Available online at www.sciencedirect.com

ScienceDirect

Journal of the Franklin Institute 357 (2020) 11042–11071

www.elsevier.com/locate/jfranklin

A nonlinear robust model predictive differential game guidance algorithm based on the particle swarm optimization

Hadi Nobahari*, Saeed Nasrollahi

Department of Aerospace Engineering, Sharif University of Technology, 1458889694, Tehran, Iran

Received 28 December 2019; received in revised form 30 June 2020; accepted 22 August 2020
Available online 1 September 2020

Abstract

Two-dimensional engagement of a pursuer and a maneuvering target, affected by matched uncertainties, is formulated as a nonlinear differential game. The uncertain guidance problem is converted into a nonlinear model predictive control problem by introducing an appropriate cost function. The objective is to calculate the best guidance commands of the pursuer and the worst possible target maneuvers simultaneously, over a receding horizon. The proposed cost function penalizes the line-of-sight rate, the pursuer acceleration, and the uncertainties. It also rewards the target maneuver. A particle swarm-based dynamic optimization algorithm is developed to solve the nonlinear model predictive differential game, affected by the uncertainties. Performance of the proposed guidance algorithm is evaluated against maneuvering and non-maneuvering targets. The algorithm is also evaluated for the cases when the pursuer has a high initial heading error, and the guidance command is constrained. The statistical performance of the proposed algorithm is evaluated using Monte Carlo simulation. Moreover, a processor in the loop experiment is performed to verify the implementation capability of the proposed algorithm. Finally, a comparison is made between the performance of the suggested algorithm with some other methods including linear- quadratic differential game, state-dependent Riccati equation-differential game, a guidance law on the basis of adaptive dynamic programming, a proportional navigation guidance improved by particle swarm optimization, a guidance algorithm based on the continuous ant colony controller, switched bias proportional navigation, and augmented proportional navigation.

* Corresponding author.
E-mail address: nobahari@sharif.edu (H. Nobahari).

https://doi.org/10.1016/j.jfranklin.2020.08.032

1. Introduction

A terminal guidance algorithm must provide fast response and high accuracy [1]. Variants of Proportional Navigation (PN) [2–4] have been used in many practical problems due to their simplicity and ease of implementation as well as their effectiveness in intercepting non-maneuvering and maneuvering targets [5–8].

Some researchers have used optimal control approaches to develop guidance laws for linearized kinematics. An optimal guidance law was introduced in [9] for a fixed or low-speed target. Another optimal guidance law was proposed in [10] for a pursuer with a small initial heading error against a low maneuvering target. Here, an estimator is used to estimate the states. Using the optimal control theory, a generalized explicit guidance algorithm was proposed in [11], to minimize the Miss Distance (MD) and the final pursuer-target relative orientation. In [12], a homing guidance law on the basis of the classical State-Dependent Riccati Equation (SDRE) was proposed for the maneuvering targets. Also, an easy-to-check equivalence condition on the solvability of SDRE was provided that reduces the online computational load. Optimal guidance laws depend on the estimated time-to-go. A sensitivity analysis has shown that the error of the estimated time-to-go deteriorates the guidance performance [13].

In the development of the guidance laws, various nonlinear control approaches have also been used, such as neural network [14], Model Predictive Control (MPC) [15,16], backstepping control [17], Sliding Mode Control (SMC) [18]. SMC is a widely used robust control scheme in the nonlinear systems containing uncertainties. However, it should be modified to be used as a guidance law, and the chattering problem should be solved. Chattering may excite high-frequency modes of the system, neglected in the modeling phase. To reduce the chattering, it is common to use a saturation function instead of the theoretically used sign function [19]; however, this approach decreases the precision of the guidance law. Moreover, the chattering of a sliding mode guidance law is partly because it considers the target maneuver as uncertainty and calculates the switching term of the guidance command based on the bounds of the target maneuver. While, a differential game guidance law solves a two-sided optimal control problem; i.e. it finds the optimal guidance command of the pursuer and the optimal target maneuver, simultaneously.

A linear pursuit-evasion Differential Game (DG) was first proposed in [20]. In [21], a Linear Quadratic DG (LQDG) guidance law was derived where the pursuer and the target trajectories are linearized along the initial Line of Sight (LOS). In [22], an optimal guidance law was derived for the end-game phase of a pursuer with dual controls. In [23] and [24], LQDG was used to provide a terminal intercept angle. A linear guidance law on the basis of the DG that considers the autopilot lag was derived in [25] and [26]. The State-Dependent Riccati Equation (SDRE) method was used in [13], to obtain a nonlinear DG guidance law for a maneuvering target. This guidance law does not depend on the time-to-go; however, it does not consider matched uncertainties. A robust DG guidance law that uses the Adaptive Dynamic Programming (ADP) was proposed in [27], for two-dimensional engagement in the presence of matched uncertainties. The performance of the ADP was evaluated against a target performing a step maneuver, while the initial heading error is small, and the pursuer has a constant velocity. A constrained ADP was used in [28] to propose a DG guidance law. Again, matched uncertainties have not been considered, and the guidance law depends on the time-to-go. A DG guidance law on the basis of the nonlinear MPC was proposed in [29], in which a gradient-descent method was used for optimization. Time-to-go and relative

distance were taken as two different cost functions; additionally, matched uncertainties were not considered.

None of the references, mentioned above, has used heuristic approaches to calculate the guidance commands. In [30], the Particle Swarm Optimization (PSO) has been utilized to design a terminal guidance law, where the objective function is the relative distance. A PSO-based guidance law has been proposed in [31], where the objective function is the LOS rate. In [32], a two-phase guidance law has been proposed, in which PN and improved PSO guidance [31] is used in the first and second phases, respectively. In [16,33,34], an integrated estimation and control algorithm on the basis of ant colony optimization has been proposed. The proposed algorithm has been used in [16,33] to simultaneously estimate the states and calculate the optimal guidance commands to nullify the LOS rate. It should be noted that none of the guidance algorithms, in the above-mentioned references, are based on the DG. Also, they do not consider uncertainties. Therefore, the literature motivates this paper to provide a heuristic robust model predictive differential game guidance algorithm.

The contributions of the work can be summarized as follows: two-dimensional engagement of a pursuer with a maneuvering target, affected by matched uncertainties, is formulated as a nonlinear DG. The uncertain guidance problem is converted into a nonlinear MPC problem by introducing a new cost function that reflects the uncertainty. The best guidance commands of the pursuer and the worst possible target maneuvers are calculated simultaneously. The proposed cost function penalizes the LOS rate, the pursuer acceleration, and the uncertainties. It also rewards the target maneuver. A particle swarm-based dynamic optimization algorithm is developed to solve the nonlinear model predictive differential game, affected by the uncertainties. A heuristic robust model predictive differential game guidance algorithm has not been given yet. The performance of the new guidance algorithm is evaluated via various numerical simulations and compared with traditional and state-of-the-art guidance laws. Moreover, a Processor in the Loop (PIL) experiment is performed, where the guidance algorithm is running on an embedded target platform, and the pursuer-target engagement is simulated in real-time. A Monte Carlo simulation is also performed to evaluate the statistical performance of the guidance algorithm.

The following sections are structured as follows: mathematical foundations of the problem are given in Section 2. The differential game problem is introduced in Section 3. The guidance problem is defined in Section 4. The guidance algorithm is described in Section 5. Numerical simulations and discussion are presented in Section 6. Finally, Section 7 is dedicated to the conclusions.

2. Mathematical foundations of the problem

Consider the following nonlinear model:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) + \mathbf{g}(\mathbf{x}(t))\mathbf{u}(t) + \mathbf{g}(\mathbf{x}(t))\mathbf{a}(\mathbf{x}(t)) \tag{1}$$

Here, $\mathbf{x}(t) \in \Re^n$ and $\mathbf{u}(t) \in \Re^m$ are the state vector and control vector, respectively. $\mathbf{f}$ and $\mathbf{g}$ are nonlinear functions and $\mathbf{a}(\mathbf{x}(t))$ models the uncertainties, matched with the control signal $\mathbf{u}(t)$, that is, it enters the system through the same channel as the control input. This type of uncertainty is called matched uncertainty. It is assumed that $\mathbf{f}(0) = 0$, and $\mathbf{a}(0) = 0$. Therefore, $\mathbf{x} = 0$ is an equilibrium point for $\mathbf{u} = \mathbf{0}$. Also, matched uncertainty is bounded as $\|\mathbf{a}(\mathbf{x}(t))\| \leq \mathbf{a}_{\max}(\mathbf{x}(t))$, where $\mathbf{a}_{\max}(\mathbf{x}(t))$ is a non-negative function.

The robust control problem is defined as finding $\mathbf{u}(t)$ such that the nonlinear model, described in Eq. (1), is asymptotically stable given the uncertainties.

The optimal control problem is defined as finding $\mathbf{u}(t)$ for the nominal model

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) + \mathbf{g}(\mathbf{x}(t))\mathbf{u}(t) \tag{2}$$

that minimizes the following objective function:

$$\int_0^{\infty} \left(\mathbf{x}^{\mathrm{T}}(t)\mathbf{Q}\mathbf{x}(t) + \mathbf{u}^{\mathrm{T}}(t)\mathbf{R}\mathbf{u}(t) + \mathbf{a}_{\max}^{\mathrm{T}}(\mathbf{x}(t))\mathbf{a}_{\max}(\mathbf{x}(t))\right)dt \tag{3}$$

where, $\mathbf{Q}$ and $\mathbf{R}$ are the state and control weighting matrices, respectively. The relation between these two control problems is explained in the subsequent proposition.

Proposition 1. *The solution of the optimal control problem (if exists), will be the solution of the robust control problem [35]. The proof of proposition 1 has been given in Appendix A.*

3. Differential game problem

The mathematical model of an uncertain nonlinear two-player zero-sum DG is given by:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) + \mathbf{g}(\mathbf{x}(t))(\mathbf{u}(t) + \mathbf{a}(\mathbf{x}(t))) + \mathbf{h}(\mathbf{x}(t))(\mathbf{v}(t) + \mathbf{b}(\mathbf{x}(t))) \tag{4}$$

where $\mathbf{x}(t) \in \Re^{\mathrm{n}}$ is the state vector of the problem, $\mathbf{u}(t) \in \Re^{\mathrm{m}}$ and $\mathbf{v}(t) \in \Re^{\mathrm{k}}$ are respectively the control vectors of the first and the second players; $\mathbf{f}$, $\mathbf{g}$, and $\mathbf{h}$ are nonlinear functions and $\mathbf{a}(\mathbf{x}(t)) = [\ a_1(\mathbf{x}(t)) \ \cdots \ a_{\mathrm{m}}(\mathbf{x}(t))\ ]^{\mathrm{T}} \in \Re^{\mathrm{m}}$ and $\mathbf{b}(\mathbf{x}(t)) = [\ b_1(\mathbf{x}(t)) \ \cdots \ b_{\mathrm{k}}(\mathbf{x}(t))\ ]^{\mathrm{T}} \in \Re^{\mathrm{k}}$ are unknown nonlinear matched uncertainties of the players. Matched uncertainties are both bounded by the known vectors $\mathbf{a}_{\mathbf{M}}(\mathbf{x}(t)) = [\ a_{1,\mathrm{M}}(\mathbf{x}(t)) \ \cdots \ a_{\mathrm{m,M}}(\mathbf{x}(t))\ ]^{\mathrm{T}}$ and $\mathbf{b}_{\mathbf{M}}(\mathbf{x}(t)) = [\ b_{1,\mathrm{M}}(\mathbf{x}(t)) \ \cdots \ b_{\mathrm{k,M}}(\mathbf{x}(t))\ ]^{\mathrm{T}}$, i.e., $\|a_i(\mathbf{x}(t))\| \le a_{i,\mathrm{M}}(\mathbf{x}(t))$ for $i = 1, .., \mathrm{m}$ and $\|b_j(\mathbf{x}(t))\| \le b_{j,\mathrm{M}}(\mathbf{x}(t))$ for $j = 1, .., \mathrm{k}$. Here, our approach is to convert the robust control problem into an optimal control problem, based on Proposition 1, and to solve it numerically on a finite horizon. For this purpose, a nominal system without uncertainty is described by

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) + \mathbf{g}(\mathbf{x}(t))\mathbf{u}(t) + \mathbf{h}(\mathbf{x}(t))\mathbf{v}(t) \tag{5}$$

The first player selects $\mathbf{u}(t)$, to regulate certain states. Instead, the second player selects $\mathbf{v}(t)$, to deviate those states from zero. These two different objectives can be merged into one objective function as follows:

$$J = \int_{t_0}^{t_0+\mathrm{T_p}} \left(\mathbf{x}^{\mathrm{T}}(t)\mathbf{Q}\mathbf{x}(t) + \mathbf{u}^{\mathrm{T}}(t)\mathbf{R}_1\mathbf{u}(t) - \mathbf{v}^{\mathrm{T}}(t)\mathbf{R}_2\mathbf{v}(t) + Q_1\right)dt \tag{6}$$

This function minimizes the tracking error and the control effort of the first player and maximizes the control effort of the second player, simultaneously.

Also, $Q_1 = \mathbf{a}_{\mathrm{M}}^{\mathrm{T}}(\mathbf{x})\mathbf{F}_{\mathrm{a}}\mathbf{a}_{\mathrm{M}}(\mathbf{x}) + \mathbf{b}_{\mathbf{M}}^{\mathbf{T}}(\mathbf{x})\mathbf{F}_{\mathrm{b}}\mathbf{b}_{\mathrm{M}}(\mathbf{x})$ reflects the uncertainties. Moreover, the positive semi-definite matrix $\mathbf{Q}$ is the state weighting matrix. The positive definite matrices $\mathbf{R}_1$ and $\mathbf{R}_2$ are the control weighting matrices. Also, $\mathrm{T_P}$ is the prediction horizon; $\mathbf{F}_{\mathrm{a}}$ and $\mathbf{F}_{\mathrm{b}}$ are weighting matrices.

Definition 1. The optimal strategy of the two players; i.e. $\mathbf{u}^*(t)$ and $\mathbf{v}^*(t)$, is the saddle point of the problem if the following relation is held for any other $\mathbf{u}(t)$ and $\mathbf{v}(t)$ [13]:

$$J(\mathbf{u}^*(t), \mathbf{v}(t)) \le J(\mathbf{u}^*(t), \mathbf{v}^*(t)) \le J(\mathbf{u}(t), \mathbf{v}^*(t)) \tag{7}$$

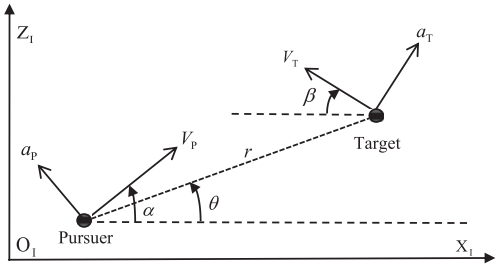


Fig. 1. Two-dimensional engagement of the players.

4. Mathematical modeling of the guidance problem

Two-dimensional engagement of the players is shown in Fig. 1. Here, $X_I - O_I - Z_I$ represents the inertial reference frame, r denotes the relative distance and θ represents the LOS angle. Flight path angle, lateral acceleration and velocity are shown by α, a_P and V_P for the pursuer and β, a_T and V_T for the target.

The gravitational acceleration can be neglected in the terminal phase compared to the pursuer acceleration [36]; therefore, the kinematic equations stating the engagement of the pursuer and the target are written as follows [13,27]:

$$\dot{r} = -V_T \cos(\theta + \beta) - V_P \cos(\alpha - \theta) \tag{8}$$

$$\dot{\theta} = \frac{V_T \sin(\theta + \beta) - V_P \sin(\alpha - \theta)}{r} \tag{9}$$

$$\dot{\alpha} = \frac{a_P}{V_P} \tag{10}$$

$$\dot{\beta} = \frac{a_T}{V_T} \tag{11}$$

A first-order autopilot is considered for the pursuer. Therefore,

$$\dot{a}_P = -\frac{1}{\tau_p} a_P + \frac{1}{\tau_p} u \tag{12}$$

where τ_p and $u$are the time constant and the input of the pursuer autopilot, respectively. The time derivative of Eqs. (8) and (9) can be stated as [13]

$$\ddot{r} = r\dot{\theta}^2 + a_P \sin(\alpha - \theta) + a_T \sin(\theta + \beta) \tag{13}$$

$$\ddot{\theta} = \frac{1}{r}\left(-2\dot{r}\dot{\theta} - a_P \cos(\alpha - \theta) + a_T \cos(\theta + \beta)\right) \tag{14}$$

In the following section, the lateral acceleration command of the target is indicated by v; components of the state vector are defined as $x_1 = r$, $x_2 = \dot{r}$, $x_3 = \dot{\theta}$, $x_4 = a_P$, $x_5 = \alpha$, $x_6 = \beta$, and $x_7 = \theta$; and Euler method, with sampling time Δt, is used to discretize the differential equations.

Definition 2. The Zero Effort Miss (ZEM) is defined as the minimum distance experienced between the players if both of them do not maneuver from the current time t onwards. ZEM can be calculated as

$$r_{\text{miss}}(t) = \frac{r^2\dot{\theta}}{\sqrt{\dot{r}^2 + (r\dot{\theta})^2}} \tag{15}$$

Therefore, regulating the LOS rate implies regulation of $r_{\text{miss}}(t)$. In the following, the proposed DG guidance algorithm is introduced. For this purpose, the discrete-time state-space model is used to formulate the algorithm.

5. Differential game guidance algorithm based on the PSO

The guidance laws based on the DG, do not require the value of target maneuver. Here, a DG guidance problem is formulated as a nonlinear dynamic optimization problem, expressed by Eqs. (5) and (6). The aim is to develop a dynamic optimization algorithm on the basis of PSO [37] to find, in an online manner, the best guidance commands of the pursuer and the worst possible maneuvers of the target, over a finite horizon.

There are many global optimization algorithms, while PSO is straightforward, and it has few parameters. Moreover, as discussed later, the convergence of the PSO can be guaranteed in certain conditions.

The proposed heuristic guidance algorithm is named as PSO-Differential Game Guidance (PSO-DGG). Fig. 2 shows the flowchart of the PSO-DGG algorithm. PSO-DGG has two loops. The first loop iterates until the end time of the game, and the second loop iterates to find the instantaneous guidance commands. At first, the parameters are set and the initial guidance commands are generated, randomly. Then, the future guidance commands are generated in a finite horizon. Next, the future states are predicted using the guidance commands. Then, the cost function is calculated for each particle. Afterward, the local and global best experiences of the particles, along with their velocity and position, are updated. The inner loop is terminated after a predetermined number of iterations, and the guidance commands are calculated utilizing a mean operator. The pseudo-code of the PSO-DGG is shown in Algorithm 1. In the following sections, more details are provided.

5.1. Initialization

PSO-DGG has some parameters, set at the beginning. These parameters are defined in the following sections. Moreover, the initial guidance commands are guessed by each particle. These initial values are guessed using a random generator with uniform probability distribution within the acceptable range of the guidance commands.

5.2. Generation of the future guidance commands

The future guidance commands are generated for particle j as

$$\mathbf{u}_j^l(k+i-1) = \mathbf{u}_j^l(k+i-2) + \boldsymbol{\upsilon}_{\mathrm{P}_j}^l(k+i-2),\ (i = 1, \ldots, \mathrm{T_p}) \tag{16}$$

$$\mathbf{v}_j^l(k+i-1) = \mathbf{v}_j^l(k+i-2) + \boldsymbol{\upsilon}_{\mathrm{T}_j}^l(k+i-2),\ (i = 1, \ldots, \mathrm{T_p}) \tag{17}$$

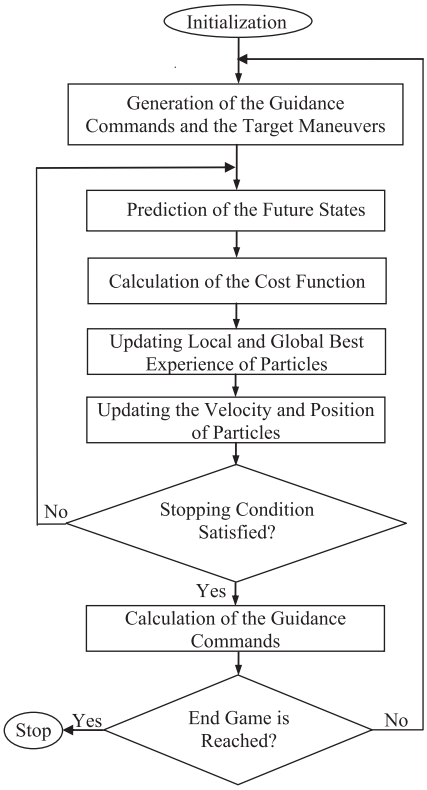


Fig. 2. Flowchart of the PSO-DGG algorithm.

Where, $\boldsymbol{\upsilon}_{\mathrm{P}}$ and $\boldsymbol{\upsilon}_{\mathrm{T}}$ are vectors of zero-mean Gaussian random variables, the standard deviation of which are $\sigma_{\upsilon_{\mathrm{P}}}$ and $\sigma_{\upsilon_{\mathrm{T}}}$, respectively; also, k and l denote the time step and the iteration number, respectively. It should be noted that $\mathbf{u}_j^1(k-1) = \hat{\mathbf{u}}(k-1)$ and $\mathbf{v}_j^1(k-1) = \hat{\mathbf{v}}(k-1)$.

5.3. Prediction of the future states

Using the model presented in Eq. (5), the future of particles are predicted as

$$\begin{aligned}\mathbf{x}_j^l(k+i) = {} & \mathbf{f}(\mathbf{x}_j^l(k+i-1)) + \mathbf{g}(\mathbf{x}_j^l(k+i-1))\mathbf{u}_l^j(k+i-1) \\ & + \mathbf{h}(\mathbf{x}_j^l(k+i-1))\mathbf{v}_j^l(k+i-1) \qquad (i = 1, \ldots, \mathrm{T_p})\end{aligned} \tag{18}$$

5.4. Calculation of the cost function

At iteration l and time step k, the cost of the j-th particle, $J_j^l(k)$, is calculated using Eq. (6) as

$$J_j^l = J_{1_j}^l + J_{2_j}^l + J_{3_j}^l - J_{4_j}^l \tag{19}$$

Algorithm 1
PSO-DGG Pseudo-Code.

Set the number of particles, number of iterations, prediction horizon, and weighting factors.
Initialize randomly the position of particles, $\mathbf{X}_j^0(k)$, for $j \in [1, N]$.
Initialize the velocity and the local and global best experience of the particles.
While (the game is not terminated)
Generate future guidance commands by Eqs. (16), and (17).
for $l = 1$ to $\mathrm{l}_{\max}$ **do**
for $j = 1$ to N **do**
Predict the future states by Eq. (18).
Calculate the cost function by Eqs. (19)–(23).
Update $\mathbf{P}_{j,best}^l(k)$ and $\mathbf{G}_{\mathrm{best}}^l(k)$.
Update the velocity and position of particles by Eqs. (24) and (25)
end for
end for
Calculate the guidance command by Eq. (27).
end while

$$J_{1_j}^l = \frac{\sum_{i=k+1}^{k+\mathrm{T_p}} \left[\mathbf{x}_{\mathrm{d}}(i) - \mathbf{x}_j^l(i)\right]^{\mathrm{T}} \mathbf{Q} \left[\mathbf{x}_{\mathrm{d}}(i) - \mathbf{x}_j^l(i)\right]}{\max_{j=1}^{\mathrm{N}} \sum_{i=k+1}^{k+\mathrm{T_p}} \left[\mathbf{x}_{\mathrm{d}}(i) - \mathbf{x}_j^l(i)\right]^{\mathrm{T}} \mathbf{Q} \left[\mathbf{x}_{\mathrm{d}}(i) - \mathbf{x}_j^l(i)\right]} \tag{20}$$

$$J_{2_j}^l = \frac{\sum_{i=k}^{k+\mathrm{T_p}-1} \mathbf{u}_j^l(i)^{\mathrm{T}} \mathbf{R}_1 \mathbf{u}_j^l(i)}{\max_{j=1}^{\mathrm{N}} \sum_{i=k}^{k+\mathrm{T_p}-1} \mathbf{u}_j^l(i)^{\mathrm{T}} \mathbf{R}_1 \mathbf{u}_j^l(i)} \tag{21}$$

$$J_{3_j}^l = \frac{\sum_{i=k+1}^{k+\mathrm{T_p}} \mathbf{a}_{\mathrm{M}_j}^l(i)^{\mathrm{T}} \mathbf{F}_{\mathrm{a}} \mathbf{a}_{\mathrm{M}_j}^l(i) + \mathbf{b}_{\mathrm{M}_j}^l(i)^{\mathrm{T}} \mathbf{F}_{\mathrm{b}} \mathbf{b}_{\mathrm{M}_j}^l(i)}{\max_{j=1}^{\mathrm{N}} \sum_{i=k+1}^{k+\mathrm{T_p}} \mathbf{a}_{\mathrm{M}_j}^l(i)^{\mathrm{T}} \mathbf{F}_{\mathrm{a}} \mathbf{a}_{\mathrm{M}_j}^l(i) + \mathbf{b}_{\mathrm{M}_j}^l(i)^{\mathrm{T}} \mathbf{F}_{\mathrm{b}} \mathbf{b}_{\mathrm{M}_j}^l(i)} \tag{22}$$

$$J_{4_j}^l = \frac{\sum_{i=k}^{k+\mathrm{T_p}-1} \mathbf{v}_j^l(i)^{\mathrm{T}} \mathbf{R}_2 \mathbf{v}_j^l(i)}{\max_{j=1}^{\mathrm{N}} \sum_{i=k}^{k+\mathrm{T_p}-1} \mathbf{v}_j^l(i)^{\mathrm{T}} \mathbf{R}_2 \mathbf{v}_j^l(i)} \tag{23}$$

In the above equations, $J_{1_j}^l$ penalizes the states tracking error, $J_{2_j}^l$ penalizes the guidance commands of the pursuer, $J_{3_j}^l$ reflects the uncertainties, and $J_{4_j}^l$ penalizes the target maneuver. Also, N is the number of particles.

5.5. Updating local and global best experiences

The local best experience of a particle, $\mathbf{P}_{j,best}^l(k)$, is the minimum cost it has experienced, so far. Moreover, the global best, $\mathbf{G}_{\mathrm{best}}^l(k)$, is the best value, experienced by all particles, so far.

5.6. Updating the velocity and position of particles

The velocity and position of particles are updated as

$$\mathbf{V}_j^{l+1}(k) = \mathrm{w}\mathbf{V}_j^l(k) + \mathrm{c}_1 r_1\left(\mathbf{P}_{j,best}^l(k) - \mathbf{X}_j^l(k)\right) + \mathrm{c}_2 r_2\left(\mathbf{G}_{\mathrm{best}}^l(k) - \mathbf{X}_j^l(k)\right) \tag{24}$$

$$\mathbf{X}_j^{l+1}(k) = \mathbf{X}_j^l(k) + \mathbf{V}_j^{l+1}(k) \tag{25}$$

where $\mathbf{V}_j^l(k)$ represents the velocity of the particle, w denotes the inertia weight, c_1 and c_2 are constants, r_1,$r_2 \in [0, 1]$ are uniform random variables, and $\mathbf{X}_j^l(k)$ is the position of particle j, defined as

$$\mathbf{X}_j^l(k) \equiv \begin{bmatrix} \mathbf{u}_j^l(k) \\ \mathbf{u}_j^l(k+1) \\ \vdots \\ \mathbf{u}_j^l(k+\mathrm{T_p}-1) \\ \mathbf{v}_j^l(k) \\ \mathbf{v}_j^l(k+1) \\ \vdots \\ \mathbf{v}_j^l(k+\mathrm{T_p}-1) \end{bmatrix} \tag{26}$$

5.7. Stopping conditions

The inner loop of the PSO-DGG stops after $\mathrm{l_{max}}$ iterations. The outer loop continues until the end of the game.

5.8. Calculation of the guidance commands

When the inner loop is terminated, particles are ranked according to their corresponding costs. Then, the guidance commands are calculated using the average position of $\mathrm{N_t}$ top particles as

$$\hat{\mathbf{X}}(k) = \frac{1}{\mathrm{N_t}} \sum_{j \in \aleph_\mathrm{t}} \hat{\mathbf{X}}_j^{l_{\max}}(k) \tag{27}$$

Where $\aleph_\mathrm{t}$ is the set of top particles, and $\mathrm{N_t} = |\aleph_\mathrm{t}| < \mathrm{N}$. The guidance commands of the top particles (elements 1, …, $\mathrm{T_p}$ of $\hat{\mathbf{X}}_j^{l_{\max}}$, $j \in \aleph_\mathrm{t}$) are used to calculate the guidance commands of the pursuer (elements 1, …, $\mathrm{T_p}$ of $\hat{\mathbf{X}}(k)$), the first element of which is applied to the pursuer in the current time. Similarly, the target maneuver of the top particles (elements $\mathrm{T_p}+1$, …, $2\mathrm{T_p}$ of $\hat{\mathbf{X}}_j^{l_{\max}}$, $j \in \aleph_\mathrm{t}$) is used to predict the target maneuver (elements $\mathrm{T_p}+1$, …, $2\mathrm{T_p}$ of $\hat{\mathbf{X}}(k)$).

6. Results and discussion

The simulation results of the PSO-DGG are presented in this section. In the first case, the velocity of the pursuer is constant. In the second case, the velocity may change through the engagement period. Moreover, the implementation capability of the PSO-DGG algorithm is investigated through a PIL experiment. Similar to [13], it is assumed that V_P, θ, and α are

Table 1
Simulation parameters for the engagement scenarios.

Case	Pursuer				Target					Uncertainties			
	x_{P0} (km)	z_{P0} (km)	V_{P0} (m/s)	α_0 (deg)	x_{T0} (km)	z_{T0} (km)	V_{T0} (m/s)	β_0 (deg)	Maneuver (m/s^2)	δ_1	δ_2	δ_3	δ_4
1a and 1 b	0	0	600	70	10	0	400	0	No maneuver	10	25	15	1
1c	0	0	600	70	10	0	400	0	50 (step)	−10	25	15	1
1d	0	0	600	70	10	0	400	0	$50\sin(\frac{\pi}{2}t)$	10	−25	15	1
1e	0	0	600	70	10	0	400	0	Random	10	−25	15	1
2a	0	0	500	70	5	0	400	0	No maneuver	10	25	−15	1
2b	0	0	500	70	5	0	400	0	50 (step)	10	25	15	−1
2c	0	0	500	70	5	0	400	0	$50\sin(\frac{\pi}{2}t)$	−10	−25	15	1

measured, and V_T, $\dot{\theta}$, and β are estimated. Also, the pursuer and the target uncertainties are modeled as in [27]. For this purpose, Eq. (14) is rewritten as [27]

$$\dot{x}_3(t) = \frac{1}{x_1(t)}\begin{pmatrix} -2x_2(t)x_3(t) - (x_4(t) + a(x_3(t)))\cos(x_5(t) - x_7(t)) \\ +(v + b(x_3(t)))\cos(x_6(t) + x_7(t)) \end{pmatrix} \tag{28}$$

Where $a(x_3(t)) = \delta_1 x_3(t)\cos(\delta_2 x_3(t) + \delta_3(x_3(t))^2)$ and $b(x_3(t)) = \delta_4 x_3(t)\sin(x_3(t))$ reflect the matched uncertainty of the pursuer, and the target, respectively. Also, $\delta_1 \in [-10, 10]$, $\delta_2 \in [-25, 25]$, $\delta_3 \in [-15, 15]$, and $\delta_4 \in [-1, 1]$ are unknown parameters [27]. Also, the controllability of the guidance problem with uncertainty has been shown in [27].

The proposed guidance algorithm is tested in both cases against different types of target maneuvers. Data on the simulation scenarios are listed in Table 1. In all cases, $\Delta t = 0.1$ *sec* and the parameters of PSO-DGG are set as follows: $\mathbf{Q} = diag(0, 0, 150, 0, 0, 0, 0)$, $\mathbf{R}_1 = 1$, $\mathbf{R}_2 = 8.5$, $\mathbf{F}_a = 1$, $\mathbf{F}_b = 1$, $T_P = 8$, $\sigma_{v_P} = 0.2$, $\sigma_{v_T} = 0.2$, $w = 0.73$,$c_1 = 1.49$, $c_2 = 1.49$, $N = 35$, $N_t = 10$, and $l = 20$. These parameters are set by trial and error. The weighting matrices, $\mathbf{Q}$, $\mathbf{R}_1$, $\mathbf{R}_2$, $\mathbf{F}_a$, and $\mathbf{F}_b$, are set based on the relative importance of different terms in Eq. (19). The prediction horizon length, T_P, affects the overshoot. The maximum value of σ_{v_P} and σ_{v_T} depends on the acceleration capability of the players. Finally, N_t and l affect the computational cost as well as the performance. The performance of the PSO-DGG algorithm is compared with LQDG [38], the State-Dependent Riccati Equation-Differential Game (SDRE-DG) [13,38], a guidance law on the basis of ADP [27], the Proportional Navigation-Improved PSO Guidance (PN-IPSOG) [32], the Continuous Ant Colony Controller (CACC) [33], the Modified Continuous Ant Colony Controller (MCAC) [16], Switched Bias Proportional Navigation (SBPN) [39], and Augmented Proportional Navigation (APN). The Results are given in the following sections.

6.1. Case1: constant-velocity pursuer

To perform simulations, the pursuer equations of motion are written as

$$\dot{x}_P = V_P\cos(\alpha) \tag{29}$$

$$\dot{z}_P = V_P\sin(\alpha) \tag{30}$$

$$\dot{\alpha} = \frac{a_{\mathrm{P}}}{V_{\mathrm{P}}} \tag{31}$$

$$\dot{a}_{\mathrm{P}} = \frac{u - a_{\mathrm{P}}}{\tau_{\mathrm{P}}} \tag{32}$$

Here, $(x_{\mathrm{P}}, z_{\mathrm{P}})$ denotes the position of the pursuer and τ_{p} is the time constant of the pursuer, defined before. Also, the target equations of motion are stated as

$$\dot{x}_{\mathrm{T}} = -V_{\mathrm{T}} \cos(\beta) \tag{33}$$

$$\dot{z}_{\mathrm{T}} = V_{\mathrm{T}} \sin(\beta) \tag{34}$$

$$\dot{\beta} = \frac{a_{\mathrm{T}}}{V_{\mathrm{T}}} \tag{35}$$

$$\dot{a}_{\mathrm{T}} = \frac{v - a_{\mathrm{T}}}{\tau_{\mathrm{T}}} \tag{36}$$

In the above equations, $(x_{\mathrm{T}}, z_{\mathrm{T}})$ is the position of the target, and τ_{T} is represents the time constant of the target. In this study, $\tau_{\mathrm{P}} = 0.1s$ and $\tau_{\mathrm{T}} = 0.1s$.

6.1.1. Case 1a: non-maneuvering target

In this case, the initial heading error is high. The engagement is depicted in Fig. 3(a). The resultant MD and the interception time are 1.95 m and 10.4 s, respectively. The deviation of the pursuer trajectory from the collision course and the lateral maneuver of the pursuer, observed in Fig. 3(a), comes from the high initial heading error and consequently, the high initial LOS rate. Fig. 3(b)-(d) show the relative velocity, the LOS rate, and the guidance command compared with the pursuer acceleration. As can be seen, the guidance command is reached to zero since the target is non-maneuvering. Moreover, after reaching the collision course, the relative velocity has reached a constant value, and the LOS rate has reached zero.

6.1.2. Case 1b: saturation of the guidance command

In this case, the guidance command is bounded within $[-400, 400]\mathrm{m/s}^2$. The engagement is illustrated in Fig. 4(a). The MD is 3.81 m and the interception time is 11 s. As expected, the MD has been increased in comparison with case 1a. Fig. 4(b)-(d) illustrates the relative velocity, the LOS rate, and the guidance command versus time. Fig. 4(d) shows that the acceleration constraint has been satisfied.

6.1.3. The processor in the loop experiment

In this section, the PSO-DGG algorithm is evaluated through a PIL experiment. PIL is used to show the implementation capability of the algorithm [40]. The schematic diagram of the experiment is illustrated in Fig. 5. Here, the initial condition of the engagement scenario and the parameters of the PSO-DGG algorithm are considered the same as before. To perform the PIL experiment, the “Simulink Real-Time” tool is utilized. Simulink Real-Time [41] enables the real-time execution of the pursuer-target simulation model. Moreover, the "run on hardware" tool compiles C code generating from the PSO-DGG algorithm and programs the hardware device (an Arduino Due module) using the Simulink Coder. Data transfer between

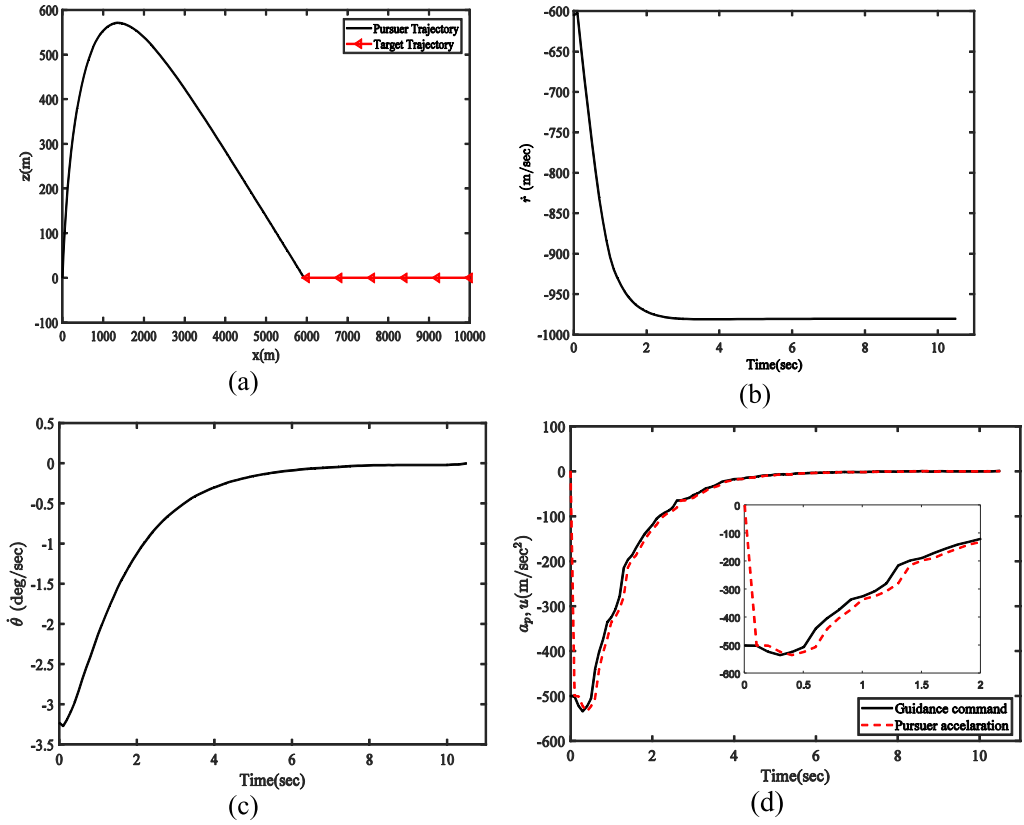


Fig. 3. The engagement scenario against the non-maneuvering target: (a) engagement trajectory; (b) relative velocity; (c) LOS rate; (d) guidance command and pursuer acceleration.

the hardware device and the pursuer-target simulation is performed via a serial link. The hardware device calculates the pursuer acceleration command and sends it back to the computer to simulate the pursuer-target engagement in the next time step. Comparison of the implementation results and the numerical simulation is shown in Fig. 6. These results demonstrate that the PSO-DGG algorithm has been successfully implemented on the processor. There are small differences between the experimental and simulation results. These differences are usually caused by practical problems such as the delay of received packets and the stochastic nature of the noise from one experiment to another.

6.1.4. Case 1c: maneuvering target (Step maneuver)

As listed in Table 1, in this case, the target performs a step maneuver, and the initial heading error is high. The engagement trajectory is depicted in Fig. 7(a). MD and interception time are 2.5 m and 13.4 s, respectively. The closing velocity ($-\dot{r}$) is first increasing and then decreasing until the end game, while it is always positive. The final divergence in Fig. 7(c) is related to the nullification of the relative distance.

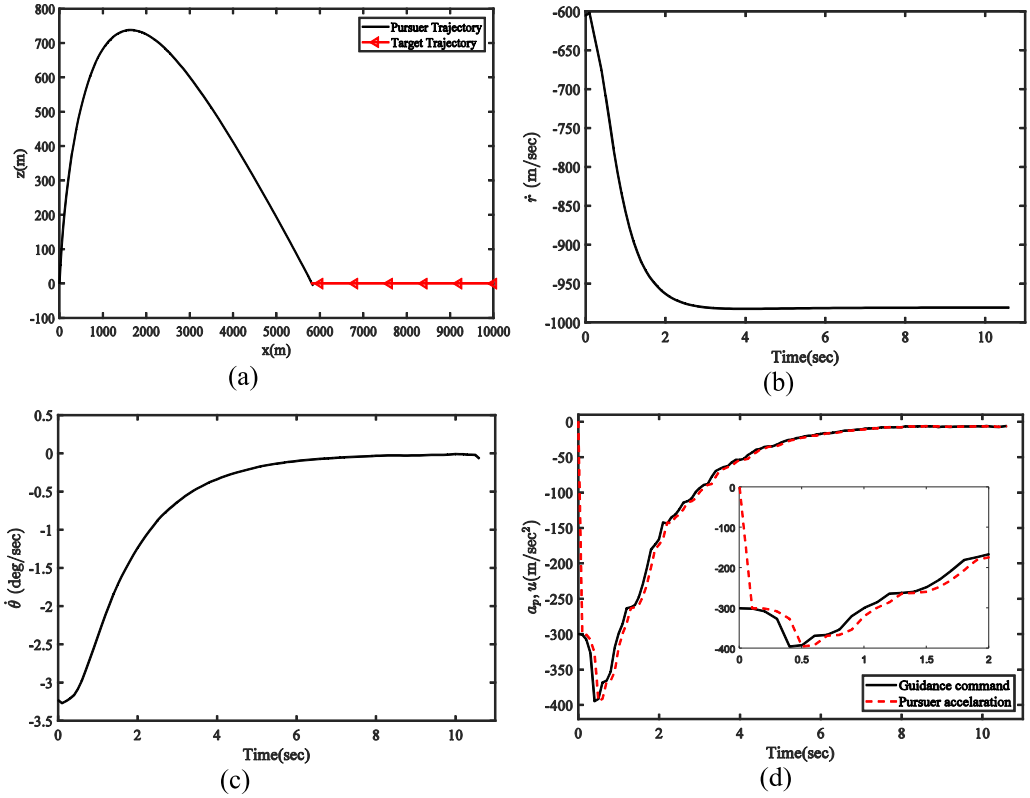


Fig. 4. The engagement scenario against the non-maneuvering target with saturation constraint on the guidance command: (a) engagement trajectory; (b) relative velocity; (c) LOS rate; (d) guidance command and pursuer acceleration.

6.1.5. Case 1d: maneuvering target (Sinusoidal maneuver)

Here, a sinusoidal maneuver is performed by the target, as described in Table 1. Simulation results are depicted in Fig. 8. MD and interception time are 2.93 m and 10.5 s, respectively. The maximum acceleration of the pursuer is 52 g. The high initial acceleration of the pursuer is due to the high initial heading error.

6.1.6. Case 1e: maneuvering target (Random maneuver)

In this scenario, a random maneuver, generated by passing white noise through a third-order Butterworth filter, as expressed in [42], is performed by the target. The noise power is set to 10,000. The generated maneuver is shown in Fig. 9, and the results of the guidance algorithm are shown in Fig. 10. In this case, the MD and the interception time are 3.05 m and 9.8 s, respectively. The maximum acceleration of the pursuer is 51 g.

6.2. Case2: variable velocity pursuer

In reality, the pursuer velocity changes during the engagement. In this section, the target is modeled by Eqs. (33)–(36). However, a more realistic model is assumed for the pursuer.

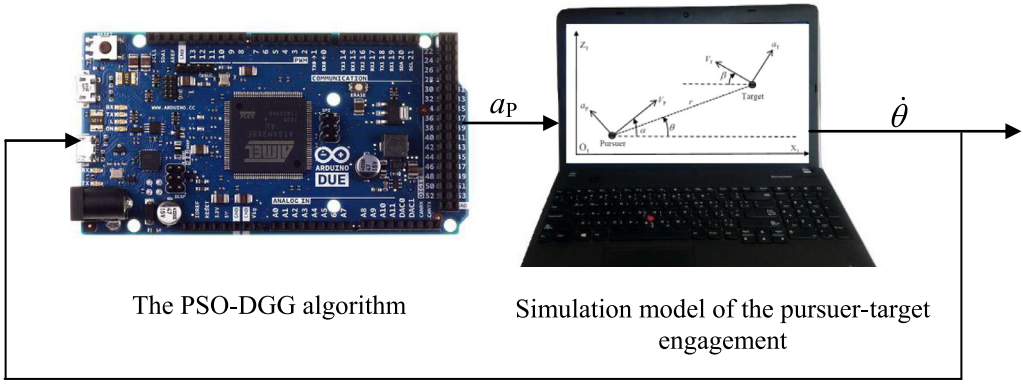


Fig. 5. Schematic diagram of the PIL experiment used to verify the PSO-DGG algorithm.

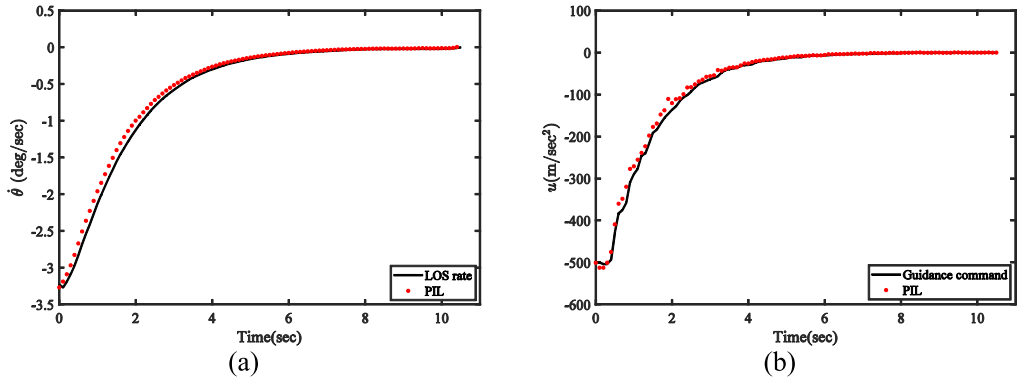


Fig. 6. Comparison of the PIL experiment with numerical simulations for the engagement scenario against the non-maneuvering target: (a) LOS rate; (b) guidance command.

The pursuer equations of motion are written as

$$\dot{x}_P = V_P \cos(\alpha) \tag{37}$$

$$\dot{z}_P = V_P \sin(\alpha) \tag{38}$$

$$\dot{\alpha} = \frac{a_P - g\cos(\alpha)}{V_P} \tag{39}$$

$$\dot{V}_P = \frac{T_P - D}{m_P} - g\sin(\alpha) \tag{40}$$

$$\dot{a}_P = \frac{u - a_P}{\tau_P} \tag{41}$$

Where T_P and D are respectively thrust and drag forces, g is the gravity acceleration and m_p is the pursuer mass. The aerodynamic drag is stated as [13,43]

$$D = D_0 + D_i \tag{42}$$

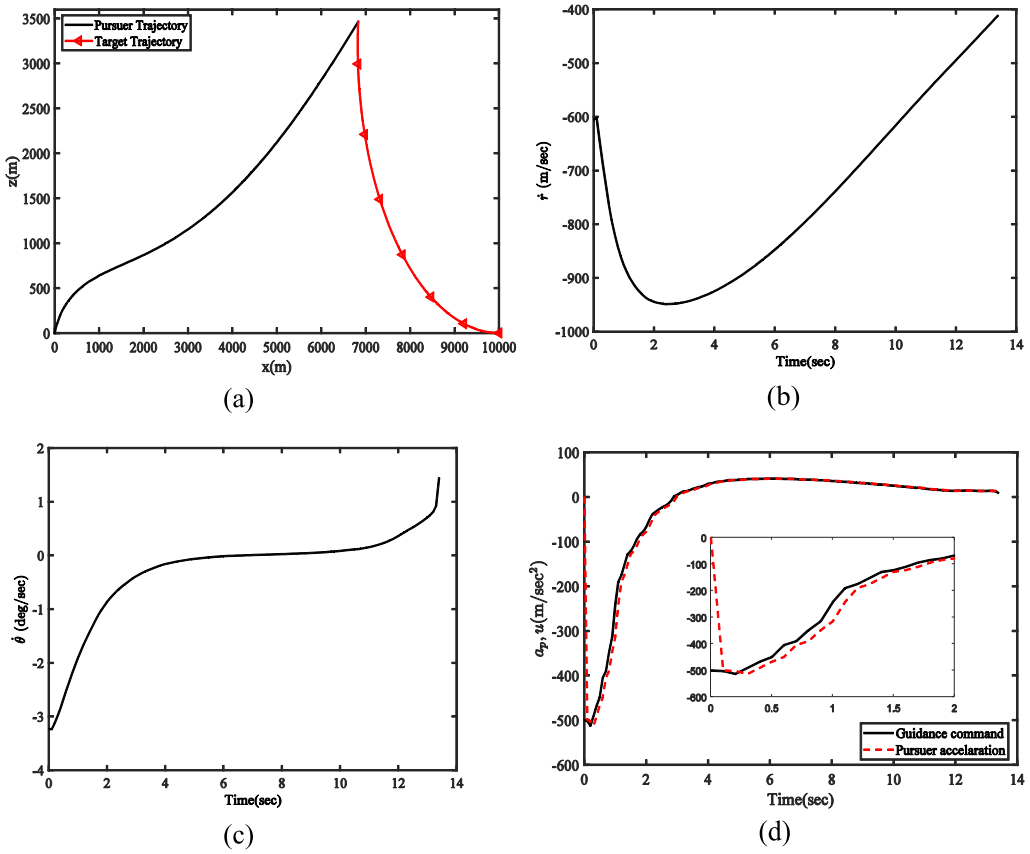


Fig. 7. The engagement against step target maneuver: (a) engagement trajectory; (b) relative velocity; (c) LOS rate; (d) guidance command compared with pursuer acceleration.

$$D_0 = C_{D_0}\bar{Q}s \tag{43}$$

$$D_i = \frac{Ka_P^2 m_P^2}{\bar{Q}s} \tag{44}$$

$$\bar{Q} = \frac{1}{2}\rho V_P^2 \tag{45}$$

Where, C_{D_0} and K represent zero-lift and induced drag coefficients, $\bar{Q}$ denotes the dynamic pressure, ρ is the air density, and $s = 1\,\mathrm{m}^2$ is the reference area. The parameters C_{D_0} and K are modeled as

$$C_{D_0} = \begin{cases} 0.02 & M < 0.93 \\ 0.02 + 0.2(M - 0.93) & 0.93 \le M < 1.033 \\ 0.04 + 0.06(M - 1.03) & 1.03 \le M < 1.1 \\ 0.0442 - 0.007(M - 1.1) & M \ge 1.1 \end{cases} \tag{46}$$

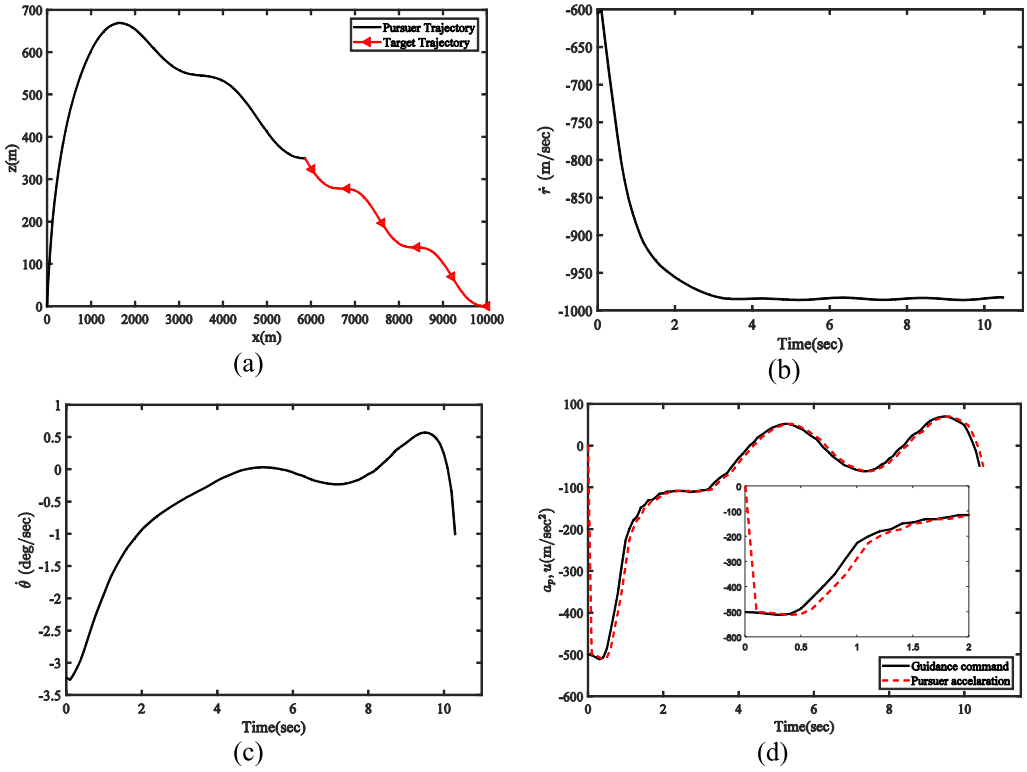


Fig. 8. The engagement against sinusoidal target maneuver: (a) engagement trajectory; (b) relative velocity; (c) LOS rate; (d) guidance command compared with pursuer acceleration.

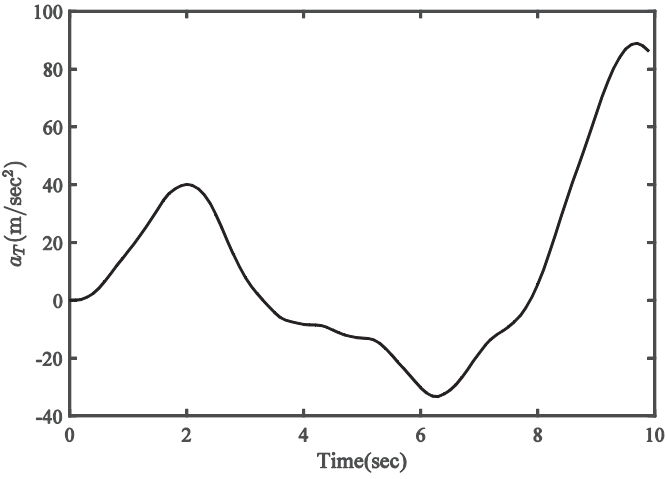


Fig. 9. The random target maneuver, generated based on [42].

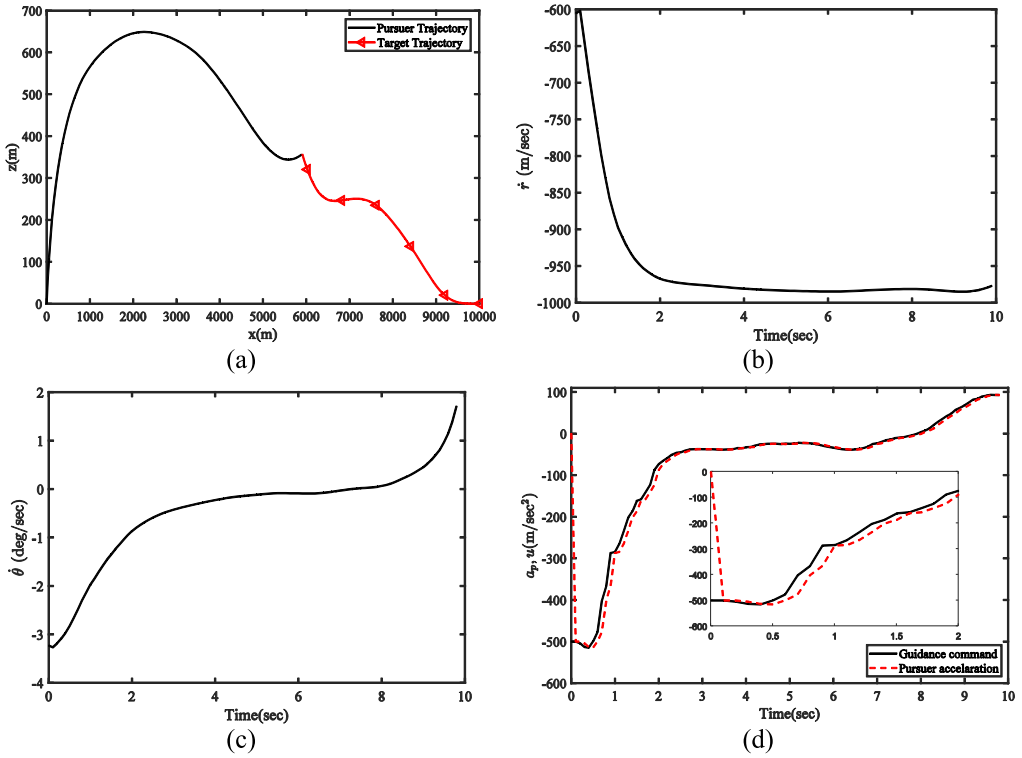


Fig. 10. The engagement against random target maneuver: (a) engagement trajectory; (b) relative velocity; (c) LOS rate; (d) guidance command compared with pursuer acceleration.

$$K = \begin{cases} 0.02 & M < 1.15 \\ 0.02 + 0.245(M - 1.15) & M \geq 1.15 \end{cases} \tag{47}$$

where, the Mach number, M, is calculated as

$$M = \frac{V_P}{\sqrt{403.2T}} \tag{48}$$

and the temperature, T, in Kelvin, is modeled as

$$T = \begin{cases} 288.16 - 0.0065 z_P & z_P < 11000\,\text{m} \\ 216.66 & z_P \geq 11000\,\text{m} \end{cases} \tag{49}$$

Also, the pursuer thrust and mass are given by

$$T_P = \begin{cases} 33000\text{N} & 0 \leq t < 1.5\text{s} \\ 7500\text{N} & 1.5\text{s} \leq t < 8.5\text{s} \\ 0\text{N} & t \geq 12\text{s} \end{cases} \tag{50}$$

$$m_P = \begin{cases} 135 - 14.53t\,\text{kg} & 0 \leq t < 1.5\text{s} \\ 113.205 - 3.331t\,\text{kg} & 1.5\text{s} \leq t < 8.5\text{s} \\ 90.035\,\text{kg} & t \geq 8.5\text{s} \end{cases} \tag{51}$$

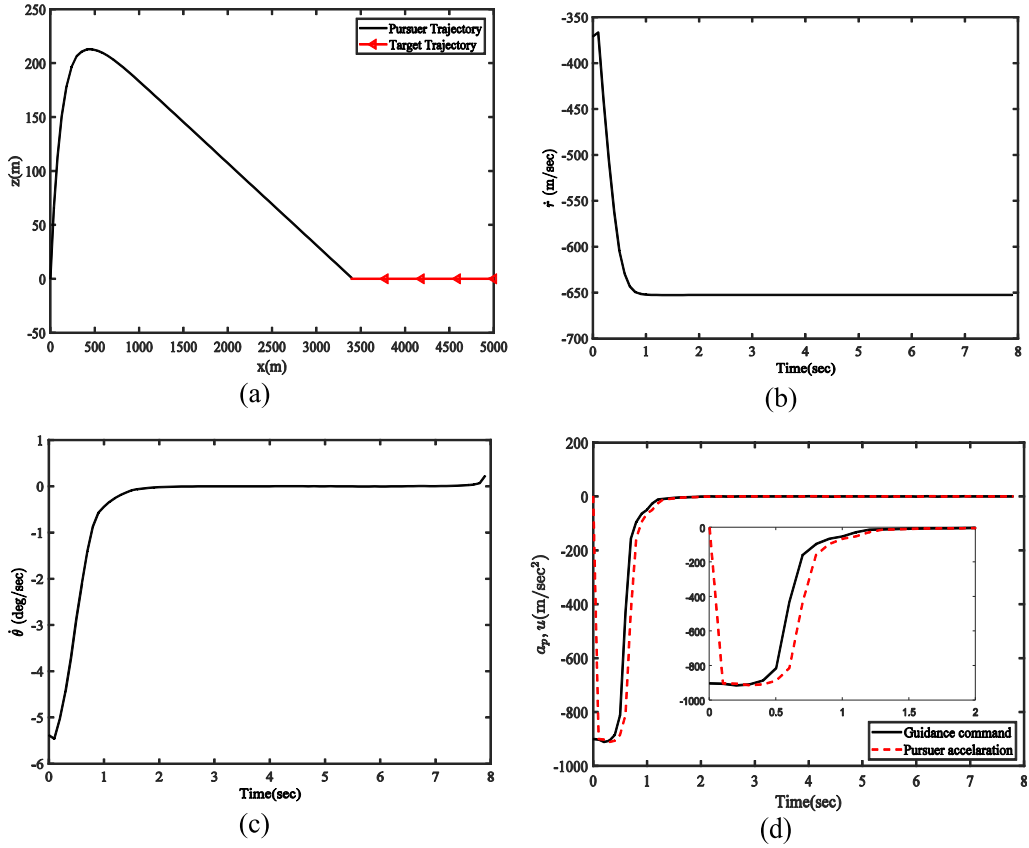


Fig. 11. Engagement of the more realistic pursuer with the non-maneuvering target: (a) engagement trajectory; (b) relative velocity; (c) LOS rate; (d) guidance command compared with pursuer acceleration.

Finally, the variation of the air density is modeled as

$$\rho(z_p) = 1.15579 - 1.058 \times 10^{-4} z_p + 3.725 \times 10^{-9} z_p^2 - 6 \times 10^{-14} z_p^3, z_p \in [0, 20000\text{m}] \quad (52)$$

6.2.1. Case 2a: non-maneuveringtarget

The engagement is shown in Fig. 11(a). The resultant MD and the interception time are 2.62 m and 7.8 s, respectively. Again, deviation of the pursuer from the collision course and consequently the lateral maneuver is due to the high initial heading error. The guidance commands shown in Fig. 11(d) is higher than those represented in Fig. 3(d) since the initial velocity of the pursuer and the relative distance have been decreased, and the initial LOS rate has been increased. Also, Fig. 12 shows the pursuer's velocity.

6.2.2. Case 2b: maneuvering target (Step maneuver)

In this scenario, a step maneuver is performed by the target. The engagement is shown in Fig. 13(a). The MD and the interception time are 3.84 m and 10.9 s, respectively. Again, the initial heading error and the target maneuver produce the LOS rate, and the pursuer performs lateral maneuver to reduce the LOS rate.

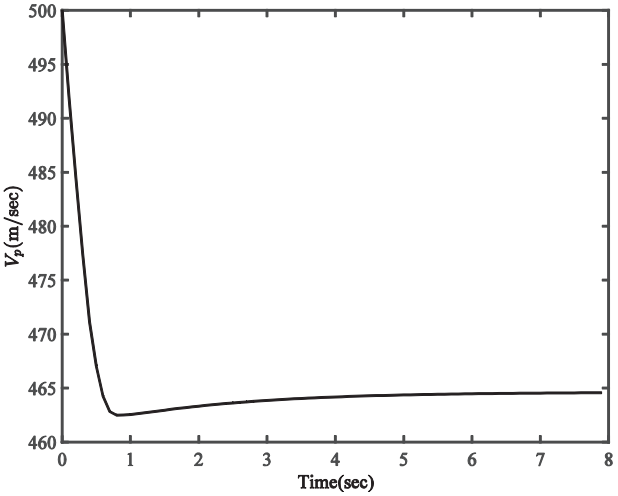


Fig. 12. Non-maneuvering target: pursuer velocity.

Table 2
Simulation data based on references [38,27,32,33,16,39].

Ref.	Pursuer						Target				
	x_{P0} (km)	y_{P0} (km)	z_{P0} (km)	V_{P0} (m/s)	α_0 (deg)	x_{T0} (km)	y_{T0} (km)	z_{T0} (km)	V_{T0} (m/s)	β_0 (deg)	Maneuver (m/s^2)
[38]	0	0	0	600	0	2.5	0	0	400	0	100 (step), $100\sin(\frac{2\pi}{2.5}t)$
[27]	0	0	0	600	30	2.5	0	0	400	55	No maneuver
[32]	0	0	2.95	686	37	5.9	4.4	3.1	275	10	10 (step)
[33]	0	0	0	913	22	3	0	0	480	130	No maneuver
[16]	0	0	0	450	25	4.2	0	2.4	200	10	30 (step)
[39]	0	0	0	500	30	4.2	0	1.5	300	140	30 (step)

6.2.3. Case 2c: maneuvering target (Sinusoidal maneuver)

In this case, the MD and interception time are 3.86 m and 7.8 s, respectively. The guidance command is higher in Fig. 14(d) than in Fig. 8(d), because, the initial velocity of the pursuer and the initial relative distance is less in case 2 than in case 1. The guidance command has not reached a constant value because the target has a sinusoidal maneuver.

In Fig. 15, a comparison is made between the performance of PSO-DGG with that of SDRE-DG and LQDG [38] in the presence of step target maneuver. The parameters, required for the simulation, are selected according to [38] as given in Table 2. The comparison results are provided in Table 3. According to these results, the values of control effort, the peak magnitude of a_p, and the MD are less for the PSO-DGG than the other methods. According to Fig. 15(a), the output of the PSO-DGG has reached a steady-state value in less than 0.5 s; however, for the LQDG, the guidance command has an increasing trend with time. It is evident that the large values of the guidance commands may damage the structure.

In Fig. 16, the comparison of the PSO-DGG algorithm with SDRE-DG and LQDG algorithms [38] is provided in the presence of a sinusoidal target maneuver, defined in Table 2. The results are provided in Table 3. According to these results, the control effort, the peak

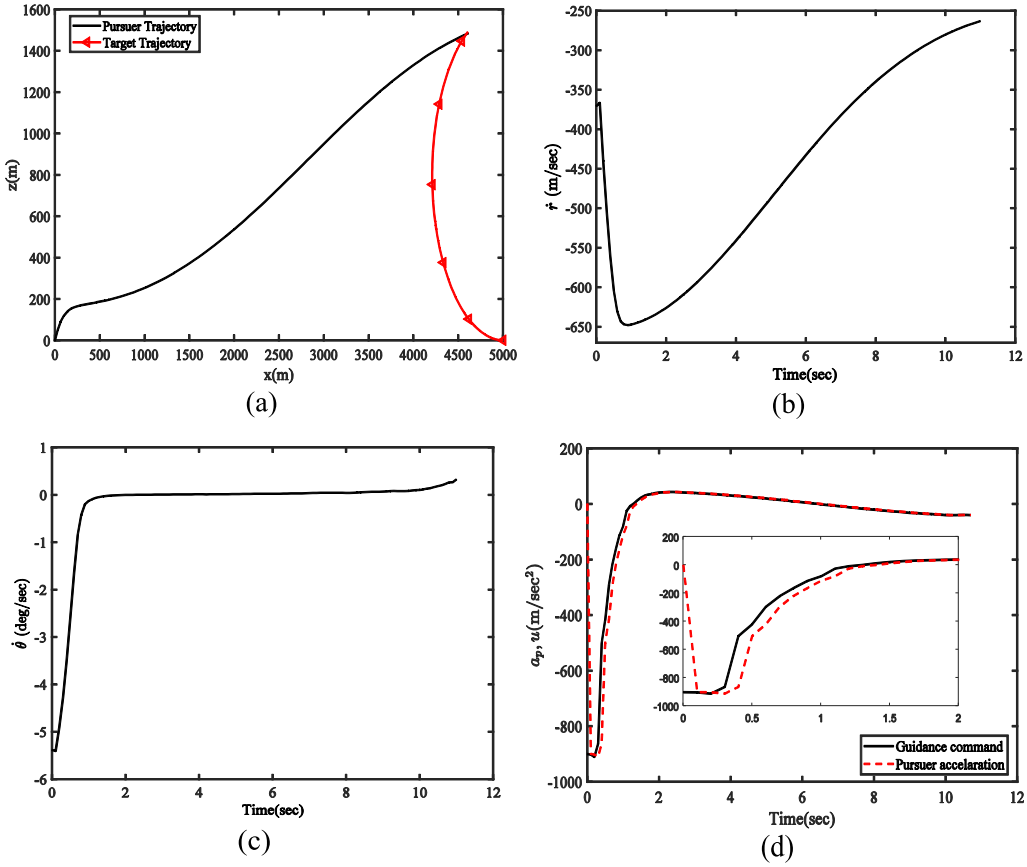


Fig. 13. Engagement of the more realistic pursuer with the target performing step maneuver: (a) engagement trajectory; (b) relative velocity; (c) LOS rate; (d) guidance command compared with pursuer acceleration.

magnitude of a_p, and the MD of PSO-DGG algorithm are lower as compared to the corresponding values of SDRE-DG and LQDG. According to Fig. 16(a), the guidance commands of the SDRE-DG and LQDG have final instability.

In Fig. 17, performance of the PSO-DGG is compared with reference [27] when the target has no maneuver. According to Table 3, the control effort, the peak magnitude of a_p, and the MD of the PSO-DGG are again lower as compared to the values reported in [27]. Moreover, in Fig. 18, a comparison is made between the performances of the PSO-DGG with that of PN-IPSOG [32]. Results, presented in Table 3, again show that the control effort and the MD of the PSO-DGG are lower as compared to the results reported in [32]. It is important to note that in the cost function of the PSO-DGG, the LOS rate, control effort, and control fluctuations are penalized; however, in [32], only the LOS rate is penalized. Furthermore, a comparison between the performance of PSO-DGG and CACC [33] is provided in Fig. 19 and Table 3. According to these results, the control effort and the MD of the PSO-DGG are slightly lower than the values reported in [33]. It should be mentioned that in CACC, in addition to the calculation of the guidance commands, the states are also estimated. As

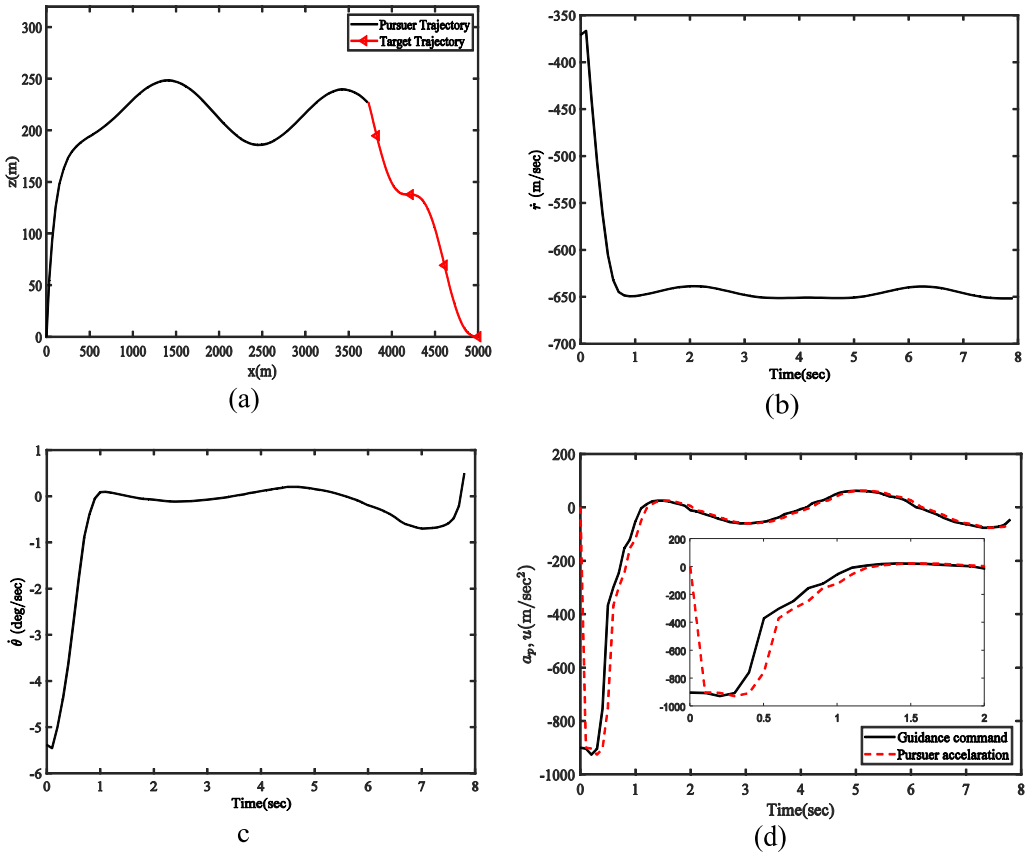


Fig. 14. Engagement of the more realistic pursuer with the target performing sinusoidal maneuver: (a) engagement trajectory; (b) relative velocity; (c) LOS rate; (d) guidance command compared with pursuer acceleration.

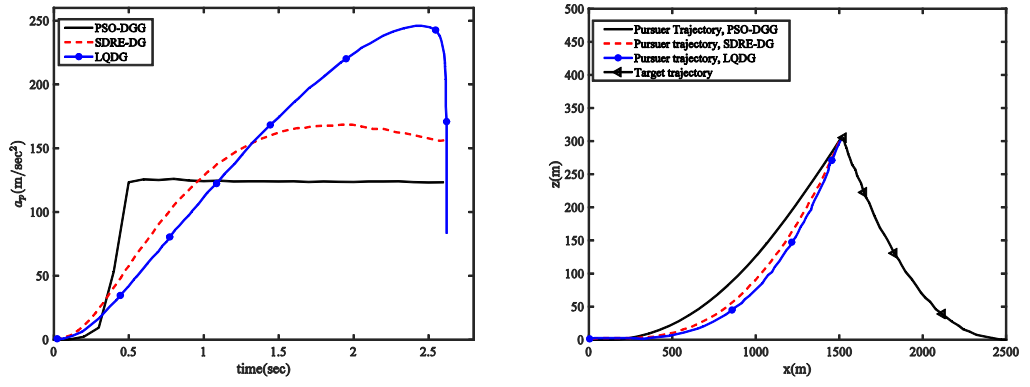


Fig. 15. Comparison with SDRE-DG and LQDG in the presence of step target maneuver: (a) guidance command; (b) engagement trajectory.

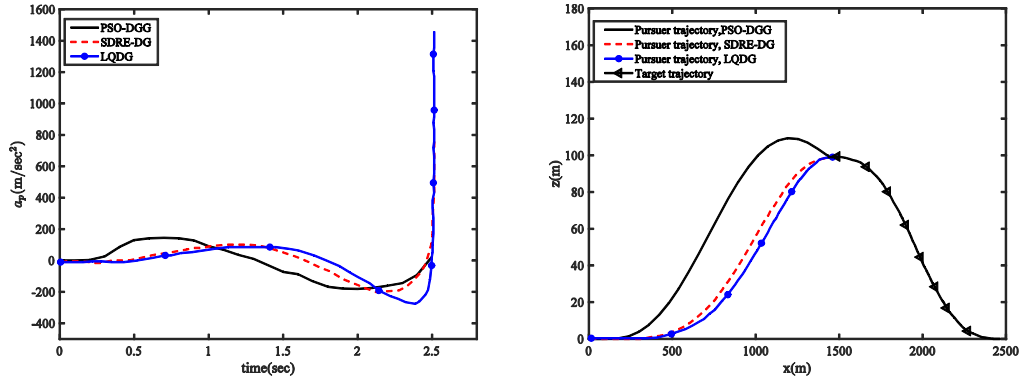


Fig. 16. Comparison with SDRE-DG and LQDG in the presence of sinusoidal target maneuver: (a) guidance command; (b) engagement trajectory.

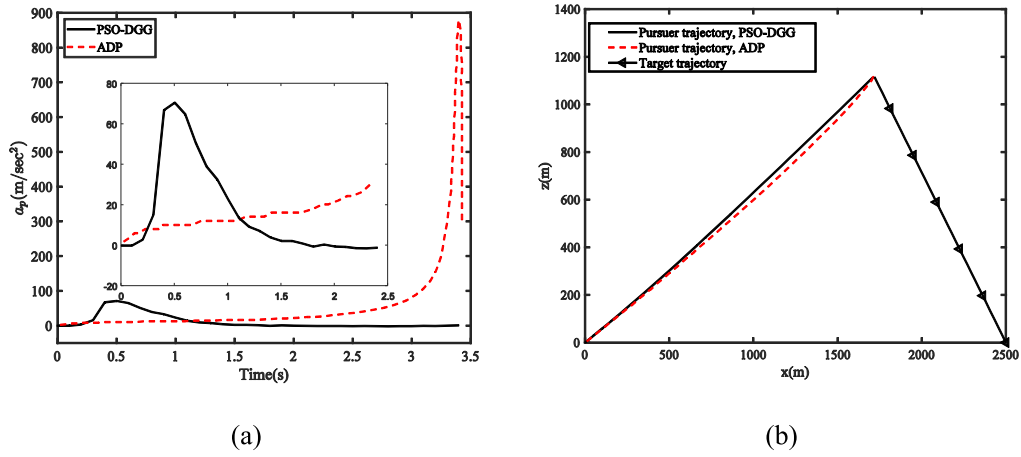


Fig. 17. Comparison with reference [27]: (a) guidance command; (b) engagement trajectory.

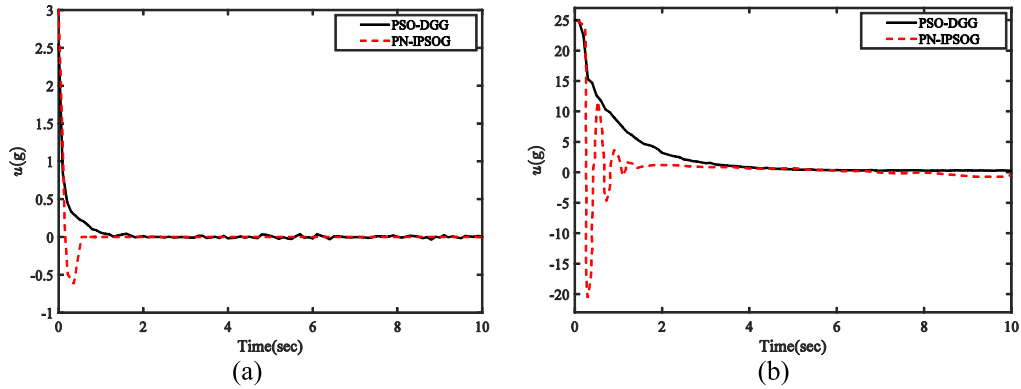


Fig. 18. Acceleration command of the PSO-DGG Compared with PN-IPSOG: (a) z-axis; (b) y-axis.

Table 3
Performance of the PSO-DGG Compared to other guidance approaches.

Guidance Law	Maneuver	Peak magnitude of a_p (m/s^2)	Interception time (s)	Control effort (m^2/s^4)	Miss distance (m)
PSO-DGG	Step	124	2.6	3.57×10^5	0.35
SDRE-DG [38]		180	2.6	9.27×10^5	Less than 0.5
LQDG [38]		240	2.6	1.91×10^6	Less than 0.5
PSO-DGG	Sinusoidal	180	2.5	3.66×10^5	0.42
SDRE-DG [38]		771	2.5	2.87×10^6	Less than 0.7
LQDG [38]		1457	2.5	2.69×10^7	Less than 0.7
PSO-DGG	No maneuver	70.6	3.4	2×10^4	0.3
ADP [27]		879	3.4	1.75×10^7	Less than 0.5
PSO-DGG	Step	250	10.1	1.24×10^5	5.72
PN-IPSOG [32]		250	10.1	2.49×10^5	7.57
PSO-DGG	No maneuver	108	5	1.89×10^4	0.90
CACC [33]		108	5	2.06×10^4	0.94
PSO-DGG	Step	250	7.9	1.97×10^4	4.35
MCACC [16]		300	7.9	2.12×10^4	5.06
PSO-DGG	Step	149	6.14	5.31×10^5	0.094
SBPN [39]		147	6.14	6.22×10^5	0.13
PSO-DGG	Step	160	8.14	1.34×10^5	9.40
APN [16]		143	8.14	2.79×10^5	13.43

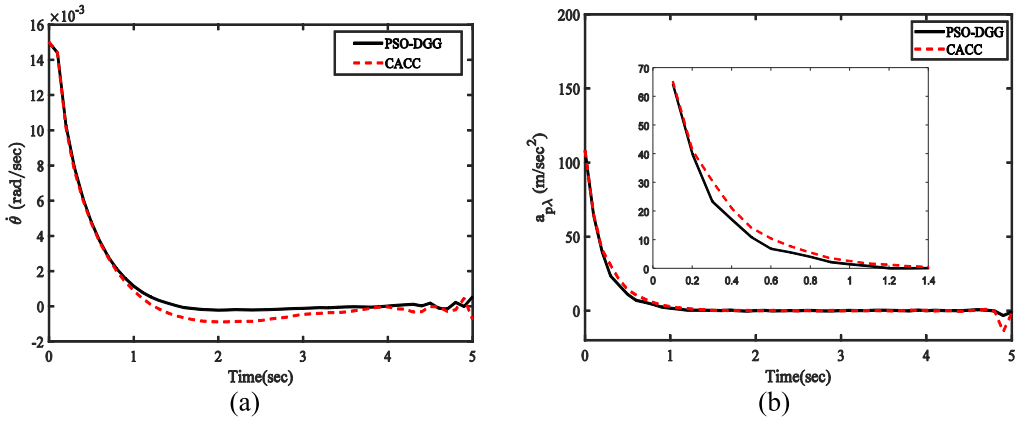
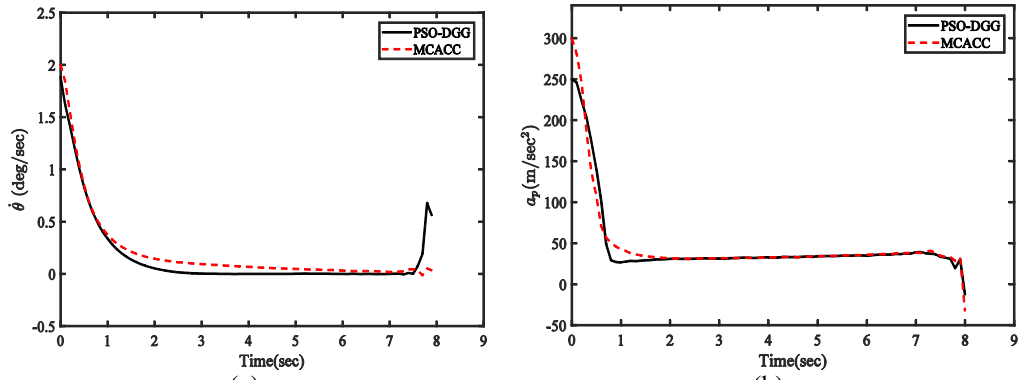


Fig. 19. Comparison with the CACC: (a) LOS rate; (b) Acceleration command.

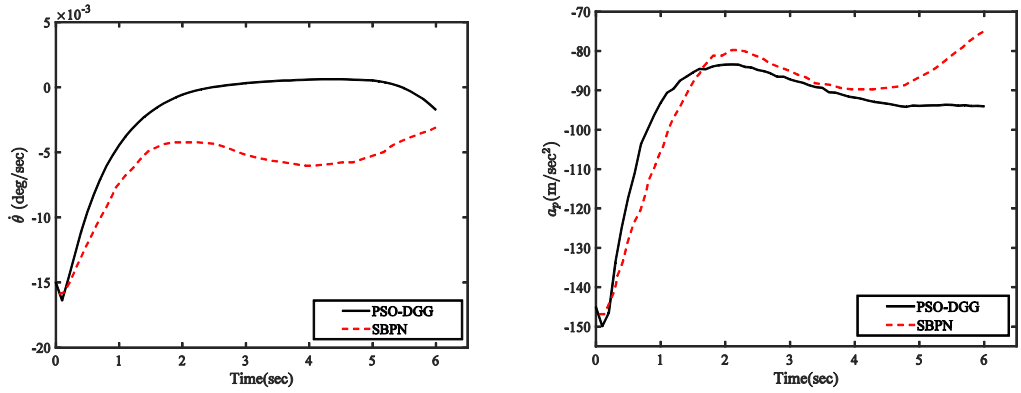
expected, the last comparison shows that PSO-DGG has no significant benefit than CACC against a non-maneuvering target.

Fig. 20 compares the performance of the PSO-DGG with the MCACC [16]. Fig. 20(a) demonstrates that the guidance command of the PSO-DGG is less than that of the MCACC. Also, the PSO-DGG nullifies the LOS rate faster than the MCACC. Fig. 21 compares the PSO-DGG with the SBPN guidance law [39]. According to Fig. 21(a), the PSO-DGG nullifies the LOS rate better than the SBPN. Fig. 22 compares the PSO-DGG with the augmented proportional navigation (APN), where the effective navigation constant is 4. This value is



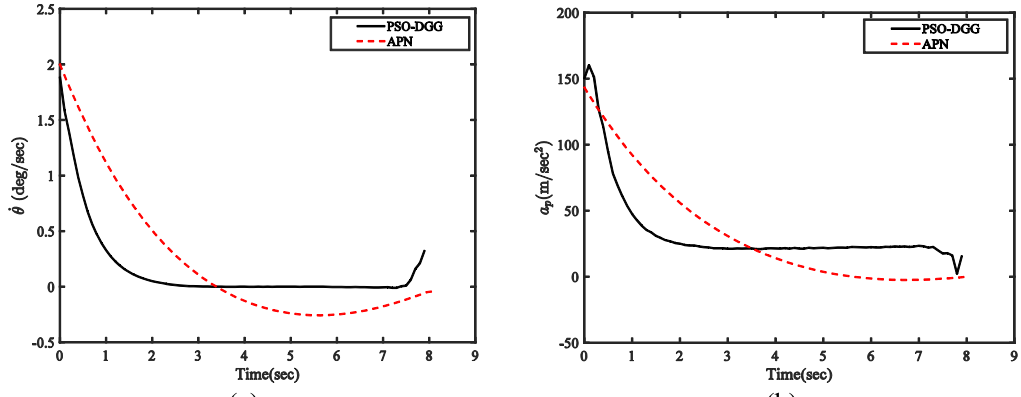
(a) (b)

Fig. 20. Comparison with the MCACC: (a) LOS rate; (b) Acceleration command.



(a) (b)

Fig. 21. Comparison with the SBPN guidance law: (a) LOS rate; (b) Acceleration command.



(a) (b)

Fig. 22. Comparison with the APN guidance law: (a) LOS rate; (b) Acceleration command.

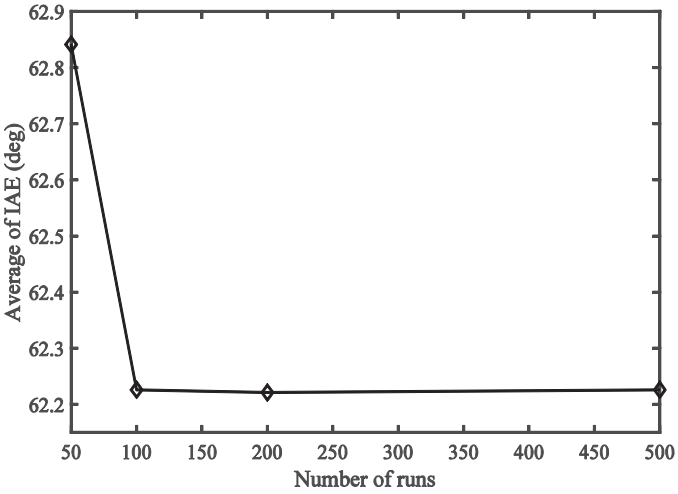


Fig. 23. Average of the IAE versus the number of runs.

determined by trial and error to minimize the MD and the control effort. The target acceleration, needed for the APN, is taken from the simulation. Fig. 22(a) shows that the PSO-DGG nullifies the LOS rate faster than the APN.

The stability and convergence of the PSO algorithm has been discussed in [44–48]. It has been shown in [45] that the particle dynamics in the PSO algorithm will be stable, and will converge if the following conditions are met:

$$w < 1 \tag{53}$$

$$c_1 + c_2 < 2(1 + w) \tag{54}$$

The parameters of the PSO-DGG algorithm have been set such that they satisfy the above conditions. It should also be noted that the PSO-DGG is a stochastic guidance algorithm. Therefore, to evaluate statistical properties, a Monte Carlo simulation [49] should be performed. For this purpose, scenario 1a, as an example, is simulated 500 times, and the Integral of Absolute Error (IAE) is calculated for the LOS rate. The average of the IAE versus the number of runs is shown in Fig. 23 for 50, 100, 200, and 500 runs. According to Fig. 23, performing 100 runs seems to be enough for convergence of the statistical properties. Stability of the PSO-DGG is also investigated numerically. For this purpose, the mean, best, worst, and standard deviation of the cost is calculated in the last sample. Results, presented in Table 4, have been calculated after simulating each scenario 100 times. These results again verify the stability of the algorithm.

Now, the sensitivity of the MD to the sampling time is investigated in Fig. 24. The results show that decreasing the sampling time from 0.2 s to 0.1 s decreases the MD, considerably. However, more decrease in the sampling time has no significant effect on the MD, while it increases the computational cost. Therefore, the selected value (0.1 s) seems acceptable.

A sensitivity analysis is also performed for the effect of the PSO-DGG parameters, on the convergence of the LOS rate. The effect of N and l_{max} is illustrated in Fig. 25. It is observed that, decreasing N from 35 to 20 causes a considerable overshoot. Also, decreasing the maximum number of iterations from 20 to 10 causes oscillation of the LOS rate.

Table 4
Mean, best, worst cost obtained for 100 runs, and the standard deviation.

Scenario	Minimum Cost			Standard deviation
	Mean	Best	Worst	
1a	0.190	0.068	0.380	0.059
1b	9.029	8.828	9.782	0.131
1c	8.982	8.694	9.754	0.104
2a	0.162	0.091	0.279	0.045
2b	7.332	6.868	8.352	0.127
2c	7.601	7.018	8.624	0.110

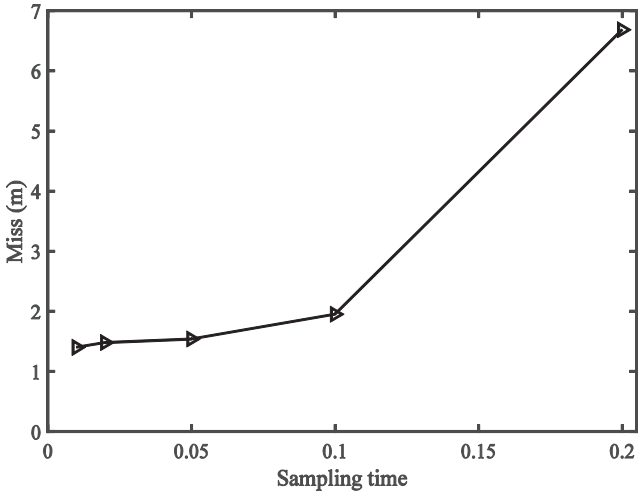


Fig. 24. The sensitivity of the miss distance to the sampling time.

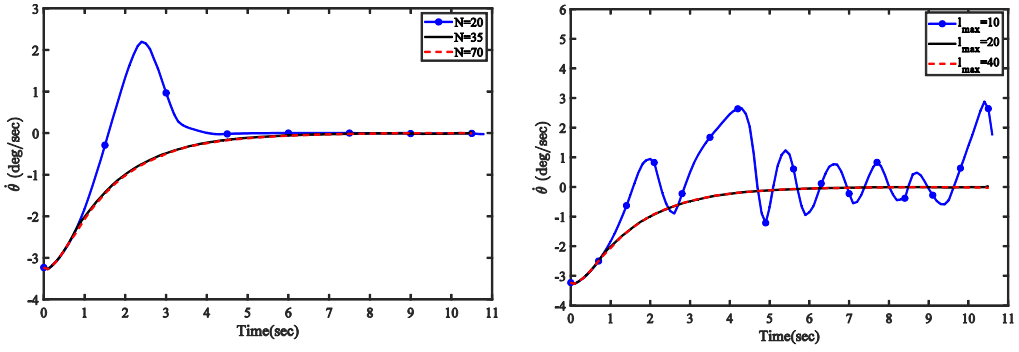


Fig. 25. Sensitivity of the PSO-DGG to the number of particles and number of iterations: (a) effect of N; (b) effect of l_{max}.

Table 5
The effect of uncertainty on the MD.

Guidance Algorithm	MD (m)			
	Uncertainty Bound			Without Uncertainty
	+10%	+20%	+30%	
PSO-DGG	3.2	4.8	35.2	2.5
CACC	4.3	8.2	70.6	2.9
MCACC	5.1	10.1	81.6	3.7
APN	7.8	14.9	93.7	4.1

Finally, the effect of uncertainties is studied, in scenario 1c, by altering the uncertain parameters, presented in Table 1, 10%, 20%, and 30% beyond the maximum/minimum values. The comparison was also performed without considering the uncertainties. The resultant MD of different guidance algorithms is obtained using Monte-Carlo simulation, and presented in Table 5. As expected, increasing the uncertainty level increases the MD. However, the PSO-DGG shows competitive and better results than the other approaches. Also, by increasing the uncertainty level up to 30%, the results deteriorate dramatically.

6.3. Computational cost

The PSO-DGG was implemented in MATLAB, and a computer with a Core 2 Duo 3 GHz CPU and 4 GByte RAM, was used for simulations. The computing time for one-step of the guidance algorithm, is approximately 0.06 s, while the sampling time of the problem is 0.1 s. It is evident that implementing the proposed algorithm in a programming language like *C++* can enormously reduce the computational burden.

7. Conclusion

In this paper, a heuristic robust model predictive differential game guidance algorithm, called PSO-DGG, was proposed. The robust guidance problem was solved by converting it into a nonlinear model predictive control problem. For this purpose, an appropriate cost function was proposed that reflects the uncertainties. In designing the guidance algorithm, first-order lag was considered for the dynamic of the pursuer. Numerical simulations were conducted for the performance evaluation of the proposed guidance algorithm at a constant velocity as well as step, sinusoidal, and random maneuvers. Performance evaluation was also tested considering the aerodynamic model of the pursuer. Numerical simulations demonstrate that the proposed guidance algorithm performs quite well when the pursuer has a high initial heading error, and there is a constraint on the guidance command. Moreover, the implementation of the PSO-DGG algorithm on a target platform was conducted successfully through the PIL experiment. Statistical performance of the PSO-DGG was investigated via Monte Carlo simulation. Finally, the performance of the PSO-DGG was compared with LQDG, SDRE-DG, ADP, a heuristic guidance algorithm, called PN-IPSOG, the CACC, MCACC, SBPN, and APN. It was shown that the miss distance and the control effort of the PSO-DGG are less than these guidance algorithms. The major drawback of the PSO-DGG is the computational load. However, it was shown that it could be implemented in real-time using a typical processor.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors gratefully acknowledge the financial support of the research office at Sharif University of Technology.

Appendix A. Proof of Proposition 1

Assume that $\mathbf{u}(t)=\mathbf{u}_0(t)$ is the solution of the optimal control problem. Consider the following positive definite Lyapunov function:

$$V(\mathbf{x}(t))=\min_{\mathbf{u}\in\Re^m}\int_0^\infty\left[\mathbf{x}^{\mathrm{T}}(t)\mathbf{Q}\mathbf{x}(t)+\mathbf{u}^{\mathrm{T}}(t)\mathbf{R}\mathbf{u}(t)+\mathbf{a}_{\max}^{\mathrm{T}}(\mathbf{x}(t))\mathbf{a}_{\max}(\mathbf{x}(t))\right]dt \tag{A.1}$$

We would like to show that Eq. (1) is asymptotically stable for $\mathbf{u}(t)=\mathbf{u}_0(t)$. For this purpose, the Lyapunov function must satisfy the Hamilton-Jacobi- Bellman (H-J-B) equation written as [35]

$$\min_{\mathbf{u}\in\Re^m}\left\{L(\mathbf{x}(t),\mathbf{u}(t))+\left(\frac{\partial V}{\partial\mathbf{x}}\right)^{\mathrm{T}}\dot{\mathbf{x}}(t)\right\}=0 \tag{A.2}$$

From equations (A.1), (A.2), and (2), the H-J-B equation can be rewritten as

$$\min_{\mathbf{u}\in\Re^m}\left\{\mathbf{x}^{\mathrm{T}}(t)\mathbf{Q}\mathbf{x}(t)+\mathbf{u}^{\mathrm{T}}(t)\mathbf{R}\mathbf{u}(t)+\mathbf{a}_{\max}^{\mathrm{T}}(\mathbf{x}(t))\mathbf{a}_{\max}(\mathbf{x}(t))+V_{\mathbf{x}}^{\mathrm{T}}\left[\mathbf{f}(\mathbf{x}(t))+\mathbf{g}(\mathbf{x}(t))\mathbf{u}(t)\right]\right\}=0 \tag{A.3}$$

If $\mathbf{u}_0(t)$ is the solution of the optimal control problem, then

$$\mathbf{x}^{\mathrm{T}}(t)\mathbf{Q}\mathbf{x}(t)+\mathbf{u}_0^{\mathrm{T}}(t)\mathbf{R}\mathbf{u}_0(t)+\mathbf{a}_{\max}^{\mathrm{T}}(\mathbf{x}(t))\mathbf{a}_{\max}(\mathbf{x}(t))+V_{\mathbf{x}}^{\mathrm{T}}\left[\mathbf{f}(\mathbf{x}(t))+\mathbf{g}(\mathbf{x}(t))\mathbf{u}_0(t)\right]=0 \tag{A.4}$$

$$2\mathbf{R}\mathbf{u}_0^{\mathrm{T}}(t)+V_{\mathbf{x}}^{\mathrm{T}}\mathbf{g}(\mathbf{x}(t))=0 \tag{A.5}$$

By using equations (1), (A.1), (A.4), and (A.5), $\dot{V}$is written as

$$\begin{aligned}
\dot{V}(\mathbf{x}(t)) &= V_{\mathbf{x}}^{\mathrm{T}}\dot{\mathbf{x}}(t)\\
&= V_{\mathbf{x}}^{\mathrm{T}}\left[\mathbf{f}(\mathbf{x}(t))+\mathbf{g}(\mathbf{x}(t))\mathbf{u}_0(t)+\mathbf{g}(\mathbf{x}(t))\mathbf{a}(\mathbf{x}(t))\right]\\
&= V_{\mathbf{x}}^{\mathrm{T}}\left[\mathbf{f}(\mathbf{x}(t))+\mathbf{g}(\mathbf{x}(t))\mathbf{u}_0(t)\right]+V_{\mathbf{x}}^{\mathrm{T}}\mathbf{g}(\mathbf{x}(t))\mathbf{a}(\mathbf{x}(t))\\
&= -\mathbf{x}^{\mathrm{T}}(t)\mathbf{Q}\mathbf{x}(t)-\mathbf{a}_{\max}^{\mathrm{T}}(\mathbf{x}(t))\mathbf{a}_{\max}(\mathbf{x}(t))-\mathbf{u}_0^{\mathrm{T}}(t)\mathbf{R}\mathbf{u}_0(t)+V_{\mathbf{x}}^{\mathrm{T}}\mathbf{g}(\mathbf{x}(t))\mathbf{a}(\mathbf{x}(t))\\
&= -\mathbf{x}^{\mathrm{T}}(t)\mathbf{Q}\mathbf{x}(t)-\mathbf{a}_{\max}^{\mathrm{T}}(\mathbf{x}(t))\mathbf{a}_{\max}(\mathbf{x}(t))-\mathbf{u}_0^{\mathrm{T}}(t)\mathbf{R}\mathbf{u}_0(t)-2\mathbf{R}\mathbf{u}_0^{\mathrm{T}}(t)\mathbf{a}(\mathbf{x}(t))\\
&= -\mathbf{x}^{\mathrm{T}}(t)\mathbf{Q}\mathbf{x}(t)-\mathbf{a}_{\max}^{\mathrm{T}}(\mathbf{x}(t))\mathbf{a}_{\max}(\mathbf{x}(t))+\mathbf{a}^{\mathrm{T}}(\mathbf{x}(t))\mathbf{a}(\mathbf{x}(t))\\
&\quad -\mathbf{u}_0^{\mathrm{T}}(t)\mathbf{R}\mathbf{u}_0(t)-2\mathbf{R}\mathbf{u}_0^{\mathrm{T}}(t)\mathbf{a}(\mathbf{x}(t))-\mathbf{a}^{\mathrm{T}}(\mathbf{x}(t))\mathbf{a}(\mathbf{x}(t))\\
&\approx -\mathbf{x}^{\mathrm{T}}(t)\mathbf{Q}\mathbf{x}(t)-\mathbf{a}_{\max}^{\mathrm{T}}(\mathbf{x}(t))\mathbf{a}_{\max}(\mathbf{x}(t))+\mathbf{a}^{\mathrm{T}}(\mathbf{x}(t))\mathbf{a}(\mathbf{x}(t))\\
&\quad -[\mathbf{R}\mathbf{u}_0(t)+\mathbf{a}(\mathbf{x}(t))]^{\mathrm{T}}[\mathbf{R}\mathbf{u}_0(t)+\mathbf{a}(\mathbf{x}(t))]\\
&\le -\mathbf{x}^{\mathrm{T}}(t)\mathbf{Q}\mathbf{x}(t)-\mathbf{a}_{\max}^{\mathrm{T}}(\mathbf{x}(t))\mathbf{a}_{\max}(\mathbf{x}(t))+\mathbf{a}^{\mathrm{T}}(\mathbf{x}(t))\mathbf{a}(\mathbf{x}(t))
\end{aligned} \tag{A.6}$$

Since $\|\mathbf{a}(\mathbf{x}(t))\| \leq \mathbf{a}_{\max}(\mathbf{x}(t))$ and $\mathbf{a}^{\mathrm{T}}(\mathbf{x}(t))\mathbf{a}(\mathbf{x}(t)) \leq \mathbf{a}_{\max}^{\mathrm{T}}(\mathbf{x}(t))\mathbf{a}_{\max}(\mathbf{x}(t))$, $\dot{V}$ is rewritten as

$$\dot{V}(\mathbf{x}(t)) \leq -\mathbf{x}^{\mathrm{T}}(t)\mathbf{Q}\mathbf{x}(t) \tag{A.7}$$

Thus, the derivative of the Lyapunov function is negative definite.

References

[1] G.M. Siouris, Missile Guidance and Control Systems, Springer Science & Business Media, 2005.
[2] N.A. Shneydor, Missile Guidance and Pursuit: Kinematics, Dynamics and Control, Elsevier, 1998.
[3] R. Yanushevsky, Modern Missile Guidance, CRC Press, 2007.
[4] Z. Paul, Tactical and strategic missile guidance, American Institute of Aeronautics and Astronautics, 2012.
[5] M. Guelman, A qualitative study of proportional navigation, IEEE Trans. Aerosp. Electron. Syst. 4 (1971) 637–643.
[6] In-Joong Ha, Jong-Sung Hur, Myoung-Sam Ko, Taek-Lyul Song, Performance analysis of PNG laws for randomly maneuvering targets, IEEE Trans. Aerosp. Electron. Syst. 26 (1990) 713–721.
[7] Pin-Jar Yuan, Shih-Che Hsu, Solutions of generalized proportional navigation with maneuvering and nonmaneuvering targets, IEEE Trans. Aerosp. Electron. Syst. 31 (1995) 469–474.
[8] T. Raghunathan, D. Ghose, Differential evolution based 3-D guidance law for a realistic interceptor model, Appl. Soft Comput. 16 (2014) 20–33.
[9] U.S.A.M. Command, R. Arsenal, Terminal Guidance for Impact Attitude Angle Constrained Flight Trajectories, IEEE Trans. Aerosp. Electron. Syst. AES-9 (1973) 852–859.
[10] T.L. Song, Impact angle control for planar engagements, IEEE Trans. Aerosp. Electron. Syst. 35 (1999) 1439–1444.
[11] E.J. Ohlmeyer, C.A. Phillips, Generalized Vector Explicit Guidance, J. Guid. Control. Dyn. 29 (2008) 261–268.
[12] L. Lin, M. Xin, Missile Guidance Law Based on New Analysis and Design of SDRE Scheme, J. Guid. Control. Dyn. 42 (2019) 853–868.
[13] R. Bardhan, D. Ghose, Nonlinear Differential Games-Based Impact-Angle-Constrained Guidance Law, J. Guid. Control. Dyn. 38 (2015) 384–402.
[14] B. Zhao, S. Xu, J. Guo, R. Jiang, J. Zhou, Integrated strapdown missile guidance and control based on neural network disturbance observer, Aerosp. Sci. Technol. 84 (2019) 170–181.
[15] R. Chai, A. Savvaris, S. Chai, Integrated missile guidance and control using optimization-based predictive control, Nonlinear Dyn 96 (2019) 997–1015.
[16] H. Nobahari, S. Nasrollahi, A terminal guidance algorithm based on ant colony optimization, Comput. Electr. Eng. 77 (2019) 128–146.
[17] J. Sun, C. Liu, Backstepping-based adaptive dynamic programming for missile-target guidance systems with state and input constraints, J. Frankl. Inst 355 (2018) 8412–8440.
[18] H. Ji, X. Liu, Z. Song, Y. Zhao, Time-varying sliding mode guidance scheme for maneuvering target interception with impact angle constraint, J. Frankl. Inst 355 (2018) 9192–9208.
[19] K.R. Babu, I.G. Sarma, K.N. Swamy, Two robust homing missile guidance laws based on sliding mode control theory, in: Proceedings of the IEEE National Aerospace and Electronics Conference, 1994, pp. 540–547.
[20] Y. Ho, A. Bryson, S. Baron, Differential games and optimal pursuit-evasion strategies, IEEE Trans. Automat. Control 10 (1965) 385–389.
[21] V. Turetsky, J. Shinar, Missile guidance laws based on pursuit-evasion game formulations, Automatica 39 (2003) 607–618.
[22] T. Shima, O.M. Golan, Linear Quadratic Differential Games Guidance Law I. Introduction, IEEE Trans. Aerosp. (2007) 1–33.
[23] V. Shaferman, T. Shima, Linear quadratic guidance laws for imposing a terminal intercept angle, J. Guid. Control. Dyn. 31 (2008) 1400–1412.
[24] X.-.Y. Xu, X.-.N. Song, Y.-.L. Cai, RETRACTED: differential game guidance law for a kinetic kill vehicle and its simulation, Simulation (2015) 003754971558883.
[25] R. Chen, J. Speyer, D. Lianos, Game-theoretic homing missile guidance with autopilot lag, AIAA Guid. Navig. (2012).

[26] N. Qi, Y. Liu, Z. Tang, Bounded differential game guidance law for interceptor with second-order maneuvering dynamics, in: Proceedings of the International Conference on Instrumentation and Measurement, Computer, Communication and Control, 2011, pp. 925–928.
[27] J. Sun, C. Liu, Q. Ye, Robust differential game guidance laws design for uncertain interceptor-target engagement via adaptive dynamic programming, Int. J. Control. 90 (2017) 990–1004.
[28] J. Sun, C. Liu, Finite-horizon differential games for missile–target interception system using adaptive dynamic programming with input constraints, Int. J. Syst. Sci. 49 (2018) 264–283.
[29] S. Kang, H.J. Kim, M.-.J. Tahk, Aerial pursuit-evasion game using nonlinear model predictive guidance, in: Proceedings of the AIAA Guidance, Navigation, and Control Conference, 2010, p. 7880.
[30] C.C. Kung, K.Y. Chen, Missile guidance algorithm design using particle swarm optimization, Trans. Can. Soc. Mech. Eng. (2013) 971–979.
[31] K.Y. Chen, Y.L. Lee, S.J. Liao, C.C. Kung, The design of particle swarm optimization guidance using a line-of-sight evaluation method, Comput. Electr. Eng. 54 (2016) 159–169.
[32] Y.L. Lee, K.Y. Chen, S.J. Liao, Using proportional navigation and a particle swarm optimization algorithm to design a dual mode guidance, Comput. Electr. Eng. 54 (2016) 137–146.
[33] H. Nobahari, S. Nasrollahi, A nonlinear estimation and control algorithm based on ant colony optimization, in: Proceedings of the IEEE Congress on Evolutionary Computation, 2016, pp. 5120–5127.
[34] H. Nobahari, S. Nasrollahi, A non-linear estimation and model predictive control algorithm based on ant colony optimization, Trans. Inst. Meas. Control (2019) 1123–1138.
[35] F. Lin, Robust Control Design: an Optimal Control Approach, John Wiley & Sons, 2007.
[36] M. Xin, S.N. Balakrishnan, E.J. Ohlmeyer, Integrated guidance and control of missiles with θ-D method, IEEE Trans. Control Syst. Technol. 14 (2006) 981–992.
[37] R.C. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in: Proceedings of the Sixth International Symposium on Micro Machine and Human Science, 1995, pp. 39–43.
[38] R. Bardhan, D. Ghose, An SDRE Based Differential Game Approach for Maneuvering Target Interception, in: Proceedings of the AIAA Guidance, Navigation, and Control Conference, 2015, p. 341.
[39] K.R. Babu, I.G. Sarma, K.N. Swamy, Switched bias proportional navigation for homing guidance against highly maneuvering targets, J. Guid. Control. Dyn. 17 (1994) 1357–1363.
[40] H. Nobahari, A. Sharifi, A hybridization of extended Kalman filter and Ant Colony Optimization for state estimation of nonlinear systems, Appl. Soft Comput. 74 (2019) 411–423.
[41] A. Kurniawan, Getting Started with Matlab Simulink and Arduino, PE Press, 2013.
[42] J.E. Kain, D.J. Yost, Command to line-of-sight guidance: a stochastic optimal control problem, J. Spacecr. Rockets. 14 (1977) 438–444.
[43] P. Kee, L. Dong, C. Siong, Near optimal midcourse guidance law for flight vehicle, in: Proceedings of the 36th AIAA Aerospace Sciences Meeting and Exhibit, 1998, p. 583.
[44] M. Clerc, J.K.-I. transactions on E. Computation, U. 2002, The particle swarm-explosion, stability, and convergence in a multidimensional complex space, IEEE Trans. Evol. Comput. 6 (2002) 58–73.
[45] I.C. Trelea, The particle swarm optimization algorithm: convergence analysis and parameter selection, Inf. Process. Lett. 85 (2003) 317–325.
[46] M. Jiang, Y. Luo, U. S.Y.-I.P. Letters, 2007, Stochastic convergence analysis and parameter selection of the standard particle swarm optimization algorithm, Inf. Process. Lett. 102 (2007) 8–16.
[47] V. Kadirkamanathan, K. Selvarajah, P.J. Fleming, Stability analysis of the particle dynamics in particle swarm optimizer, IEEE Trans. Evol. Comput. 10 (2006) 245–255.
[48] K. Yasuda, N. Iwasaki, G. Ueno, E. Aiyoshi, Particle swarm optimization: a numerical stability analysis and parameter adjustment based on swarm activity, IEEJ Trans. Electr. Electron. Eng. 3 (2008) 642–659.
[49] R.Y. Rubinstein, D.P. Kroese, Simulation and the Monte Carlo Method, John Wiley & Sons, 2016.