Enhancement of Particle Swarm Optimization by Stabilizing Particle Movement

Hyunseok Kim, Seongju Chang, and Tae-Gyu Kang

We propose an improvement of particle swarm optimization (PSO) based on the stabilization of particle movement (PM). PSO uses a stochastic variable to avoid an unfortunate state in which every particle quickly settles into a unanimous, unchanging direction, which leads to overshoot around the optimum position, resulting in a slow convergence. This study shows that randomly located particles may converge at a fast speed and lower overshoot by using the proportional-integral-derivative approach, which is a widely used feedback control mechanism. A benchmark consisting of representative training datasets in the domains of function approximations and pattern recognitions is used to evaluate the performance of the proposed PSO. The final outcome confirms the improved performance of the PSO through facilitating the stabilization of PM.

Keywords: Particle swarm optimization, proportional-integral-derivative control, neural network.

I. Introduction

Particle swarm optimization (PSO) has been applied successfully in many application areas over the last decade and is known to be particularly effective in solving large-scale nonlinear optimization problems [1]. PSO is an evolutionary computation algorithm [2] and is used in the simulation of graceful but unpredictable choreography of bird flocks.

As shown in Fig. 1, the main concept of PSO is to match the nearest-neighbor velocity (acceleration) and to weigh the acceleration using a random term (craziness) [2]. In particular, the stochastic variable called "craziness" is used to avoid the unfortunate state of every particle quickly settling into a unanimous, unchanging direction. Such a uniform (0, 1) distributed random number can lead to overshoot around the optimum position, resulting in slow convergence. Hence, it is necessary that each particle track the global best position securely while preserving the characteristics.

The proportional-integral-derivative (PID) approach is a well-recognized feedback control mechanism [3]. The advantages of the PID approach include its easy implementation and the fact that only three parameters need to be adjusted: the proportional term, K_P, depends on the present error; the integral term, K_I, is subject to an accumulation of past errors; and the derivative term, K_D, is a prediction of future errors [4]. Furthermore, the PID approach is a sufficient way to track the output of a system toward a desired set-point with fast convergence, low overshoot, and low steady-state error under the condition that the three parameters are adjusted to the optimum values for the desired control response. This study therefore uses a PID approach to support each particle to securely move toward the global best position, which could result in a significant performance enhancement of the PSO.

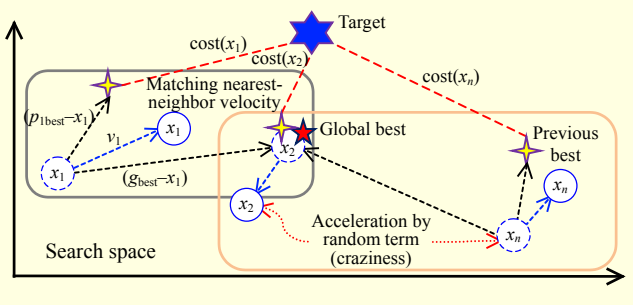


Fig. 1. Conceptual framework of PSO with nearest neighbor velocity matching and craziness [2].

Manuscript received Apr. 25, 2013; revised May 15, 2013; accepted June 3, 2013.

This work was supported by the IT R&D program of KEIT [10040037, An intelligent IT lighting system development using interactive information exchange in complex space].

Hyunseok Kim (phone: +82 42 860 5762, hertzkim@etri.re.kr) and Tae-Gyu Kang (tgkang@etri.re.kr) are with the IT Convergence Technology Research Laboratory, ETRI, Daejeon, Rep. of Korea.

Seongju Chang (schang@kaist.ac.kr) is with the Department of Civil and Environmental Engineering, KAIST, Daejeon, Rep. of Korea.

© 2013 ETRI

II. PSO and Particle Movement

PSO optimizes a problem by utilizing individual particles moving around the search space according to (1) and (2) over its velocity and position [5]-[7]. Each particle position is adjusted to the best position found and is updated as better positions are found by other particles. As the model is iterated, the particles can ascertain the best position found thus far. The length of vectors x_i and v_i equals the dimensionality of the search space.

$$v_i[n] = \chi \cdot \left\{ \begin{array}{c} \omega \cdot v_i[n-1] \\ +c_1 \cdot rand_1 \cdot (p_{i_{\text{best}}}[n-1] - x_i[n-1]) \\ +c_2 \cdot rand_2 \cdot (g_{\text{best}}[n-1] - x_i[n-1]) \end{array} \right\}, \quad (1)$$

$$x_i[n] = x_i[n-1] + v_i[n], \quad (2)$$

$$Z_k \approx 10 + 2 \cdot \sqrt{\text{dimension}}. \quad (3)$$

The number of particles can be determined by (3) as described in [6]. In (1), ω is the inertial constant that dynamically adjusts the velocity, and c_1 and c_2 are cognitive and social constants, respectively, which change the velocity of particles toward the previous best position and the global best position. The previous best position is found through a cost evaluation for its own previous best positions, as in (4). The global best position is selected through a cost evaluation for all $p_{i_{\text{best}}}$ as in (5). In addition, the constriction factor defined as (6) is introduced to further speed up the convergence [7].

$$p_{i_{\text{best}}}[n] = \begin{cases} x_i[n] \text{ if } \text{cost}(x_i[n]) \le \text{cost}(p_{i_{\text{best}}}[n-1]), \\ p_{i_{\text{best}}}[n-1] \text{ otherwise}, \end{cases} \quad (4)$$

$$g_{\text{best}}[n] = \left\{ p_{j_{\text{best}}}[n] \,|\, \forall i : \text{cost}(p_{j_{\text{best}}}[n]) \le \text{cost}(p_{i_{\text{best}}}[n]) \right\}, \quad (5)$$

$$\chi = \frac{2}{|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}|}, \text{ where } \varphi = c_1 + c_2, \ \varphi > 4. \quad (6)$$

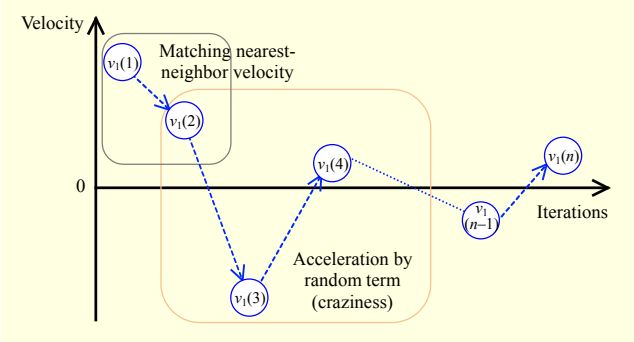


Fig. 2. Zigzagging tendency activated by random term.

According to the main idea of PSO, the position of each particle is updated based on a term attracting the particle toward its own previous best position and the global best position. Essentially, this simple rule creates a synchrony of movement [2]. Thus, the PSO uses uniform random numbers $rand_1$ and $rand_2$ as stochastic variables, that is, craziness, to avoid an unfortunate state in which all particles quickly settle into an unchanging direction. As shown in Fig. 2, the presence of random numbers increases the zigzagging tendency around the optimum position and slows down the convergence [8]. The distance error between each particle and the global best position can be oscillated and therefore makes it difficult to meet the termination criteria. Thus, a new approach is required to move the particles smoothly to prevent a premature convergence to non-optimal positions.

III. Stabilizing Particle Movement of PSO

We propose a novel way to change the position of a particle with reduced oscillations using the PID approach described through (7). Velocity adjustments are made based on the acceleration through the distance error calculated with its own current position and the best previous position compared with the global best position, respectively. In addition, two uniform distributed random numbers are used to preserve craziness. The updated velocity is regarded as a momentum factor to move each particle and is considered the present error in our PID control. Thus, the updated position of each particle is calculated through the multiplication between the velocity and the three terms of the PID (K_P, K_I, and K_D). We select the three terms through trial and error operations.

Table 1. Tunable and default parameters.

Training parameter	Value
Swarm size (Z_k)	Tuned
Minimum position (*min_pos*)	1.2×min (LM-NN)
Maximum position (*max_pos*)	1.2×max (LM-NN)
Inertial constant (ω)	1
Cognitive constant (c_1)	2.05
Social constant (c_2)	2.05
Use constriction factor	True
Clip the particle position	True
Maximum initial velocity	1
Minimum norm of velocity	0.05
Proportional term (K_P)	0.5 (fixed)
Integral term (K_I)	0.4 (fixed)
Derivative term (K_D)	0.3 (fixed)

$$x_i[n] = x_i[n-1] + \left\{ \begin{array}{c} K_P \cdot v_i[n] \\ +K_I \cdot \sum_{k=0}^{n} v_i[k] \\ +K_D \cdot \left(v_i[n] - v_i[n-1]\right) \end{array} \right\}. \quad (7)$$

The dynamic range of each particle position needs to be limited to prevent a swarm from becoming unstable or exploding. We set the range of each particle position within the two parameters: *max_pos* and *min_pos*, which are calculated from the output of the precedent Levenberg-Marquardt neural network (LM-NN) training method and experimental trials. We set the initial global best position to the optimal configuration trained by the precedent LM-NN method in the same way as in [9]. Consequently, randomly spread particles can more precisely assure a global optimal solution than LM-NN. Table 1 shows the prevailing default parameters used in [5] through [10] and the tunable parameters determined from each experiment. The stabilized PSO in which individual particles are evaluated and move is in a better position to smoothly find the best optimal solution with no local minima. The proposed PSO converges faster and has a higher performance than before. Moreover, it is attractive because there are very few parameters to adjust and it can be easily implemented in the real world.

IV. Simulation Outcomes

The performance evaluation for the proposed PSO is carried out to investigate whether the suggested method is sufficiently robust to improve the stability of particle movement (PM) and the speed of convergence with different experimental datasets used to train a neural network. For this exploration, we use exemplary datasets acquired from MATLAB: *Engine*, *Simplefit*, *Iris*, *Glass*, *House*, *Abalone*, *Bodyfat*, *Building*, *Chemical*, *Cancer*, *Thyroid*, and *Wine* [11]. These datasets are representative training datasets in the fields of function fitting and pattern recognition and classification. To ensure fair comparison and reliable performance evaluation of the proposed approach, the three terms of the PID are fixed as follows: K_P=0.5, K_I=0.4, and K_D=0.3. Other parameters remain the same for both the ordinary PSO and the proposed PSO. The optimization process is supposed to be terminated when the updated velocity of all particles is within the minimum criteria: 0.05. To compare the stability, the total sum absolute error (TSAE) and mean absolute error (MAE) are calculated as

$$TSAE = \sum_{n=\text{iterations}} \sum_{i=\text{particles}} |v_i[n]|, \quad (8)$$

$$MAE_{n=\text{iterations}} = \frac{\sum_{i=\text{particles}} |v_i[n]|}{\text{number of particles}}. \quad (9)$$

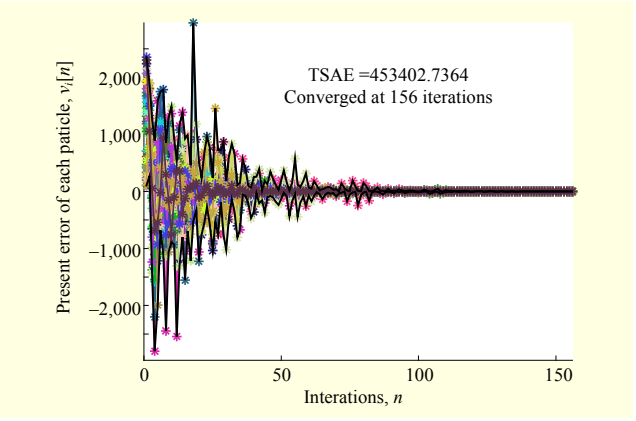


Fig. 3. Unstable PMs of ordinary PSO.

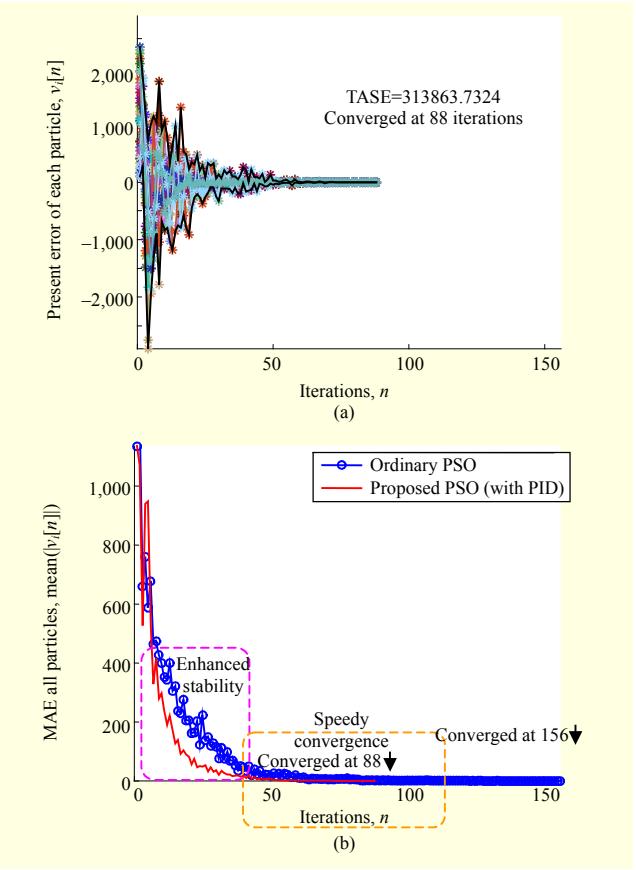
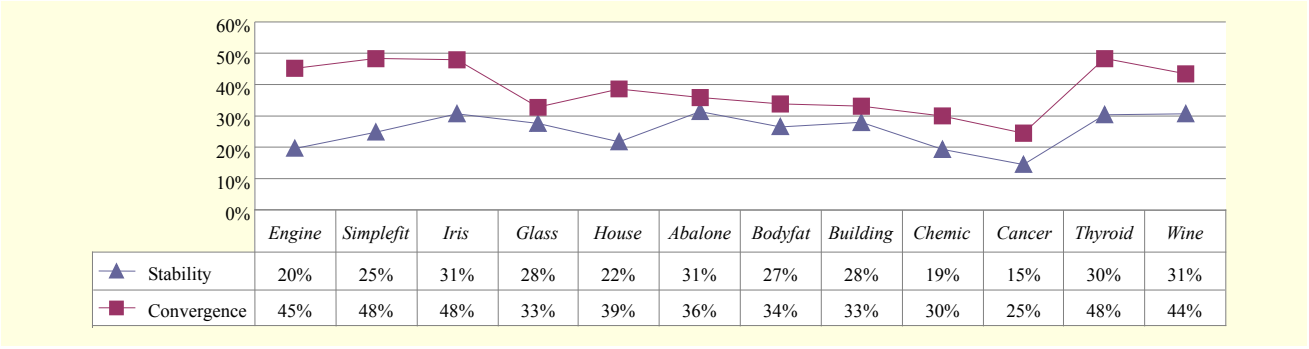


Fig. 4. (a) Improvement in stabilization of PMs. (b) Comparison between ordinary and proposed PSO.

As the result of stabilization, the time needed for convergence is shorter than that needed for ordinary PSO. Thus, the number of iterations required for the convergence is counted for the comparison of the speed of convergence.

We firstly evaluate the movement of each particle during the process using a *Wine* dataset consisting of classifiers derived from three wineries in relation to thirteen constituents found through chemical analysis. As shown in Fig. 3, the zigzagging

ETRI Journal, Volume 35, Number 6, December 2013
http://dx.doi.org/10.4218/etrij.13.0213.0197



	Engine	Simplefit	Iris	Glass	House	Abalone	Bodyfat	Building	Chemic	Cancer	Thyroid	Wine
Stability	20%	25%	31%	28%	22%	31%	27%	28%	19%	15%	30%	31%
Convergence	45%	48%	48%	33%	39%	36%	34%	33%	30%	25%	48%	44%

Fig. 5. Graph of results of enhancement for stable PMs (blue triangle) and speedy convergence (red rectangle).

movements of each particle are not trivial and slowly converge at 156 iterations. The improvement in the stabilization of each particle by using the proposed PSO is shown in Fig. 4(a): a particle can rapidly converge at 88 iterations. The result clearly demonstrates that the performance of the proposed PSO is better than the ordinary PSO, as illustrated in Fig. 4(b).

The stability indicates the overshoot or undershoot value, calculated by (10). How quickly all particles converge is calculated by (11). The performance of the proposed PSO is an improvement over that of the ordinary PSO, as shown in Fig. 5: approximately 25% average enhanced stability and 38% faster convergence.

$$\text{Stability enhancement} = (1 - \frac{\text{TSAE of proposed PSO}}{\text{TSAE of ordinary PSO}}), \quad (10)$$

$$\text{Convergence enhancement} = (1 - \frac{\text{Iterations of proposed PSO}}{\text{Iterations of ordinary PSO}}), \quad (11)$$

Through all evaluative processes, the proposed PSO proves to be suitable to support each particle to securely track the global best and preserve the characteristics necessary to prevent every particle from quickly settling into the local minima.

V. Conclusion

We presented a novel approach for the improvement of PSO through stabilizing PM based on a widely spread PID feedback control mechanism. The proposed PSO can perform better than ordinary PSO in accordance with the enhancement in stability and convergence. The Kalman filter, a useful algorithm for calculating the noise-reduction path over the long term [12] can also support an object with noise for following a target position well. We believe our research can greatly contribute to the performance enhancement of an ordinary PSO, especially in such aspects as the minimization of overshoots or undershoots, convergence acceleration, and the diversified use of the PSO algorithm. In particular, the proposed PSO could be successfully applied to the problem domains demanding a fast evolutionary programming algorithm such as tracking dynamic systems and tackling multi-objective optimization.

References

[1] R.C. Eberhart and Y. Shi, "Guest Editorial Special Issue on Particle Swarm Optimization," *IEEE Trans. Evolutionary Comput.*, vol. 8, 2004, pp. 201-203.

[2] J. Kennedy and R. Eberhart, "Particle Swarm Optimization," *Proc. IEEE Int. Conf. Neural Netw.*, vol. 4, Nov./Dec. 1995, pp. 1942-1948.

[3] M. Willis, "Proportional-Integral-Derivative Control," Department of Chemical and Process Engineering, University of Newcastle, 1999. http://lorien.ncl.ac.uk/ming/pid/PID.pdf

[4] M. Araki, "PID Control," *Control Syst., Robotics, Automation*, vol. 2, 2002.

[5] Y. Shi and R. Eberhart, "Parameter Selection in Particle Swarm Optimization," *Evolutionary Programming VI*, 1998, pp. 591-600.

[6] Y. Shi, "Particle Swarm Optimization: Developments, Applications and Resources," *Proc. Congress Evolutionary Comput.*, vol. 1, 2001, pp. 81-86.

[7] M. Clerc, "The Swarm and the Queen: Towards a Deterministic and Adaptive Particle Swarm Optimization," *Proc. Congress Evolutionary Comput.*, vol. 3, 1999, pp. 1951-1957.

[8] I.C. Trelea, "The Particle Swarm Optimization Algorithm: Convergence Analysis and Parameter Selection," *Inf. Process. Lett.*, vol. 85, no. 6, 2003, pp. 317-325.

[9] H. Kim and S. Chang, "High-Resolution Touch Floor System Using Particle Swarm Optimization Neural Network," *IEEE Sensors J.*, vol. 13, no. 6, June 2013, pp. 2084-2093.

[10] R.C. Eberhart and Y. Shi, *Computational Intelligence: Concepts to Implementations*, Morgan Kaufmann, 2007.

[11] Matlab, *Sample Data Sets*. Available: http://www.mathworks.co.kr/kr/help/nnet/gs/sample-data-sets.html

[12] J. Seok et al., "A Novel Method for Bitrate Control within Macroblocks Using Kalman and FIR Filters," *ETRI J.*, vol. 33, no. 4, Aug. 2011, pp. 641-644.