long as shown in fig 8.

II. Set the coefficient of (S^2) equal to 1 and the coefficient of (S) equal to 4 try to change the constant coefficient from 1to 40 with step 20 at the same time change the numerator from 1 to 80 with step 40. We observe decrease the rise time (RT) where the system takes short time to reach the steady state as showing in figures 9,10.

III. Keeps the coefficient of (S^2) equal to 1 and the constant equal to 40, try to change the coefficient of (S). When increasing in value of a coefficient of (from 4 to 8) (S) the peak overshoot (OV), and amplitude (AM) decrease as showing in figures 11, 12 so that the system got good response (takes short time to reach the steady state).

IV. However, to reduce the swing, rise time (RT), and reach quickly to steady state by set the coefficient of (S^2) equal to 1, the coefficient of (S) equal to 8 and change the constant from (1to 40 with step 20) and numerator from (1to 80 with step40), until get the optimal response for system as showing in figures

B. PSO Algorithm

PID controller tuning with Particle Swarm Optimization

Step 1: Initialize the PSO algorithm by setting the number of particles (n), the number of iterations (L),dimension of the problem(dim), PSO parameters are c1 = c2 = 2,i=0 ,and PSO moment of inertia w=0.9,Fitness=i*ones(n,L),

Step 2: fitness=0*ones (n, bird_setp);

Step 3: Initialize the parameter
R1 = rand (dim, n); R2 = rand (dim, n);
Current_fitness =0*ones (n, 1);

Step 4: Initializing swarm and velocities and position
Current_position = 10*(rand(dim, n)-.5);
Velocity = .3*randn (dim, n); Local_best_position = current_position;

Step 5: Evaluate initial population
i-th particle of the population (loop for i = 1:n , end) current_fitness (i) = tracklsq (current_position(:,i)); local_best_fitness = current_fitness; [global_best_fitness,g] = min (local_best_fitness) ;
i-th particle of the population (loop for i = 1:n , end) globl_best_position (:,i) = local_best_position(:,g) ;

Step 6: Velocity update Velocity = w *velocity + c1*(R1.*(local_best_position-current_position)) + c2*(R2.*(globl_best_position-current_position));

Step 7: Swarm update Current_position = current_position + velocity;

Step 8: Evaluate anew swarm (M*ain Loop)* Iter = 0 ;(Iterations ,,counter) While (iter < bird_setp) Iter = iter + 1;
(Loop for i = 1:n,) current_fitness (i) = tracklsq (current_position (:,i)) ;
(Loop for i = 1: n, end) If current_fitness (i) less than local_best_fitness (i) local_best_fitness (i) = current_fitness (i); local_best_position (:,i) = current_position(:,i) ; End if [current_global_best_fitness,g] = min (local_best_fitness); If current_global_best_fitness less than global_best_fitness global_best_fitness = current_global_best_fitness; (Loop for i = 1: n, end) Globl_best_position(:,i) = local_best_position(:,g); End if Velocity = w *velocity + c1*(R1.*(local_best_position-current_position)) + c2*(R2.*(globl_best_position_current_position)); current position = current position + velocity; sprint ('The value of interaction iter %3.0f ', iter); End (end of while loop) Xx=fitness (:, 50); [Y,I] = min(xx); current_position(:,I)

IV. CONCLUSION

The Design is carried out in MATLAB and has been observed that the system response is improved by setting the transfer function parameters until we reach an optimal response of the system after implementing the value of PID constants obtained from the fuzzy rule method (FRMs) and PSO. The proposed methodology gives better performance in the rise time; peak overshot and the steady-state error. The response observed from the present PSO controller has a slight over shoot that can be further improved by setting the value of the parameter coefficients of fourth order transfer function by veering the constant value between 0.5to 0.75. However, it is observed from the simulation that the PID (PSO PID and FRM PID) controller performs improved in FRM PID (the rise time (RT), settling and steady state) better than PSO PID.

REFERENCES

[1] W. R. Hwang and W. E. 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, pages 1383–1388. IEEE Computer Society, June 26-29 1994.

[2] Zulfatman and M. F. Rahmat. Application of self-tuning fuzzy PID controller on industrial hydraulic actuator using system identification approach. International Journal on Smart Sensing and Intelligent Systems, 2(2), June 2009

[3] Seema Chopra, R. Mitra, and Vijay Kumar. Auto tuning of fuzzy PI type controller using fuzzy logic. International journal of computational cognition, 6(1), Mar. 2008.

[4] R.Swaroop, B. george and . P.K, Sadhu. A novel design for automatic tuning of PID controller using sugeno based fuzzy logic. International journal of computational cognition, September 2010. 8(3)

[5] Jihong Lee. On methods for improving performance of PI-type fuzzy logic controllers. IEEE Transactions On Fuzzy Systems, 1(4), Nov. 1993.

[6] K. J. Aström, "Intelligent tuning," in Adaptive Systems in Control and Signal Processing, L. Dugard, M. ˝Saad, and I. D. Landau, Eds. Oxford, U.K.: Pergamon, 1992, pp. 360–370.

[7] K. J. Aström, C. C. Hang, P. Persson, andW. K. Ho, "Toward intelligent PID control," Automatica, vol. 28, pp. 1–9, 1992. [8] G. Chen, "Conventional and fuzzy PID controllers: An overview," Int. J. Intell. Control Syst., vol. 1, pp. 235–246, 1996.

[8] J. Carvajal, G. Chen, and H. Ogmen, "Fuzzy PID controller: Design,performance evaluation, and stability analysis," Inform. Sci., vol. 123, pp. 249–270, 2000.

[9] W. M. Tang, G. Chen, and R. D. Lu, "A modified fuzzy PI controller for a flexible-joint robot arm with uncertainties," Int. J. Fuzzy Sets Syst., vol. 118, pp. 109–119, 2001.

[10] Kennedy, J. and Eberhart, R., "Particle Swarm Optimization," IEEE International Conference on Neural Networks, pp. 1942_1948 (1995).

[11] Shi, Y. and Eberhart, R., "A Modified Particle Swarm Optimizer," IEEE Congress on Evolutionary Computation, May, pp. 69_73 (1998).

[12] Shi, Y. and Eberhart, R. C., "Empirical Study of Particle Swarm Optimization," IEEE Congress on Evolutionary Computation, July, pp. 1945_1950 (1999).