Computer control software design model based on particle swarm optimization and PID algorithm

Qian Xinjie, Hu Guixiang, Fu Qiulin, Yang Bo
Yibin Vocational and Technical College, Yibin Sichuan, 644003

Keywords: computer control; Particle Swarm; software design

Abstract. The optimal design for computer control software is studied. Since computer control software is susceptible to interference during the control process, a computer control software design model based on improved PID algorithm is proposed. The PID algorithm is combined with particle swarm algorithm to calculate PID control parameters, which is viewed as evolutionary particles of the particle population, and given a certain flight speed in the search space, the speed of the particles will be adjusted iteratively and dynamically in accordance with the experience of population's evolution calculation, in order to achieve computer control software design. The simulation results show that the proposed algorithm applied for computer control software design, can improve the control precision and meet the actual needs of computer control.

Introduction

During practical computer control software design process, the controlled object processes features, like uncertainty, nonlinearity and time-varying [1]. The traditional PID linear control method based on accurate mathematical model have the defects at the aspects of systems' adaptivity and robust control [2,3]. In order to realize the adaptive control of the system with above characteristics, domestic and foreign scholars have proposed a variety of nonlinear control methods, including fuzzy control, expert system control and neural network control method [4-6]. Therefore, the computer control software design methods have become a hot issue requiring to be researched in control field, and concentrated widespread concern of many experts and scholars.

Principle of computer control software design method

The improved PSO algorithm combined with PID control algorithm, can obtain the improved PID control algorithm, computer control software design is achieved with that algorithm. The principle is as follows:

Improved Particle Swarm Optimization.

When assignment is performed for particle swarm inertia coefficient, if the mapping function is a linear decline, the extreme points of the algorithm may not be the extreme point of real dynamic systems, the particle velocity v_{ij} will be affected to deviate from the extreme point of the current environment by the larger ω, thus benefits to have a quick access to local minima search, the extreme of the entire algorithm is better. However, if the inertia coefficient ω is executed at a certain stage of the algorithm, reducing the diversity of particles of particle swarm, it will inevitably lead to more difficult to find the global extreme convergence point for the particles. Assuming that the particle space of j-th particle in the i-dimensional space at the t-th iteration calculation is $d_{ij}(t)$, then

$$d_{ij}(t) = |x_{ij}(t) - g_{besti}(t)| \tag{1}$$

When calculating the actual particle spacing, due to the aggregation properties of the particle velocity, particles will produce tight sets when classified according to speed, if the particle spacing is small, the population is easy to fall into local minima, however, the convergence is difficult when the spacing is larger. Therefore, after grouping by particle distance, the degree of particles' clustering is recorded, when the degree of particles' clustering is high, the smaller the distance between particles, and the inertia factor should be increased, when the degree of particles' clustering is lower, the coefficient of inertia is reduced to increase the convergence degree of the adaptive algorithm.

Assuming that

$$\begin{cases} d_{mean}(t) = \frac{[\sum_{j=1}^{n}\sum_{i=1}^{d} d_{ij}(t)]}{(n*d)} \\ d_{max}(t) = |max[d_{ij}(t)]| \\ k = \frac{[d_{max}(t)-\overline{d(t)}]}{d_{max}(t)} \end{cases} \quad (2)$$

Where $d_{mean}(t)$ is the average particle spacing of particle swarm, $d_{max}(t)$ is the maximum particle distance, k is the clustering degree of the current particles' state, the value is $[0,1]$, then during the particle weight calculation process, the inertia weights should be adjusted as follows:

$$\begin{cases} w(t) = w * w_{start} & k \geq \alpha \\ w(t) = w * \frac{1}{w_{end}} & k < \beta \end{cases} \quad (3)$$

Where, $\{\alpha, \beta\}$ is the inertia factor adjustment parameter.

Fast tuning for PID control parameters.

The above improved particle swarm algorithm combined with PID control algorithm, fast tuning for PID control parameters is necessary. PID control parameters is viewed as evolutionary particles of the particle population, and given a certain flight speed in the search space, the speed of the particles will be adjusted iteratively and dynamically in accordance with the experience of population's evolution calculation, when the algorithm converges to the optimal objective at a certain speed, the value of the particles at this time is given to PID control parameters. The algorithm structure of fast tuning for PID control parameters with evolutionary computation is shown in Figure 1:

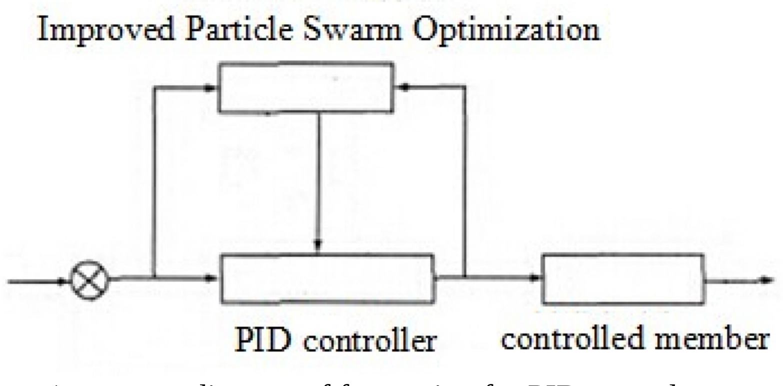

Figure 1 structure diagram of fast tuning for PID control parameters

The detailed steps of above fast tuning for PID control parameters are as follows:

(1) Parameter initialization

Including the initialization of weights of particles, learning parameters, and the number of iterations times,etc. and the velocity and position of particle are assigned;

(2) Population initialization

The population composed of n particles randomly, and each particle is set to correspond to three parameters of the PID controller;

(3) calculation of the fitness value of particles

The weight of particles is mapped and corrected, and its fitness value is calculated;

(4) determination of the particle

When the adaptive value of the particles compared to the best position historically, if it is superior, the current historical best position will be updated; when the adaptive value of the particles compared to global historical best particle, if it is superior, the current position is saved as the global optimum position;

(5) Update of the particles

The speed and position of the particle is updated iteratively;

(6) determination conditions

Calculation is tested to determine whether stopping condition is met, if yes, the iteration should be stop, otherwise turn to (3);

According to the method outlined above, it is possible to combine PID algorithm and improved particle swarm algorithm, so as to achieve computer control software design.

Example Simulation analysis

The computer control software design model based on the improved PID algorithm is applied on the low-voltage power line carrier communication platform, to achieve adaptive control for network traffic, and assist Y-NET protocol stack to improve the quality of data transmission services, so as to ensure minimal cell loss ratio and maximum utilization of network resources. Computer control software design model can accomplish the desired level by be tracking and adjusting dynamically adjusted the queue length on carrier platform, and avoid waste of network resources caused by cell loss or low traffic, due to excessive traffic.

The system input *r(k)* is set as the desired queue length, its value is determined according to the nature of the business, the available bandwidth and buffer capacity (indicated by B) and other factors, in the application, the desired queue length is taken as 256Bytes. $e(k)=r(k)-y(k)$is the queue variation, $u(k)$ is the output control of the controller, *y (k) is* output queue length, C is the service rate (available bandwidth). According to Y-NET protocol stack and control theory, C is assigned as 5Mbps, and regarded as interference; τ is the loop delay of the bottleneck between the associated nodes (transmission delay and switching delay). Thus, the queuing model of the local node the can be described by the following formula:

$$y(k+1) = Sat_B\{y(k) + \sum_{i=1}^{k} T[u(k-\tau_i) - C]\} \tag{4}$$

Wherein, $\mathrm{Sat}_B = \begin{cases} B, (x > B) \\ x, (0 < x \le B) \\ 0, (x \le 0) \end{cases}$.

The transfer function is built based on queue model

$$G_s(s) = \frac{se^{-\tau s} - C}{s^2} \tag{5}$$

Under the same simulation conditions, with different algorithms to perform computer control, their response time were compared. The results obtained are shown below:

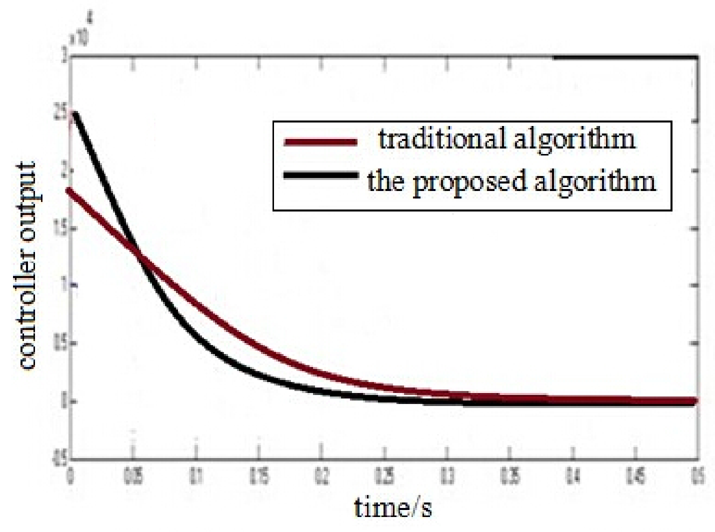

Figure 2. Comparison of response time with different algorithms

It can be seen from the above chart that the proposed algorithm has a stronger capability of adjusting the flow of platform, the output response time of funning is shorter, so the ability to prevent congestion is stronger, which enables to reduce packet loss while taking full advantage of the buffering capacity.

Conclusion

This paper presents a computer control software design model based on improved PID algorithm. The PID algorithm is combined with particle swarm algorithm to calculate PID control parameters, which is viewed as evolutionary particles of the particle population, and given a certain flight speed in the search space, the speed of the particles will be adjusted iteratively and dynamically in accordance with the experience of population’s evolution calculation, in order to achieve computer control software design. The simulation results show that the proposed algorithm applied for computer control software design, can improve the control precision and meet the actual needs of computer control.

References

[1] Ken Yeh, Cheng-Wu Chen, DC Lo, Kevin FR Liu. Neural-network fuzzy control for chaotic mass damper systems with time delays. *Vibration and Control*. Vol.18, no.6,pp.785-795,2012.2-3

[2] Xu Liancheng. Intelligent constant temperature control system based on PROTEUS simulation [J]. Electronics World, 2013, (12):156-156.

[3] Liu Kun, Shi Jianfei. Constant Temperature Control System Based on Fuzzy-PID [J]. Science & Technology Information, 2013, (19):2-2.

[4] G. Cheng and K. Peng, “Robust composite nonlinear feedback control with application to a servo positioning system,” *IEEE Trans. Ind. Electron.*, vol. 54, no. 2, pp. 1132–1140, Apr. 2007.22

[5] Z. R. Radakovic, V. M. Milosevic,S. B. Radakovic. Application of Temperature Fuzzy Controller in an Indirect Resistance Furnace [J]. Applied Energy, 2002, 73:167-182.

[6] Jen Yang. Chen. Rule Regulation of Sliding Mode Controller Design: Direct Adaptive Approach [J]. Fuzzy Sets and Systems, 2001,120:159-168