Automatika

Journal for Control, Measurement, Electronics, Computing and Communications

ISSN: 0005-1144 (Print) 1848-3380 (Online) Journal homepage: https://www.tandfonline.com/loi/taut20

UAV autonomous collision avoidance approach

Renke He, Ruixuan Wei & Qirui Zhang

AUTOMATIKA, 2017
VOL. 58, NO. 2, 195–204
https://doi.org/10.1080/00051144.2017.1388646

REGULAR PAPER

OPEN ACCESS

UAV autonomous collision avoidance approach

Renke He[a], Ruixuan Wei[b] and Qirui Zhang[b]

[a]Information and Navigation College, Air Force Engineering University, Xi'an City, China; [b]Aeronautics and Astronautics Engineering College, Air Force Engineering University, Xi'an City, China

ABSTRACT

The conventional sense-and-avoid collision avoidance mode of UAV (unmaned aerial vehicle) lacks applicability and timeliness in a multi-threat environment. In this paper, a new efficient collision avoidance approach for uncertain threat environments derived from the idea of autonomous mental development is proposed. The proposed collision avoidance pattern consists of a sensory layer, a logic layer and a development layer. The threat information is sensed using the sensory layer, and the path planning approach in the logical layer is applied to the output configuration of UAV. In the development phase, the developmental networks approach is used for online learning, training and updating the logical layer so as to form the sense–action mapping, which is stored as the "basic experience" for UAV executing the avoidance manoeuvre. In the implementation phase, the command is executed by matching the sensing information and action base. The simulation results show that the proposed approach has better timeliness compared to the conventional approaches.

ARTICLE HISTORY
Received 20 September 2016
Accepted 29 September 2017

KEYWORDS
UAV; development networks; collision avoidance

1. Introduction

With a wide application of aeronautical techniques in the field of economy and military, the number of Unmanned Aircraft Systems (UASs) increases significantly in airspace [1–3], while high-rise buildings, birds, complex air conditions, etc., have posed a significant challenge to air safety. The UASs must attain the same or higher security level compared with a manned aircraft [4]. Many researchers have focused on the UAV collision avoidance technique as a key for ensuring air safety. The Autonomous Mental Development (AMD) [5] method can simulate the mental development process of human-beings, which can also be applied in the decision-making of UAV collision avoidance in a complex threat environment.

Sensing and Avoidance (S&A) is a major pattern for threat avoidance. The aircraft's collision avoidance methods can be divided into the following categories: (1) the method of resolving the guidance law based on geometrical relationships [6–8]. This method calculates the guidance law of avoidance manoeuvre according to the relative distance, speed, acceleration, angle, etc., between the aircraft and threats, but it is difficult to apply in complex threat airspace. (2) The second method is based on real-time path planning [9–12]. With the development of research in this field, many path-planning approaches with significant improvement in timeliness have been proposed, such as artificial potential-field approaches, A* algorithm, artificial heuristic approaches and sampling-based path-planning approaches, etc., which can be applied to common path-planning problems in dynamic environments. However, the timeliness of such methods is still questionable due to the lack of computation efficiency and applicability in complex or high-dimensional dynamic environments. (3) The third approach is based on decision mechanism [13–15]. The decision-making approaches based on recognition-primed decision-making, Markov decision process and Bayesian decision theory are capable of implementing decision-making deduction considering real-time information. Thus, they have obvious advantages in terms of making threat avoidance strategy. However, the current researches mainly focus on how to make a better decision under certain rules [16], so it is flawed when dealing with complex situations beyond the known rules. Aiming at the current challenges in S&A techniques, it is necessary to introduce a learning mechanism to solve the above-mentioned problems [17].

Recently, the concept of AMD has been extended profoundly to robotic [18,19], particularly it is applied in visual development [20,21], language and behaviour learning techniques of robots [22,23]. The main idea is to simulate the mental development process of an infant, and the development of brain is realized by constant incremental learning through profound interaction with the environment, and storage of the learned knowledge through certain methods [5]. Unlike those approaches where the UAVs are programmed considering the mission, this idea proposed an approach for self-organization and incremental learning in a non-specific mission.

In this paper, a threat avoidance pattern that consists of a sensory layer, a logic layer and a development

CONTACT Renke He lnzrds@163.com

layer is proposed. The logic layer accomplishes the initial global configuration space planning by path-planning method and implements real-time local re-planning in the sensory region when the threat is approaching, where the threat information is detected by the sensor layer. Subsequently, the avoidance manoeuvre under normal threat conditions is adopted as the fundamental experience, and the mapping of sense–action is formed through online learning and training, which is based on the developmental networks (DNs) [24]. Therefore, the number of neurons constantly grows by learning, training and implementing the process. Moreover, the "experience" is also constantly accumulated while dealing with different types of threats. Finally, the collision avoidance development process of UAVs is accomplished.

2. Sensory layer and logic layer

2.1 Mapping model

Sense information is obtained by the sensory layer; configuration information of the UAV is the output of the logic layer. As shown in Figure 1, the local sensing and path-planning outputs are the inputs in the DN, the network map the inputs and manoeuvre behaviour bank, UAV behaviour is output in manoeuvre bank. In the network, the input (sense)–output (behaviour) relationship sample formed by the sensory layer and the logic layer, mapping would be implemented by adjusting the weights of neurons in the DN. In the beginning, the behaviour bank has nothing, which constantly stores the mapping of sense–behaviour as experience in the process of DN training.

The sensor in the sensory layer is an onboard lidar with a detection range of $\boldsymbol{d}(t)$, where t is the time (real-time data). Assume that the sensor is able to acquire all information regarding the environment, and $\boldsymbol{d}(t) = (d_1(t), d_2(t), \ldots, d_n(t))$, where $d_i(t)$ represents the shortest distance between the UAV and the threat in a certain direction, $\boldsymbol{v}(t)$ is the current velocity of the UAV, $\theta(t)$ represents the output heading angle of the path planning. The simplified configuration of the UAV can be set as

$$\boldsymbol{q}(t) = (x_t, y_t, \theta(t)) \tag{1}$$

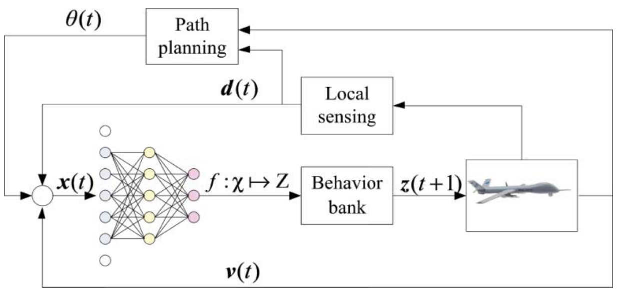


Figure 1. Collision avoidance frame.

where (x_t, y_t) is the current position of the UAV. The heading angle $\theta(t)$ is calculated as follows:

$$\theta(t) = \arctan(v_{yt}/v_{xt}) \tag{2}$$

where v_{xt}, v_{yt} are the velocity projection along x-axis and y-axis, respectively. Assuming that the value of velocity is constant, the next configuration is given as follows:

$$\boldsymbol{q}(t+1) = (x_{t+1}, y_{t+1}, \theta(t+1)) \tag{3}$$

The coordinates of the next position are

$$\begin{aligned} x_{t+1} &= x_t + v(t)\cos\theta(t)\cdot\Delta t \\ y_{t+1} &= y_t + v(t)\sin\theta(t)\cdot\Delta t \end{aligned} \tag{4}$$

The state of current environment is denoted as

$$\boldsymbol{x}(t) = (\boldsymbol{d}(t), \boldsymbol{v}(t), \theta(t)) \in \chi \tag{5}$$

The mapping of sense–behaviour can be represented as: $f:\chi \mapsto Z$, and the next output can be calculated as follows:

$$\boldsymbol{z}(t+1) = f(\boldsymbol{x}(t)) \tag{6}$$

2.2 The logic layer

The logic layer outputs the control command through path-planning method. In this paper, the collision avoidance is considered as a decision-making process of a local area. Making use of the dual-planning idea that contains local and external planning within the sensory range, the avoidance problem of unknown type of threats is solved.

Set $\boldsymbol{G}=(\boldsymbol{V}, \boldsymbol{C})$ as a grid map with weights, where $\boldsymbol{V}$ denotes the set of grid points, and $\boldsymbol{C}$ represents the cost space (the Euclidean distance is set as the heuristic function). Every grid point $g \in \boldsymbol{G}$ (except the boundary grid) is directly linked to eight neighbouring grids, denoted as Neighbour(g). The set of paths from the start node s to the target node d can be represented by

$$\boldsymbol{P}(\boldsymbol{G}; s, d) = \{(g_0, g_1, \ldots, g_N) \mid g_{n+1} \in \text{Neighbour}(g_n), \\ g_0 = g, g_N = d, n = 0, \ldots, N\}$$

And the cost of path is

$$\mathrm{c}(g_0, g_1, \ldots, g_N) = \sum_{n=0}^{N} \mathrm{c}(g_n) \tag{7}$$

$k(x)$ is defined as the path with minimum cost from a particular node x to a target node d, and is given as

$$k(x) = \begin{cases} 0 & x = Td \\ \min_{y\in\text{Neighbour}(x)}\{k(y) + \mathrm{c}(x)\} & \text{otherwise} \end{cases}$$

For a certain node x, the sensory range is set as $L(x)$, where $L(x) \in \boldsymbol{G}$. $\boldsymbol{B}(x)$ represents the margin region of

$L(x)$, but it does not belong to $L(x)$.

$$\boldsymbol{B}(x) = \{z \mid z \in \text{Neighbour}(L(x))\} \cap \{z \mid z \notin L(x)\}$$

Set x_{cur} as the current position of the UAV. For any $x \in L(x_{\text{cur}})$, the initial cost is set as $k_0(x)$. When a threat obstacle is sensed in the original path, the modified cost function is given as follows:

$$k(x) = \min\left\{k_0(x), \min_{y \in B(x_{\text{cur}})}[c(x,y) + k(y)]\right\} \tag{8}$$

where $c(x,y)$ represents the cost from x to a certain node y in the sense margin. Choosing the node y in margin with minimum cost, thus

$$y = \arg\min_{y \in B(x_{\text{cur}})}[c(x_{\text{cur}},y) + k(y)] \tag{9}$$

The path with minimum cost from x_{cur} to a node y in margin is the local planned path.

After accomplishing the local path planning, the configuration command is the output and is applied to satisfy the requirement of emergency collision avoidance. Then, the path from a node in the margin to the target node is calculated, which finally realizes the dual-planning. The computation cost of local paths in the sensory region is much less than those of the global paths. The delay caused by computation is minimum, and hence, it is more suitable for real-time threat avoidance.

3. Collision avoidance method

Based on the definition of sensory layer and logic layer mentioned above, the DN is used to connect both the layers in this section. The sample of knowledge is formed by mapping the sensory behaviour from the sensory layer to the logic layer. Then, it is trained using DN by adjusting the weights between neurons to form the mapping relationship and extend the bank to realize the development.

3.1 Algorithm of the developmental layer

As shown in Figure 2, the DN is composed of a sensory layer X, an internal brain area Y and a motor area Z.

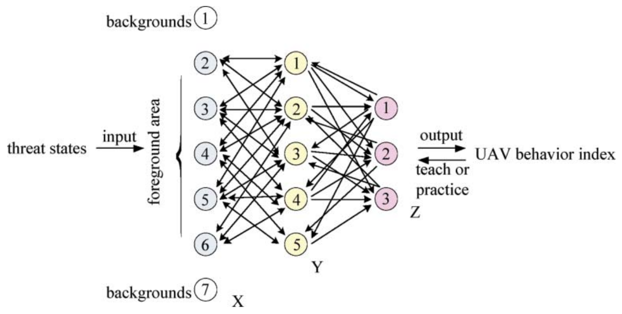


Figure 2. Developmental networks.

The sensory input and effector output can be connected through the hidden networks in the brain. The networks among the three layers are connected bi-directionally. The layers X and Y are connected by bottom–up synapses, while the layers Y and Z are connected by top–down ones. In the incremental learning phase, the effector layer serves as an input layer, conducting the learning process through interactions with the outside world. Otherwise, it is treated as an output layer.

The dimensions of X, Y, Z are determined according to the actual input and output of DN. In this paper, the inputs are parameters of threat, i.e. $\boldsymbol{v}(t)$, $\theta(t)$, $\boldsymbol{d}(t)$, the outputs determine which behaviour would be selected in the bank. The complete process is shown in Figure 1. The procedure can be explained as follows:

Initial state is at $t = 0$. For any area $A \in \{X, Y, Z\}$, initialize its renewal part $\boldsymbol{N} = (\boldsymbol{V}, \boldsymbol{G})$ and the responsive vector $\boldsymbol{r}$. $\boldsymbol{V}$ includes the weights of all synapses and $\boldsymbol{G}$ includes the ages of all neurons.

At $t = 1, 2, \ldots$, the task of collision avoidance is initiated in the simulation environment. The inputs are provided for network training; iterative computation is carried out in area A, and the corresponding function is as follows:

$$(\boldsymbol{r}', \boldsymbol{Q}') = f(\boldsymbol{b}, \boldsymbol{d}, \boldsymbol{Q}) \tag{10}$$

where $\boldsymbol{b}$ represents the bottom–up vector, $\boldsymbol{t}$ denotes the top–down vector and $\boldsymbol{r}'$ is the response vector. The update method is $\boldsymbol{N} \leftarrow \boldsymbol{N}'$ and $\boldsymbol{r} \leftarrow \boldsymbol{r}'$. The bottom–up and top–down vectors are in the Y-layer. The process mentioned above shows the weight update of the network nodes in the UAV collision avoidance process.

DN constantly performs the prediction for the next time Z. If the prediction value turns out to be a wrong match, it would be directly replaced by the measured value at the next moment in the learning process. That is to say, the wrong outputs from Z lead to the wrong behaviour of the UAV, and eventually, it will be corrected in the learning process. Hence, the right outputs are generated by path-planning modular.

For the weight vector $\boldsymbol{v} = (\boldsymbol{v_b}, \boldsymbol{v_t})$ in area A, its response vector is

$$\boldsymbol{r}(\boldsymbol{v_b}, \boldsymbol{b}, \boldsymbol{v_t}, \boldsymbol{t}) = \frac{\boldsymbol{v_b}}{\|\boldsymbol{v_b}\|}\frac{\boldsymbol{b}}{\|\boldsymbol{b}\|} + \frac{\boldsymbol{v_t}}{\|\boldsymbol{v_t}\|}\frac{\boldsymbol{t}}{\|\boldsymbol{t}\|} = \dot{\boldsymbol{v}}\dot{\boldsymbol{p}} \tag{11}$$

where $\dot{\boldsymbol{v}} = (\boldsymbol{v_b}/\|\boldsymbol{v_b}\|, \boldsymbol{b}/\|\boldsymbol{b}\|)$ is the response vector after normalization, $\dot{\boldsymbol{p}} = (\boldsymbol{b}/\|\boldsymbol{b}\|, \boldsymbol{t}/\|\boldsymbol{t}\|)$ is the output vector after normalization. The deviations of directions of $\dot{\boldsymbol{v}}$ and $\dot{\boldsymbol{p}}$ reflect the degree of matching.

The top-k competition mechanism is adopted among the neurons within area A. When $k = 1$, the winner neuron j is defined as

$$j = \arg\max_{1 \le i \le c} r(\boldsymbol{v_{bi}}, \boldsymbol{b}, \boldsymbol{v_{ti}}, \boldsymbol{t}) \tag{12}$$

Only the winner neurons can be activated and update their weights of synapses, while the other neurons would stay in their inhibitory states. The Hebbian learning algorithm is adopted in the learning process of neurons. For each activated neuron j, the synapse vector is updated by the following mechanism with the increment of an activating age:

$$v_j \leftarrow \omega_1(n_j)v_j + \omega_2(n_j)r_j\dot{\boldsymbol{p}} \tag{13}$$

where $\omega_2(n_j)$ is the learning rate of neuron j at activating age n_j, $\omega_1(n_j)$ is the maintaining rate and $\omega_1(n_j) + \omega_2(n_j) = 1$. It can be described briefly as

$$v_j^{(i)} = \frac{i-1}{i}v_j^{(i-1)} + \frac{1}{i}\dot{p}(t_i), i = 1, 2, \ldots, n_j \tag{14}$$

where t_i is the activation time of neuron j. After the learning phase, mapping is formed between the neurons in Y and Z layers based on the inputs from the sensory area. Therefore, the whole training of collision avoidance is completed.

3.2 Threat avoidance method

DN is capable of performing incremental learning and internal network mapping through interactions with the outside world. In the development phase, import the threat states from the sensory layer into the logic layer, and then execute the mapping of sensory input and output behaviours. Thus, neurons were activated to form an "experience" of collision avoidance. In the execution phase, the UAV behaviour is exported directly based on the mapping between sensing information and behaviour bank. Such a recurrent process of development and implementation can realize the growth of the behaviour bank. The flow-chart of the proposed algorithm is shown in Figure 3.

In the algorithm, first, N samples are provided as input for training. Then, ask for z_j, if it is not matching, the learning rate $\omega_2(n_j)$ is decreased. Decrease $\boldsymbol{v}_j$ and return for the next matching. If it is matching, the learning rate $\omega_2(n_j)$ is increased, and $\boldsymbol{v}_j$ and z_j' are updated. In the execution phase, based on the inputs from the sensory layer, search for the matching of z_j. If so, the UAV behaviour is output in the bank; otherwise, execute path planning to generate a new path to resolve the collision avoidance, and put the new result into the sample.

4. Simulation and analysis

To test the performance of the proposed method, different simple situations are designed to train the DN. The simulation is conducted in Matlab 7.0 using Windows XP, Intel Core i3, 3.3 GHz platform.

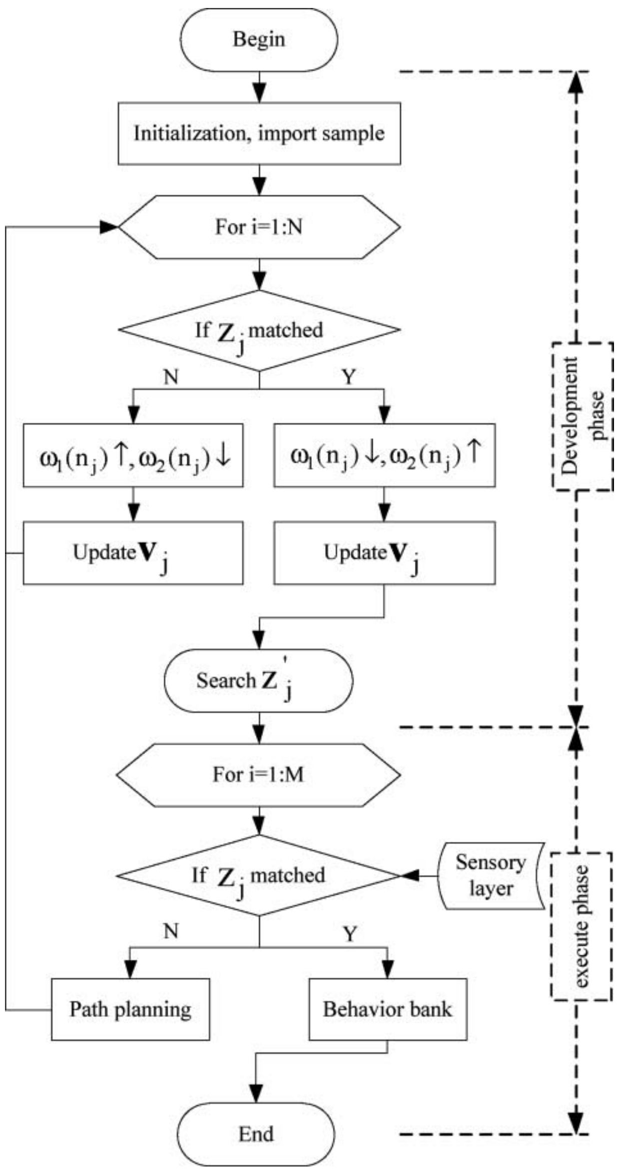


Figure 3. The algorithm flow-chart.

4.1 Training of the developmental networks

First, eight different simple situations are applied for training the basic avoidance behaviour of UAVs. The considered environment size is 10 km × 10 km, with the beginning and ending depots as (0.5,4) and (9,4), respectively. The threat model is

$$P = \begin{cases} \sqrt{(x - x_t) + (y - y_t)} & r < r_d \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

where r_d values are 1 km and 0.6 km, (x_t, y_t) is the location of the threat.

A different number of network neurons are set in the DN to determine the relationship between the activated age and the number of neurons.

As shown in Figure 4, the paths are exported by path planning according to different situations; then, online learning is implemented. The primary behaviour bank and samples are determined through

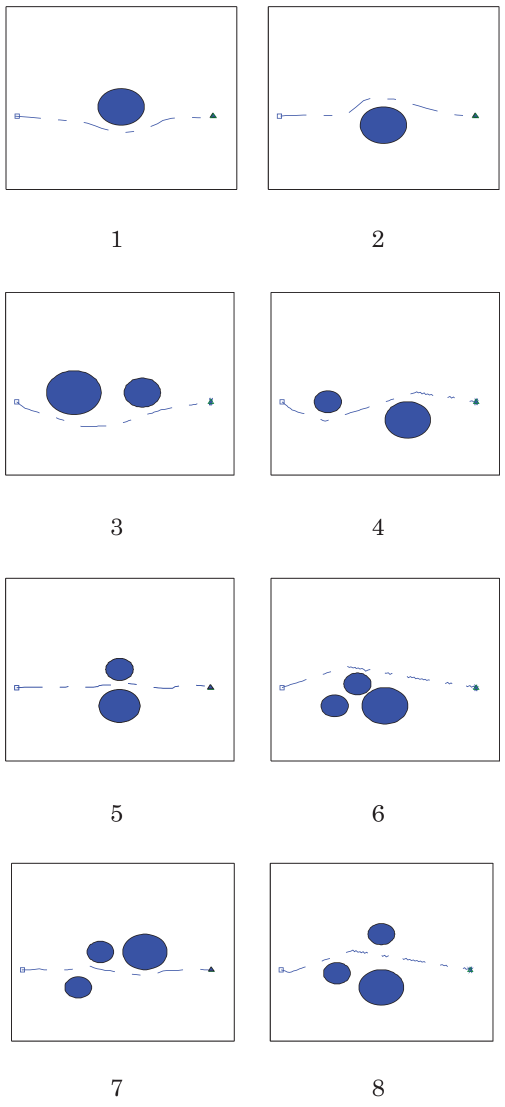


Figure 4. Training sample.

training, which forms the basic mapping of sense–behaviour.

In the development process, the number of neurons is closely related to the activation ages. As shown in Figure 5, under the top-k mechanism, when the number of neurons is relatively small, the activation time of neurons in the same area increases significantly, and the updating rate of synapsis is relatively high, which means that the mapping relationships of neurons within this area is relatively large. When the number of neurons is large, the probability that the neurons are to be activated in the current scenario increases, while the neurons in other areas are inhibitory at the same time. This shows that neurons in different areas correspond to different scenarios, and it is unnecessary to repeatedly activate a certain group of neurons.

4.2 Validation of the algorithm

The scenarios with known static threats, unknown pop-up static threats and dynamic threats are set to validate the proposed algorithm. The threat model is

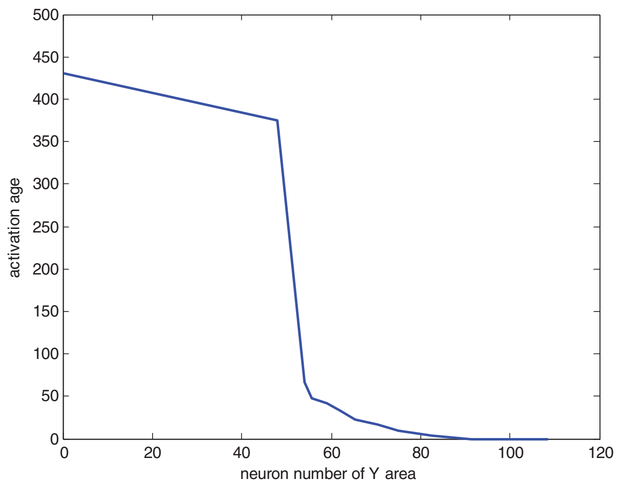


Figure 5. Developmental networks neurons.

considered as a hemisphere,

$$P = \begin{cases} \sqrt{(x - x_t) + (y - y_t) + (z - z_t)} & r < r_d \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

where r_d is 2.5 km, and (x_t, y_t, z_t) is the location of the threat.

The area size of environment is 20 km × 20 km × 4 km. The parameter settings of scenario-1 are shown in Table 1.

As shown in Figure 6, the hemispheres are threats, and the red lines are the flight paths of the UAV. At t = 25 s, the UAV was avoiding the first static threat, and the dynamic threat was moving at the same time. At t = 50 s, a threat suddenly turned up in the sensory region of the UAV. Thus, the UAV tried to match the state of the threat with its behaviour bank to avoid collision. At t = 75 s, a dynamic threat ran into the sensory region; the UAV started the third avoidance action. The heading angle of the UAV in scenario-1 is shown in Figure 7. This paper reserved the track of the dynamic threat to observe the threat situations and the UAV's behaviour of avoidance. The result shows that the proposed algorithm has good adaptability to such kinds of environment.

In scenario-2, nine static threats and two dynamic threats are set to test the real-time avoidance ability of the proposed approach under multi-threat conditions. The threat model is the same as the one in Section 4.1, and the r_d is set randomly. The size of the environment is 20 km × 20 km. Parameter settings are shown in Table 2.

As shown in Figure 8, the UAV encountered with threat1 and threat2 during the flight. The UAV and

Table 1. Initial parameters in scenario-1.

	UAV	Static threat	Pop-up threat	Dynamic threat
Starting point	(1,2,0)	(4,5,0)	(10,6,0)	(11,15,0)
Velocity (m/s)	200	0	0	80

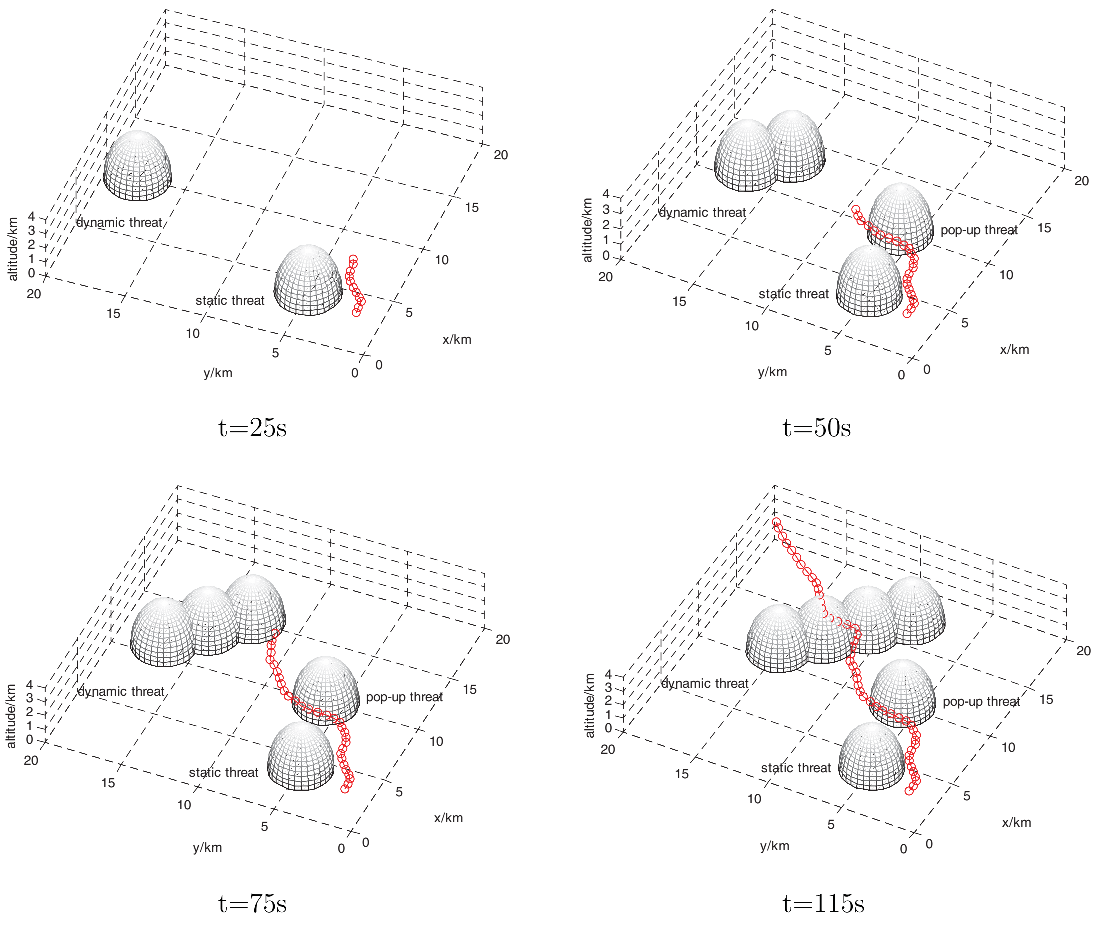


Figure 6. Threat avoidance process.

threats started to fly at the same time, but their velocities are quite different. At t = 80 s, threat1 came into the range of perception, the UAV tried to match the state of the threat with its behaviour bank and output the avoidance command for the first time. At t = 248 s, threat2 came into the sensory range executing the same process again. As shown in Figure 8, in order to show the time parameter, we set the z-axis to represent time, and removed some of the threat images which interrupt the sight of view. Figure 9(a) shows the

Table 2. Initial parameters in scenario-2.

	Starting point	Velocity (m/s)	Beginning time (s)	Encountered time (s)
UAV	(2,2)	175	0	
Dynamic threat1	(5,20)	124	0	80
Dynamic threat2	(40,47)	68	0	248

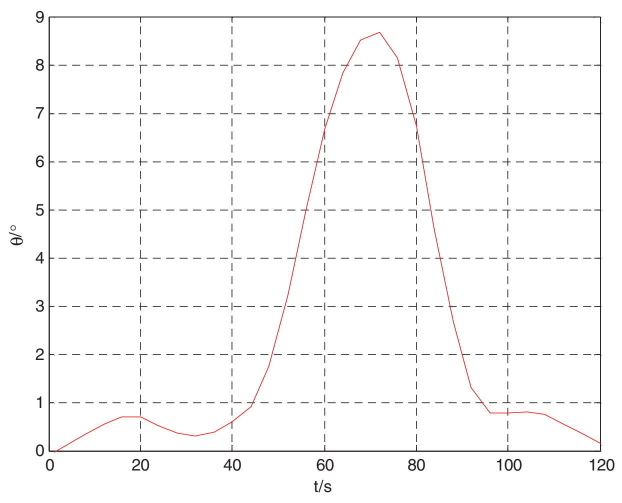


Figure 7. Heading angle in scenario-1.

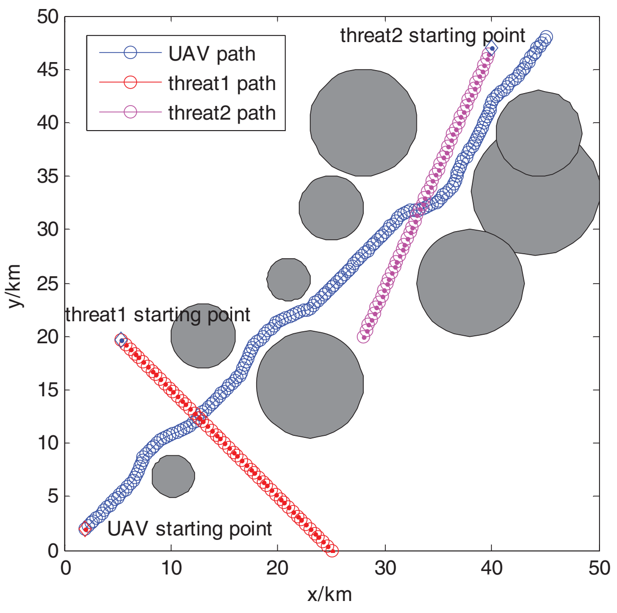
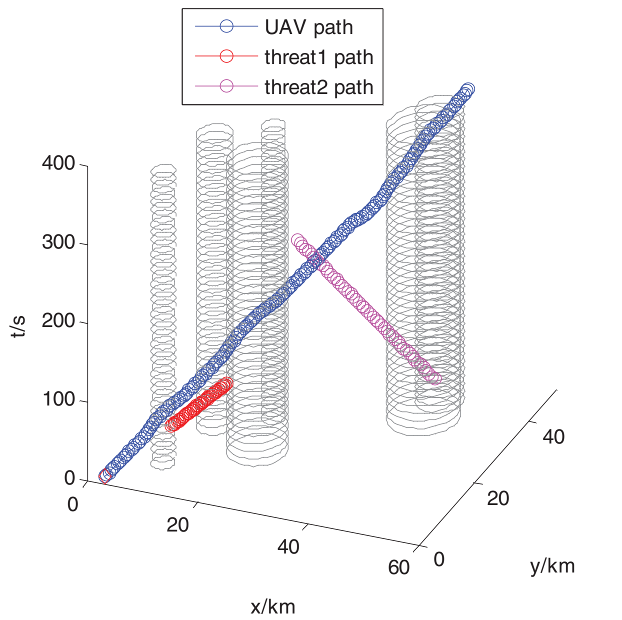
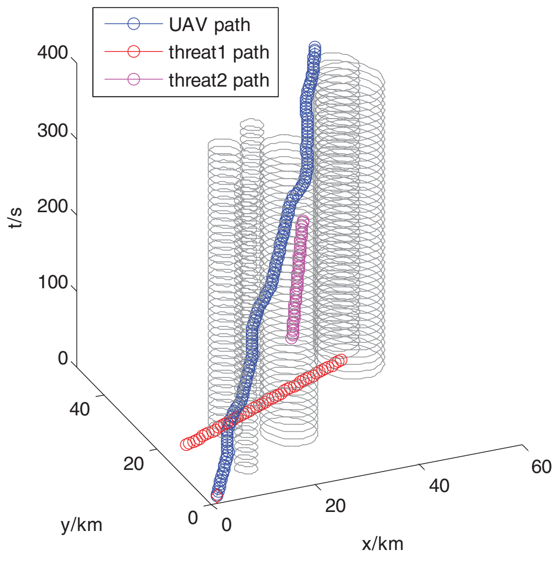


Figure 8. Scenario-2.



(a) left view



(b) right view

Figure 9. Scenario-2 simulation in 3D.

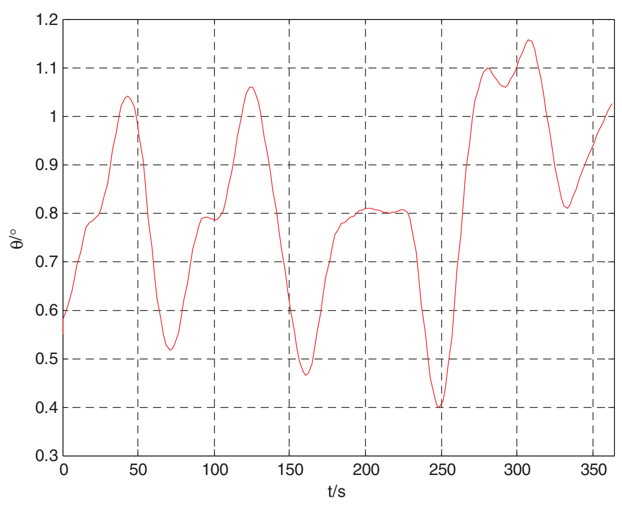


Figure 10. Heading angle in scenario-2.

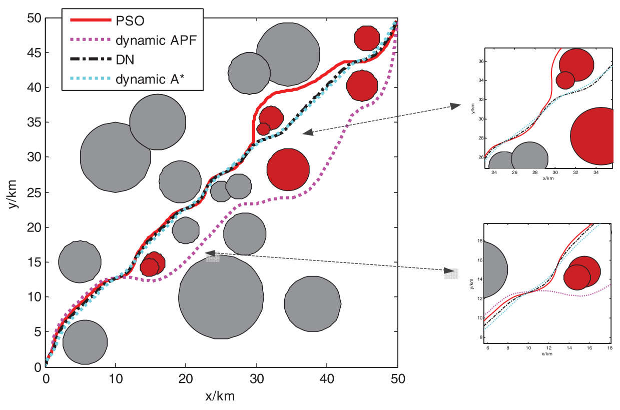
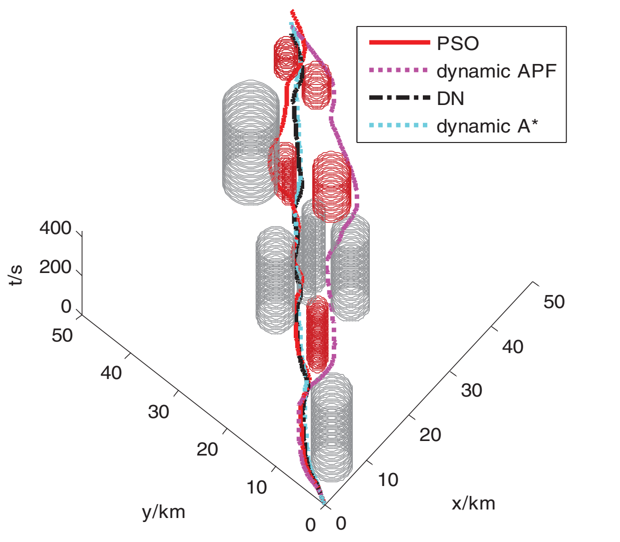
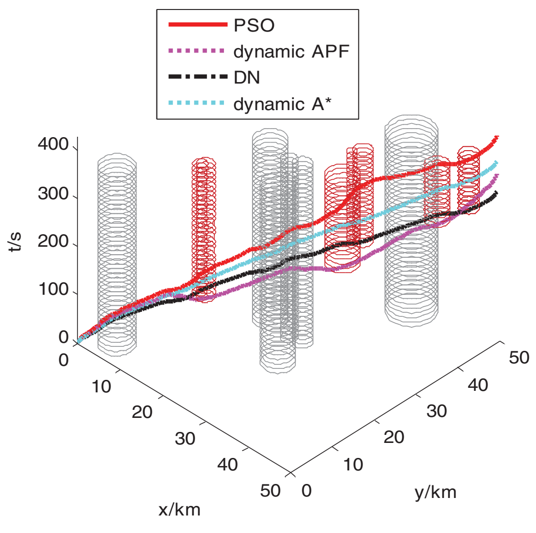


Figure 11. Collision avoidance comparison.

process of avoiding threat1. It can be seen from the z-axis that they have not intersected in time. Figure 9(b) shows the similar process for threat2. The heading angle of the UAV in scenario-2 is shown in Figure 10.



(a) Oblique view



(b) left view

Figure 12. Collision avoidance comparison in 3D.

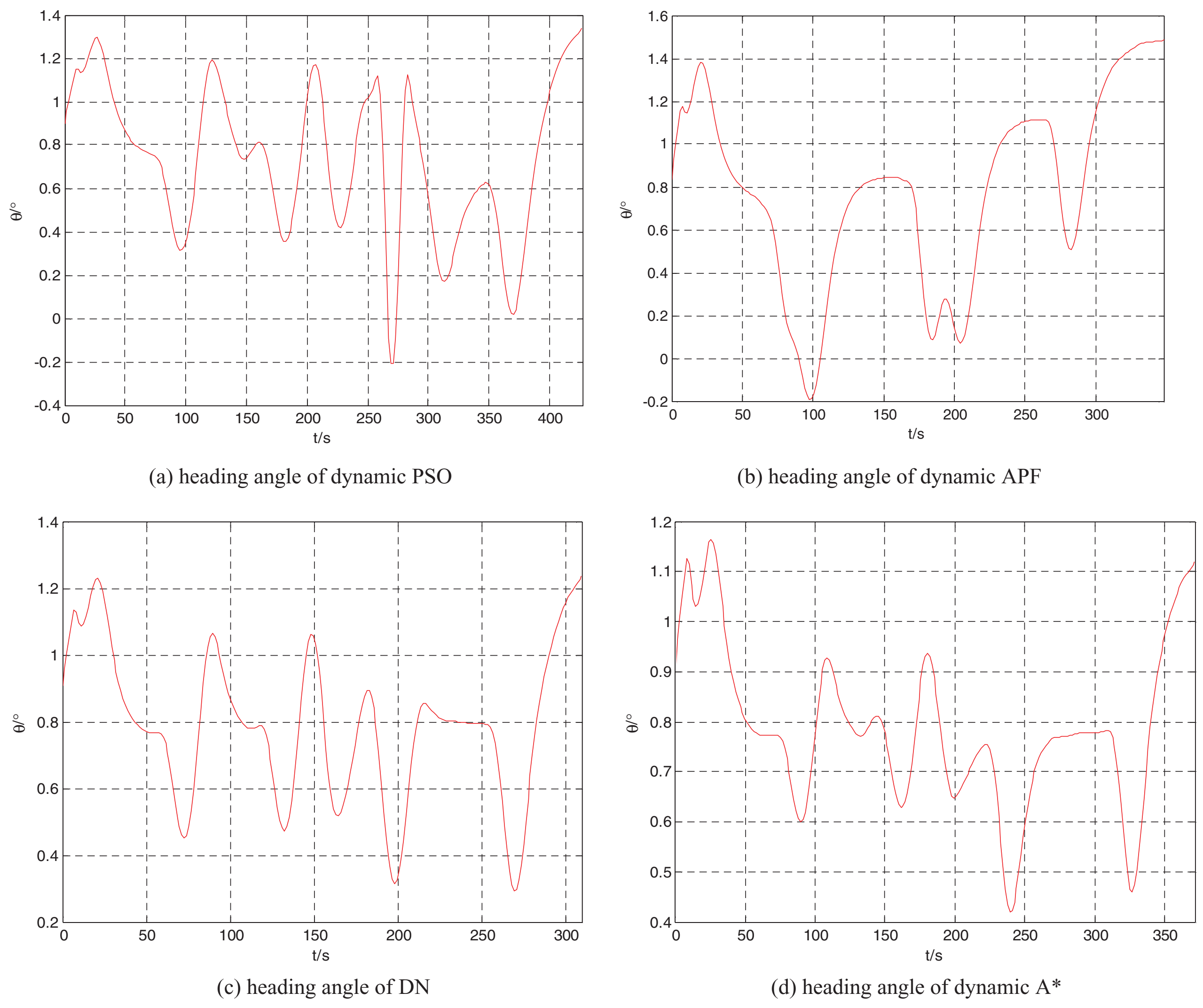


Figure 13. Heading angle of algorithms.

Results of different scenarios validate the feasibility of the proposed algorithm.

4.3 Performance comparison

To show the performance differences, the proposed algorithm was compared with dynamic Particle Swarm Optimization (PSO) [25], dynamic A* algorithm [26] and dynamic Artificial Potential Field (APF) [27] for the same cases. The environment's size is considered as 50 km × 50 km for this case. A total of 13 static threats and 7 pop-up threats are set. The threat model is the same as the one in Section 4.1, and the r_d value is set randomly. The starting point is (0,0) and the end point is (50,50). The results are shown in the following.

As shown in Figure 11, the static threats and the pop-up threats appeared at different time. The paths generated by the four algorithms are entirely different. The three-dimensional oblique view of Figure 11 is shown in Figure 12(a), whereas Figure 12(b) indicates the corresponding left view. To illustrate the paths clearly, some of the threat images which interrupt the view have been removed. It can be seen that in the initial phase, the computational differences between four paths are minor. After encountering the first pop-up threat, the difference began to increase, and the directions of different paths turn out to deviate. With the change in threats, the difference of computational time and path's direction becomes more and more apparent. Figure 13 shows the heading angle of the UAV in different algorithms. The simulation results are summarized in Table 3.

As shown in Table 3, the shortest path was generated by the dynamic A^* algorithm, while the longest path was generated by dynamic APF. The path generated by the DN is a bit longer than that of the dynamic A^* method. The shortest computation time is for the DN algorithm; dynamic APF's computation time is longer than DN's algorithm, and that for dynamic PSO is the longest one. The proposed algorithm is fine enough in terms of calculation time and the cost of

Table 3. Simulation results.

Algorithm	Dynamic PSO	Dynamic APF	DN	Dynamic A^*
Calculation time (s)	427	348	310	372
Path length (km)	75.4	77.6	72.3	71.2

path's length, and it can adapt to a workspace that exists pop-up threats by making avoidance behaviour in time.

5. Conclusions

In this paper, the concept of AMD is introduced. Also, the UAV collision avoidance method based on DN is proposed by combining the collision avoidance strategy of the UAV with the DN. Unlike the conventional sense–avoidance methods, the proposed method enables the UAV to obtain fundamental experience of collision avoidance by learning and training, as well as the online expansion of knowledge. For different types of threats, neurons in the DN realize growth and development by adjustment and updation of the synapses' weights. The simulation results show that the proposed method based on DN has better timeliness in threat avoidance.

The disadvantage of the proposed method is that it is unable to confirm the area directly where neurons need to be activated, which will affect the computation time to a considerable extent. The future work will focus on the improvement of the sensory layer in the DN by classifying different types of threats when sensing information from the outside world, which can be applied to determine certain areas of activated neurons directly.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This study was supported by the National Natural Science Foundation of China [grant number 61573373], [grant number 61673392], [grant number 61703420].

References

[1] Gundlach J. Designing unmanned aircraft systems: a comprehensive approach. Reston (VA): American Institute of Aeronautics and Astronautics Reston; 2012.
[2] Angelov P. Sense and avoid in UAS: research and applications. Chichester (UK): Wiley; 2012.
[3] Valavanis KP, Vachtsevanos GJ. Handbook of unmanned aerial vehicles. Dordrecht: Springer; 2015.
[4] Prats X, Delgado L, Ramírez J, et al. Requirements, issues, and challenges for sense and avoid in unmanned aircraft systems. J Aircraft. 2012;49(3): 677–687.
[5] Weng J, Mcclelland J, Pentland A, et al. Artificial intelligence – autonomous mental development by robots and animals. Science. 2001;291(5504): 599–600.
[6] Peng J, Luo W, Liu W, et al. A suboptimal and analytical solution to mobile robot trajectory generation amidst moving obstacles. Auton Robot. 2015; 39(1): 1–23.
[7] Melega M, Lazarus S, Savvaris A, et al. Multiple threats sense and avoid algorithm for static and dynamic obstacles. J Intell Robot Syst. 2015;77:215–228.
[8] Yang X, Alvarez LM, Bruggemann T. A 3D collision avoidance strategy for UAVs in a non-cooperative environment. J Intell Robot Syst. 2013;70:315–327.
[9] Davis J, Perhinschi M, Wilburn B, et al. Development of a modified Voronoi algorithm for UAV path planning and obstacle avoidance. In: AIAA Guidance, Navigation, and Control Conference; 2012 Aug 13–16; Minneapolis, MN. Reston (VA): American Institute of Aeronautics and Astronautics; 2012. p. 1–12.
[10] Zhou S, Wang J, Jin Y. Route planning for unmanned aircraft based on ant colony optimization and Voronoi diagram. Second International Conference on Intelligent System Design and Engineering Application. 2012 Jan 6–7; Sanya, China. New York: IEEE; 2012. p. 732–735.
[11] Lin L, Goodrich MA. Hierarchical heuristic search using a Gaussian mixture model for UAV coverage planning. IEEE Trans Cybern. 2014;44(12): 2532–2544.
[12] Israelsen J, Beall M, Bareiss D, et al. Automatic collision avoidance for manually tele-operated unmanned aerial vehicles. In: Proceedings of the IEEE International Conference on Robotics & Automation; 2014 May 31–Jun 7; Hong Kong. New York: IEEE; 2014. p. 6638–6643.
[13] Mikaelian T, Rhodes DH, Nightingale DJ, et al. A logical approach to real options identification with application to UAV systems. IEEE Trans Syst Man Cybern A. 2012;42(1): 32–47.
[14] Bertuccelli LF, Wu A, How JP. Robust adaptive Markov decision processes:planning with model uncertainty. IEEE Control Syst Mag. 2012;32(5): 96–109.
[15] Bethke B, How JP, Vian J. Group health management of UAV teams with applications to persistent surveillance. In: Proceedings of the American Control Conference; 2008 Jun 11–13; Seattle, WA. New York: IEEE; 2008. p. 3145–3150.
[16] Vachtsevanos G, Tang L, Drozeski G, et al. From mission planning to flight control of unmanned aerial vehicles:strategies and implementation tools. Annu Rev Control. 2005;29(1): 101–115.
[17] Yu X, Zhang Y. Sense and avoid technologies with applications to unmanned aircraft systems: review and prospects. Prog Aerosp Sci. 2015;74:152–166.
[18] Asada M, Hosoda K, Kuniyoshi Y, et al. Cognitive developmental robotics: a survey. IEEE Trans Auton Mental Dev. 2009;1(1): 12–34.
[19] Stoytchev A. Some basic principles of developmental robotics. IEEE Trans Auton Mental Dev. 2009;1(2): 122–130.
[20] Solgi M, Weng J. Developmental stereo: emergence of disparity preference in models of the visual cortex. IEEE Trans Auton Mental Dev. 2009;27(2): 123–127.
[21] Paslaski S, Vandam C, Weng J. Modeling dopamine and serotonin systems in a visual recognition network. In: International Joint Conference on Neural Networks; 2011 Jul 31–Aug 5; San Jose, CA. New York: IEEE; 2011. p. 3016–3023.
[22] Murata S, Namikawa J, Arie H. Learning to reproduce fluctuating time series by inferring their time-dependent stochastic properties: application in robot learning via tutoring. IEEE Trans Auton Mental Dev. 2013;5(4): 298–310.
[23] Althaus N, Mareschal D. Modeling cross-modal interactions in early word learning. IEEE Trans Auton Mental Dev. 2013;5(4): 288–297.
[24] Weng J. Brain-like networks logically reason and optimally generalize. In: International Joint Conference on

Neural Networks; 2011 Jul 31–Aug 5; San Jose, CA. New York: IEEE; 2011. p. 2983–2990.

[25] Karimi J, Pourtakdoust SH. Optimal maneuver-based motion planning over terrain and threats using a dynamic hybrid PSO algorithm. Aerosp Sci Technol. 2013;26(1):60–71.

[26] Likhachev M, Ferguson DI, Gordon GJ, et al. Anytime dynamic A*: an anytime, replanning algorithm. In: International Conference on Automated Planning and Scheduling; 2005 Jun 5–10; Monterey, CA. Reston (VA): AIAA press; 2005. p. 262–271.

[27] Latombe JC. Robot motion planning. Boston, MA: Springer; 1991. Chapter 7, Potential field methods; p. 295–355.