Novel Technique for PID Tuning by Particle Swarm Optimization

*S. Easter Selvan, Sethu Subramanian, S. Theban Solomon

Abstract – **An attempt has been made by incorporating some special features in the conventional particle swarm optimization (PSO) technique and its usefulness was tested in a common control application involving PID controller tuning. This algorithm differs from the existing PSO methods in the inclusion of unbiased search approach of the swarm particles without establishing inter-particle communication in the initial phase itself, built-in acceleration mechanism of the particles trapped in the local minima of the objective function, preserving the initial population without reproducing those with favourable costs by cluster unification and finer search phase to identify the global minimum. The results of the aforementioned application in comparison with practical methods are quite encouraging.**

Index terms – **population, swarm particle, local minimum, cost, Euclidean distance.**

I. Introduction

SWARMING behaviour is exhibited in some of the organisms in search of conducive environmental conditions for sustenance. Optimization techniques using analogy of swarming principle have been adopted to solve a variety of engineering problems in the past decade.

Ant colony search algorithm (ACO) which mimicries the behaviour of ant, metaphor in exchanging information through pheromone was developed by Dorigo[4]. Swarming strategies in bird flocking and fish schooling are used in the particle swarm optimization technique (PSO) introduced by Eberhart and Kennedy which can be computationally inexpensive [1]. Parsopoulos and Vrahatis attempted to improve the search efficiency in PSO by performing two stage transformation of the objective function which eliminates and elevates the neighbourhood of the local minima [2]. Alternative runs and tumbles in *E-coli* bacteria found in the human intestine constitute chemotaxis and this foraging mechanism was imitated by Kevin Passino for solving optimization problem in control system [3].

That, PSO is suitable for handling combinatorial optimization problems and both discrete and continuous variables as well [7] and easy to implement with few parameters to adjust [8], enables the realization of this technique for our control application.

In the earlier PSO algorithms, each particle of the swarm is accelerated by its best previous position and towards the best particle in the entire swarm. Here, the underlying assumption is that each particle in the swarm remembers the best position already visited and also it is informed about the best particle position. In our approach attempt has been made to closely follow the searching strategy adopted by organisms in nature, favouring the capacity to remember the previously visited positions by each individual and avoiding the bias introduced in the search process by steering them towards the best in the swarm in the initial stage itself. After letting the particles to search adequate number of times in the solution space independently for the best possible positions, they are attracted to the basin containing the best particle by establishing proper communication among them about the search environment.

The PSO algorithm discussed here has been tested for a control application – tuning PID controller for optimal settling time of the plant transfer function and the results are reported to be encouraging.

This paper is organized as follows: Section II details the generation of 3 dimensional solution space by approximate PID tuning method. Unbiased search for optimal controller settings is discussed in Section III. Unification of clusters based on mutual Euclidean distances between the particles is dealt with in Section IV. Section V explains the selection process of the 'fittest' value through fine search mechanism. Experimental results are reported in Section VI and Section VII concludes the paper.

*S. Easter Selvan is Senior Lecturer in the department of Elecronics and Communication Engineering in Karunya Institute of Technology, Tamil Nadu, India. [easterselvan@rediffmail.com]
Sethu Subramanian and S. Theban Solomon are with Electronics and Communication Engineering Department, Karunya Institute of Technology.

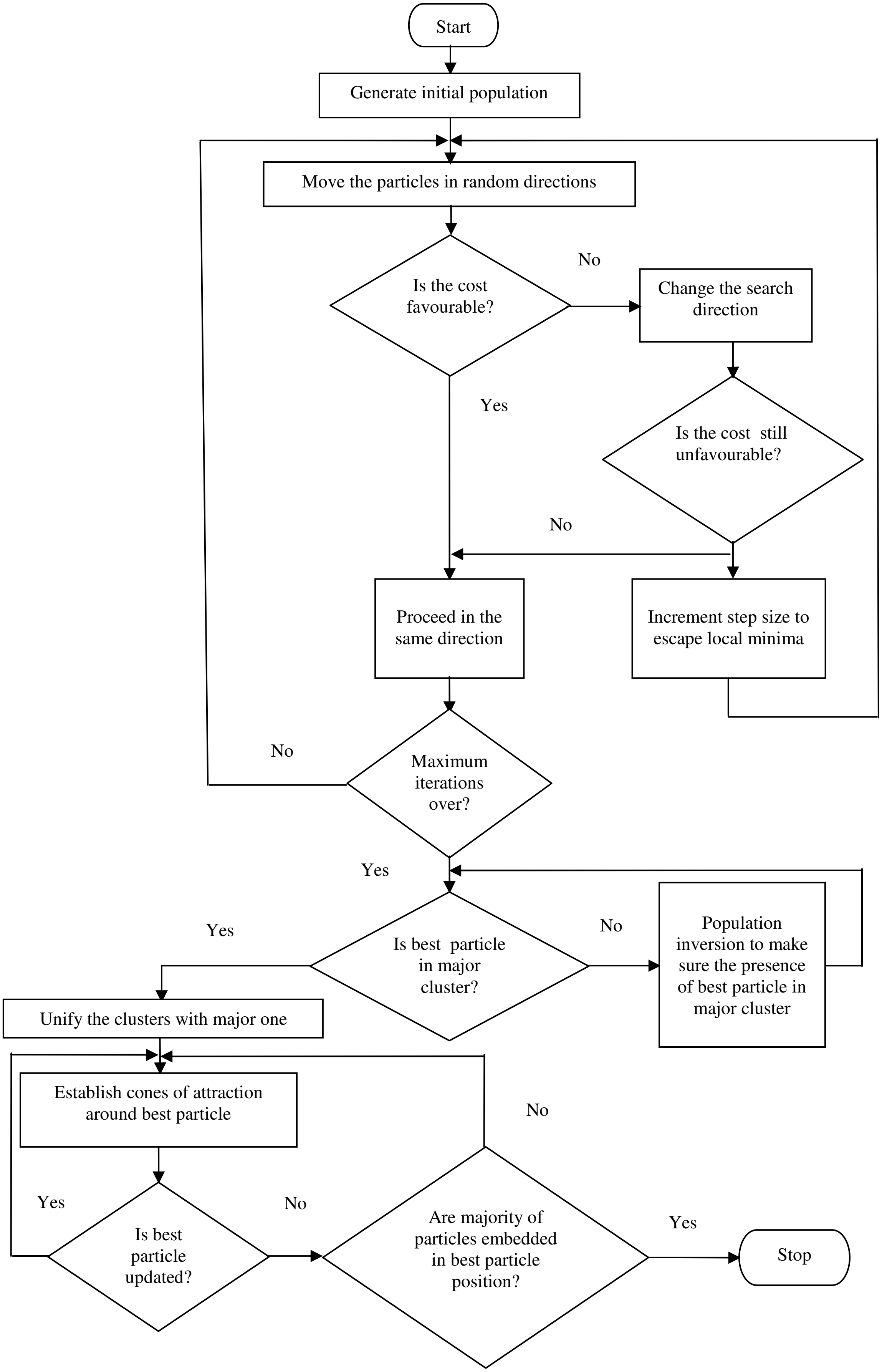

Fig. 1. General flowchart of suggested PSO

II. GENERATION OF 3 DIMENSIONAL SOLUTION SPACE

The prime objective of this work is to test the performance of the particle swarm optimization algorithm developed, incorporating special features such as preserving the initial population of the particles without reproducing, unification of generated clusters and fine search mechanism to select the 'fittest' in a common control problem.

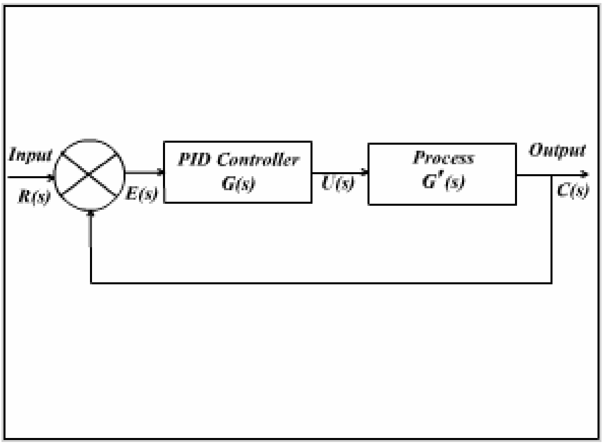

Fig. 2. Block diagram of a plant with PID controller

Attempt has been made to achieve globally minimal settling time in the step response of a process which is cascaded with PID controller by tuning the k_p, k_i and k_d values in the transfer function of the controller stated as

$$G(s) = \frac{U(s)}{E(s)} = k_p + \frac{k_i}{s} + k_d s \tag{1}$$

where

k_p : proportional gain

k_i : integral gain

k_d : differential gain

$E(s)$: Laplace transform of the input to the controller

$U(s)$: Laplace transform of the output of the controller

Usually, the choice of the controller coefficients is by approximate methods, which in turn will not guarantee globally optimal solution for control applications. To circumvent the uncertainty due to the above trial and error methods for tuning the PID, a novel approach is sought which makes use of a known evolutionary technique – particle swarm optimization.

The 3 dimensional solution space is generated based on the possible range of values of k_p, k_i and k_d derived through the Ziegler-Nichols method after ensuring the presence of all the poles of the transfer function confined to the left half of the s plane and populated with N number of particles whose initial positions are randomly generated. The initial position of the i^{th} particle of the swarm can be represented by a 3 dimensional vector,

$$x_i^{in} = (x_{i1}, x_{i2}, x_{i3})^T \tag{2}$$

Let p_{max}, p_{min}, i_{max}, i_{min}, d_{max} and d_{min} be the extreme values of the axes in the 3 dimensional solution space.

$$\begin{aligned} x_{i1} &= rand \times p_{max} \\ x_{i2} &= rand \times i_{max} \\ x_{i3} &= rand \times d_{max} \end{aligned} \tag{3}$$

thereby $p_{min} < x_{i1} < p_{max}$, $i_{min} < x_{i2} < i_{max}$ and $d_{min} < x_{i3} < d_{max}$. *rand* is a vector of random numbers in [0.0, 1.0].

III. UNBIASED SEARCH FOR OPTIMAL SOLUTION

Let the cost of the i^{th} particle at the current position be stated as

$$T_S(x_{i1}^t, x_{i2}^t, x_{i3}^t)$$

The next assumed position of the particle can be computed using the random unit vector and the step size of the particle movement determined based on the maximum distance by which two particles confined to the solution space can be separated.

$$x_i^{t+1} = x_i^t + \psi \times S_s \times \phi_{unit} \tag{4}$$

where

x_i^{t+1} : next position of the i^{th} particle of the swarm

x_i^t : current position of the i^{th} particle

ψ : experimentally determined constant for efficient search and fast convergence

S_s : distance by which the consecutive positions of the particle are separated

ϕ_{unit} : randomly oriented unit vector

ψ, S_s and ϕ_{unit} are expressed as follows:

$$\psi = \begin{cases} 3 & if \quad iter \le 0.75 \times \max\_iter \\ & else \\ & 1 \end{cases} \tag{5}$$

where

$iter$: the current iteration number

max_*iter*: the total number of iterations to be performed

$$S_s = \frac{D_{SS}}{\gamma} \tag{6}$$

where D_{SS} is the maximum Euclidean distance measured between two particles confined to the solution space. For example, in Fig. (3) the value of D_{SS} is $\sqrt{p_{\max}^2 + d_{\max}^2}$ and γ is chosen to be 100. The choice of γ is critical. If γ is too small, and if the optimum value lies in a valley with steep edges, the search will tend to jump out of the valley. On the other hand, if γ is too large, convergence will be slow, but if the search finds a local minimum, it will typically not deviate too far from it.

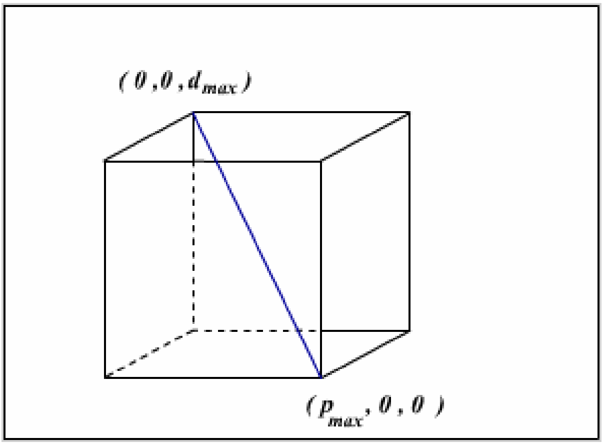

Fig. 3. Maximum distance of separation in the solution space

$$\phi_{unit} = \frac{\phi\ (i)}{\sqrt{\phi^T\ (i) + \phi\ (i)}} \tag{7}$$

where $\phi(i) \in \Re^3$ with each element $\phi_m(i)$, m = *1*, *2*, *3*, a random number generated in [-1, 1]. The cost of the i^{th} particle at the new position x_i^{t+1} is compared with the cost at the previous position x_i^t to decide the direction of movement in the next course of search as the particle is assumed to have built-in memory. If the present cost is lower than the previous, the particle steps forward in the same direction specified by the unit vector in the previous position. On the other hand, the unfavourable cost at the new position reverts the particle back to the previous position, presumably because the particle would have landed in the previous step in one of the local minima in the solution space. The particle as a result gets 'locked up' in the valley until an attempt in some other random direction enables the particle to find a favourable cost.

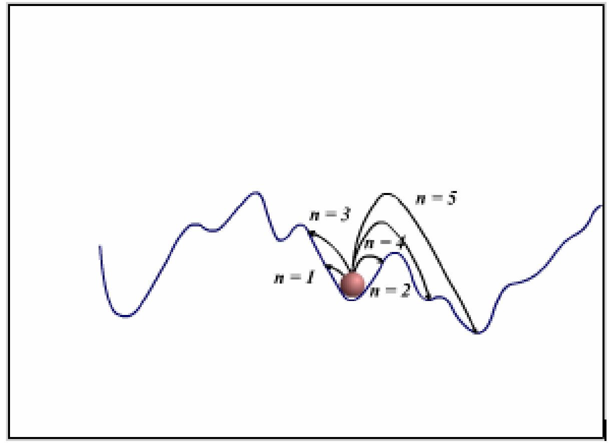

Fig. 4. Stepping pattern of a particle 'locked up' in local minimum

The 'lock up' mechanism is mathematically represented as follows. If $f(x)$ is a real valued objective function yielding the cost pertaining to the particle position $x \in S$, where S is a non-empty compact set termed as solution space and $f(\tilde{x})$ is the cost at the position $\tilde{x}$ corresponding to a local minimum, a transformation is performed to map $f(x)$ to $g(x)$ so that the particle ignores the positions in the solution space S wherein the costs are equal to or higher than the one at the recently visited local minimum.

The transformed function $g(x)$ is defined as

$$g(x) = f(x) + \gamma\ [\text{sgn}(f(x) - f(\tilde{x})) + 1] \times [\text{sgn}(f(\tilde{x}) - f(x)) + 2] \tag{8}$$

where

$$\gamma = \frac{\max f(x) - f(x)}{2} \tag{9}$$

$$\text{sgn}(x) = \frac{2}{1 + \exp(-\lambda x)} - 1 \tag{10}$$

In equation (9), $\max f(x)$ represents maximum cost assigned to the unfavourable positions in the solution space thereby hindering the visits of the particle to those positions. In equation (10), λ is a real number in [0.0, 1.0] and it can also be represented as

$$\text{sgn}(x) = \begin{cases} -1, & when \quad x < 0 \\ 0, & when \quad x = 0 \\ 1, & when \quad x > 0 \end{cases} \tag{11}$$

The 'locked up' particle in the local minimum due to the transformed function $g(x)$ takes linearly increasing steps to escape from the valley iteratively until it succeeds in finding another position with a lower cost compared to the local minimum. The modified step size of the 'locked up' particle in the local minimum is stated as

$$x_i^{l+n} = x_i^{l} + n(\psi \times S_s \times \phi_{unit}) \quad (12)$$

where

x_i^{l+n} : position of the 'locked up' particle in the n^{th} attempt to step out from the local minimum position x_i^{l}

x_i^{l} : position of the i^{th} particle 'locked up' in the local minimum

n : number of attempts made by the i^{th} particle to escape the local minimum position.

At the end of the specified number of iterations, based on the structure and the location of the basins with reduced cost, particle clusters are formed in the solution space.

IV. Unifying the Clusters in the Potential Space

Depending upon the cost function, one or more well defined or contiguous particle clusters can be identified along with a few stray particles in the solution space, which are to be unified with reference to the location of the cost-wise best particle in the entire population. The strategy of unifying the clusters varies based on the criteria whether the cluster under consideration encloses the best particle or not. The Euclidean distance measure is extended to 3 dimensional case to compute the distance of separation d_i between i^{th} particle position in the solution space and the rest of the particles in the population. A measure of particle dispersion $d_{average}$ in the solution space is derived by computing the average of the sum of mutual distances by which all the particles in the population are separated from the rest.

$$d_{average} = \frac{\displaystyle\sum_{i=1}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N}\left[\sum_{m=1}^{3}\left(x_m^{i}-x_m^{j}\right)^2\right]^{1/2}}{N} \quad (13)$$

where x_m^{i} is the m^{th} coordinate of the solution space corresponding to the i^{th} particle position and N is the population size.

The best particle is concluded to be located in the thickly populated cluster if the sum of distances between the best particle and the rest of the population is smaller than the measure of particle dispersion, $d_{average}$. Otherwise, the best particle is to be in any one of the surrounding clusters with thin population.

$$d_{best\_particle} = \sum_{\substack{i=1\\ i\neq b}}^{N}\left[\sum_{m=1}^{3}\left(x_m^{b}-x_m^{i}\right)^2\right]^{1/2} \quad (14)$$

where x_m^{b} is the m^{th} coordinate of the solution space corresponding to the best particle position.

For the above mentioned two cases, the unification of clusters is handled by the underlying strategies.

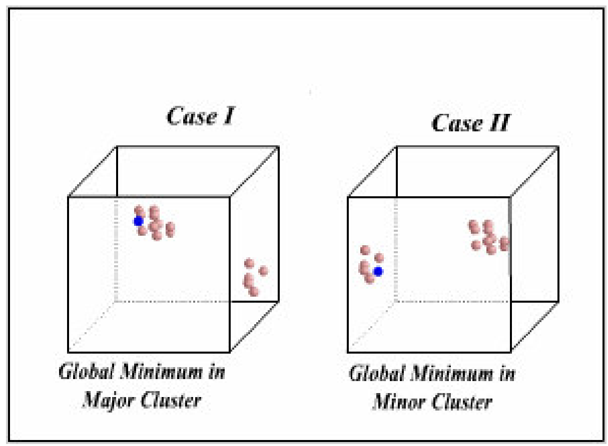

Fig. 5. Two cases arising during cluster unification process

Case I: $d_{best\_particle} < d_{average} \Rightarrow$ best particle in the thickly populated (major) cluster.

All the particles in the solution space whose Euclidean distances from the best particle are more than $r_{cluster}$ defined in equation (15) are dislodged towards the best particle in the direction and by the magnitude specified in equation (16).

$$r_{cluster} = \frac{\displaystyle\sum_{i=1}^{k}\left[\sum_{m=1}^{3}\left(x_m^{b}-x_m^{i}\right)^2\right]^{1/2}}{k\eta} \quad (15)$$

where k is the number of particles in the neighbourhood of the best particle and η is the shrinking factor. The values of k and η set to be 10 and 5 respectively in the experiments conducted, yielded better results.

Case II: $d_{best\_particle} > d_{average} \Rightarrow$ best particle is located in the surrounding.

Those particles whose Euclidean distances from the best particle are less than the limiting value in equation (15), are moved towards the latter in the direction and by the magnitude given in equation (16).

$$\vec{leap} = D_i\left\{1-\exp\left(-d_{useful}\right)\right\}\left\{\sum_{m=1}^{3}\left[x_m^i - x_m^b\right]^2\right\}^{1/2} \tag{16}$$

where

$leap$: step size and direction of the i^{th} particle in the cluster unification phase

D_i : Euclidean distance between the i^{th} particle and the best particle in the cluster

$$d_{useful} = 0.1 + \frac{2.2}{d_{range}}\left(D_i - d^*_{\min}\right) \tag{17}$$

for linearly mapping the actual distance of separation into a useful range to determine the step size.

Here, $d_{range} = d^*_{\max} - d^*_{\min}$ (18)

where

$d^*_{\max}$ and $d^*_{\min}$ are the maximum and the minimum distances of separation between the best particle and the rest of the particles lying beyond $r_{cluster}$ defined in equation (15) respectively. The particles are iteratively dislodged towards the cluster enclosing the best particle until all the particles are separated from the best one by distances less than $r_{cluster}$ thereby unifying the scattered clusters in the solution space.

V. Selection of the Fittest Particle

Unlike the conventional techniques, wherein the particles having unfavourable costs are discarded and those with favourable costs are reproduced, the unification of particle clusters allow us to use the same set of population for intensive search process to select the 'fittest' particle position in the optimal solution space. The i^{th} particle other than the best one is made to assume different positions on the surface of the virtual sphere centered at the i^{th} particle position, whose radius is the Euclidean distance between this and the best particle.

$$x_i^{t+1} = x_i^t + \left\{\sum_{m=1}^{3}\left[x_m^b - x_m^i\right]^2\right\}^{1/2} \phi_{unit} \tag{19}$$

Everytime, as the particles assume new positions, it is ensured to update the best particle by comparing the costs corresponding to these positions with the previously selected best particle cost. Simultaneously, the best particle in a given instant is assumed to 'diffuse attractant' towards the rest of the particles in the cluster, which leads to establishment of 'cones of attraction' with axes connecting the best particle and the rest in the population. Subject to the condition that the angle subtended by the vector joining the i^{th} particle to the best one and the vector joining the present and the next positions of the i^{th} particle lies within θ degrees,

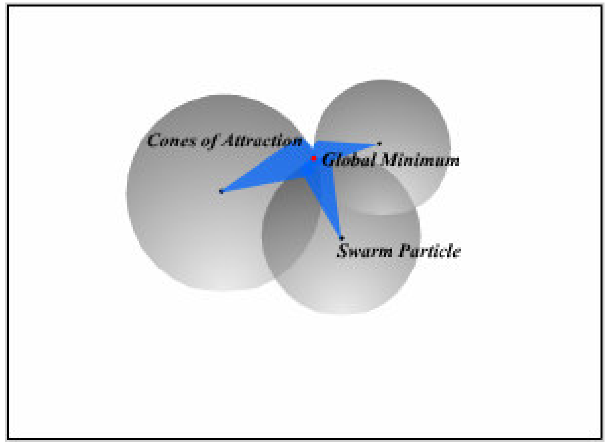

Fig. 6. Fine search mechanism by establishing 'cones of attraction'

the particle under consideration is absorbed by the best particle. This process is explained by the following equation.

$$x_i^{t+1} = \begin{cases} x_b^t \quad \text{if} \quad \cos^{-1}\left(x_1x_2 + y_1y_2 + z_1z_2\right) \le \theta \\ \quad\quad \textit{else} \\ x_i^t + \left\{\sum_{m=1}^{3}\left[x_m^b - x_m^i\right]^2\right\}^{1/2} \phi_{unit} \end{cases} \tag{20}$$

where

(x_1, y_1, z_1) is the unit vector joining the i^{th} particle and the best particle at that instant. (x_2, y_2, z_2) is the unit vector joining the i^{th} particle's present and the next positions in the solution space. The search process continues until sizable population gets absorbed by the best particle whose cost is claimed to be globally minimal. Experimentally, the critical angle θ within which particle absorption takes place is selected as 5 degrees and the criteria for terminating the fine search is to attract minimum of 50% of the total population to the best particle position.

VI. Experimental Results and Discussion

The performance of the algorithm developed was tested with transfer functions of systems of order ranging from three to six. The cost function here is the settling time in terms of system time constant which decides the performance of any industrial process. The closed loop PID controller cascaded with the process was tuned for the values k_p, k_i and k_d first by using Ziegler-Nichols method and then by our modified PSO algorithm. Corresponding settling time was computed in both cases. In all the cases tested, the settling times obtained by PSO were much less than those values by the approximate method, as indicated in the table.

The initial unbiased search phase was proven to be very efficient, as towards the end, most of the particles in the entire swarm could be dragged into regions of close proximity to the globally minimal position in the solution space. To visualise this, the same algorithm was modified to tune PI controller and the final positions of majority of the particles in the swarm were confined to the basin wherein the settling time is the lowest as depicted in the surface plot in figure 8(a).

For a typical case, the results are shown in 4 different stages to narrate the swarming behaviour of the particles in the population.

S. No	Transfer Function	ZIEGLER-NICHOLS			T_{S1}	PSO			T_{S2}
		K_D	K_I	K_P	(τ)	K_D	K_I	K_P	(τ)
1	$\frac{5}{s^4+3s^3+7s^2+5s}$	0.6939	0.3603	0.9042	11.7143	0.0383	0.0016	0.2453	2.33
2	$\frac{s+5}{s^4+17s^3+60s^2+5s+5}$	0.8913	0.2805	2.8662	83.2857	3.8007	0.1013	0.1137	1.6667
3	$\frac{s+5}{s^4+17s^3+60s^2+10s}$	2.8373	0.0881	0.4423	37.6667	3.8286	0.0011	0.7524	3.2667
4	$\frac{(s+1)(s^2+s+1)}{(s+1)^2(s+2)^2(s+3)^2}$	0.1062	2.3530	273.779	637	3.8359	4.0	0.1069	3.5231
5	$\frac{6}{s^4+3s^3+4s^2+3s+1}$	0.7386	0.3385	0.4352	96.5	0.0286	0.0466	0.0715	1.6279
6	$\frac{250s+500}{s^3+12s^2+100s+0}$	0.0212	11.7765	1.92	19.25	11.3854	4.6026	11.4145	1
7	$\frac{300(s+100)}{s(s+10)(s+40)}$	0.0575	4.35	0.1450	214	1.9072	0.1470	1.0625	1
8	$\frac{4.19}{s(s+1)(s+5)}$	0.5905	0.4234	1.2672	10.2222	0.4610	0.0131	1.3679	3.25
9	$\frac{60}{(s+1)(s+2)(s+5)}$	0.1778	1.4057	1.2384	16.2857	0.3002	0.4833	0.7099	1.8462
10	$\frac{1.6}{s^2+2.584s+1.6}$	0.1096	2.2811	12.4533	33.5	0.12	0.9626	1.2828	2.15

Table 1. Comparison of settling time obtained by Ziegler-Nichols method and PSO algorithm

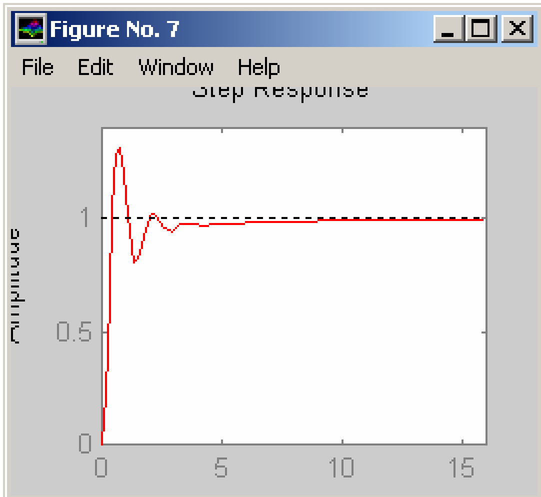

Fig. 7 (a) Step response of a plant whose transfer function is $\dfrac{1.6}{s^2 + 2.584s + 1.6}$ and cascaded with PID controller tuned by Ziegler-Nichols method

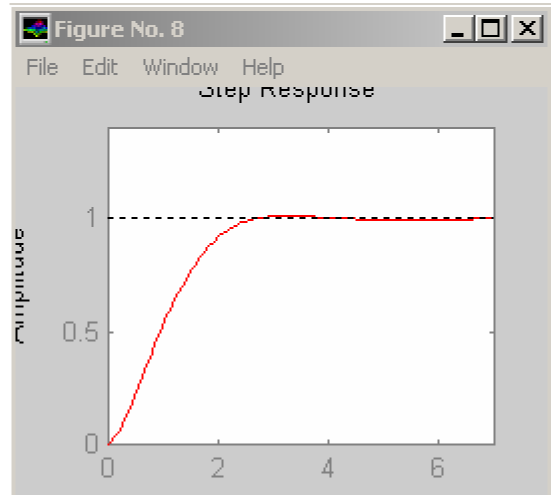

Fig. 7(b) Step response of the plant cascaded with PID controller tuned by Paticle Swarm Optimization method

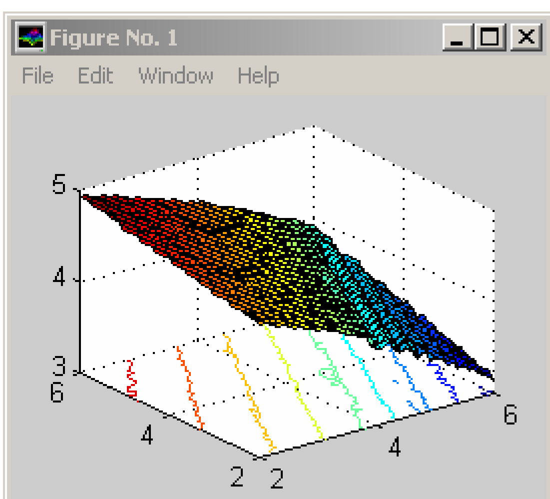

Fig. 8 (a) The surface plot of the objective function used to compute settling time for a plant whose transfer function is $\dfrac{(s+1)(s^2+s+1)}{(s+1)^2(s+2)^2(s+3)^2}$ cascaded with PI controller

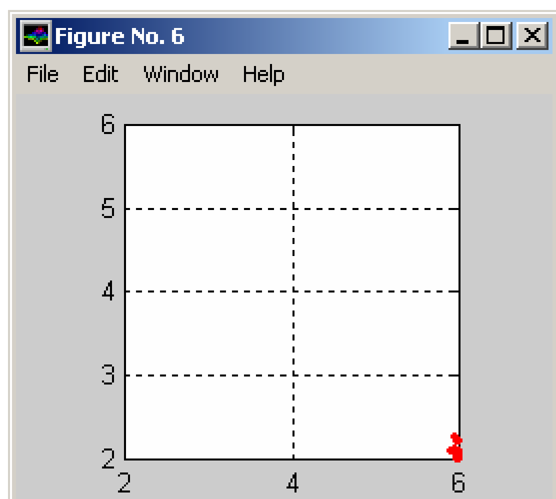

Fig. 8 (b) The settlement of the swarm particles in the basin with global minimum value depicted in fig. 8 (a) in the two dimensional solution space after completing unbiased search for optimal values.

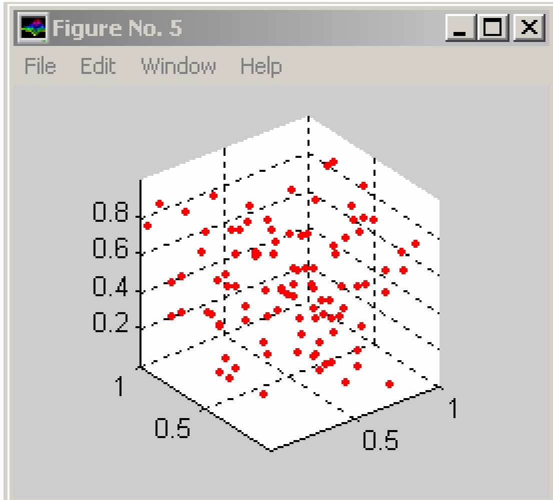

Fig. 9 (a) Generation of population (100) inside solution space for tuning PID controller cascaded with the plant whose transfer function is $\frac{6}{s^4+3s^3+4s^2+3s+1}$

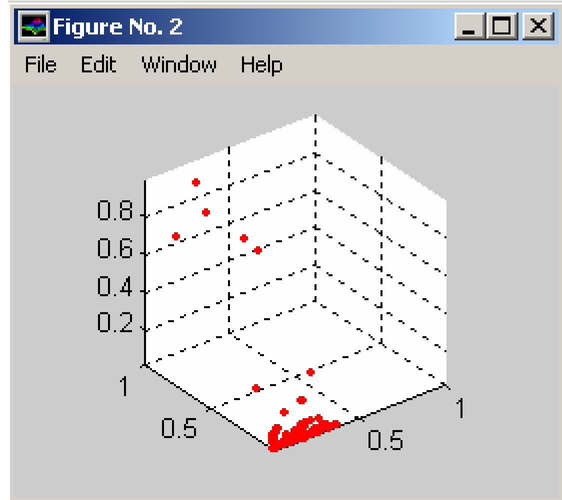

Fig. 9 (b) Cluster formation as a result of unbiased search by the particles for optimal values

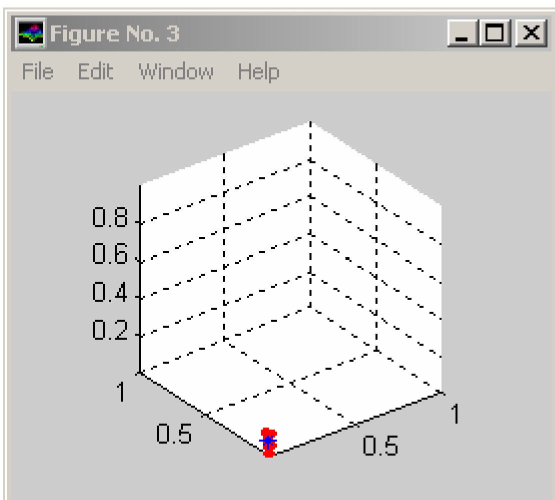

Fig. 9 (c) Unification of particle clusters enclosing best particle whose position is indicated in blue

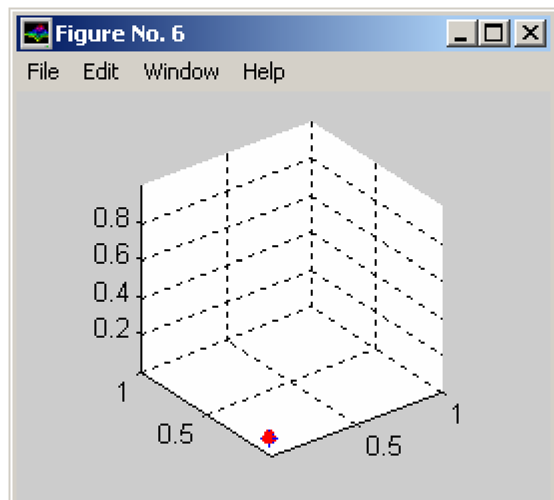

Fig. 9 (d) Appearance of the particles in the solution space after embedding the majority of the particles on the globally best one during the fine search phase.

VII. Conclusion

In this paper, the modified PSO algorithm was implemented in PID controller tuning and the results obtained for different plant transfer functions conformed to the theoretical predictions. Encouragingly, the settling time values for the PID controller coefficients selected by PSO were seemingly the best compared to the choice made by conventional tuning method.

In an attempt to evaluate the performance of this method, the formation of distinct particle clusters in the solution space was reported in 80% of the cases which is correlated to the structure of the surface plot indicating the cost at different locations for a given objective function. Precisely, the existence of well defined basins led to better results and faster convergence. However, the subsequent cluster unification algorithm tends to overcome the difficulty by reorienting even the isolated particles towards potentially optimum location, which might be encountered in remote cases wherein the costs at different locations of the solution space are not extreme to form valleys.

REFERENCES

[1] Kennedy, J., and Eberhart, R. C. (1995). Particle Swarm Optimization. Proc. IEEE International conference on Neural Networks (Perth, Australia), IEEE Service center, Piscataway, NJ, pp. IV: 1942 – 1948.

[2] K.E. Parsopoulos and M.N. Vrahatis. Recent approaches to global optimization problems through particle swarm optimization. Natural Computing 1: 235 – 306, 2002.

[3] Passino K.M. "Biomimicry of Bacterial Foraging for Distributed Optimization and Control", IEEE Control Systems Magazine, Vol. 22, No. 3, pp. 52 – 67, June 2002.

[4] A. Colorni, M. Dorigo, and V. Maniezzo, "Distributed Optimization by Ant Colonies", Proc. of First European Conference on Artificial Life, pp. 134 – 142, Cambridge, MA: MIT Press 1991.

[5] Y. Shi, R.C. Eberhart, "Emperical study of particle swarm optimization," Proc. 1999 congress on Evolutionary Computation, July 6 – 9 1999, pp. 1945 – 1950.

[6] B. Porter and A. H. Jones, "Genetic tuning of digital PID controllers", Electronic Letters, Vol. 28, pp. 843 – 844, 1992.

[7] http://homepage2.nifty.com/fukuyama-yoshikazu/EcTutorial(Chap5Found of PSO).pdf

[8] http://web.ics.purdue.edu/~hux/tutorials.shtml