JOURNAL OF AERONAUTICS AND SPACE TECHNOLOGIES
JULY 2012 VOLUME 5 NUMBER 4 (1-14)

A NEW PARTICLE SWARM OPTIMIZATION METHOD FOR THE PATH PLANNING OF UAV IN 3D ENVIRONMENT

Y. Volkan PEHLIVANOGLU

Turkish Air Force Academy
Aeronautics and Space Engineering Dept.
Yesilyurt/İstanbul/Turkey
vpehlivan@hho.edu.tr

Received: *13th January 2012,* ***Accepted:*** *27th July 2012*

ABSTRACT
Particle swarm optimization (PSO) method is relatively a new population-based intelligence algorithm and exhibits good performance in optimization problems. However, during the optimization process, the particles become more and more similar, and gather into the neighborhood of the best particle in the swarm, which makes the swarm prematurely converged possibly around the local solution. PSO technique can be augmented with an additional mutation operator that provides diversity and helps prevent premature convergence on local optima. In this paper, mathematical analysis of a basic PSO is reissued and a diversity concept is evaluated in commonly used PSO algorithms including constriction factor PSO, inertial weight PSO, Gaussian mutation PSO, and a new vibrational mutation PSO combining the idea of mutation strategy related to periodicity. New algorithm is tested and compared with selected PSO algorithms. The comparative experiments have been conducted on a wide range of nonlinear functions and a path planning problem of unmanned aerial vehicle (UAV) in three-dimensional (3D) terrain environment. The results give insight into how mutation operator effects the nature of the diversity and show that the addition of a mutation operator with a periodicity concept can significantly enhance the optimization performance.

Key words: *PSO, Diversity, Periodicity, Path planning.*

3 BOYUTLU ORTAMDA YOL PLANLAMASI İÇİN KULLANILABİLECEK GELİŞMİŞ PARÇACIK SÜRÜ ENİYİLEME YÖNTEMİ

ÖZET
Parçacık sürü eniyileme yöntemi nispeten yeni bir nüfus temelli yapay zekâ algoritması olup, eniyileme problemlerinde oldukça iyi performans sergileyebilmektedir. Bununla berber eniyileme süreci esnasında sürü içersindeki parçacıklar gittikçe birbirlerine benzemekte ve çoğunlukla da sürü içersindeki en iyi parçacık etrafında öbeklenmektedirler. Bu öbeklenme ise eniyileme sürecini genel çözüm yerine yerel çözümle sonlandırabilmektedir. Bununla beraber yöntem ilave mutasyon operatörü kullanılarak geliştirilebilir ve bu sayede sürü içersindeki çeşitlilik arttırılarak yerel çözüm yerine genel çözüme ulaşılabilir. Bu makalede öncelikle yöntemin matematiksel temelleri gözden geçirilmiş, çeşitlilik kavramı üzerinde durularak çeşitliliğin bazı parçacık sürü algoritmalarındaki davranışları incelenmiştir. Ayrıca bu çözümlemeye dayalı olarak geliştirilen periyodik mutasyon uygulamaları yeni parçacık sürü yönteminde uygulamaya konulmuştur. Geliştirilen yeni yöntemin verimliliğini göstermek için değişik test fonksiyonları ile yol planlama problemleri çözülerek farklı algoritmalarla karşılaştırmalara gidilmiştir. Elde edilen sonuçlar periyodik mutasyon uygulamalarının etkilerini ve bu sayede elde edilen yüksek verimliliği teyit eder niteliktedir.

Anahtar Kelimeler: *PSO, Çeşitlilik, Periyodiklik, Yol planlaması.*

1. INTRODUCTION

As in other evolutionary algorithms, Particle Swarm Optimization (PSO) method is a population-based stochastic optimization algorithm that originates from "nature". Often these algorithms may require more cost function evaluations than comparable gradient-based algorithms. They, however, provide attractive characteristics, such as ease of implementation for both continuous and discrete problems, efficient use

of large numbers of parallel processors, no requirement for the continuity in response functions, and more robust solution generations for searching global or near global solutions. PSO algorithms search the optimum within a population called "swarm." It benefits from two types of learning, such as "cognitive learning" based on an individual's own history and "social learning" based on swarm's own history accumulated by sharing information among all particles in the swarm. Since its development in 1995 by Kennedy *et al.* [1], it has attracted significant attention. Most of the investigations on this topic are related to either the mathematical analysis focusing on how it works, or improving the PSO to get faster and more reliable solutions. The impetus for the latter is typically the trapping of the solution candidates to local optima, or the so-called premature convergence.

An optimization algorithm prematurely converges to a local optimum if it is no longer able to explore other sections of the search space than the examined area and there at least another region exists that contains a solution superior to the currently discovered one. The main reason for the premature convergence might be the lack of diversity. Diversity is essential for a healthy search; and mutations are the basic operators to provide the necessary variety within a swarm. Apart from designing new mutation operators, researchers have put relatively less effort into investigating how to apply mutation operators during the process and determining what kind of diversity should be provided within the swarm in PSO process. Therefore, after having a close scrutiny of the diversity concept based on quantification and qualification studies, new mutation strategy and operator are improved to provide beneficial diversity within the swarm. This new approach is called vibrational PSO. They were applied to selected benchmark test functions and path planning problem of unmanned aerial vehicle (UAV) in three-dimensional (3D) terrain environment. As demonstrated by these test cases, it is observed that new algorithms outperform the current popular algorithms.

Definition of basic PSO

Let s be the swarm size, d be the particle dimension space, and each particle of the swarm has a current position vector $\boldsymbol{X}_i$, current velocity vector $\boldsymbol{V}_i$, individual best position vector $\boldsymbol{P}_i$ found by particle itself. The swarm also has the global best position vector $\boldsymbol{P}_g$ found by any particle during all prior iterations in the search space. Assuming that the function f is to be minimized and describing the following notations in t^{th} iteration, then the definitions are as follows:

$$\boldsymbol{X}_i(t) = \left(x_{i,1}(t), x_{i,2}(t), \dots, x_{i,d}(t)\right), \quad (1)$$

$$x_{i,j}(t) \in R^d, i = 1,2, \dots, s$$

where each dimension of a particle is updated using the following equations:

$$\begin{aligned} v_{i,j}(t) &= v_{i,j}(t-1) \\ &+c_1 r_1 \left(p_{i,j}(t-1) - x_{i,j}(t-1)\right) \\ &+c_2 r_2 \left(p_{g,j}(t-1) - x_{i,j}(t-1)\right) \end{aligned} \quad (2)$$

$$x_{i,j}(t) = x_{i,j}(t-1) + v_{i,j}(t) \quad (3)$$

In Eq. (3), c_1 and c_2 denote constant coefficients, r_1 and r_2 are elements from random sequences in the range of (0, 1). The parameter c_1 controls the influence degree of a "cognitive" part of an individual, and c_2 determines the effect of a "social" part of the swarm. The personal best position vector of each particle is computed using the following expression:

$$\boldsymbol{P}_i(t) = \begin{cases} \boldsymbol{P}_i(t-1) & if\ f(\boldsymbol{X}_i(t)) \geq f(\boldsymbol{P}_i(t-1)) \\ \boldsymbol{X}_i(t) & if\ f(\boldsymbol{X}_i(t)) < f(\boldsymbol{P}_i(t-1)) \end{cases} \quad (4)$$

Then, the global best position vector is found by

$$\boldsymbol{P}_g(t) = arg \min_{\boldsymbol{P}_i(t)} f(\boldsymbol{P}_i(t)) \quad (5)$$

Improved PSO algorithms in the literature

PSO may have some issues related to convergence speed, prematurely converged solutions, deficient accuracy or lack of diversity. Numerous modifications have been proposed so far to overcome these issues [2]. These may be broadly divided into two categories as focusing on hardware or software improvements. The hardware focus is related to parallel computing. The software studies may be further classified as hybridization with other search algorithms, inspiration from other stochastic-based algorithms, rearranging and manipulating the reproduction descriptions mainly related to the velocity equation, parameter and neighborhood topology manipulations within the reproduction phase, and finally the swarm sizing and grouping.

Since each particle's cost function can be evaluated independently in the swarm, PSO is ideally suited for synchronized or unsynchronized execution on a cluster of computers in parallel. Integrating PSO algorithm with other search algorithms is called hybridization. A common hybrid application is to use PSO with another population-based algorithm such as a genetic algorithm (GA). Yet another technique is to use a gradient-based algorithm as an integrated part. Manipulating or rearranging Eq. (2) and Eq. (3) is an important part of algorithm improvements. In these manipulations, the velocity equation is extended with

an extra term or shortened depending on the approach. Parameter re-descriptions constitute another bundle of modification packages. Since Shi *et al.* [3] suggested the use of an inertia weight *(w)* multiplying with the previous velocity in Eq. (2), the parameter number to be adjusted is increased to three. The inertia weight determines the effect of a previous velocity. The common PSO parameters, w, c_1, c_2 may be constant, linearly or nonlinearly changeable, let's say periodically, adaptively, chaotic or randomly changeable depending on the time or other reference such as cost function value, velocity, or metropolis criteria. Neighborhood topology originates from the selection of position vectors $\boldsymbol{P}_i$ and $\boldsymbol{P}_g$. In the original algorithm, the particle is going to be attracted by individual best position and global best position. It means no other individual best position vectors will have an attractive effect to individual itself and also the best one will never be changed at that iteration. At this point, different neighborhood topology descriptions are introduced. Population itself is also studied in many different aspects. Dynamic swarm sizing or dividing the swarm into subgroups may be promising to solve some of the optimization problems. Inspirations from other stochastic-based algorithms such as GA, simulated annealing, or opposition-based learning algorithms are other popular methodologies to improve the basic PSO algorithm. Commonly used GA reproduction operators including selection, crossover, elitism, and mutation can be deployed in PSO algorithm architecture.

Among GA operators, mutation is the most utilized. Because the drawback of PSO is due to lack of diversity, which forces the swarm particles to converge to the position found so far which may be a local optimum. Without an effective diversity enhancing mechanism, the optimization algorithm may not be able to efficiently explore and exploit the search space. Mutation operators introduce new particles into a swarm by manipulating a current particle, thus adding diversity into the swarm and probably preventing stagnation of the search in local optima. However, the mutation application procedure brings some new adjustments to the algorithm such as the criteria of mutation applications, the position where mutation will be applied, and the selection of random probability distribution sequence. Mutation probability percentages, similarity and closeness to each other in terms of Euclidian distance, and describing diversity thresholds may be the criteria for mutation applications. Gaussian, Cauchy, Beta distributions, chaotic distribution based on logistic maps, or mixed types such as cloud distribution are possible random distribution sequences. Randomly selected current positions, velocities, or even individual best position and global best position vectors may be mutated [4-14]. However, elitism concept may be required in the case of mutating the individual best or global best position vectors.

Although mutations provide diversity within the swarm, they may also cause peripheral stagnation search. Therefore, the type of mutation operator and its application strategy are important decision parts for the mutation applications. Apart from designing new mutation operators, researchers have put relatively less effort into investigating how to apply mutation operators during the process and determining what kind of diversity should be provided within the swarm in PSO process. Before moving forward to extensions of PSO, it would be better to reissue the basic PSO algorithm and its mathematical basis.

2. MATHEMATICAL ANALYSIS OF PSO ALGORITHM

Basic PSO

Ozcan and Mohan [15] analyzed the updating Eq. (2) and (3) in an analytical way. Let's reissue the analytical analysis to go a step further. For the sake of simple mathematical analysis, assume that $c_1.r_1$ and $c_2.r_2$ are constants which are equal to φ_1 and φ_2, respectively, and $\boldsymbol{P}_i\ (t)$, $\boldsymbol{P}_g\ (t)$ are also constants, such as p_i, and p_g, respectively. Then, Eq. (2) and (3) become,

$$v_{i,j}(t) = v_{i,j}(t-1) + \varphi_1\left(p_i - x_{i,j}(t-1)\right) + \varphi_2\left(p_g - x_{i,j}(t-1)\right) \quad (6)$$

$$x_{i,j}(t) = x_{i,j}(t-1) + v_{i,j}(t) \quad (7)$$

by getting $x_{i,j}(t\text{-}1)$ using Eq. (7) and substituting it into Eq. (6) we get the following particle path equation

$$x_{i,j}(t) - (2-\varphi_1-\varphi_2)x_{i,j}(t-1) + x_{i,j}(t-2) = \varphi_1 p_i + \varphi_2 p_g \quad (8)$$

$$x - \alpha\dot{x} + \ddot{x} = \beta \quad (9)$$

$$\alpha = (2-\varphi_1-\varphi_2) \quad (10)$$

$$\beta = \varphi_1 p_i + \varphi_2 p_g \quad (11)$$

This is a linear non-homogeneous second order differential equation. The general solution of Eq. (8) can be derived by getting complementary and particular solutions. The resulted solution including initial boundary conditions can be found such that

$$x(t) = \chi\tau^t + \varepsilon\phi^t + \gamma \quad (12)$$

where

$$\chi = \left(0.5 - \frac{(\varphi_1+\varphi_2)}{2\eta}\right)x_{i,j}^{(0)} + \frac{v_{i,j}^{(0)}}{\eta} + \frac{\beta}{2\eta} - \frac{\beta}{2(\varphi_1+\varphi_2)} \quad (13)$$

$$\varepsilon = \left(0.5 + \frac{(\varphi_1 + \varphi_2)}{2\eta}\right) x_{i,j}^{(0)} - \frac{v_{i,j}^{(0)}}{\eta} - \frac{\beta}{2\eta} + \frac{\beta}{2(\varphi_1 + \varphi_2)} \quad (14)$$

$$\tau = (\alpha + \eta)/2 \quad (15)$$

$$\phi = (\alpha - \eta)/2 \quad (16)$$

$$\eta = \sqrt{\alpha^2 - 4} \quad (17)$$

$$\gamma = \beta/(\varphi_1 + \varphi_2) \quad (18)$$

This result may expose several different cases depending on α value, specifically the sum of φ_1 and φ_2 values. Value of η becomes a complex number for $0< \varphi_1 + \varphi_2 <4$. Converting ϕ and τ into polar forms and substituting them into Eq. (12) we can obtain the following simplified equations:

$$x_{i,j}(t) = \kappa \sin(\theta t) + \lambda \cos(\theta t) + \gamma \quad (19)$$

$$\theta = tan^{-1}(\|\eta\|/|\alpha|) \quad (20)$$

$$\kappa = (2v_{i,j}^{(0)} - (\varphi_1 + \varphi_2)x_{i,j}^{(0)} + \beta)/\|\eta\| \quad (21)$$

$$\lambda = x_{i,j}^{(0)} - \gamma \quad (22)$$

This is the general form for the particle path. We can make some assumptions such that there is only one dimension, $p=p_i=p_g$, $\varphi_1 + \varphi_2= \varphi$ and $2>\varphi>0$ or $4>\varphi>2$, $x^{(0)}=x_0$, $v^{(0)}=v_0$. Then, the path of any particle in the swarm is directed by the following equations:

$$x(t) = A_1 \sin(\theta t) + A_2 \cos(\theta t) \quad (23)$$

$$A_1 = (2v_0 - \varphi x_0 - \varphi p)/(\sqrt{|\varphi^2 - 4\varphi|}) \quad (24)$$

$$A_2 = x_0 - p \quad (25)$$

$$\theta = tan^{-1}(\sqrt{|\varphi^2 - 4\varphi|}/|2 - \varphi| \quad (26)$$

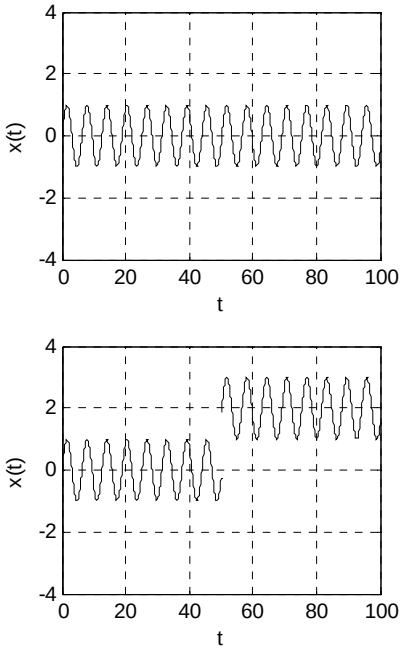


Figure 1. Trajectories of a particle in a simplified PSO algorithm with Eq. (23) and Eq. (27), respectively.

As Clerc and Kennedy [16] concluded, the particle as seen in discrete time "surfs" on an underlying continuous foundation of sine waves, and each time it jumps from one wave to another by using random numbers in the periphery of a random number distribution. At this point, let's proceed with the Eq. (23) and add redefined Kronecker delta function as follows:

$$x(t) = \begin{cases} x(t) + A_n\delta & \delta = 1 \; if \; t \geq nf, n = 1,2,... \\ x(t) & \delta = 0 \; if \; t < nf \end{cases} \quad (27)$$

here f is the impulse period/frequency and A_n is the third amplitude which is either negative or positive. The resulting graphs for Eq. (23) and Eq. (27) with positive A_n and f with 50 can be plotted in Fig. 1. The purpose of adding an impulse depending on the time frame is to catch the big wave, or in other words, to extend the search domain from one local area to another local area for the particle. The effect of an impulse will be analyzed in further sections.

Improved PSO algorithms

When a new algorithm is proposed into literature, it is expected to be compared with the current state of art algorithms on commonly used benchmark test functions. Obviously, it is difficult to make a fair comparison, because each algorithm may have peculiar tuning, which results in important differences on the cost functions. However, well-defined and straightforward algorithms may provide a relatively good inference. Three well known PSO algorithms are selected as comparative optimization algorithms. These are constriction factor PSO (c-PSO), linearly decreased inertial weight PSO (w-PSO), and Gaussian mutation based PSO (g-PSO).

c-PSO The particle swarm with a constriction factor is introduced by Clerc [16], which investigated the use of a parameter called the constriction factor. With the constriction factor K, the particle velocity and position dimensions are updated via:

$$v_{i,j}(t) = K((v_{i,j}(t-1) + c_1 r_1(p_{I,j}(t-1) - x_{i,j}(t-1)) + c_2 r_2(p_{g,j}(t-1) - x_{i,j}(t-1))) \quad (28)$$

$$K = \frac{2}{\left|2 - \psi - \sqrt{\psi^2 - 4\psi}\right|}, \quad \psi = c_1 + c_2, \psi > 4 \quad (29)$$

$$x_{i,j}(t) = x_{i,j}(t-1) + v_{i,j}(t)$$

A particularly important contribution of this factor is that if it is correctly chosen, it guarantees the stability of PSO without the need to bind the velocities. Typically, values of *2.05* are used for c_1 and c_2, making ψ is equal to *4.1* and K is equal to *0.729*.

w-PSO Shi and Eberhart [17] introduced the idea of a time-varying inertia weight. The idea was based on the control of the diversification and intensification behavior of the algorithm. The velocity is updated in accordance with the following expressions:

$$\begin{gathered} v_{i,j}(t) = w(t)v_{i,j}(t-1) \\ +c_1 r_1\left(p_{I,j}(t-1) - x_{i,j}(t-1)\right) \\ +c_2 r_2(p_{g,j}(t-1) - x_{i,j}(t-1)) \\ x_{i,j}(t) = x_{i,j}(t-1) + v_{i,j}(t) \end{gathered} \quad (30)$$

The inertia weight, w, is decreased linearly starting from initial point, w_{ini}, and ending to last point, w_{end}, related to maximum iteration number, T. Normally, the starting value of the inertia weight is set to *0.9* and the final to *0.4*. However, we tuned them to [0.6, 0.2] range for better performance.

g-PSO First Gaussian mutation based PSO algorithm is introduced by Higashi and Iba [4], which use a mutation operator that changes a particle dimension value using a random number drawn from a Gaussian distribution. A particle is selected for mutation using a mutation rate that is linearly decreased during a run. However, it is improved by several researchers. Pant *et al.* [18] developed Gaussian mutation operator application technique for updating the position of the swarm particles. The mutation operator is activated only when the diversity of the swarm becomes less than a certain threshold, d_{low}. The velocity is updated via Eq. (30) in addition to the following criteria:

$$\begin{gathered} if\ d < d_{low} \\ then\ x_{i,j}(t) = x_{i,j}(t-1) + \xi\ rand \end{gathered} \quad (31)$$

$$d = \frac{1}{s}\sum_{i=1}^{s}\sqrt{\sum_{j=1}^{d}\left(x_{i,j}(t-1) - \bar{x}_j(t-1)\right)^2} \quad (32)$$

$$\bar{x}_j(t-1) = \sum_{i=1}^{s} x_{i,j}(t-1)\,/s \quad (33)$$

where ξ is the scaling factor, *rand* is a random number generated by Gaussian distribution. The diversity description given by Eq. (32) is based on the differences between the average dimensional position and the current dimensional particle positions.

v-PSO. The traditional general form of the mutation which was applied in the previous PSO algorithm can be written as $x_{i,j}(t) = g(x_{i,j}(t))$; where g is the mutation operator providing the offspring vector. Instead of this strict form of a mutation operator it can be described including mutation strategy as

$$x_{i,j}(t) = F(g(x_{i,j}(t)), fr) \quad (34)$$

where F is the generalized mutation function, fr is a user defined application frequency. Right after the elitism application originating from GA, in every fr^{-1} period of the generations applying the mutation operator to all particle dimensions of the whole swarm, individuals in the population spread throughout the design space. Mutation operator is given by

$$\begin{gathered} x_{i,j}(t) = x_{i,j}(t)[1 + A(0.5 - rand)\delta] \\ i = 1,2,\dots,s \\ j = 1,2,\dots,d \\ \delta = \begin{Bmatrix} 1 & if\ t = nfr, n = 1,2,\dots \\ 0 & if\ t \neq nfr \end{Bmatrix} \end{gathered} \quad (35)$$

where A is a user defined scale factor called an amplitude and it may be selected as a fixed number or computed during the iterations, *rand* is a real random number specified by random number generator in accordance with $N[0, 1]$. In the applications Gaussian probability density function is used. However other density functions can also be used in Eq. (35). The aim of designing such a mutation strategy is to catch the big wave for escaping from local traps and getting the correct search pattern. The velocity and the positions are updated via Eq. (30) except the iterations corresponding to the mutation period.

Comparison of c-PSO and v-PSO in a simple case

Before moving to test bed of benchmark functions it may be beneficial to test c-PSO and v-PSO algorithms in case of 3D multi-modal Rastrigin test function which is given in Table 1. The experimental set up for c-PSO is given as; S is equal to *5*, G is equal to *200*, and for v-PSO; S is equal to *5*, G is equal to *200, f* is equal to *10, A* is equal to *5,* elite count is equal to *3, c_1* is equal to 1.5, c_2 is equal to 2, w is equal to *0.05*. Both algorithms are run 40 times and the results are averaged. The resulted plots can be seen in Fig. 2 and 3. We can see the efficiency of periodic mutations by looking at averaged best/swarm f_{min} values versus generations in Fig. 2. The best value resulted by c-PSO at generation with 200 can be provided by v-PSO at the generation by 70. The accuracy at the end of maximum generation for c-PSO is about 10^0 grades. However the accuracy given by v-PSO is about 10^{-1} grades. The other interesting point is the fluctuation of average swarm objective function values related to mutation frequency. Although the dispersion of the swarm is high at the start of each period the swarm is getting closer to global optimum at the end of the periods.

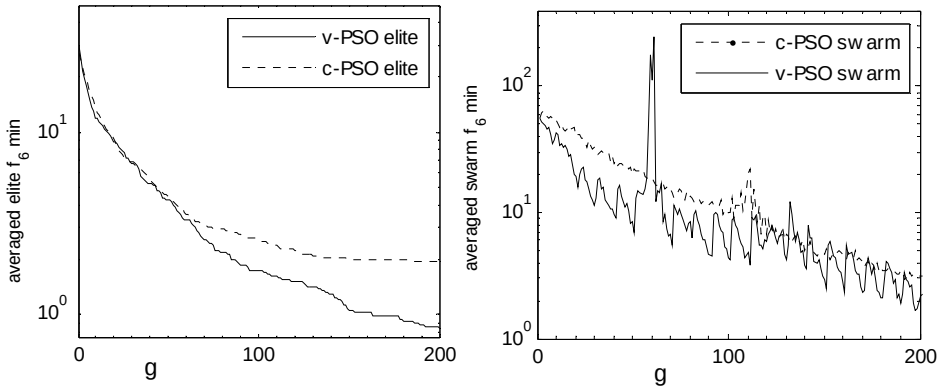


Figure 2. Averaged elite/swarm objective function changes versus generations for c-PSO and v-PSO algorithms.

The swarm's trajectories in 4-dimensional domain can be seen in Fig. 3. In plots the x axis of particle is ridden on the generations (g) with a certain range. Totally 40 runs including 40,000 particle positions are plotted in each figure. The dispersions of the particles at each mutation period are clearly observed during the generations. The dispersions are getting smaller while the iterations are getting forward.

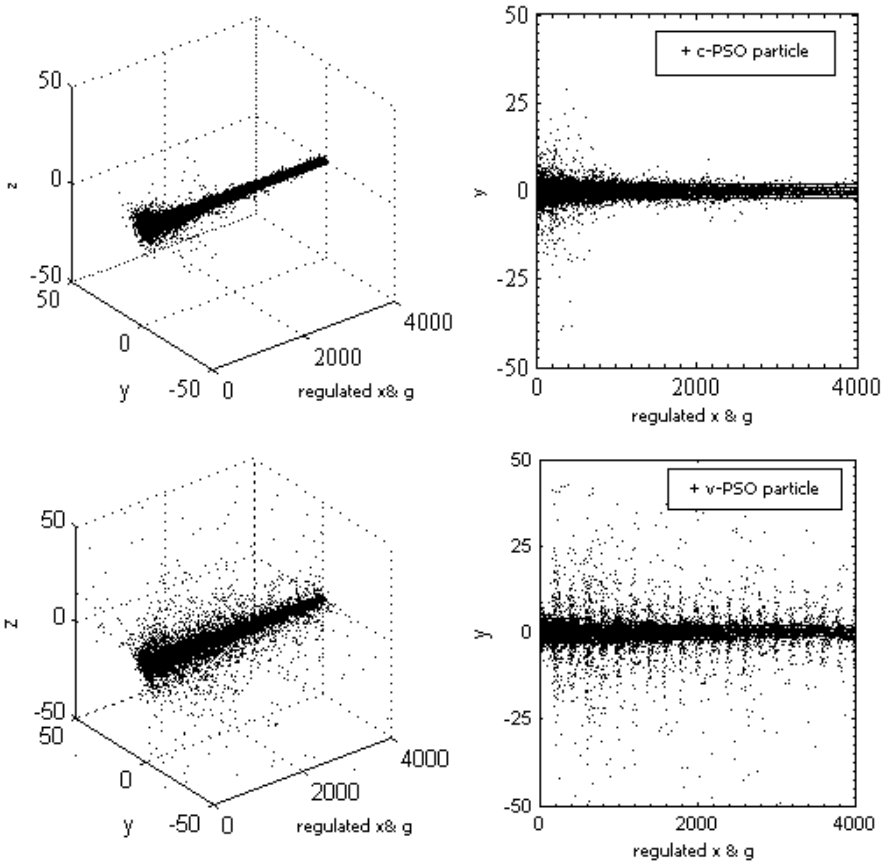


Figure 3. Swarm particle trajectories in 4-D domain for c-PSO and v-PSO.

Comparison of improved PSOs based on diversity and kinetic energy

The description of swarm diversity may be based on the difference between the average dimensional value and particle's dimensional values or between the elite individual in the swarm and other particle's dimensional values. The selected diversity is given by Eq. (32). Kinetic energy is constructed via particle's velocity values and it is given by Eq. (34). It is designed to indicate swarm energy required to escape from the tackles.

$$Ke(t) = \sum_{i=1}^{S} \sum_{j=1}^{D} \frac{x_{i,j}^2(t)}{2} \qquad (36)$$

The swarm diversity and kinetic energy changes versus time domain for 20 dimensional Rastrigin function problem are plotted in Fig. 4 for each selected PSO algorithm. The swarm size and maximum iteration number are fixed to 20 individuals and 10,000 generations, respectively for each algorithm. All algorithms are run 40 times and the results are averaged. The diversity of c-PSO linearly decreases during the generations and after a while swarm becomes unique and the kinetic energy of the swarm is also decreased to low band periphery. The diversity of v-PSO is also decreased during the iterations but having a difference. The degradation is smaller than c-PSO and the diversity band is big enough to extend the search ability of the swarm. The similarity between w-PSO and g-PSO is clearer until a certain iteration step. This is normal because g-PSO is the same as w-PSO until the diversity value decreases to threshold diversity value. At that time the Gaussian mutation operator takes role and keeps the diversity in a certain levels. The same level stability can be seen in kinetic energy change. w-PSO loses its diversity capability later than c-PSO.

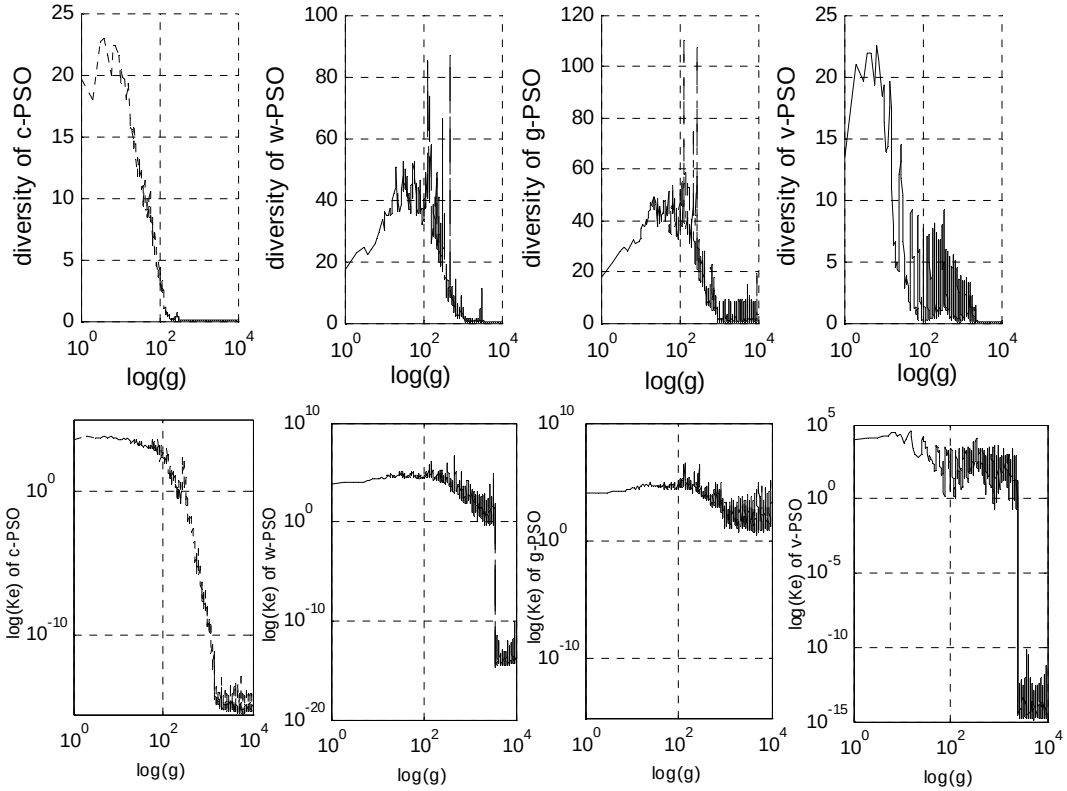


Figure 4. Diversity and kinetic energy change versus generations for selected PSO algorithms.

Inference with wavelet analysis in g-PSO

Describing a threshold value for the diversity has a big effect on the swarm diversity. However it may cause some concerns related to convergence. The swarm diversity is directly related to the threshold value. Small threshold value may prevent the swarm having big enough diversity to escape from local optima. On the other hand, larger threshold diversity may result in divergence from the right path. However it brings some important points such as periodicity and amplitude. The diversity change versus generations for different threshold diversity values can be seen in Fig. 5. The selected threshold values are 10^{-6}, 10^{-4}, 10^{-1}, and 10^{0}. According to the plots the threshold diversity value controls the significance of mutation application and the amplitude of the swarm. While the threshold diversity increases the amplitude decreases, the periodicity starts earlier. This feature has a direct effect on the averaged best individual value. Fig. 6 shows the relationship between averaged best individual values versus threshold diversity. The relationship between diversity threshold and averaged best individual is nonlinear. This is because the threshold value has a double effect on both mutation application and amplitude at the same time.

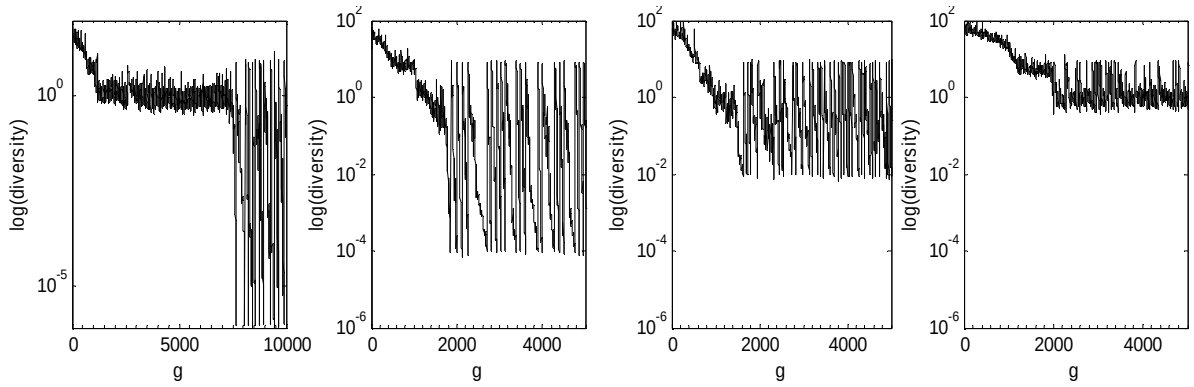


Figure 5. The diversity changes versus generations for different diversity thresholds.

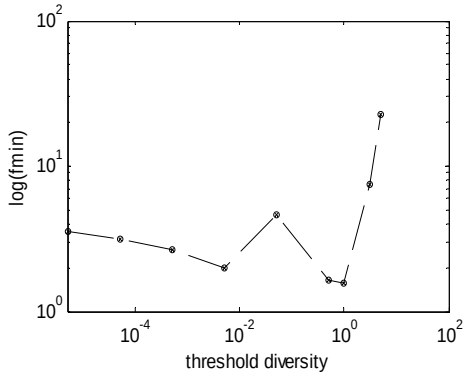


Figure 6. Averaged best individual and threshold diversity changes in g-PSO algorithm.

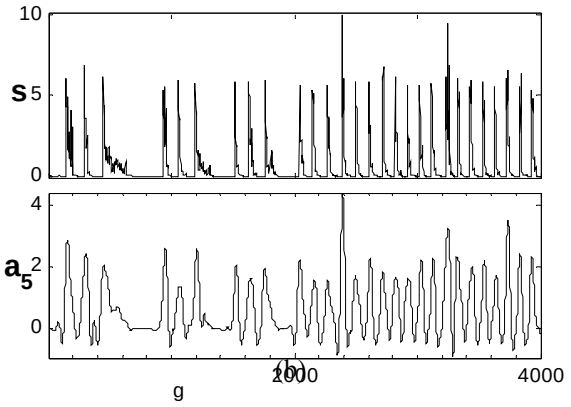
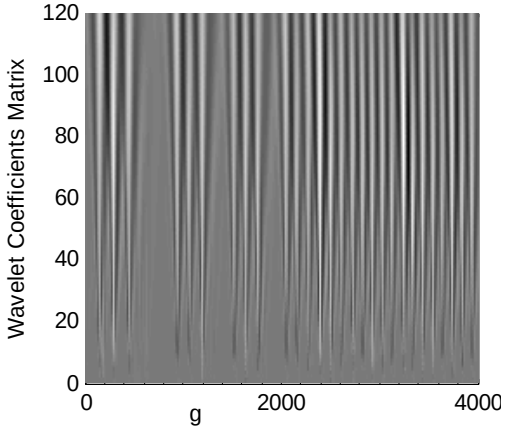


Figure 7. The wavelet analysis of the diversity signal in g-PSO algorithm (a) coefficient matrix, (b) time series.

The significance of mutation application can be searched in more details. The diversity change for the first threshold value (10^{-6}) can be analyzed by using db4 wavelets at 5-level [19]. The resulted discrete wavelet transform for the diversity series can be seen in Fig. 7. Plot 7 shows that the main signal caries periodic signal and noises. The periodicity is clearly seen after 2000th iteration in the approximation signal which caries low frequency signals and has 90 percentages of the whole signal power and on the wavelet coefficients matrix.



(a)

The deficiency of g-PSO originates from the double effect of threshold diversity. However it brings two important concepts which are periodicity and amplitude change related to generations. The description of v-PSO given in Eq. (35) pacifies the deficiency of Gaussian mutation application and facilitates the algorithm to control the periodicity and amplitude with different parameters which are the frequency and the amplitude.

3. APPLICATIONS AND RESULTS

At first, v-PSO and the comparative PSO algorithms are tested using test functions including unimodal and multi-modal benchmark functions. Secondly, v-PSO algorithm and the best comparative PSO algorithm, g-PSO, are applied to path planning problem of UAV in 3D terrain environment.

Comparative results using test functions

The selected test bed is given in Table 1. The test procedure includes two different bundles. The first bundle contains the algorithm sensitivity to different function dimensions as *D* is equal to 10, 20, and 30 with fixed generation size as 10,000 and fixed swarm size as 20. The second bundle contains the behaviors of the algorithms related to different swarm sizes such

as 10, 20, and 30 with fixed dimensions as *D* is equal to 30, and fixed function evaluations of 200,000. All algorithms are run 100 times and the results are averaged. Peculiar settings are the following: c_1 and c_2 are equal to 2.05 for c-PSO; c_1 and c_2 are equal to 2; w_{ini} and w_{end} are equal to 0.6, 0.2, respectively for w-PSO; c_1 and c_2 are equal to 2, w_{ini} and w_{end} are equal to 0.6, 0.2, respectively, D_{low} is equal to 0.5 for g-PSO; c_1 is equal to 1.5, c_2 is equal to 2, *w* is equal to 0.05, *fr* is equal to 10, *A* is equal to 1, elite count is equal to 3 for v-PSO.

Fixed iteration rate-fixed swarm size

The resulting values are shown in Table 2. The performance of four algorithms with different dimensions is tabulated in terms of means and averaged CPU times for six test functions. The test runs were executed on an off-the-shelf laptop computer that has Intel Centrino Duo processor with 64-bit accuracy. The mean value is calculated in accordance with 95% confidence interval ratio. The best results among the four algorithms are shown in bold. The averaged global best individual values versus generations for *D* is equal to 30 are shown in Fig. 8.

The best performance belongs to v-PSO in all test cases. The accuracy and the efficiency superiorities of v-PSO can be clearly seen in Fig. 8. It converges within 10^2 or 10^3 generation number to the acceptable objective function values. However, the other algorithms converge within 10^4 generation number to the same or worse objective function values.

Fixed function evaluation rate-fixed dimension rate

The averaged best function values versus swarm sizes which are *10*, *20*, and *30* for *D* which is equal to *30* are shown in Fig. 9 and in Table 3. The mean value is calculated in accordance with *95%* confidence interval ratio. The best results among the four algorithms are shown in bold. The similar results are observed comparing with the previous analysis. v-PSO has the best performance in all function optimizations. Fig. 9 shows an interesting result in the relationship between the performance and the swarm size. Generally, v-PSO has similar performances for all swarm sizes. It means v-PSO provides a better efficiency in lower swarm sizes. However, the other algorithms' performances significantly depend on the swarm size. Typically, larger swarm size means better performance for them except v-PSO. The main reason for this result is that v-PSO provides enough diversity within the swarm even in a low swarm size. The other algorithms need a larger swarm size to have enough diversity in the swarm.

Table I: Definition of test functions.

f	Test function	*f(x)*	search range	*x**	*f(x*)*
f_1	Ackley	$-20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right)-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right)+20+e$	$[-30, 30]^D$	[0,0,…,0]	0
f_2	Cosine mixture	$\sum_{i=1}^{n}x_i^2-0.1\sum_{i=1}^{n}\cos(5\pi x_i)$	$[-1, 1]^D$	[0,0,…,0]	-0.1n
f_3	Exponential	$-\exp\left(-0.5\sum_{i=1}^{n}x_i^2\right)$	$[-1, 1]^D$	[0,0,…,0]	-1
f_4	Griewank	$1+\frac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)$	$[-600, 600]^D$	[0,0,…,0]	0
f_5	Rastrigin	$10n+\sum_{i=1}^{n}[x_i^2-10\cos(2\pi x_i)]$	$[-5.12, 5.12]^D$	[0,0,…,0]	0
f_6	Schwefel	$418.9829n-\sum_{i=1}^{n}x_i\sin\left(\sqrt{\lvert x_i\rvert}\right)$	$[-500, 500]^D$	[420.9687…,420.9687]	0

Table II: The first test bundle results.

		c-PSO		**w-PSO**		**g-PSO**		**v-PSO**	
f	*D*	mean	t_{CPU}	Mean	t_{CPU}	mean	t_{CPU}	mean	t_{CPU}
f_1	10	0.1040±0.0659	1.612	2.5e-15±9.9e-17	1.5219	3.258e-4±4.5e-5	1.634	**1.84e-15±2.9e-16**	1.717
	20	1.7684±0.2544	1.964	0.0231±0.0322	1.7922	0.0013±1.8e-4	1.95	**2.84e-15±1.5e-16**	1.957
	30	3.8334±0.3806	2.406	0.0266±0.0374	2.163	0.0027±3.7e-4	2.307	**4.93e-15±3.4e-16**	2.296
f_2	10	-0.9483±0.0158	1.557	-0.9970±0.0041	1.4688	-0.9998±3.1e-5	1.568	**-1±0**	1.644
	20	-1.533±0.0485	1.938	-1.9512±0.0162	1.74	-1.9979±2.1e-4	1.883	**-2±0**	1.921
	30	-2.0261±0.0692	2.318	-2.8167±0.0286	2.11	-2.9942±5.1e-4	2.255	**-3±0**	2.26
f_3	10	-1±2.2e-18	1.468	-1±0	1.371	-1±1.5e-6	1.422	**-1±0**	1.55
	20	-1±2.15e-17	1.78	-1±1.7e-17	1.593	-0.9999±9.2e-6	1.625	**-1±3e-18**	1.771
	30	-1±5.14e-14	2.136	-1±5.4e-18	1.877	-0.9998±1.9e-5	1.901	**-1±1e-17**	2.039
f_4	10	0.0915±0.0093	2.174	0.0742±0.0075	2.075	0.0549±0.0046	2.125	**0.0209±0.006**	2.251
	20	0.0586±0.0478	2.577	0.0291±0.005	2.396	0.0222±0.0041	2.483	**0.0026±0.002**	2.553
	30	0.2238±0.1616	3.043	0.0153±0.0035	2.835	0.0166±0.004	2.946	**8.8568e-4±0.001**	2.991
f_5	10	8.6760±0.7852	2.133	5.1837±0.5259	2.055	0.0398±0.0389	2.111	**0±0**	2.21
	20	38.4849±2.1134	2.635	23.7496±1.6545	2.483	2.8259±0.6476	2.537	**0±0**	2.621
	30	84.6011±4.6683	3.146	53.5088±2.5935	3.004	14.1188±2.24	3.033	**5.6843e-16±1e-15**	3.068
f_6	10	836.04±61.8981	2.732	842.7311±59.72	2.185	815.1561±64.3	2.232	**620.8131±50.4**	2.371
	20	2.375e+3±112.69	4.222	2.1577e+3±101	3.093	2.1798e+3±120.9	3.021	**1.3384e+3±68.5**	3.128
	30	4.146e+3±177.38	5.451	3.62e+3±158.35	3.945	3.47e+3±149.3	3.781	**2.1395e+3±103.3**	3.971

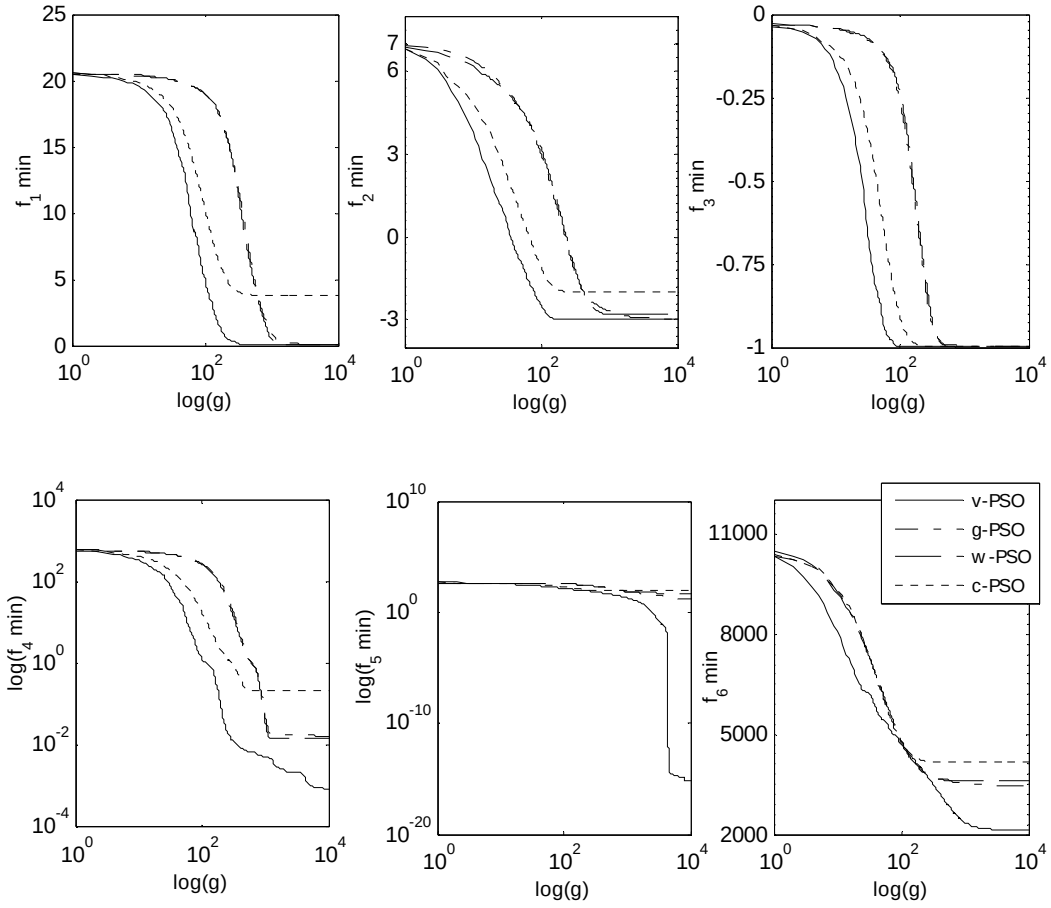


Figure 8. Averaged best function values versus generations for test functions.

Table III: Second Test Bundle Results.

f	*P*	c-PSO	w-PSO	g-PSO	v-PSO
		mean	Mean	Mean	mean
f_1	10	9.412±0.6149	1.5913±0.4344	0.0803±0.096	**4.583e-15±3e-16**
	20	4.6684±0.4644	0.0902±0.067	0.0026±3.7e-4	**4.9738e-15±3e-16**
	30	2.2994±0.2776	0.0116±0.023	0.0027±3.9e-4	**5.0449e-15±3e-16**
f_2	10	-1.4844±0.0918	-2.6113±0.05	-2.996±3.3e-4	**-3±0**
	20	-1.9673±0.0767	-2.833±0.03	-2.9937±6.1e-4	**-3±0**
	30	-2.1842±0.071	-2.8714±0.03	-2.9928±7.3e-4	**-3±0**
f_3	10	-1±8.8e-5	-1±1e-17	-0.9999±1e-5	**-1±1e-17**
	20	-1±2e-13	-1±3e-18	-0.9998±1.74	**-1±1e-17**
	30	-1±1e-16	-1±2e-18	-0.9997±2e-5	**-1±1e-17**
f_4	10	0.8154±0.275	0.0339±0.01	0.0223±0.005	**0.0016±0.002**
	20	0.1233±0.064	0.0167±0.003	0.0155±0.003	**0.0011±0.001**
	30	0.0431±0.01	0.0153±0.003	0.0153±0.003	**5.9021e-4±0.001**
f_5	10	104.3609±5.25	69.9753±3.63	12.1629±2.31	**0±0**
	20	81.8152±3.98	52.8919±2.42	13.5722±1.77	**1.1369e-15±1e-15**
	30	73.6367±3.7	44.8626±2.45	15.4020±1.98	**5.6843e-16±1e-15**
f_6	10	4.6531e+3±142	3.9629e+3±161	3.6985e+3±194	**2.2246e+3±123**
	20	3.978e+3±149	3.5837e+3±147	3.5119e+3±188	**2.1888e+3±110**
	30	3.7372e+3±140	3.4045e+3±134	3.5136e+3±152	**2.2258e+3±107**

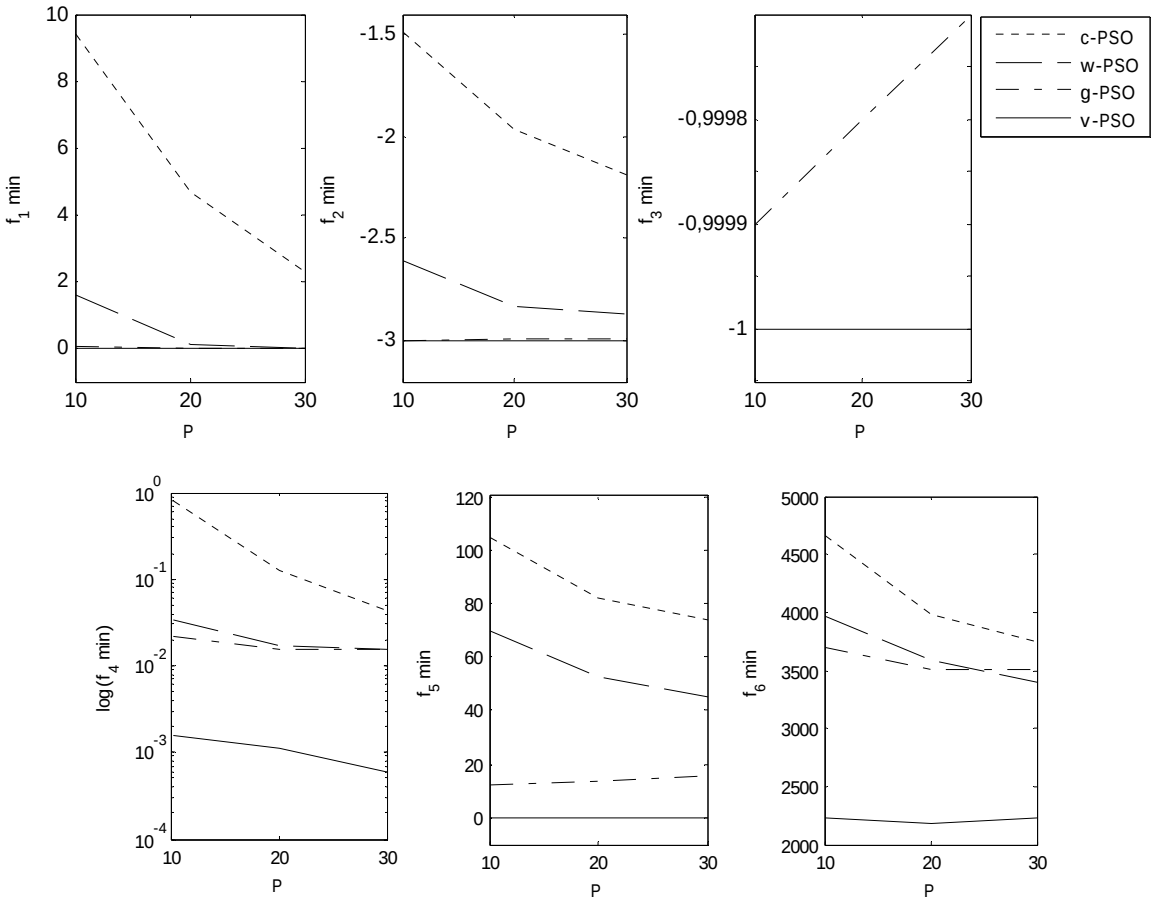


Figure 9. Averaged best function values versus swarm sizes for selected test functions.

Path planning problem

The number of applications that consider the use of autonomous UAVs is increased for civilian or military purposes. There are many activities that must be carried out by a UAV system to enable the execution of the autonomous navigation. These activities are mapping and modeling the environment, path planning, and flight control systems. Path planning is one of the most important problems in the navigation

process. Its objective is to find out the optimal flight path starting from the departure location and ending in an arrival location in the proper duration during which UAV is able to accomplish the pre-arranged task and avoid the hostile threats. The selection of an appropriate algorithm in every stage of the path planning process is very important. Optimal path planning heavily relies on time consuming optimization techniques such as numerical computation. It is usually solved offline based on the known information before takeoff. To demonstrate, v-PSO and g-PSO are applied and compared in path planning problems in a 3D sinusoidal terrain model.

Path representation

Bezier curves [20] have been widely adopted when computing smooth, dynamically feasible trajectories for UAVs. A smooth path is an important feature for UAV flight dynamics because aerial vehicles cannot fly on line segments like land vehicles. Bezier curve is defined by

$$\begin{aligned} x_1(d) &= \sum_{i=0}^{n} B_i^n d^i (1-d)^{n-i}\, x_{1,i} \\ x_2(d) &= \sum_{i=0}^{n} B_i^n d^i (1-d)^{n-i}\, x_{2,i} \\ x_3(d) &= \sum_{i=0}^{n} B_i^n d^i (1-d)^{n-i}\, x_{3,i} \\ B_i^m &= n!/i!\,(n-1)! \end{aligned} \tag{37}$$

where d is the step size parameter of the curve whose values vary uniformly between [0,1], n is the number of control points, ($x_{1,i}$, $x_{2,i}$, $x_{3,i}$) are the coordinates of the i^{th} control point which define the path coordinates ($x_1(d)$, $x_2(d)$, $x_3(d)$).

Terrain modeling

The function of trigonometric based terrain modeling is given by

$$\begin{aligned} (x_1, x_2) = \sin(x_2 + b_1) + b_2 \sin(x_1) \\ + b_3 \cos\left(b_4 \sqrt{(x_1)^2 + (x_2)^2}\right) \\ + b_5 \cos(x_2) \\ + b_6 \sin\left(b_6 \sqrt{(x_1)^2 + (x_2)^2}\right) \\ + b_7 \cos(x_1) \end{aligned} \tag{38}$$

where b_i are real constant numbers. The terrain model is depicted in Fig. 10. Sinusoidal terrain model is a mathematically continuous model.

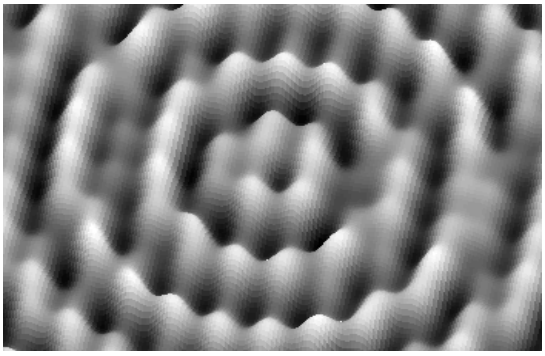


Figure 10. Sinusoidal terrain model.

Fitness function

The evaluation function of an individual measures the cost of a candidate path. The fitness function is designed to accommodate three different terms: minimize the distance flown, maintain a smooth trajectory preventing sharp turns, and satisfy the clearance providing the safe distance for UAV from terrain. We have proposed a linear combination of these three factors. The general description of the optimization problem is given below

$$\min f = \sum_{i=1}^{3} a_i G_i \tag{39}$$

subject to

$$\begin{gathered} x_3^{curve} - x_3^{surface} < S_d \\ \theta < \theta_S \\ x_3^{curve} < A^L \end{gathered}$$

where G_i is the term connected to various concerns, a_i is weighting constant, and A^L is the altitude limitation for UAV. The first concern is the length of the curve G_1 and is calculated by the given expression

$$\begin{aligned} G_1 = \sum_{i=1}^{dn-1} [(x_{1,i+1} - x_{1,i})^2 + (x_{2,i+1} - x_{2,i})^2 \\ + (x_{3,i+1} - x_{3,i})^2]^{1/2} \end{aligned} \tag{40}$$

where dn is the curve discretization number; $x_{1,i}$, $x_{2,i}$, and $x_{3,i}$ are the discrete coordinates of the path curve. The second concern, G_2, is the passing ratio of the curve through the terrain boundary and is calculated by the given expression:

$$G_2 = \sum_{i=1}^{dn} if\ x_{3,i}^{curve} - x_3^{surface} < S_d,\quad punishment + 1 \tag{41}$$

where S_d is a safe distance determined by user, x_3^{curve} is the discrete path curve coordinate, and $x_3^{surface}$ is the terrain model coordinate. This expression penalizes the curve that passes through the solid boundary. So, the penalty is proportional to the number of discretised curve points located under the solid

surface. The third concern, G_3, is the curvature angle ratio of the curve which is calculated by the given expression:

$$G_3 = \sum_{i=1}^{n-1} if\ \theta_{i,i+1} < \theta_S,\quad punishment + 1 \qquad (42)$$

where $\theta_{i,i+1}$ is the angle between the extension of the line segment connecting Bezier control points i and $i+1$, θ_S is the safe turning angle determined by user for the UAV. This concern is designed to prevent the aerial vehicle from exceeding the lateral and vertical acceleration limits, because the flight envelope determines the maximum radius of turns for flying objects. The weight constants, a_i, are determined experimentally. In test cases, a_2 is selected as a high number.

Test case and results

The test case is given in Fig. 11. The mission of UAV is to depart from the departure point and arrive at the arrival point under certain conditions. Two particle swarm algorithms are tested in this case. These algorithms are v-PSO and g-PSO, and the features of these algorithms are tabled in Table 4. The data for fitness function is given in Table 5.

Table IV: Particle swarm algorithms' features.

Algorithm	c_1	c_2			
g-PSO	2	2	w_{ini} 0.6	d_{low}	ξ
			w_{end} 0.2	5	0.1
v-PSO	2	2	w_{ini} 0.6	fr	A
			w_{end} 0.2	20	0.1

Table V: Fitness function data.

S_D	θ_T	A^L	a_1	a_2	a_3
0.05	60º	5	5	10	1

All algorithms are run 30 times. The averaged best individual fitness function values versus generations are plotted in Fig. 12. According to this figure, v-PSO outperformed g-PSO. v-PSO reaches the fitness value of 458 at 318th generation, while g-PSO reaches the same value at 500th generation. Therefore, v-PSO decreases 36% the required generation number.

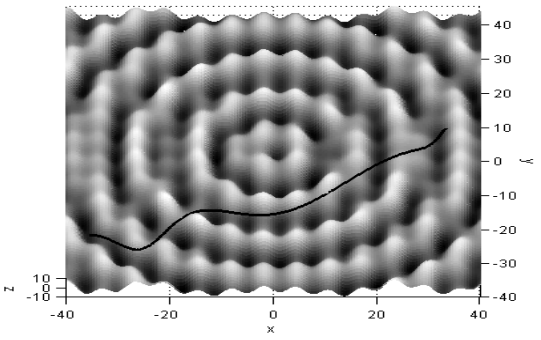


Figure 11. An example path constructed by v-PSO.

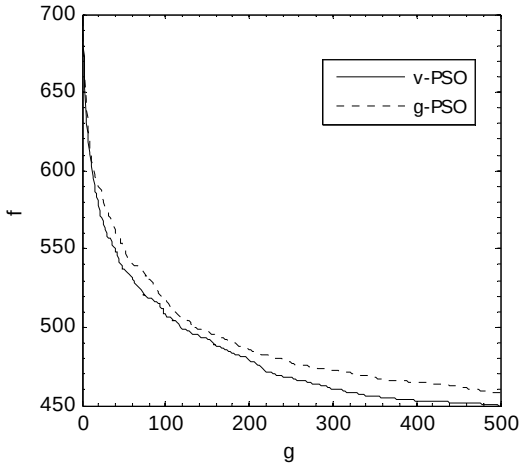


Figure 12. The fitness values of algorithms versus generations.

4. CONCLUSIONS

We have re-observed that in the general case, particle "surfs" on sine waves in nature. A particle seeking an optimal location attempts to catch another wave randomly, manipulating its frequency and amplitude. However manipulations are not enough to catch the big wave directing the global solution. During the optimization process, the swarm looses its diversity and the particles become more and more similar, and gather into the neighborhood of the best particle in the swarm, which makes the swarm premature convergence probably around the local solution. Although mutation applications provide diversity in the swarm diversity without continuous impetus disappears during the generations. A mutation with a separate amplitude and periodicity seems to help the particle to "jump" onto another big wave. Vibrational PSO is a good example to catch the big wave toward the global optimum. Its efficiency is observed in multi-modal test functions and a path planning problem. v-PSO is usually decreased the required objective function evaluations down to %65 or less depending on the function nature and also gives more accurate results than other comparative algorithms. Current selected algorithms seem to be dependent on the population size. However v-PSO is almost independent from population size. It shows similar performances although the population size is increased. The other algorithms increase their performances when the population size is increased under the same problem conditions. This is because the population size directly affects the diversity in the population and these algorithms pasifize the lack of diversity within the big sized populations. v-PSO does not need more individuals because the periodic mutations provide enough diversity in the swarm. However, the diversity provided by vibrational mutation operator is a global random diversity. In some cases this type of diversity may not be enough to catch the correct wave. In addition to global

random diversity the local random diversity combining with elite particle in the swarm may help fast convergence. For further step local diversity in terms of randomness and controlled phenomena promises faster and more accurate results in long but real optimization processes.

5. REFERENCES

[1] Eberhart, R. C., Kennedy, J.: A new optimizer using particle swarm theory. Proc. 6th Int. Symp. Micromachine Human Sci., Nagoya, Japan, pp. 39–43, (1995)

[2] Valle, Y., Venayagamoorthy, Mohagheghi, G. K., S., Hernandez, J.-C., Harley, R. G.: Particle swarm optimization: basic concepts, variants and applications in power systems. IEEE Trans. Evol. Comput., vol. 12, no. 2, pp. 171-195, (2008)

[3] Shi, Y., Eberhart, R.: Parameter selection in particle swarm optimization. Proc. 7th Annual Conf. on Evol. Prog., pp. 591-601, (1998)

[4] Higashi, H., Iba, H.: Particle swarm optimization with Gaussian mutation. Proc. of the IEEE Swarm Intell. Symp., pp. 72 – 79, (2003)

[5] Stacey, A., Jancic, M., Grundy, I.: Particle swarm optimization with mutation. Proc. of the IEEE Congr. Evol. Comput., pp. 1425-1430, (2003)

[6] Pant, M., Thangaraj, R., Singh, V. P., Abraham, A.: Particle swarm optimization using Sobol mutation. IEEE 1st Int. Conf. on Emerging Trends in Eng. and Tech., pp. 367-372, (2008)

[7] Andrews, P. S.: An investigation into mutation operators for particle swarm optimization. IEEE Congr. Evol. Comput., pp. 1044-1051, (2006)

[8] Zavala, A. E., Aguirre, A. H., Diharce, E. R. V., Rionda, S. B.: Constrained optimization with an improved particle swarm optimization algorithm. Int. J. Intell. Comput. and Cybern., vol. 1, no. 3, pp. 425-453, (2008)

[9] Jia, D., Li, L., Zhang, Y., Chen, X.: Particle swarm optimization combined with chaotic and Gaussian mutation. Proc. of the 6th World Congr. Intell. Cont. Auto., pp. 3281-3285, (2006)

[10] Wu, X., Cheng, B., Cao, J., Cao, B.: Particle swarm optimization with normal cloud mutation. Proc. on the 7th World Congr. Intell. Cont. Auto., pp.2828-2832, (2008)

[11] Chen, J., Ren, Z., Fan, X.: Particle swarm optimization with adaptive mutation and its application research in tuning of PID parameters. 1st Int.. Symp. Syst. Cont. Aerospace and Astronautics, pp.990-994, (2006)

[12] Yang, M., Huang, H., Xiao, G.: A novel dynamic particle swarm optimization algorithm based on chaotic mutation. IEEE 2nd Int. Workshop on Knowledge Discovery and Data Mining, pp.656-659, (2009)

[13] Liu, J., Fan, X., Qu, Z.: An improved particle swarm optimization with mutation based on similarity. 3rd Int.. Conf. Natural Comput., pp. 824-828, (2007)

[14] Biao, C., Zhishu, L., Die, F., Jian, H., Peng, O., Qing, L.: Mutated fast convergent particle swarm optimization and convergence analysis. IEEE 1st Int. Conf. on Intell. Netw. Intell. Syst., pp. 5-8, (2008)

[15] Ozcan, E., Mohan, C. K.: Particle swarm optimization: Surfing the waves. Proc. 1999 Congr. Evol. Comput., Washington, DC, pp. 1939–1944, (1999)

[16] Clerc, M., Kennedy, J.: The particle swarm-explosion, stability, and convergence in a multidimensional complex space. IEEE Trans. Evol. Comput., vol. 6, no. 1, pp. 58–73, (2002)

[17] Shi, Y., Eberhart, R.: A modified particle swarm optimizer. Proc. of the World Congr. Comput. Intell., pp. 69–73, (1998)

[18] Pant, M., Radha, T., Singh, V. P.: A New Diversity Based Particle Swarm Optimization using Gaussian Mutation. Int. J. Math. Modeling, Simulation and Appl., 1(1), pp. 47-60, (2008)

[19] Misiti, M., Misiti, Y., Oppenheim, G., Poggi, J-M.: Wavelets and Their Applications, ISTE Ltd., USA, pp. 8-12, (2007)

[20] Farin, G.: Curves and Surfaces for Computer Aided Geometric Design; A Practical Guide, Academic Press, Inc., pp. 41-42, (1993)

VITAE

Asst.Prof.Dr.Lt.Col. Y. Volkan PEHLIVANOGLU

He obtained his B.S degree from Aeronautical Engineering Department at Istanbul Technical University (ITU) in 1993, B. S. S degree from International Relations Department at Istanbul University (IU) in 1998, M.S degree from Aerospace Engineering Department at Old Dominion University (ODU in USA) and ASTIN in 2006, and PhD degree from Aerospace Engineering Department at Old Dominion University in 2010. His research interests are fluid dynamics, design and optimization methods.