MISSILE GUIDANCE ALGORITHM DESIGN USING PARTICLE SWARM OPTIMIZATION

Chien-Chun Kung[1] and Kuei-Yi Chen[2]
[1]*Department of Mechatronic, Energy and Aerospace Engineering, Chung Cheng Institute of Technology, National Defense University, Taoyuan County, Taiwan*
[2]*School of Defense Science, Chung Cheng Institute of Technology, National Defense University, Taoyuan County, Taiwan*
E-mail: cckung@ndu.edu.tw; kueiYi.chen@gmail.com

ICETI 2012-P2015_SCI
No. 13-CSME-84, E.I.C. Accession 3542

ABSTRACT
This paper presents a PSO guidance (PSOG) algorithm design for the pursuit-evasion optimization problem. The initialized particles are randomly set within the guidance command solution space and the relative distance is taken as the objective function. As the PSOG algorithm proceeds, the iteration will execute until the global optimum is reached. Two pursuit-evasion scenarios show that the PSOG algorithm has satisfied performance in execution time, terminal miss distance, time of interception, final stage turning rate and robust pursuit capability.

Keywords: missile guidance law, PSO algorithm, pursuit-evasion.

CONCEPTION D'UN ALGORITHME D'OPTIMISATION PAR ESSAIM DE PARTICULES POUR LE GUIDAGE DE MISSILES

RÉSUMÉ
Cet article présente la conception d'un algorithme à essaim de particules (PSOG) pour le problème d'optimisation de poursuite-évasion. Les particules initialisées sont réglées de manière aléatoire à l'intérieur de l'espace de commande de guidage, et la distance relative l'objectif. Pendant que l'algorithme d'essaim de particules procède, l'itération se poursuit jusqu'à ce qu'il parvienne à l'optimum global. Deux scénarios de poursuite-évasion montrent que l'algorithme PSOG présente une performance satisfaisante dans le temps d'exécution, l'écart de déviation, le temps d'interception, le taux de virage à l'étape finale, et la capacité de poursuite robuste.

Mots-clés : algorithme à essaim de particules ; poursuite-évasion ; optimum global ; capacité de poursuite robuste.

1. INTRODUCTION

In air intercept, a missile's navigation system calculates the relative position of the target to determine the flight path, and guides the missile to track its target effectively. Nowadays, many guidance laws have been proposed to intercept a maneuverable target, including PN, APN and GPN [1–3]. With the concept to minimize the performance index, e.g., the terminal miss distance, the optimal feedback guidance law was derived using optimal control theory [4], transfer function and acceleration constraint of a missile [5].

In recent years, solving optimization problems by using artificial intelligence techniques has become increasingly popular. To solve an optimal feedback guidance law, artificial intelligence strategies demonstrate excellent effects on the complex, nonlinear, dynamic, and even multi-objective pursuit-evasion system. For example, Lin et al. [6] developed a preliminary guidance and control design for guided missiles via a genetic searching approach. Choi et al. [7] derived a NN (Neural Network) guidance law from two-dimensional pursuit-evasion games. For improving interception performance, additional pattern scenario selection and hybrid guidance which combines NN guidance law and PN guidance have been designed. Li et al. [8] designed an optimal fuzzy logic controller of guidance law which is based on PN guidance law, and optimized the number of fuzzy rules by employing genetic algorithm. Omar et al. [9] designed an integrated fuzzy guidance law which consists of three fuzzy controllers.

Since the novel population-based stochastic optimization search algorithm, the particle swarm optimization (PSO), has been introduced in 1995 by Kennedy and Eberhart [10], the effective technique for optimization has been applied to solve varieties of optimization problems, including neural networks [11], multiple objective optimization [12], etc. Furthermore, Sang et al. [13] combine Lyapunov stability theory and the PSO algorithm.

PSO algorithm has a flexible and well-balanced evolutionary mechanism, a group of particles, adjusting their positions in solution space according to their own experience and their neighbors to enhance and adapt to the global and local exploration and exploitation abilities within a short calculation time. The algorithm does not require that the objective functions and the constraints have to be differentiable and continuous. In addition, the advantage of PSO algorithm is that this algorithm has fast convergence to the optimal solution, and easy to implement. Especially, PSO algorithm only has a few parameters to adjust. A particle swarm optimization is such an algorithm that seems to be suitable for application to nonlinear, dynamic, and multi-objective pursuit-evasion optimization problems.

In this paper, we are proposing a new PSO guidance algorithm design to search fast, global optimal guidance commands each within a short-range pursuit period. Using the point-mass model for a missile, the three-dimensional guidance commands along the pitch-axis and yaw-axis need to be determined. The PSO guidance algorithm thus is designed for starting with a reasonably sized swarm (about 40–100 particles) where the particles are initialized with a random distribution within the guidance command solution space. As the initial conditions of both missile and target are given, the particles will tend to cluster towards a global optimum with iterations proceeding. If the PSO guidance algorithm can work (with the convergence at the end of the run), a pursuit-evasion simulation system is performed to verify our proposed algorithm.

This paper is organized as follows: Section 2 give an overview of the PSO algorithm; Section 3 introduce the proposed PSO guidance algorithm; Section 4 presents simulation results for two cases of engagement scenarios and verifies the effectiveness of the proposed method; finally Section 5 presents concluding remarks.

2. BRIEF REVIEW OF PSO ALGORITHM

The basic idea of the PSO algorithm is that individuals (particles) learn from their own experiences, but also by mimicking the most successful behavior of the population. The population is called swarm. Let n denote the swarm size. Each particle $1 \leq i \leq n$ represents a possible solution to the optimizing problem at hand,

and is described by the position vector Q_i in the search space. The ith particle velocity vector is represented by V_i, which is usually clamped to a maximum velocity $V_{\max}$ specified by the user. The local best position achieved by the particle so far is indicated $Pbest_i$ and the best particle in the swarm is denoted $Gbest$.

The particles moving into the search space for a pre-defined number of iterations. The particles evaluate their positions according to certain fitness functions in each time step and adjust their own velocities and positions by sharing memories of their best positions so far, $Pbest_i$, and the best position of all particles in the swarm up to the current iteration, $Gbest$, according to

$$V_i^{new} = wV_i^{old} + c_1 rand_1(Pbest_i - Q_i^{old}) + c_2 rand_2(Gbest_i - Q_i^{old}) \tag{1}$$

$$Q_i^{new} = V_i^{new} + Q_i^{old} \tag{2}$$

The first term of Eq. (1) showing the particle diversity provides the previous velocity as momentum to "fly" through the search space, where w is the inertial weight factor. The second and third terms are the cognition and social parts, where c_1 and c_2 are the self-confidence factor and swarm-confidence factor used to scale the contribution of cognitive and social components, respectively. The cognitive component of a particle takes the best position found so far by this particle as the desired input to make the particle move toward its own best positions. The social component represents the collaborative behavior of the particles to find the global optimal solution. The social component always pulls the particles toward the best position. The $rand_1$ and $rand_2$ are two random numbers uniformly distributed in the range of [0, 1] employed to realize random exploration of particles.

At the start of the PSO process, the initial positions and velocities of particles are generated randomly and most of the particles will be distributed far away from the intended global optimum. The fitness function of each particle is then evaluated according to the defined objective function. At each iteration, the new velocity of each particle is calculated according to Eq. (1) and the updated position for the next fitness function evaluation is calculated according to Eq. (2). If a particle finds a better position than the previously found best position, e.g., $Pbest_i$ or $Gbest$, its location is stored in memory. As the algorithm proceeds, these particles will migrate to a region surrounding the global optimum by updating each generation. Later iterations will serve to fine tune the approach until the global optimum is reached.

3. MECHANISM OF PSO GUIDANCE ALGORITHM

Traditionally, many techniques including classical and modern guidance laws, e.g., Pursuit [14], PN [1], differential game [15], optimal control [4] and PID guidance [16] have been applied to the missile guidance problem. However, it is difficult to include all the constraints, uncertainties and nonlinearities in a single problem formulation. In concept, the PSO algorithm is simple, few in parameters, easy to implement and capable of jumping local optima. The algorithm is suitable for applying to nonlinear, dynamic, and multi-objective pursuit-evasion problems in air combats.

This paper proposes a new opinion to employ PSO to optimize the guidance for an air-to-air missile each within a short-range period. The period is designed according to the engagement scenarios. As the relative distance is far, the period is set loose; when the relative distance is short, the period is designed tight. The inputs of the PSO guidance algorithm are the dynamic states of target and missile and the outputs are the guidance commands including pitch and yaw accelerations.

According to the symbol definitions of PSO algorithm described in the previous section, let us denote the swarm size where each particle $1 \leq i \leq n$ is characterized by the particle position vector Q_i, the particle velocity vector V_i, and the local best position achieved by the particle so far $Pbest_i$. Moreover, let $Gbest$ denote the best particle in the swarm. The PSO guidance algorithm can be stated as follows and as per the block diagram shown in Fig. 1.

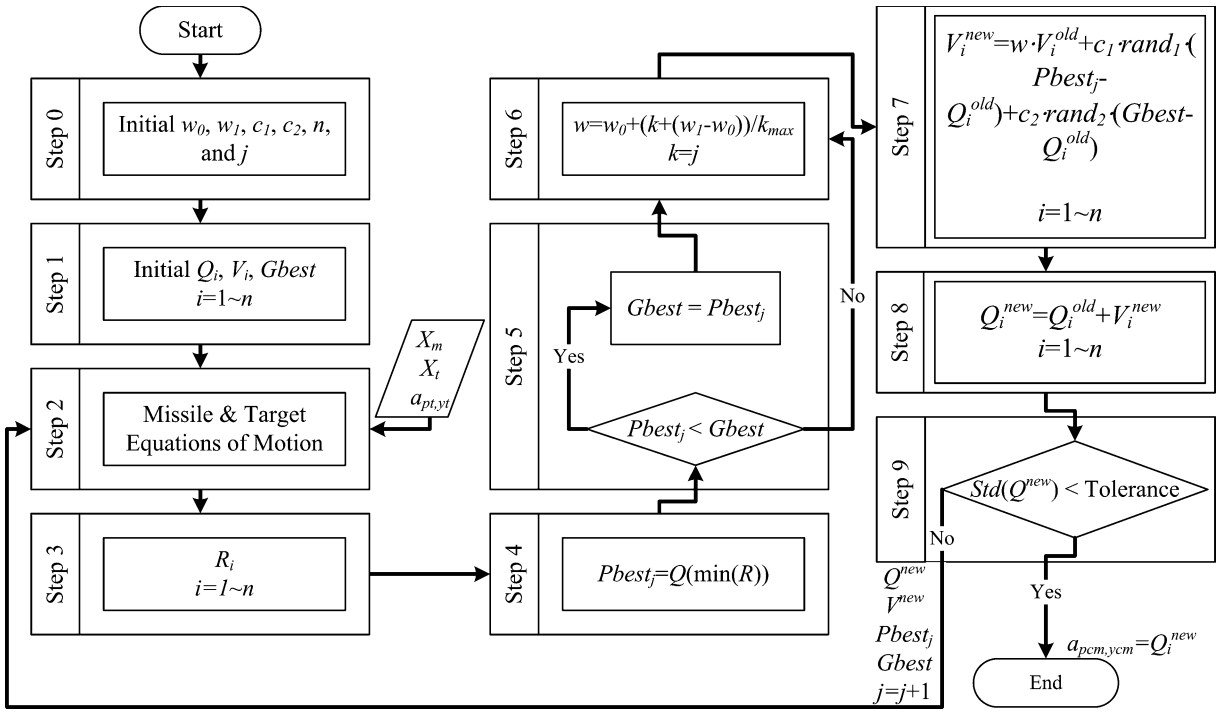

Fig. 1. The flow chart of PSO guidance algorithm.

Step 0: Choose the boundary values of inertial weight factor w_0 and w_1 ($w_0 < w_1$). Preset the self-confidence factor and swarm-confidence factors c_1, c_2, population size n and step counter j.

Step 1: Initialize population randomly with the location and velocity of the ith particle, Q_i and V_i for $1 \leq i \leq n$. The particles are initialized with a random distribution within the guidance command solution space. Initialize the best particle with location $Gbest$ in the population the best location $Pbest_i$ found so far with a copy of Q_i for each particle $i, i = 1,2,3\ldots,n$.

Step 2: The point-mass models of the dynamics of missile and target are employed as follows:

$$\dot{V} = \frac{1}{m}(T - D) - g\sin\gamma \tag{3}$$

$$\dot{a}_p = \frac{(a_{pc} - a_p)}{\tau} \tag{4}$$

$$\dot{a}_y = \frac{(a_{yc} - a_y)}{\tau} \tag{5}$$

$$\dot{\gamma} = \frac{(a_p - g\cos\gamma)}{V} \tag{6}$$

$$\dot{\Psi} = \frac{a_y}{V\cos\gamma} \tag{7}$$

$$\dot{x} = V\cos\gamma\cos\Psi \tag{8}$$

$$\dot{y} = V\cos\gamma\sin\Psi \tag{9}$$

$$\dot{z} = V\sin\gamma \tag{10}$$

$$D = k_{d1}V^2 + k_{d2}\frac{a_p^2 + a_y^2}{V^2} \tag{11}$$

From the above equations, we can get missile's and target's x-, y-, z-coordinates x_m, y_m, z_m and x_t, y_t, z_t, respectively; the velocity V_m and V_t; the flight path angle γ_m and γ_t; and the azimuth angle Ψ_m and Ψ_t. The movements of missile and target are controlled by the pitch and yaw accelerations (a_{pm}, a_{ym}) and (a_{pt}, a_{yt}), respectively. How to optimize the accelerations (a_{pm}, a_{ym}) by PSO algorithm is the focal point of this paper.

Let X_m stand for the vector of missile states, $X_m = [x_m, y_m, z_m, V_m, \gamma_m, \Psi_m, a_{pm}, a_{ym}]^T$. The updated X_m is the initial condition of the above dynamic Eqs. (3)–(11) for every time interval. The guidance commands a_{pcm} and a_{ycm} are set by the ith particle position vector Q_i. Then we can predict the missile's behavior with the ith particle position by solving the dynamic Eqs. (3)–(10) within a defined short-range period. In this paper, the Runge–Kutta numerical analysis method is used to get the solutions. Assuming the tactic of the target is invariable at the short-range period, the target's behavior can be predicted by estimating the target states X_t and accelerations a_{pt} and a_{yt} using Eqs. (3)–(10). Then, we can obtain the relative range between the missile and the target by transforming the 3-D Cartesian coordinates into the spherical coordinates. The predicted pursuit-evasion histories of all particles will be recorded in step 2.

Step 3: Pursuit-evasion is a nonlinear constrained multi-objective optimization problem where final time, energy consumption, and miss distance are usually treated as objectives. In this paper, the relative distance is taken into account to be the objective function. The fitness function is then evaluated according to the defined objective function. Apply the predicted histories in the above step to evaluate all fitness functions.

Step 4: The best location $Pbest_i$ found in this paper is different from the classical PSO algorithm. The $Pbest_j$ is defined as the best location of all particles in the swarm on the jth generation, i.e.,

$$Pbest_j = Q(\min(R)) \tag{12}$$

where Q is the position set of all particles.

Step 5: If $Pbest_j \leq Gbest$, then set $Gbest = Pbest_j$.

Step 6: The inertial weight factor is evaluated by

$$w = w_0 + \frac{j(w_1 - w_0)}{j_{\max}} \tag{13}$$

Step 7: Update the velocity V_i of each particle according to Eq. (1) and is confined within the range of $[-V_{\max}, V_{\max}]$.

Step 8: Update the location Q_i of each particle according to Eq. (2).

Step 9: Check if the standard deviation of the updated particle positions Q is less than the defined tolerance or not. If the standard deviation is greater than the tolerance, return to step 2 until termination criterion is met. If the standard error is less than the tolerance, output the missile's pitch and yaw guidance accelerations a_{pcm} and a_{ycm}.

A numerical simulation environment is constructed to validate the PSO guidance algorithm. The simulation processes are completely built on the Matlab/Simulink environment and is outlined as follows (see Fig. 2):

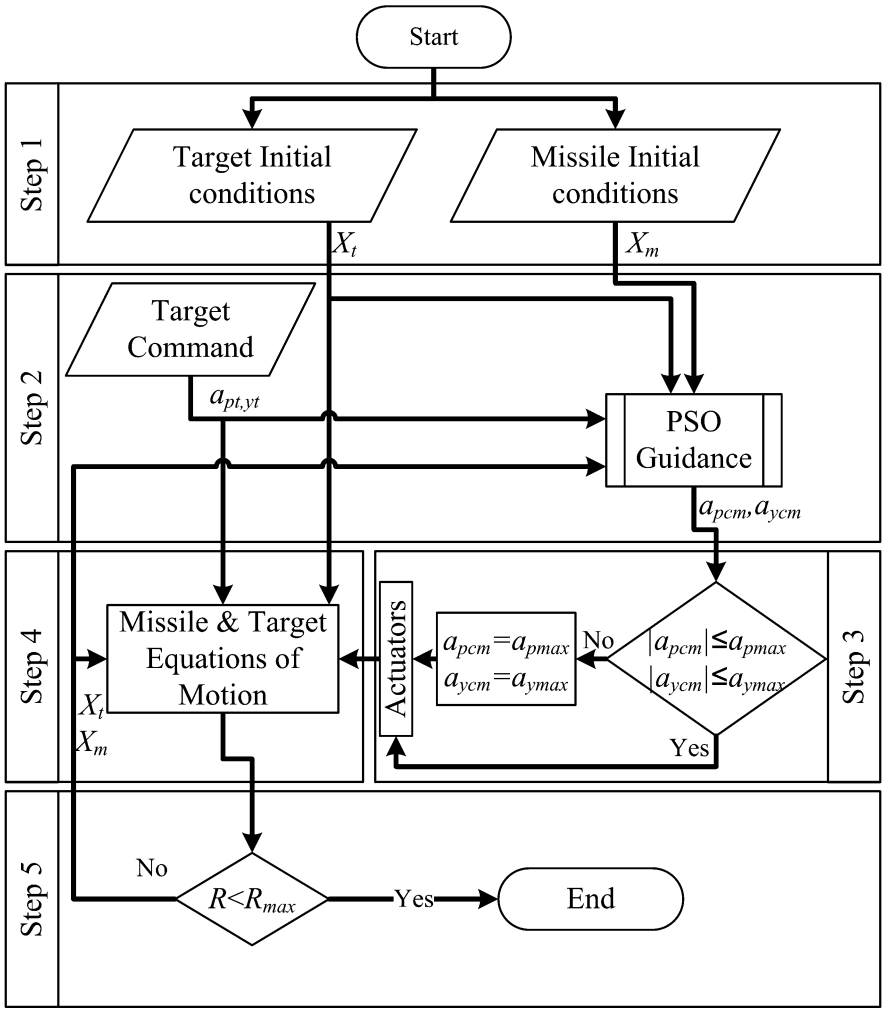

Fig. 2. The pursuit-evasion flow chart for validating PSO guidance algorithm.

Step 1: Choose the initial conditions of missile and target, X_m and X_t, respectively. The initial conditions include the x-, y-, z-coordinates, the velocities, the flight path angles and the azimuth angles.

Step 2: Firstly, the target's commands a_{pct} and a_{yct} are determined by target's pilot. The output acceleration values of a_{pct} and a_{yct} through actuators are expressed in Eqs. (4) and (5). After obtaining the target's acceleration commands (a_{pt}, a_{yt}), we incorporate the information with X_m and X_t in the evaluating process of PSO guidance algorithm. With reference to the PSO guidance algorithm shown in Fig. 1, the PSO guidance commands a_{pcm} and a_{ycm} are then calculated according to the algorithm.

Step 3: Here, the determined PSO guidance commands a_{pcm} and a_{ycm} must be matched within an acceptable bound $a_{p\max}$ and $a_{y\max}$; otherwise, the commands should maintain the saturation values $a_{p\max}$ and $a_{y\max}$.

Step 4: Substitute the acceleration commands of missile and target into the equations of motion in Eqs. (3)–(11) to obtain the new states X_m and X_t through integration.

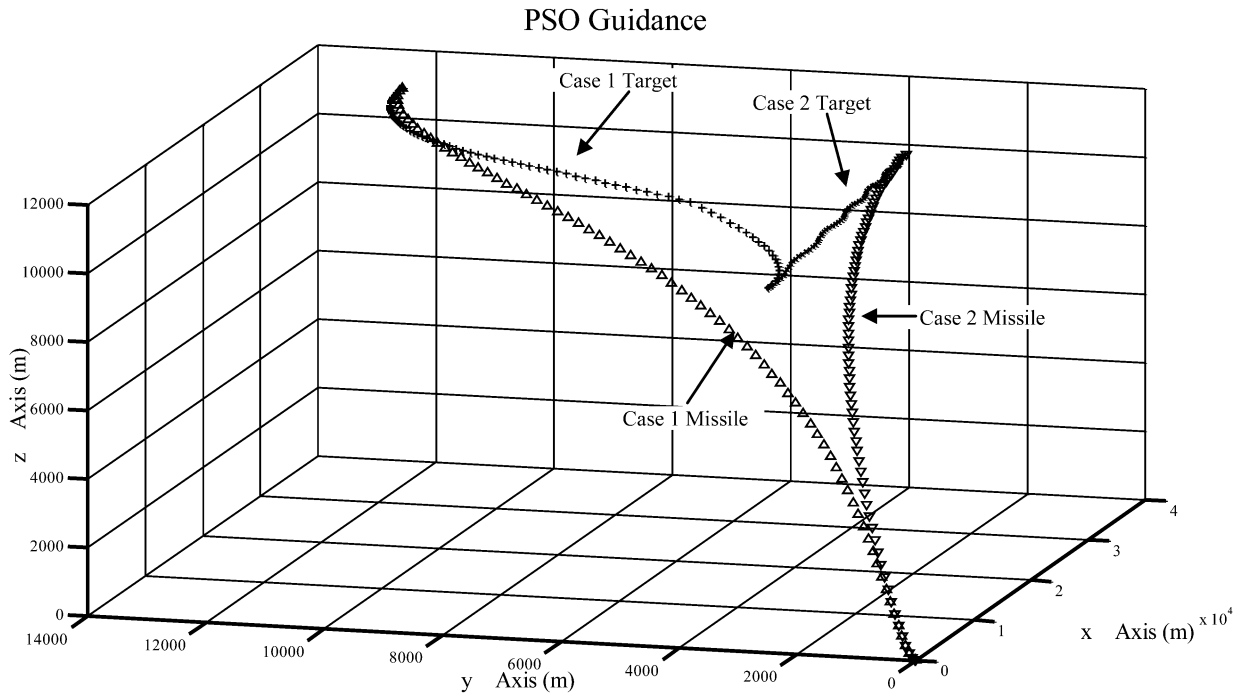

Fig. 3. The simulation results of the two pursuit-evasion scenarios.

Step 5: Compute the missile-to-target range R. If R is greater than the required kill radius $R_{\max}$, then feed the new states X_m and X_t back to step 2 until termination criterion is met.

Compared with the existing AI guidance laws, e.g. NN guidance [7], fuzzy guidance [8], the proposed method has a simple architecture to implement and excellent capability to search global optimal solution within a short calculation time. More specifically, there are only four kinds of parameters, including inertial weight factor, self-confidence factor, swarm-confidence factor and population size need to select.

4. SIMULATION RESULTS

Implementation of this proposed guidance algorithm has been completed. This section discusses the details of the implementations and compares their results. To evaluate our proposed algorithm, the target took two kinds of evasion strategies as follows:

- Case 1: The target curved to the left at first, and then changed to fly straight and upwardly. At last, the target turned right along a curved path (see Fig. 3, case 1 target).
- Case 2: The target evaded along the S-curve (see Fig. 3, case 2 target).

The PSO guidance algorithm was implemented to use 100 particles and run for maximum 500 generations. The inertial weight factor lower boundary w_0 was set to 0.5 and the upper boundary w_1 was set to 1.5. The self-confidence factor c_1 and the swarm-confidence factor c_2 were set to 1. The convergent tolerance was set to 10^{-3}.

The particles and velocities were initialized randomly within the guidance command solution space. In the simulations, we used the Euclidean distance to be evaluating function. The kill radius was set to 15 m and referred to the AIM-7 Sparrow Missile. If the relative distance of missile and target were less than 15 m, it would be an effective attack. If the missile missed the target, the simulation would stop at the 1000th

Table 1. The initial condition of missile and target.

Item	X axis [m]	Y axis [m]	Z axis [m]	Initial speed [m/sec]	Mass [kg]	Integral step [sec]
Missile	0	0	0	1	1135	0.1
Target	5000	3000	10000	779	20626	0.1

Table 2. The data of simulation results.

Performance Index			Scenario 1	Scenario 2
PSO Iteration Execution time(sec)			0.2	0.2
Miss distance(m)			10.4	6.6
Time of interception(sec)			74.9	75.7
Pitch acceleration (m/sec^2)		Maximum	35	35
		Average	7.0	6.6
Yaw acceleration (m/sec^2)		Maximum	35	34.5
		Average	5.3	3.6
Final stage (1000 m~15 m) turning rate (rad/sec)	$\dot{g}$	Maximum	0.027	0.007
		Average	0.009	0.005
	$\dot{Y}$	Maximum	0.062	0.059
		Average	0.052	0.029

iterations. Two pursuit-evasion scenarios are illustrated in Fig. 3. The escaped target case 1 is shown in plus trajectory and in block trajectory for case 2. The trajectories of missile are plotted by triangle symbol. The initial conditions are shown in Table 1.

Table 2 collates the data of simulations of the two scenarios. The two missile distances, 10.4 m and 6.6 m ($<$ 15 m), demonstrate the validity of the proposed guidance. On the other hand, in the runs where the implementations were able to converge on the global optimum within the short-range pursuit period, it took PSO algorithm about 0.2 sec iteration execution time on average to converge. In the two pursuit-evasion scenarios, the PSO guidance law takes satisfactory time to track (74.9 and 75.7 sec). It should be pointed out that S-curve is highly maneuverable of disengaging from missile tracking, so it widens the time of interception. These values show that PSOG algorithm has higher tracking ability (less tracking time) and accuracy (less miss distance). The maximum accelerations, 35 g, for the two scenarios are less than the limitation, 40 g and the averages of acceleration are accepted. Moreover, the missile with PSOG needs less pitch and yaw accelerations which can be observed by the final stage turning rate ($<$ 0.07 rad/sec). It is considered to be practical as PSOG is used for intercepting with the requirement of ending the attack in a tail chase. To this end, it is clear that the PSO guidance algorithm exhibits the robust pursuit capability to different escape strategies. The PSOG results displayed satisfied performance in execution time, miss distance, time of interception, and robust pursuit capability.

5. CONCLUSION

In this work, we presented a new PSO guidance algorithm and demonstrated the simulation results. The particles of the PSO guidance algorithm are initialized with a random distribution within the guidance command solution space. As the algorithm proceeds, these particles will migrate to the local optimum until the global optimum is reached. With the states of missile and target as the input, the PSO guidance algorithm outputs the global optimal guidance commands. From the simulation results, the PSO guidance algorithm exhibited satisfied performance in execution time, miss distance, time of interception, and robust pursuit capability. The proposed method seems to show a lot of promise as a novel population-based stochastic op-

timization search guidance algorithm. Additional target evasion strategies and more complicated objective functions need to be used for further evaluation of this technique. Furthermore, since the parameters of PSO is sensitive to the convergence performance and result, more studies on the effect of the parameters and the optimal tuning are still in progress at present.

REFERENCES

1. Imado, F. and Miwa, S., "Fighter evasive maneuvers against proportional navigation missile", *Journal of Aircraft*, Vol. 23, No. 11, pp. 825–830, 1986.
2. Siouris, G.M., "Comparison between proportional and augmented proportional navigation", *Nachrichten-technische Zeitschrift (NTZ-Communications Journal)*, pp. 278–280, 1974.
3. Yang, C.D., Hsiao, F.B. and Yeh, F.B., "Generalized guidance law for homing missiles", *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 25, No. 2, pp. 197–212, 1989.
4. Bryson, A.E., Jr. and Ho, Y.C., *Applied Optimal Control: Optimization, Estimation and Control*, Taylor & Francis, 1975.
5. Rusnak, I. and Meir, L., "Optimal guidance for acceleration constrained missile and maneuvering target", *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 26, No. 4, pp. 618–624, 1990.
6. Lin, C.L. and Lai, R.M., "Parameter design for a guidance and control system using genetic approach", *Aerospace Science and Technology*, Vol. 5, No. 6, pp. 425–434, 2001.
7. Choi, H.L., Tahk, M.J. and Bang, H., "Neural network guidance based on pursuit-evasion games with enhanced performance", *Control Engineering Practice*, Vol. 14, No. 7, pp. 735–742, 2006.
8. Li, S.Q. and Yuan, L.Y., "Design of fuzzy logic missile guidance law with minimal rule base", in *Proceedings of Sixth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD'09)*, Vol. 6, pp. 176–180, 2009.
9. Omar, H.M. and Abidob, M.A., "Designing integrated guidance law for aerodynamic missiles by hybrid multi-objective evolutionary algorithm and tabu search", *Aerospace Science and Technology*, Vol. 14, No. 5, pp. 356–363, 2010.
10. Kennedy, J. and Eberhart, R., "Particle swarm optimization", in *Proceedings of IEEE International Conference on Neural Networks 1995*, Vol. 4 , pp. 1942–1948, 1995.
11. Kiranyaz, S., Ince, T., Yildirim, A. and Gabbouj, M., "Evolutionary artificial neural networks by multi-dimensional particle swarm optimization", *Neural Networks*, Vol. 22, No. 10, pp. 1448–1462, 2009.
12. Chien, H.F., Lin, J.H. and Chao, Y.N., *Missile Flight Mechanics*, Beijing Institute of Technology Press, 2000.
13. Sang, D., Min, B.M. and Tahk, M.J., "Impact angle control guidance law using Lyapunov function and PSO method", in *Proceedings of SICE, 2007 Annual Conference*, pp. 2253–2257, 2007.
14. Kung, C.C., Chiang, F.L. and Chen, K.Y., "Design a three-dimensional pursuit guidance law with feedback linearization method", *World Academy of Science, Engineering and Technology*, No. 55, pp. 136–141, 2011.
15. Isaacs, R., *Differential Games: A Mathematical Theory with Applications to Warfare and Pursuit, Control and Optimization*, Dover, 1999.
16. Lin, C.M., Hsu, C.F., Chang, S.K., and Wai, R.J., "Guidance law evaluation for missile guidance systems", *Asian Journal of Control*, Vol. 2, No. 4, pp. 243–250, 2000.