Design of Fuzzy Logic Controller Based on Differential Evolution Algorithm

Li Shuai[1] and Sun Wei[2]

[1] School of Mechatronic Engineering and Automation, Shanghai University, Shanghai, 200072
lishuaicumt@126.com

[2] College of Information & Electrical Engineering, China University of Mining & Technology, Xuzhou, 221008
sw3883204@163.com

Abstract. In order to overcome the deficiency of fuzzy control algorithm, an adaptive fuzzy logic controller is proposed. In this method, the differential evolution algorithm (DE) was employed to optimize parameters of fuzzy controller: quantitative factor and proportional factor , they were designed as individuals of DE population, and evaluated using the fitness function provided until the termination condition was fulfilled. Then the selected parameter values were sent back to fuzzy logic controller. Simulation results concerning two-tank system show that the DE optimized fuzzy controller has good adaptability, as well as it`s effectiveness, which provides a new approach to improve fuzzy control system.

Keywords: Fuzzy Logic Control, Differential Evolution Algorithm, Optimization.

1 Introduction

In1974, the British scholar E. H. Mamdani applied fuzzy logic control to boilers and steam engines controls, which indicated the birth of fuzzy control theory. Now fuzzy logic controllers have been widely applied to industrial process controls, and show to be a more accurate and efficient method. Although both fuzzy control theory and technology have been developed rapidly, there are still some shortcomings such as how to choose control rules, as well as how to adjust quantitative factors and proportional factor and so on. Some authors [1], [2] presented adaptive fuzzy controllers, which can be adjusted automatically by changing certain parameters such as shape and location of suitable membership functions. Pintu Chandra Shill et al [3] used QGA to optimize fuzzy control rules and membership function. However, the performance of the fuzzy controller mostly depends on human experience, and requires tedious trial and error processes, and it can’t quickly converge to the optimal values.

Differential Evolution (DE) was proposed by R. Storn and K. Price in 1995[4]. As a new evolutionary computing technique, DE has some good properties, and it is simple, robust yet efficient in solving the global optimization problem. J. Vesterstrom

M. Fei et al. (Eds.): LSMS/ICSEE 2014, Part II, CCIS 462, pp. 18–25, 2014.

made a comparative study of differential evolution, evolutionary algorithms and particle swarm optimization on numerical benchmark problems, the results show that, DE outperforms PSO and other evolutionary algorithms[5].

In this paper, we proposed an optimized fuzzy logic controller based on differential evolution algorithm, through the quantitative factor and proportional factor to enhance the optimal design of fuzzy controller performance, and applied it to two-tank system.

2 Differential Evolution Algorithm

Differential Evolution (DE) is an evolutionary algorithm, which optimize problems by cooperation among individuals within populations and competition to. To apply a DE algorithm, we should generate a population, in which all individual initial values are chosen at random within bounds set by the user. Each individual is real-coded-dimensional vector depending on the problems. In the evolutional process, vectors in every generation undergo evolution through natural selection. Every vector crossover with a donor vector generated by mutating to form a trial vector, if cost function of the trial vector is less than that of the old ones, the trial vector replace the old to form next generation. The process will not end unless the termination condition is satisfied[6].Usually, the performance of a DE algorithm depends on three variables: the population size NP, the mutation scaling factor F, and the crossover rate CR. Fig.1 shows steps of the DE algorithm[7].

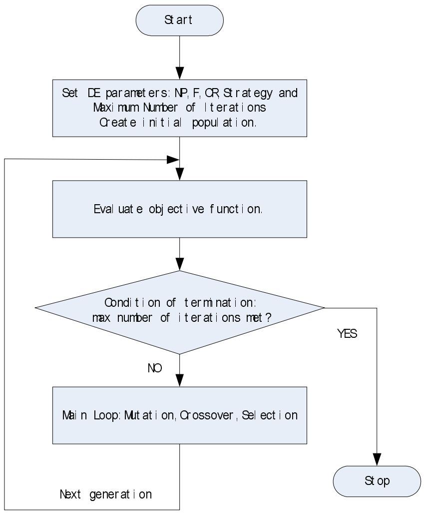

Fig. 1. Steps of the differential evolution algorithm

3 Fuzzy Logic Control Based on Differential Evolution Algorithm

3.1 The Design of Individual

Differential evolution is a parallel direct search method, it utilizes NP vectors, each of dimension D, which is the number of decision variable in the optimization problem. The DE vector $x_{i,j}$ is:

$$x_{i,j}(i = 1,2,..., NP;\ j = 1,2,..., D) \tag{1}$$

where i is the individual index, j is dimension index.

In this paper, we choose quantitative factors k_e、k_{ec} and proportional factor k_u to constitute the individual x [8]:

k_e	k_{ec}	k_u

Fig. 2. The structure of individual

where k_e and k_{ec} denote quantitative factor of inputs, k_u denotes proportional factor of output. All these are based on real-coded.

The initial population is selected randomly in the bounds defined for each variable x. These bounds are specified by the user according to the optimization problem. After initialization, DE performs several vector operations, in a process called evolution[9].

3.2 Differential Evolution Operations

There are three DE operations: mutation, crossover, and selection. The initial value of vector x is selected randomly. Mutation and crossover are used to generate new vectors called trial vector, and selection then determines which of the vectors will survive into the next generation[10].

(a) Mutation Operation

Mutation for each vector creates a mutant vector:

$$V_{i,G+1} = X_{i,G} + F * \left(X_{r1,G} - X_{r2,G}\right) \tag{2}$$

where the indexes r_1 and r_2 represent the random and mutually different integers generated within range $[1, NP]$ and also different from index i. F is a real number ($F \in [0,2]$) which controls the amplification of the difference vector ($x_{r1,G}$- $x_{r2,G}$).

(b) Crossover Operation

Through crossover operation, the target vector is mixed with the mutated vector, to yield the trial vector, following scheme:

$$U_{i,G+1} = \left(u_{1i,G+1}, u_{2i,G+1}, \ldots, u_{Di,G+1}\right) \tag{3}$$

Where

$$u_{ji,G+1} = \begin{cases} V_{ji,G+1} & \begin{array}{l} if\left(rand(j) \le CR\right) \\ or(j = mbr(i)), \end{array} \\ X_{ji,G} & \begin{array}{l} if\left(rand(j) > CR\right) \\ or(j \ne mbr(i)) \end{array} \end{cases}$$

for *i=1,2,…,NP*, *j=1,2,…,D*, *rand(j)* is selected uniformly randomly within *[0,1]*, *mbr(i)* represents a random integer within *[1,D]* , which is responsible for the trial vector containing at least one parameter from the mutant vector. *CR∈[0,1]*, which is a crossover parameter presenting the probability of creating parameters for trial vector from a mutant vector.

(c) Selection Operation

The selection operation selects the vector which will survive to be a member of the next generation. Which vector will be selected depending on the fitness value to evaluate the performance of the controller, it is necessary to define a fitness function. The target of our controller is to evaluate the dynamic and static characteristic of control system, such as rapid response, short adjusting time, small overshoot and small stable error etc., so the following fitness function is proposed:

$$x_{i,G+1} = \begin{cases} u_{i,G+1}, & f\left(u_{i,G+1}\right) < f\left(x_{i,G}\right) \\ x_{i,G}, & f\left(u_{i,G+1}\right) \ge f\left(x_{i,G}\right) \end{cases} \tag{4}$$

for *f(x)* is the fitness function.

3.3 The Fitness Function

To evaluate the performance of the controller, it is necessary to define a fitness function. The target of our controller is to evaluate the dynamic and static characteristic of control system, such as rapid response, short adjusting time, small overshoot and small stable error etc., so the following fitness function is proposed:

$$f_i = at_s + b|\sigma| + \sum\left(ce^2 + du^2\right) \tag{5}$$

for t_s represents regulating time. $\sigma$is the overshoot. *e* indicates the difference between the desired value and the actual value. *u* indicates the output of fuzzy controller, and *a*, *b*, *c* and *d* are constant coefficients.

3.4 The Algorithmic Step

The proposed overall tuning process works in following ways[11][12]:

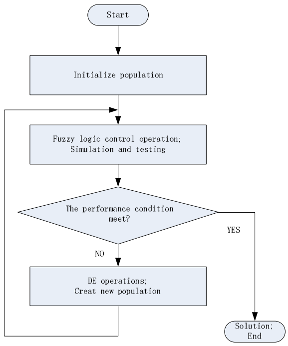

Fig. 3. Flowchart of DE based fuzzy control system

4 Simulation Results and Comparative Analysis

In this section, we applied the proposed method to the two-tank system [13]. Fig. 4 gives the structure. The experimental device includes of two water tanks T_1 and T_2, reservoir, junction valve V_1, leak valves V_2, V_3 and V_4, as well as execution structures and sensors. Water is draw from the reservoir by pump P_1, and through V_3 into water tank T_1, then through V_1 into water tank T_2, finally through V_2 go back to the reservoir. The leak valve V_4 is usually closed. We take the tank T_2 liquid level h_2 as the controlled variable.

Based on the mass conservation condition and Bernoulli law, the two-tank system model is:

$$A\frac{dh_1}{dt} = Q - Q_{12},\ A\frac{dh_2}{dt} = Q_{12} - Q_{20} \tag{6}$$

for $Q_{12} = \mu_1 S \mathrm{sgn}(h_1 - h_2)(2g|h_1 - h_2|)^{1/2}$ is the water speed from T_1 to T_2, $Q_{20} = \mu_2 S(2gh_2)^{1/2}$ is the water speed from T_2 to the reservoir, *sgn(z)* is the sign function of z, *A* is the cross sectional area of the tank, *S* for the pipe cross-sectional area, *g* is the acceleration due to gravity, μ_1, μ_2 are the flow coefficient, the values are as follows: $A = 6.3585\times10^{-3}\,\mathrm{m}^2$, $S == 6.3585\times10^{-5}\,\mathrm{m}^2$, $\mu_1 = 0.083$, $\mu_2 = 0.1133$.

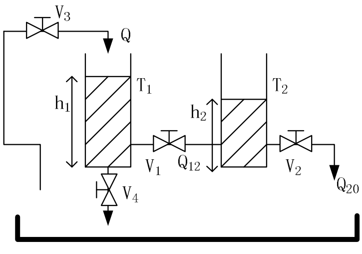

Fig. 4. The structure of two-tank system

For double tank water level control system, the aim is making liquid level of tank T2 reach the set value. Tank 1 and 2's initial liquid level is set to 0. To design fuzzy controller for double tank water level control system, we choose the error signal e(the difference between desired and actual value of h_2) and the error derivative signal de as input variables, and the control signal u (water speed Q) as the output variable. Five fuzzy sets were used for each input/output variable, like [NM NS ZE PS PM]. Then DE is used to optimize k_e, k_{ec} and k_u. In the optimal process, each vector is set as 3 dimensions and adopted the real-coded (look Fig. 2). DE population scale is NP=30, and mutation probability is F=0.6, crossover probability is CR=0.5, the maximum evolution generation is 30. The fitness function parameters are a=1, b=3, c=2, d=2. The sampling period is ts=0.01s, and The setting value of Tank 2's liquid level is : h_{20}=200mm for t=0~500s, h_{20}=350mm for t=501~1000s, h_{20}=160mm for t=1001~1600s. Fig. 5 shows the performance comparison curves.

From the simulation results, we know that, fuzzy controller has better performance than the PID controller. When the fuzzy controller is optimized by the DE algorithm, the output of the system became much better than the fuzzy controller, both in overshoot and in rapid respond time. So the method proposed in this paper is more effective.

To verify the anti-interference capability, we open the V_4 valve to add a disturbance at t=40s when the system is stable, allowing water to flow from the leak valve for some time, and then close the valve. It cost 30s for the system became stable, as shown in Fig. 6. So, it has good performance to overcome disturbance. After 30 generation, the best fitness value is 0.54784, Fig. 7 shows the fitness curve. Table 1 gives the comparison of the parameter values before and after optimization.

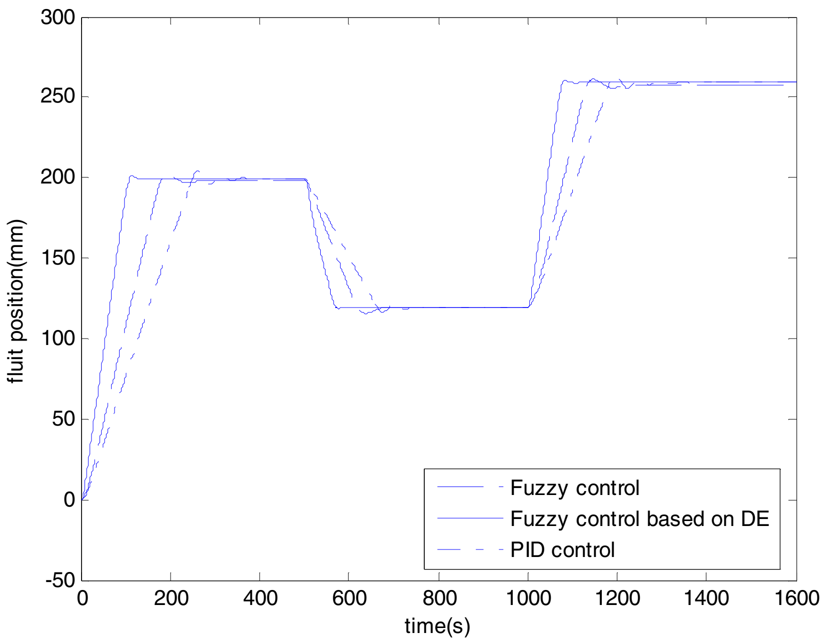

Fig. 5. Output response curves

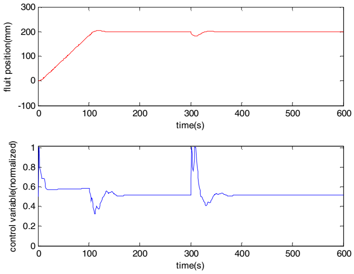

Fig. 6. The curves of output and control variable with disturbance

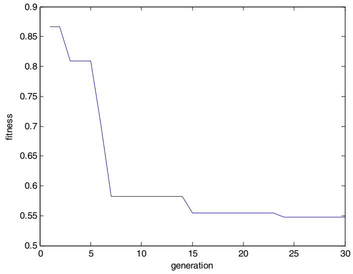

Fig. 7. Fitness function curve

Table 1. The parameter values before and after optimization

	k_e	k_{ec}	k_u
before	*30*	*1*	*1.2*
after	*32.106*	*0.69367*	*1.9472*

5 Conclusions

In this paper, we propose insight into the design of simplified tuning a fuzzy logic controller. The differential evolution (DE) algorithm is employed. The proposed method is able to optimize input and output weights k_e , k_{ec} and k_u. We applied the DE-based fuzzy logic method to deal with two-tank system, experimental results show that the proposed controller has better performance than existing fuzzy control system.

References

1. Huiwen, Y.W., Chen, D.Z.: Adaptive fuzzy logic controller with rule-based changeable universe of discourse for a nonlinear MIMO system. In: Proceedings of International Conference on Intelligent Systems Design and Applications (2005)

2. Jou, J.M., Chen, P.-Y., Yang, S.-F.: An adaptive fuzzy logic controller: its VLSI architecture and applications. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 8, 52 (2000) ISSN:1063-8210
3. Shillt, P.C., Akhand, M.A.H., Muraset, K.: Fuzzy Logic Controller for an Inverted Pendulum System using Quantum Genetic Optimization. In: Proceedings of 14th International Conference on Computer and Information Technology (ICCIT), pp. 22–24 (2011)
4. Storn, R., Price, K.: Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces. J. Global Optimize 11, 341–359 (1997)
5. Vesterstrom, J., Thomsen, R.: A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems. In: Proceedings of the IEEE Congress on Evolutionary Computation, pp. 1980–1987 (2004)
6. Cheng, S.-L., Hwang, C.: Optimal Approximation of Linear Systems by a Differential Evolution Algorithm. Transactions on Systems, Man, and Cybernetics—Part A: System and Humans 31(6) (November 2001)
7. Roger, L.S.T., Srinivas, M., Rangaiah, G.P.: Global optimization of benchmark and phase equilibrium problems useing differential evolution
8. Guo, X., Zhou, Y., Gong, D.: Optimization of fuzzy sets of fuzzy control system based on hierarchical genetic algorithms. In: Proceedings of IEEE TENCON 2002 (2002)
9. Brest, J., Zumer, V., Maucec, M.S.: Self-adaptive differential evolution algorithm in constrained real-parameter optimization. In: 2006 IEEE Congress on Evolutionary Computation, pp. 16–21 (2006)
10. Liu, J.L.: On setting the control parameter of the differential evolution method. In: Proc. 8th Int. Conf. Soft Computing (MENDEL 2002), pp. 11–18 (2002)
11. Shillt, P.C., Akhand, M.A.H., Muraset, K.: Fuzzy logic controller for an inverted pendulum system using quantum genetic optimization. In: Proceedings of 14th International Conference on Computer and Information Technology (ICCIT 2011), pp. 22–24 (2011)
12. Wonohadidjojo, D.M., Kothapalli, G., Hassan, M.Y.: Position control of electro-hydraulic actuator system using fuzzy logic controller optimized by particle swarm optimization. International Journal of Automation and Computing 10(3), 181–193 (2013)
13. Cheong, F., Lai, R.: Designing a hierarchical fuzzy logic controller using the differential evolution approach. Applied Soft Computing 7, 481–491 (2007)