Intelligent and Robotic Systems manuscript No.
(will be inserted by the editor)

A Decentralized Fuzzy Learning Algorithm for Pursuit-Evasion Differential Games with Superior Evaders

Mostafa D. Awheda · Howard M. Schwartz

Received: date / Accepted: date

Abstract In this paper, we consider a multi-pursuer single-superior-evader pursuit-evasion game where the evader has a speed that is similar to or higher than the speed of each pursuer. A new fuzzy reinforcement learning algorithm is proposed in this work. The proposed algorithm uses the well-known Apollonius circle mechanism to define the capture region of the learning pursuer based on its location and the location of the superior evader. The proposed algorithm uses the Apollonius circle with a developed formation control approach in the tuning mechanism of the fuzzy logic controller (FLC) of the learning pursuer so that one or some of the learning pursuers can capture the superior evader. The formation control mechanism used by the proposed algorithm guarantees that the pursuers are distributed around the superior evader in order to avoid collision between pursuers. The formation control mechanism used by the proposed algorithm also makes the Apollonius circles of each two adjacent pursuers intersect or be at least tangent to each other so that the capture of the superior evader can occur. The proposed algorithm is a decentralized algorithm as no communication among the pursuers is required. The only information the proposed algorithm requires is the position and the speed of the superior evader. The proposed algorithm is used to learn different multi-pursuer single-superior-evader pursuit-evasion games. The simulation results show the effectiveness of the proposed algorithm.

Keywords Fuzzy control · Reinforcement learning · Pursuit-evasion differential games · Apollonius circles.

1 Introduction

The prevention, detection and response to intruders who want to cross a perimeter can be a security concern for individuals, companies or countries. Due to the development of robotic systems in recent years, there has been an increase in demand for autonomous guarding robots based security applications. The objective of this kind of security applications is to prevent intruders from performing harmful actions against some persons or territories of strategic importance such as important infrastructure, transportations and international borders [9]. Pursuit-evasion games and guarding a territory games have been widely used to facilitate such type of security applications. In this work, we are interested in pursuit-evasion games because of their extensive applications in the real world such as surveillance and tracking, search and rescue, locating and capturing hostile intruders, localizing and neutralizing environmental threads, and collision avoiding systems in intelligent transportation systems [2,26,28,32,33].

Pursuit-evasion games have been widely studied in the literature [4,5,10–38,43]. In a pursuit-evasion game, the pursuer wants to capture the evader in a short time, and the evader wants to escape from the

Mostafa D. Awheda
Department of Systems and Computer Engineering
Carleton University
1125 Colonel By Drive, Ottawa, ON, K1S 5B6, Canada
E-mail: mawheda@sce.carleton.ca

Howard M. Schwartz
Department of Systems and Computer Engineering
Carleton University
1125 Colonel By Drive, Ottawa, ON, K1S 5B6, Canada
E-mail: schwartz@sce.carleton.ca

pursuer or prolong the capture time [28,29]. The pursuit-evasion game was first proposed in [45] as a one-pursuer one-evader game. However, in recent years, multi-player pursuit-evasion games have received more attention [28,32,33]. One of the approaches used in the literature to optimize the performance of the pursuers in multi-player pursuit-evasion games is the hierarchical decomposition framework. This approach decomposes the multi-player pursuit-evasion game into small games between pursuers and evaders. That is, one-pursuer one-evader games, or multi-pursuer single-superior-evader games [25,32]. We mean by the superior evader that the evader that has a speed which is similar to or higher than the speed of all pursuers. Cooperation among pursuers is another approach that is used in the literature to optimize the performance of the pursuers in multi-player pursuit-evasion games and to make them act as a whole to perform their mission task. One of the popular techniques used to facilitate cooperation among pursuers in a multi-player pursuit-evasion game is the formation control [25], which shapes the relative position and the orientation of pursuers.

1.1 **Related work**

A number of articles [4,5,12,19,22–24] investigated pursuit-evasion games with slow evaders, where the capture of the evader is always guaranteed. However, in real-world applications, evaders may run with speed similar to or higher than the speed of pursuers. In such cases, the capture of the faster (superior) evader may require more than one pursuer. This has led many researchers to investigate multi-pursuer single-superior-evader pursuit-evasion games and propose a number of different techniques so that one or some of the pursuers can capture the superior evader [25–28,33–35,37,38]. These techniques use different mechanisms such as hierarchical decomposition approaches and formation control approaches to tune the pursuers. However, all these techniques are deterministic approaches and involve no learning in the tuning mechanisms of the pursuers. In real-world applications, pursuers may need to adapt to changing environments.

There is no much work in the literature addressing the learning in multi-player pursuit-evasion differential games with superior evaders. In [29,30], the authors proposed different learning techniques for multi-player pursuit-evasion games. However, the learning techniques proposed in [29,30] are not suitable for pursuit-evasion differential games because they only work with games that have discrete state and action spaces. In addition, the algorithm proposed in [29] is only applicable for multi-player pursuit-evasion games with slow evaders (i.e. evaders with speed slower than the speed of the pursuers). The only other article to consider the use of learning in multi-player pursuit-evasion differential games with superior evaders is proposed in [36]. The authors of this article propose their algorithm based on three basic behaviors of each pursuer (namely Move-to-goal, Avoid-obstacle and Hunting), and learning is only involved in one behavior (Hunting). The authors of this article structure the learning part of their algorithm based on the Q-learning algorithm. Because the Q-learning algorithm requires discrete state and action spaces, the authors used a state-space reduction mechanism to deal with the continuous spaces. However, in many real-world applications, a priori discretization of the action space may not be useful [39]. In addition, using a coarse discretization of the state or action space may lead to a poor performance [40–42].

1.2 **Main contribution**

In this work, we consider multi-pursuer single-superior-evader pursuit-evasion differential games. Our objective is to make the pursuers in the multi-pursuer single-superior-evader pursuit-evasion differential game learn their strategies so that one or some of the learning pursuers can capture the superior evader. We develop a formation control mechanism and use it from the learning point of view to develop a decentralized learning algorithm that can be used in multi-pursuer single-superior-evader pursuit-evasion differential games. This work is the first piece of work that presents a decentralized learning algorithm to capture a superior evader in a pursuit-evasion differential game without using any type of discretization for the state or action spaces. We establish a fast and robust learning algorithm that directly incorporates the idea of Apollonius circles and the developed formation control mechanism in the reward function of the learning algorithm.

The proposed algorithm uses the well-known Apollonius circle as a mechanism to define the capture region of the learning pursuer based on its location and the location of the superior evader. Based on the defined capture region, the proposed algorithm uses a developed formation control approach to construct the reward function of the learning pursuer. The proposed algorithm uses this reward function to tune the FLC of the learning pursuer by the residual gradient fuzzy actor critic learning (RGFACL) algorithm proposed in [4]. The developed formation control approach used by the proposed algorithm guarantees that the pursuers are distributed around the superior evader in order to avoid collision between pursuers. It also makes the Apollonius circles of each two adjacent pursuers intersect or be at least tangent to each other so that the capture of the superior evader can occur. The proposed algorithm is a decentralized algorithm as no communication among the pursuers is required. The only information the proposed algorithm requires is the position and the speed of the superior evader. We evaluate the proposed algorithm over a number of multi-pursuer single-superior-evader pursuit-evasion differential games, and the results validate the proposed algorithm.

This paper is organized as follows: Preliminary concepts and the problem definition are presented in Section 2. Section 3 presents the RGFACL algorithm. The proposed algorithm is introduced in Section 4. The simulation and results are presented in Section 5.

2 Preliminary Concepts and Problem Definition

2.1 Fuzzy Inference Systems

The fuzzy inference systems (FISs) used in this work are zero-order Takagi-Sugeno (TS) FISs [44] with constant consequents. Each fuzzy system consists of L rules. The inputs of each rule are n fuzzy variables; whereas the consequent of each rule is a constant number. Each rule l $(l = 1, ..., L)$ has the following form,

$$R_l : \text{IF } s_1 \text{ is } F_1^l, \; \ldots, \text{ and } s_n \text{ is } F_n^l \qquad \text{THEN } z_l \; = \; k_l \tag{1}$$

where s_i, $(i = 1, ..., n)$, is the ith input state variable of the fuzzy system, n is the number of the input state variables, and F_i^l is the linguistic value of the input s_i at the rule l. Each input s_i has h membership functions. The variable z_l represents the output variable of the rule l, and k_l is a constant that describes the consequent parameter of the rule l. In this work, Gaussian membership functions are used and each membership function (MF) is defined as follows,

$$\mu^{F_i^l}(s_i) = \exp\Big(- (\frac{s_i - m}{\sigma})^2 \Big) \tag{2}$$

where σ and m are the standard deviation and the mean, respectively.

In each FIS used in this work, the total number of the standard deviations of the membership functions of its inputs is defined as H, where $H = n \times h$. In addition, the total number of the means of the membership functions of its inputs is H. Thus, for each FIS used in this work, the standard deviations and the means of the membership functions of the inputs are defined, respectively, as σ_j and m_j, where $j = 1, ..., H$. We define the set of the parameters of the membership functions of each input, $\Omega(s_i)$, as follows,

$$\begin{gathered}\Omega(s_1) = \{(\sigma_1, m_1), (\sigma_2, m_2), ..., (\sigma_h, m_h)\}\\ \Omega(s_2) = \{(\sigma_{h+1}, m_{h+1}), (\sigma_{h+2}, m_{h+2}), ..., (\sigma_{2h}, m_{2h})\}\\ .\\ .\\ .\\ \Omega(s_n) = \{(\sigma_{(n-1)h+1}, m_{(n-1)h+1}), (\sigma_{(n-1)h+2}, m_{(n-1)h+2}), ..., (\sigma_H, m_H)\}\end{gathered} \tag{3}$$

The output of the fuzzy system is given by the following equation when we use the product inference engine with singleton fuzzifier and center-average defuzzifier [1].

$$Z(\mathbf{s}) = \frac{\sum_{l=1}^{L}\left[\left(\prod_{i=1}^{n}\mu^{F_i^l}(s_i)\right)k_l\right]}{\sum_{l=1}^{L}\left(\prod_{i=1}^{n}\mu^{F_i^l}(s_i)\right)} = \sum_{l=1}^{L}\Phi_l(\mathbf{s})k_l \tag{4}$$

where $\mathbf{s} = (s_1, ..., s_n)$ is the state vector, $\mu^{F_i^l}$ describes the membership value of the input state variable s_i in the rule l, and $\Phi_l(\mathbf{s})$ is the normalized activation degree (normalized firing strength) of the rule l at the state $\mathbf{s}$ and is defined as follows:

$$\Phi_l(\mathbf{s}) = \frac{\prod_{i=1}^{n}\mu^{F_i^l}(s_i)}{\sum_{l=1}^{L}\left(\prod_{i=1}^{n}\mu^{F_i^l}(s_i)\right)} = \frac{\omega_l(\mathbf{s})}{\sum_{l=1}^{L}\omega_l(\mathbf{s})} \tag{5}$$

where $\omega_l(\mathbf{s})$ is the firing strength of the rule l at the state $\mathbf{s}$ and it is defined as follows,

$$\omega_l(\mathbf{s}) = \prod_{i=1}^{n}\mu^{F_i^l}(s_i) \tag{6}$$

We define the set of the parameters of each firing strength of each rule in each FIS, $\Omega(\omega_l)$, as follows,

$$\begin{aligned}
\Omega(\omega_1) &= \{(\sigma_1, m_1), (\sigma_{h+1}, m_{h+1}), ..., (\sigma_{(n-1)h+1}, m_{(n-1)h+1})\}\\
\Omega(\omega_2) &= \{(\sigma_1, m_1), (\sigma_{h+1}, m_{h+1}), ..., (\sigma_{(n-1)h+2}, m_{(n-1)h+2})\}\\
&.\\
&.\\
&.\\
\Omega(\omega_h) &= \{(\sigma_1, m_1), (\sigma_{h+1}, m_{h+1}), ..., (\sigma_H, m_H)\}\\
\Omega(\omega_{h+1}) &= \{(\sigma_1, m_1), (\sigma_{h+2}, m_{h+2}), ..., (\sigma_{(n-1)h+1}, m_{(n-1)h+1})\}\\
&.\\
&.\\
&.\\
\Omega(\omega_L) &= \{(\sigma_h, m_h), (\sigma_{2h}, m_{2h}), ..., (\sigma_H, m_H)\}
\end{aligned} \tag{7}$$

2.2 The Pursuit-Evasion Differential Game

The pursuit-evasion game is defined as a differential game [45]. In this game, the goal of the pursuer is to capture the evader in a minimum time whereas the goal of the evader is to escape from the pursuer. The model of the pursuit-evasion differential game is shown in Fig. (1). The equations of motion of the pursuer and evader robots are given as follows [46,47],

$$\begin{aligned}
\dot{x}_\kappa &= V_\kappa\cos(\theta_\kappa)\\
\dot{y}_\kappa &= V_\kappa\sin(\theta_\kappa)\\
\dot{\theta}_\kappa &= \tfrac{V_\kappa}{L_\kappa}\tan(u_\kappa)
\end{aligned} \tag{8}$$

where κ represents both the pursuer "p" and the evader "e", (x_κ, y_κ) represents the position of the robot κ, θ_κ represents the orientation of the robot κ, L_κ is the wheelbase of the robot κ, u_κ is the robot κ's steering angle, $u_\kappa \in [-u_{\kappa_{max}}, u_{\kappa_{max}}]$, and V_κ is the robot κ's speed.

The pursuer captures the evader if the distance d between them is less than the capture radius d_c, where the distance d is defined as follows

$$d = \sqrt{(x_e - x_p)^2 + (y_e - y_p)^2} \tag{9}$$

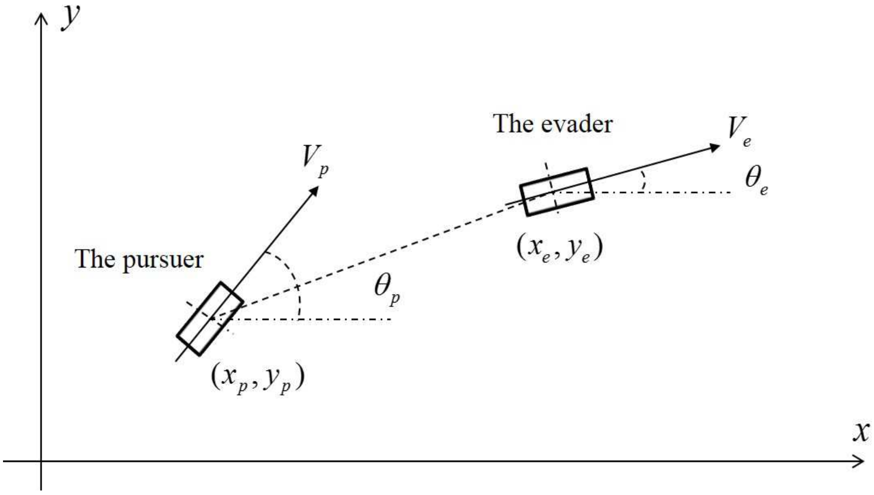

Fig. 1 Pursuit-evasion model

2.3 **Apollonius Circles**

The Apollonius circle was first presented in [45]. Consider a single pursuit-evasion differential game with a pursuer P that has a constant speed V_p and a superior evader E that has a constant speed V_e, where $V_e > V_p$. Let us also consider that the positions of the pursuer P and the evader E are (x_p, y_p) and (x_e, y_e), respectively. Fig. (2) shows the Apollonius circle created by the pursuer P and the evader E. The pursuer captures the evader if the distance between them is less than a small amount d_c, $\|(x_p, y_p) - (x_e, y_e)\| \leq d_c$. Fig. (2) shows the capture region of the pursuer and the evasion region of the evader. If the evader moves in a direction that is inside the capture region of the pursuer (the region covered by the angle $\angle AEB$ in Fig. (2)), the capture of the evader by the pursuer will be guaranteed if the pursuer is well tuned. On the other hand, if the evader moves into its evasion region (the region that is not covered by the angle $\angle AEB$ in Fig. (2)), the evader will be able to escape from the pursuer if the evader is well tuned.

The point C, (x_c, y_c), in Fig. (2) is a random point that is located on the Apollonius circle and such that $\gamma = \frac{\|\overrightarrow{PC}\|}{\|\overrightarrow{EC}\|} = \frac{V_p}{V_e}$, and $\gamma < 1$. The centre of the Apollonius circle, O_{AC}, and its radius, R_{AC}, can be defined as follows [27,35],

$$O_{AC} = \Big(\frac{x_p - \gamma^2 x_e}{1-\gamma^2}, \frac{y_p - \gamma^2 y_e}{1-\gamma^2}\Big) \tag{10}$$

$$R_{AC} = \frac{\gamma\sqrt{\left(x_p - x_e\right)^2 + \left(y_p - y_e\right)^2}}{1-\gamma^2} \tag{11}$$

As shown in Fig. (2), if the evader moves into the capture region of the pursuer (towards point C for example) and gets captured by the pursuer, then by the triangle property, the capture condition is defined as follows [25,26,33],

$$\frac{\sin\alpha}{\sin\beta} = \frac{V_e}{V_p} \tag{12}$$

where α is the absolute value of the angle difference between the pursuer's direction and its line of sight (LOS) to the evader, and β is the absolute value of the angle difference between the evader's direction and its LOS to the pursuer.

Thus,

$$\beta = \sin^{-1}\Big(\frac{V_p}{V_e}\sin\alpha\Big) \tag{13}$$

Or,

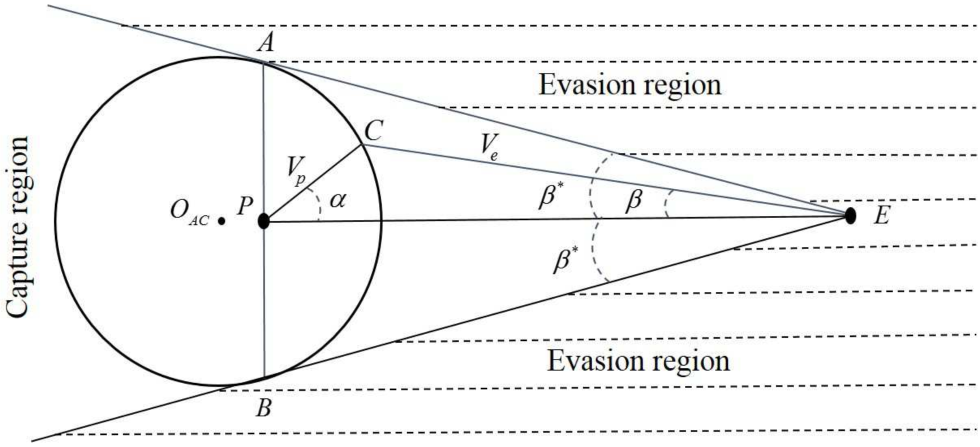

Fig. 2 Apollonius circle created by the pursuer P and the evader E.

$$\beta^* = \sin^{-1}\left(\frac{V_p}{V_e}\right) \tag{14}$$

where $\beta^* = \max(\beta)$.

Hence, in a multi-player pursuit-evasion differential game with $V_e > V_p$, the condition required by the pursuer to capture the superior evader is defined as follows,

$$\beta \leq \sin^{-1}\left(\frac{V_p}{V_e}\right) \tag{15}$$

In other words, as long as the evader's angle β is not greater than β^*, the capture of the superior evader E by the pursuer P will be possible. Fig. (2) shows that the capture region of the pursuer P covers $2\beta^*$. In other words, the pursuer P can capture the superior evader E if the evader moves with $\beta \leq \beta^*$ from its LOS to the pursuer in the direction of either the boundary line EA or the boundary line EB.

As can be seen in Fig. (2), the Apollonius circle defines the capture region of the pursuer and the escape region of the evader. The boundary lines (threshold lines) EA and EB separate between the capture region of the pursuer and the escape region of the evader. Fig. (2) also shows that when the evader moves with an angle β such that $\beta = \beta^*$ in the direction of the boundary line (threshold line) EA, the pursuer has to move with an angle $\alpha = \frac{\pi}{2}$ (in counter-clockwise direction) in order to satisfy the capture condition of Eq. (12) and capture the evader at the point A. In addition, when the evader moves with an angle β^* in the direction of the boundary line (threshold line) EB, the pursuer has to move with an angle $\alpha = \frac{\pi}{2}$ (in clockwise direction) in order to capture the evader at the point B. We define the points A and B as the furthest capture points (threshold capture points) as they are located on the threshold lines that separate between the capture region of the pursuer and the escape region of the evader. They also cost the pursuer the longest time to capture the evader.

2.4 **Problem Definition**

In this paper, we consider a pursuit-evasion differential game with N pursuers and one superior evader E. Each pursuer P_o, $(o = 1, ..., N)$, has a constant speed V_{p_o}, and all pursuers have the same speed. The evader, on the other hand, has a constant speed V_e, where $V_e > V_{p_o}$. It is important to mention here that although we only now consider multi-pursuer single-superior-evader pursuit-evasion differential games with $V_e > V_{p_o}$, our work is also applicable to multi-pursuer single-superior-evader pursuit-evasion differential games with $V_e = V_{p_o}$. This is more illustrated in the subsection (4.3). The positions of the pursuers and the evader are $(x_{p_o}(t), y_{p_o}(t))$ and $(x_e(t), y_e(t))$, respectively. We assume that each pursuer P_o knows the position of the evader at time t. We also assume that the constant speed of the evader is known to each pursuer P_o. The capture of the superior evader occurs if the distance between the pursuer P_o and the evader E is less than or equal to a small specific amount d_c, $\|(x_{p_o}, y_{p_o}) - (x_e, y_e)\| \leq d_c$.

Eq. (11) shows that when γ is small (the evader is much faster than the pursuer), the radius of the Apollonius circle will be small and the evader will have more paths to escape. Therefore, to capture the superior evader, more pursuers are needed to surround the superior evader. The pursuers have to be distributed around the superior evader to construct a polygon whose vertices are the pursuers' positions [25, 27, 33]. The Apollonius circles of each two adjacent pursuers have to intersect or be at least tangent to each other [27, 35]. This represents the most important condition required in a multi-pursuer single-superior-evader pursuit-evasion differential game in order to capture the superior evader by one or some of the pursuers. Thus, a formation control approach that controls the angle distributions of the pursuers around the superior evader has to be integrated with the control strategy of each pursuer. This is to guarantee that the pursuers are distributed around the superior evader and the Apollonius circles of each two adjacent pursuers intersect or are at least tangent to each other.

In a multi-pursuer single-superior-evader pursuit-evasion differential game with $V_e > V_{p_o}$, each pursuer P_o covers $2\beta^*$ of the evader's movement. Thus, the minimum number of the pursuers needed to surround the superior evader in order to capture it is defined as follows [25, 27, 34],

$$N = \left[\frac{2\pi}{2\beta^*}\right]_+ = \left[\frac{2\pi}{2\sin^{-1}\left(\frac{V_p}{V_e}\right)}\right]_+ = \left[\frac{\pi}{\sin^{-1}\left(\frac{V_p}{V_e}\right)}\right]_+$$

Thus,

$$N = \left[\frac{\pi}{\sin^{-1}\left(\frac{V_p}{V_e}\right)}\right]_+ \tag{16}$$

where $[.]_+$ defines the smallest integer number that is greater than or equal to $[.]$.

From Eq. (16), we have

$$N \geq \frac{\pi}{\sin^{-1}\left(\frac{V_p}{V_e}\right)}$$

Thus,

$$\sin^{-1}\left(\frac{V_p}{V_e}\right) \geq \frac{\pi}{N}$$

Or,

$$\frac{V_p}{V_e} \geq \sin\left(\frac{\pi}{N}\right) \tag{17}$$

Hence, if $\frac{V_p}{V_e} \geq \sin\left(\frac{\pi}{N}\right)$ and the Apollonius circles of each two adjacent pursuers intersect or are at least tangent to each other all the time, there will be at least one pursuer that would satisfy the capture condition of Eq. (12) and will then be able to capture the superior evader.

3 The Residual Gradient Fuzzy Actor Critic Learning (RGFACL) Algorithm

Different fuzzy learning algorithms that can be applied to pursuit-evasion differential games are proposed in the literature [4–8]. In this work, the proposed algorithm uses the RGFACL algorithm as the RGFACL algorithm is shown in [4] to be robust and has a quick convergence speed. The RGFACL algorithm uses three fuzzy inference systems (FISs); one is used as an actor (fuzzy logic controller, FLC), and the other two FISs are used as critics [4]. The critics are used to estimate the value functions $V_t(\mathbf{s}_t)$ and $V_t(\mathbf{s}_{t+1})$ of the same learning agent at two different states $\mathbf{s}_t$ and $\mathbf{s}_{t+1}$, respectively. The input parameters of the actor and the critics are the means and the standard deviations of the Gaussian membership functions (MFs) of their inputs. On the other hand, the output parameters of the actor and the critics are the constant consequent parameters of their fuzzy rules. To simplify notations, we define the input and the output parameters of the actor and the critics as ψ^A and ψ^C, respectively. The temporal difference error, $\varDelta_t$, is defined as follows,

$$\varDelta_t = r_t + \gamma V_t(\mathbf{s}_{t+1}) - V_t(\mathbf{s}_t) \tag{18}$$

where r_t is the immediate reward of the learning agant, and γ is a discount factor.

The mean square error, E, of the temporal difference error, $\varDelta_t$, is defined as follows,

$$E = \frac{1}{2}\varDelta_t^2 \tag{19}$$

The RGFACL algorithm updates the input and the output parameters of the critics, ψ^C, as follows [4],

$$\psi_{t+1}^C = \psi_t^C - \alpha\frac{\partial E}{\partial \psi_t^C} \tag{20}$$

where α is a learning rate for the parameters of the critics. On the other hand, the term $\frac{\partial E}{\partial \psi_t^C}$ is updated as follows [4],

$$\frac{\partial E}{\partial \psi_t^C} = \varDelta_t\Big[\gamma\frac{\partial V_t(\mathbf{s}_{t+1})}{\partial \psi_t^C} - \frac{\partial V_t(\mathbf{s}_t)}{\partial \psi_t^C}\Big] \tag{21}$$

From Eq. (21), Eq. (20) can be rewritten as follows,

$$\psi_{t+1}^C = \psi_t^C - \alpha\big[r_t + \gamma V_t(\mathbf{s}_{t+1}) - V_t(\mathbf{s}_t)\big] \ . \ \big[\gamma\frac{\partial}{\partial \psi_t^C}V_t(\mathbf{s}_{t+1}) - \frac{\partial}{\partial \psi_t^C}V_t(\mathbf{s}_t)\big] \tag{22}$$

The derivatives $\frac{\partial V_t(\mathbf{s}_t)}{\partial \psi_t^C}$ are defined as follows,

$$\frac{\partial V_t(\mathbf{s}_t)}{\partial k_l} = \varPhi_l(\mathbf{s}_t) \tag{23}$$

$$\frac{\partial V_t(\mathbf{s}_t)}{\partial \sigma_j} = \frac{2(s_i - m_j)^2}{\sigma_j^3} \times \sum_{l=1}^{L}\xi_{j,l}\frac{k_l - V_t(\mathbf{s}_t)}{\sum_l \omega_l(\mathbf{s}_t)}\omega_l(\mathbf{s}_t) \tag{24}$$

$$\frac{\partial V_t(\mathbf{s}_t)}{\partial m_j} = \frac{2(s_i - m_j)}{\sigma_j^2} \times \sum_{l=1}^{L}\xi_{j,l}\frac{k_l - V_t(\mathbf{s}_t)}{\sum_l \omega_l(\mathbf{s}_t)}\omega_l(\mathbf{s}_t) \tag{25}$$

where,

$$s_i = \begin{cases} s_1 & \text{if} \quad (\sigma_j, m_j) \in \varOmega(s_1) \\ s_2 & \text{if} \quad (\sigma_j, m_j) \in \varOmega(s_2) \\ . & \\ . & \\ s_n & \text{if} \quad (\sigma_j, m_j) \in \varOmega(s_n) \end{cases} \tag{26}$$

and,

$$\xi_{j,l} = \begin{cases} 1 & \text{if} \quad (\sigma_j, m_j) \in \varOmega(\omega_l) \\ 0 & \text{if} \quad (\sigma_j, m_j) \notin \varOmega(\omega_l) \end{cases} \tag{27}$$

The derivatives $\frac{\partial V_t(\mathbf{s}_{t+1})}{\partial \psi_t^C}$ are defined as follows,

$$\frac{\partial V_t(\mathbf{s}_{t+1})}{\partial k_l} = \varPhi_l(\mathbf{s}_{t+1}) \tag{28}$$

$$\frac{\partial V_t(\mathbf{s}_{t+1})}{\partial \sigma_j} = \frac{2(s_i' - m_j)^2}{\sigma_j^3} \times \sum_{l=1}^{L}\xi_{j,l}\frac{k_l - V_t(\mathbf{s}_{t+1})}{\sum_l \omega_l(\mathbf{s}_{t+1})}\omega_l(\mathbf{s}_{t+1}) \tag{29}$$

$$\frac{\partial V_t(\mathbf{s}_{t+1})}{\partial m_j} = \frac{2(s_i' - m_j)}{\sigma_j^2} \times \sum_{l=1}^{L}\xi_{j,l}\frac{k_l - V_t(\mathbf{s}_{t+1})}{\sum_l \omega_l(\mathbf{s}_{t+1})}\omega_l(\mathbf{s}_{t+1}) \tag{30}$$

where,

$$s'_i = \begin{cases} s'_1 & \text{if} \quad (\sigma_j, m_j) \in \Omega(s_1) \\ s'_2 & \text{if} \quad (\sigma_j, m_j) \in \Omega(s_2) \\ . & \\ . & \\ s'_n & \text{if} \quad (\sigma_j, m_j) \in \Omega(s_n) \end{cases} \tag{31}$$

where s'_i is the ith input state variable of the state vector $\mathbf{s}_{t+1}$.

The input and the output parameters of the actor, ψ^A, are updated as follows [4],

$$\psi^A_{t+1} = \psi^A_t + \beta \Delta_t \frac{\partial u_t}{\partial \psi^A_t} \left[\frac{u_c - u_t}{\sigma_n} \right] \tag{32}$$

where β is a learning rate for the actor parameters, u_c is the output of the actor with a random Gaussian noise. The derivatives of the output of the FLC (the actor), u_t, with respect to the input and the output parameters of the FLC can be calculated by replacing $V_t(\mathbf{s}_t)$ with u_t in Eq. (23), Eq. (24) and Eq. (25) as follows,

$$\frac{\partial u_t}{\partial k_l} = \Phi_l(\mathbf{s}_t) \tag{33}$$

$$\frac{\partial u_t}{\partial \sigma_j} = \frac{2(s_i - m_j)^2}{\sigma_j^3} \times \sum_{l=1}^{L} \xi_{j,l} \frac{k_l - u_t}{\sum_l \omega_l(\mathbf{s}_t)} \omega_l(\mathbf{s}_t) \tag{34}$$

$$\frac{\partial u_t}{\partial m_j} = \frac{2(s_i - m_j)}{\sigma_j^2} \times \sum_{l=1}^{L} \xi_{j,l} \frac{k_l - u_t}{\sum_l \omega_l(\mathbf{s}_t)} \omega_l(\mathbf{s}_t) \tag{35}$$

4 The Proposed Algorithm

In this work, we develop a decentralized learning algorithm for multi-pursuer single-superior-evader pursuit-evasion differential games, where the speed of the evader is similar to or higher than the speed of each pursuer. This is the first time that anyone has shown that they can develop a system of learning to capture a superior evader in a pursuit-evasion differential game without using any type of discretization for the state or the action space. There have been some work reported in the literature to capture superior evaders [25–28,33–35,37,38]. However, these techniques are deterministic approaches as no learning is involved in these techniques. In real world applications, pursuers may need to adapt to changing environments. Our objective in this work is to make the pursuers in a multi-pursuer single-superior-evader pursuit-evasion differential game learn their strategies so that one or some of the learning pursuers can capture the superior evader. The proposed algorithm uses the RGFACL algorithm proposed in [4] to tune the input and the output parameters of the fuzzy logic controller (FLC) of the learning pursuer P_o. The proposed algorithm is a decentralized algorithm as each learning pursuer P_o has its own learning algorithm and no communication among the pursuers is required. The only information the proposed learning algorithm of the pursuer P_o requires is the position and the speed of the superior evader. The proposed algorithm uses the well-known Apollonius circle to define the capture region of the learning pursuer P_o based on its location and the location of the superior evader. The capture region of the pursuer P_o is used with a developed formation control approach to construct the reward function of the learning pursuer P_o, as will be illustrated later on in this section, so that one or some of the learning pursuers can capture the superior evader. This reward function is used in the tuning mechanism of the FLC of the learning pursuer P_o. The formation control mechanism used in the reward function of the learning pursuer P_o guarantees that the pursuers are distributed around the superior evader in order to avoid collision between the pursuers. In addition, the formation control mechanism makes the Apollonius circles of each two adjacent pursuers intersect or be at least tangent to each other so that the capture of the superior evader can occur.

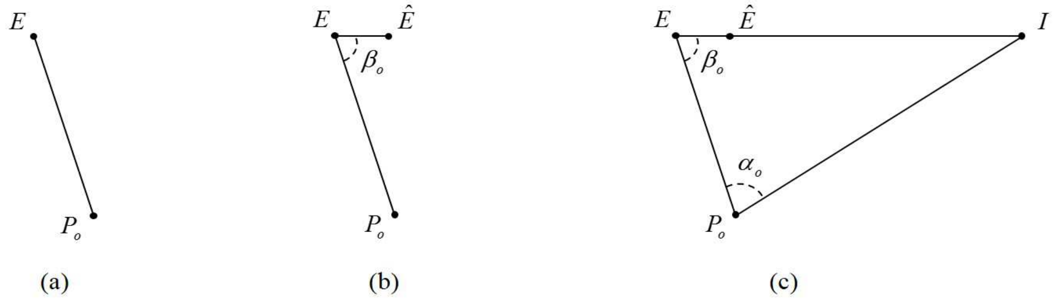

Fig. 3 Geometric illustration of defining the action for the pursuer P_o

4.1 The action u_{p_o} of the pursuer P_o

Fig (3-a) shows the pursuer P_o and the evader E. We use a Kalman filter to estimate the future position of the evader at the next time step, $\hat{E}$. Fig (3-b) shows the pursuer P_o and the evader at its real and estimated future positions. We define the angle β_o as the absolute value of the angle difference between the direction of the evader (the direction of the vector $\overrightarrow{E\hat{E}}$) and the direction of the evader's LOS to the pursuer P_o (the direction of the vector $\overrightarrow{EP_o}$). That is,

$$\beta_o = |\bar{\beta}_o| \tag{36}$$

where

$$\bar{\beta}_o = \angle\overrightarrow{E\hat{E}} - \angle\overrightarrow{EP_o} \qquad \text{and} \qquad -\pi \leq \bar{\beta}_o \leq \pi \tag{37}$$

Fig (3-c) shows the angle α_o that the pursuer P_o should select in order to satisfy the capture equation defined in Eq. (12). That is, the angle that describes the direction $\theta_{\alpha_o} = \angle\overrightarrow{P_o I}$ that the pursuer P_o should take in order to capture the evader at the future capture point I. To find the direction θ_{α_o}, we first define the vector $\overrightarrow{P_o\acute{I}}$; the vector $\overrightarrow{P_o\acute{I}}$ has the same length of the vector $\overrightarrow{P_o E}$ and the same direction of the vector $\overrightarrow{P_o I}$. We use the angle α_o and the vector $\overrightarrow{P_o E}$ to define the vector $\overrightarrow{P_o\acute{I}}$ as follows,

$$\overrightarrow{P_o\acute{I}} = R(\bar{\alpha}_o) \times \overrightarrow{P_o E} \tag{38}$$

where $R(\bar{\alpha}_o)$ is a rotation matrix that is used to perform rotation in Euclidean space and it is defined as follows,

$$R(\bar{\alpha}_o) = \begin{bmatrix} \cos(\bar{\alpha}_o) & -\sin(\bar{\alpha}_o) \\ \sin(\bar{\alpha}_o) & \cos(\bar{\alpha}_o) \end{bmatrix} \tag{39}$$

and,

$$\bar{\alpha}_o = \begin{cases} -\alpha_o & \text{if } \bar{\beta}_o > 0 \\ \alpha_o & \text{otherwise} \end{cases} \tag{40}$$

The direction of rotation will be counter-clockwise if the angle $\bar{\alpha}_o$ is positive. On the other hand, the direction of rotation will be clockwise if the angle $\bar{\alpha}_o$ is negative.

Let $\overrightarrow{P_o\acute{I}} = (x_{dir}, y_{dir})$. Thus, the direction θ_{α_o}, the direction of the vector $\overrightarrow{P_o\acute{I}}$, is defined as follows,

$$\theta_{\alpha_o} = \angle\overrightarrow{P_o\acute{I}} = \tan^{-1}\left(\frac{y_{dir}}{x_{dir}}\right) \tag{41}$$

Hence, the action u_{p_o} selected by the pursuer P_o at each sampling period is defined as follows,

$$u_{p_o} = \theta_{\alpha_o} - \theta_{p_o} \tag{42}$$

where θ_{p_o} is the direction (orientation) of the pursuer P_o.

4.2 **The developed formation control approach**

Different formation control approaches that are applied to multi-pursuer single-superior-evader pursuit-evasion differential games are presented in the literature [26,27,33,34]. The formation control approaches control the angle distributions (the relative position and the orientation) of the pursuers around the superior evader so that the superior evader can be captured by one or some of the pursuers. The formation control approach presented in [27] is a centralized approach as it requires cooperation between pursuers. This is because the pursuer that can intercept the evader has to broadcast the interception point to the other pursuers, whose responsibilities are to head up towards that interception point in order to shrink the enclosed polygon (whose vertices are the pursuers' positions) and contain the evader. The formation control approach presented in [34] is a decentralized approach as no communication among pursuers is required. The strategy of each pursuer, in this formation control approach, is either to intercept the evader if the evader moves into the capture region of the pursuer or to move towards a virtual target if the evader moves into its escape region. The virtual target is defined as the threshold point (either point A or point B in Fig. (2) based on the direction of the evader), where it is most likely for the evader to cross into the capture region of the pursuer. However, when the angle $\beta_o > \frac{\pi}{2}$, moving towards a virtual target by the pursuer is not the best choice to contain the evader into the polygon constructed by the pursuers. The formation control approach presented in [26,33] is also a decentralized approach as no communication among pursuers is required. The strategy of each pursuer, when following this formation control approach, is either to intercept the superior evader if the evader moves into the capture region of the pursuer or to move in parallel with the evader if the evader moves into its escape region. However, this strategy guarantees the capture of the superior evader only in the case that the superior evader and the pursuers have the same speed ($V_e = V_{p_o}$). When $V_e > V_{p_o}$, this strategy may fail to make one or some of the pursuers capture the superior evader.

The formation control approach proposed in this work is a modified version of the formation control approaches presented in [26,27,33,34]. The proposed formation control approach used in this work is a decentralized approach. The strategy of the pursuer P_o, in this formation control approach, is to intercept the evader if the evader moves into the capture region of the pursuer. However, if the evader moves into its escape region, the strategy of the pursuer P_o is either to move to a virtual target or to move in parallel with the evader. The pursuer P_o moves to a virtual target if the angle β_o is such that $\sin^{-1}\left(\frac{V_{p_o}}{V_e}\right) < \beta_o \leq \frac{\pi}{2}$, where the virtual target is the threshold point (either point A or point B in Fig. (2)). On the other hand, the pursuer P_o moves in parallel with the evader if the angle β_o is such that $\beta_o > \frac{\pi}{2}$. This formation control approach will guarantee that the pursuers are distributed around the superior evader in order to avoid collision between pursuers. This formation control approach will also shrink the enclosed polygon constructed by the pursuers and make the pursuers contain the evader so that the Apollonius circles of each two adjacent pursuers intersect or are at least tangent to each other. This will make one or some of the pursuers capture the superior evader.

4.3 **The reward function r_{p_o} of the pursuer P_o**

In this work, we set the rewards of the learning pursuer P_o based on the developed formation control mechanism illustrated above so that one or some of the learning pursuers can capture the superior evader. The speed of the pursuer P_o is defined by V_{p_o}, and the speed of the evader E is defined by V_e. We assume that the speed of the evader is known to each pursuer. At each sampling period, the inputs to the FLC of the pursuer P_o are the angle β_o and its derivative $\dot{\beta}_o$. Based on its inputs, the FLC of the pursuer P_o selects an angle α_o which is then used to calculate the action u_{p_o} as in Eq. (42).

The pursuer P_o will be rewarded based on the region that the evader moves into; the capture region of the pursuer P_o or the evasion region of the evader shown in Fig. (2). It is important to mention here that the capture region of the pursuer P_o and the corresponding escape region of the evader are updated regularly every time step. Consequently, the reward function of the pursuer P_o will change every time step, depending on the angle α_o selected by the pursuer P_o. We will first describe how to reward the pursuer P_o when the evader moves into the capture region of the pursuer P_o. That is, the evader moves with an angle β_o such that $\beta_o \leq \sin^{-1}\left(\frac{V_{p_o}}{V_e}\right)$. The reward function of the pursuer P_o, in this case, is constructed based on the triangle property defined in Eq. (12). We use the capture equation defined in Eq. (12) as a mechanism to reward the learning pursuer P_o at every time step. That is, if the angle α_o selected by the pursuer P_o satisfies the triangle property (the capture equation) of Eq. (12), the pursuer

P_o will be rewarded with a positive payoff (+1 for example). On the other hand, if the angle α_o selected by the pursuer P_o does not satisfy the triangle property (capture equation) of Eq. (12), the pursuer P_o will be punished with a negative payoff (-1 for example). This is shown in Fig. (4-a). Hence, the reward function of the pursuer P_o when the evader moves into the capture region of the pursuer P_o is defined as follows,

$$r_{p_o} = \begin{cases} +1 & \text{if } \alpha_o \in \big[\chi - \epsilon_a, \chi + \epsilon_a\big] \ \& \ \beta_o \le \sin^{-1}\big(\frac{V_{p_o}}{V_e}\big) \\ -1 & \quad \text{otherwise} \end{cases} \tag{43}$$

where $\chi = \sin^{-1}\big[\frac{V_e}{V_{p_o}}\sin(\beta_o)\big]$, and ϵ_a is a very small constant that defines the angle tolerance.

On the other hand, when the evader moves into its evasion region, the pursuer P_o will not be rewarded based on the capture equation defined in Eq. (12). In this case, the evader moves with an angle β_o such that $\beta_o > \sin^{-1}\big(\frac{V_{p_o}}{V_e}\big)$. Thus, the pursuer P_o has to work with the other pursuers to surround (enclose) the evader. This can be done by following the developed formation control illustrated above. Thus, the strategy of the pursuer P_o, in this case, is either to move to a virtual target (the threshold point) or to move in parallel with the evader. The pursuer P_o has to move to a virtual target (the threshold point A or B) if the evader moves with an angle β_o such that $\sin^{-1}\big(\frac{V_{p_o}}{V_e}\big) < \beta_o \le \frac{\pi}{2}$. On the other hand, the pursuer P_o has to move in parallel with the evader if the angle β_o is such that $\beta_o > \frac{\pi}{2}$. We will first define the reward function of the pursuer P_o when the evader moves into its escape region with an angle β_o such that $\sin^{-1}\big(\frac{V_{p_o}}{V_e}\big) < \beta_o \le \frac{\pi}{2}$. As illustrated in Subsection (2.3), the pursuer moves to the threshold points (either point A and B) when the angle α_o selected by the pursuer P_o is such that $\alpha_o = \frac{\pi}{2}$. This is shown in Fig. (4-b) and Fig. (4-c). Thus, the reward function in this case is defined as follows,

$$r_{p_o} = \begin{cases} +1 & \text{if } \alpha_o \in \big[\frac{\pi}{2} - \epsilon_a, \frac{\pi}{2} + \epsilon_a\big] \ \& \ \sin^{-1}\big(\frac{V_{p_o}}{V_e}\big) < \beta_o \le \frac{\pi}{2} \\ -1 & \quad \text{otherwise} \end{cases} \tag{44}$$

When the evader, on the other hand, moves into its escape region with an angle β_o such that $\beta_o > \frac{\pi}{2}$, the pursuer P_o has to move in parallel with the evader. Thus, if the angle α_o selected by the pursuer P_o makes the pursuer P_o move in parallel with the evader, the pursuer will be rewarded; otherwise the pursuer P_o will be punished. To make the pursuer P_o move in parallel with the evader, the angle α_o selected by the pursuer P_o has to such that $\alpha_o + \beta_o = \pi$. This is shown in Fig. (4-d). Thus, the reward function of the pursuer P_o when the evader moves into its evasion region with an angle β_o such that $\beta_o > \frac{\pi}{2}$ is defined as follows,

$$r_{p_o} = \begin{cases} +1 & \text{if } \alpha_o + \beta_o \in \big[\pi - \epsilon_a, \pi + \epsilon_a\big] \ \& \ \beta_o > \frac{\pi}{2} \\ -1 & \quad \text{otherwise} \end{cases} \tag{45}$$

Hence, from Eq. (43) to Eq. (45), the reward function of the pursuer P_o when the evader moves into either the capture region of the pursuer P_o or its evasion region can be defined as follows,

$$r_{p_o} = \begin{cases} +1 & \text{if } \alpha_o \in \big[\chi - \epsilon_a, \chi + \epsilon_a\big] \ \& \ \beta_o \le \sin^{-1}\big(\frac{V_{p_o}}{V_e}\big) \\ +1 & \text{if } \alpha_o \in \big[\frac{\pi}{2} - \epsilon_a, \frac{\pi}{2} + \epsilon_a\big] \ \& \ \sin^{-1}\big(\frac{V_{p_o}}{V_e}\big) < \beta_o \le \frac{\pi}{2} \\ +1 & \text{if } \alpha_o + \beta_o \in \big[\pi - \epsilon_a, \pi + \epsilon_a\big] \ \& \ \beta_o > \frac{\pi}{2} \\ -1 & \quad \text{otherwise} \end{cases} \tag{46}$$

where $\chi = \sin^{-1}\big[\frac{V_e}{V_{p_o}}\sin(\beta_o)\big]$.

Special case: When the speed of the evader is such that $V_e = V_{p_o}$, the Apollonius circle becomes a straight line and the term $\beta_o \le \sin^{-1}\big(\frac{V_{p_o}}{V_e}\big)$ in Eq. (46) becomes $\beta_o \le \frac{\pi}{2}$. Thus, Eq. (46) can be rewritten as follows,

$$r_{p_o} = \begin{cases} +1 & \text{if } \alpha_o \in \big[\chi - \epsilon_a, \chi + \epsilon_a\big] \ \& \ \beta_o \le \frac{\pi}{2} \\ +1 & \text{if } \alpha_o + \beta_o \in \big[\pi - \epsilon_a, \pi + \epsilon_a\big] \ \& \ \beta_o > \frac{\pi}{2} \\ -1 & \quad \text{otherwise} \end{cases} \tag{47}$$

The strategy learned by the pursuer P_o when using the reward function defined in Eq. (47) indicates that the pursuer P_o will either intercept the evader if the evader moves into the capture region of the pursuer or move in parallel with the evader if the evader moves into its escape region. This strategy is similar to the strategy that was presented in [26,33] in the case that the superior evader and the pursuers

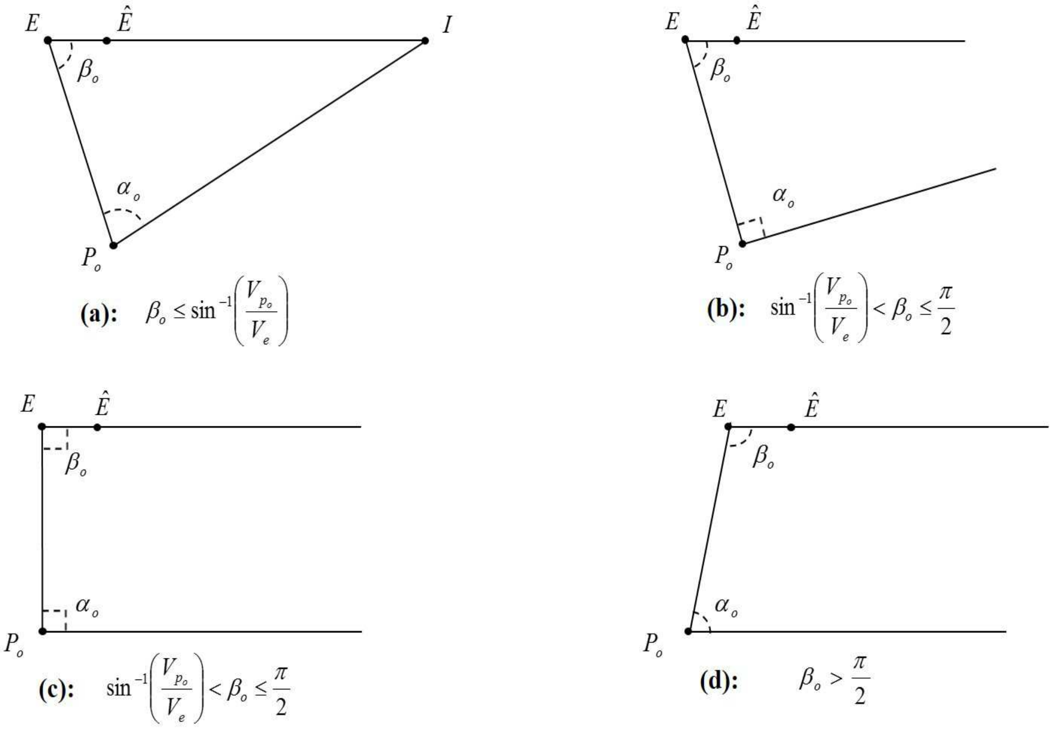
E
Ê
I
β_o
α_o
P_o
(a): $\beta_o \le \sin^{-1}\left(\frac{V_{p_o}}{V_e}\right)$
E
Ê
β_o
α_o
P_o
(b): $\sin^{-1}\left(\frac{V_{p_o}}{V_e}\right) < \beta_o \le \frac{\pi}{2}$
E
Ê
β_o
α_o
P_o
(c): $\sin^{-1}\left(\frac{V_{p_o}}{V_e}\right) < \beta_o \le \frac{\pi}{2}$
E
Ê
β_o
α_o
P_o
(d): $\beta_o > \frac{\pi}{2}$

Fig. 4 Geometric illustration of the reward function mechanism for the pursuer P_o

have the same speeds (i.e. $V_e = V_{p_o}$). Hence, the reward function mechanism constructed based on the formation control approach developed in this work will also be applicable in the case that the superior evader and the pursuers have the same speeds.

5 Simulation and Results

In this section, we evaluate the proposed algorithm on three different multi-pursuer single-superior-evader pursuit-evasion differential games where the speed of the evader is similar to or higher than the speed of each pursuer. In each game, each pursuer P_o is learning its control strategy so that one or some of the learning pursuers can capture the superior evader. The evader is also learning its control strategy so that it can reach a specific target (x_e^T, y_e^T) before it is being captured by one or some of the pursuers. It is important to mention here that our objective is not to design an optimal strategy for the superior evader. Our objective is only to evaluate our algorithm when the evader is an intelligent superior evader. Thus, we assume that the priority of learning for the evader is to learn how to reach its target. However, if the distance between the evader and the nearest pursuer is less than a specific distance (tolerance distance, d_{tol}), the priority of learning for the evader becomes to escape from that pursuer.

5.1 **Simulation setup**

As illustrated in the previous section, for each pursuer P_o, $(o = 1, ..., N)$, we define the angle β_o as the absolute value of the angle difference between the evader's direction and its LOS to the pursuer P_o. In addition, we define the state $\mathbf{s}_t$ for the pursuer P_o by two input variables which are the angle β_o and its derivative $\dot{\beta}_o$. Five Gaussian membership functions (MFs) are used to define the fuzzy sets of each input to the FISs of the proposed learning algorithm of each pursuer P_o. On the other hand, the state $\mathbf{s}_t$ of the evader is defined by two input variables, δ_e and its derivative $\dot{\delta}_e$, where δ_e is the angle difference between the direction of the evader and the direction of its target (x_e^T, y_e^T). However, if the distance between the evader and the nearest pursuer, d_{p_o}, is less than the tolerance distance d_{tol}, δ_e is defined as the angle difference between the direction of the evader and the direction of the LOS of that nearest pursuer to the evader. Thus, the reward function of the superior evader is defined as follows,

$$r_e = \begin{cases} \Delta_e(t) & \text{if } d_{p_o} > d_{tol} \\ -\Delta_{p_o}(t) & \text{if } d_{p_o} \leq d_{tol} \text{ \& } d_{p_o} \leq d_{p_j} \text{ \& } o \neq j \end{cases} \tag{48}$$

where,

$$\Delta_e(t) = D_e(t) - D_e(t+1)$$

$$D_e(t) = \|(x_e, y_e) - (x_e^T, y_e^T)\|$$

$$d_{p_o} = \|(x_{p_o}, y_{p_o}) - (x_e, y_e)\|$$

$$\Delta_{p_o}(t) = D_{p_o}(t) - D_{p_o}(t+1)$$

$$D_{p_o}(t) = \|(x_{p_o}, y_{p_o}) - (x_e, y_e)\|$$

$$d_{p_j} = \|(x_{p_j}, y_{p_j}) - (x_e, y_e)\|$$

Five Gaussian membership functions (MFs) are used to define the fuzzy sets of each input to the FISs of the RGFACL algorithm of the evader. The wheelbases and the maximum steering angles of the pursuers and the evader are set as follows, $L_{p_o} = L_e = 0.5$ m and $u_{p_o}^{max} = u_e^{max} = 0.8$ rad. The parameters of the learning algorithms of the pursuers and the evader are set as those parameters in [4]. The sampling time is defined as $T = 0.05$ s, whereas the capture radius is defined as $d_c = 0.5$m. The tolerance distance is defined as $d_{tol} = 10$ m. The number of episodes is set to 200, whereas the number of steps (in each episode) is set to 2000.

Game 1: In this game, the speed of the evader is similar to the speed of each pursuer (i.e. $V_e = V_{p_o} = 1$ m/s). We use three pursuers, (i.e. $N = 3$ and $o = 1, 2, 3$). The evader starts its motion from the position $(x_e, y_e) = (0, 0)$ with an initial orientation $\theta_e = 0$ rad. The pursuer P_1 starts its motion from a random position at $d_1 \angle \theta_{d_1}$; d_1 is the distance between the pursuer P_1 and the origin $O = (0, 0)$, and θ_{d_1} is the angle that describes the direction of the vector $\overrightarrow{OP_1}$. Thus, the random position of the pursuer P_1 is defined as $(x_{p_1}, y_{p_1}) = \big(d_1 \cos\theta_{d_1}, d_1 \sin\theta_{d_1}\big)$. The initial orientation of the pursuer P_1 is defined as $\theta_{p_1} = \tan^{-1}\big(\frac{-y_{p_1}}{-x_{p_1}}\big)$. Similarly, the pursuers P_2 and P_3 start their motions from random positions but taking into account the angle distributions of the pursuers around the evader required by the formation control. Since we have three pursuers, we define the initial positions of the pursuers P_2 and P_3 as $(x_{p_2}, y_{p_2}) = \big(d_2 \cos(\theta_{d_1} + 2\pi/3), d_2 \sin(\theta_{d_1} + 2\pi/3)\big)$ and $(x_{p_3}, y_{p_3}) = \big(d_3 \cos(\theta_{d_1} + 4\pi/3), d_3 \sin(\theta_{d_1} + 4\pi/3)\big)$, respectively. The distances d_2 and d_3 are randomly selected. The initial orientations of the pursuers P_2 and P_3 are defined as $\theta_{p_2} = \tan^{-1}\big(\frac{-y_{p2}}{-x_{p2}}\big)$ and $\theta_{p_3} = \tan^{-1}\big(\frac{-y_{p3}}{-x_{p3}}\big)$, respectively.

Game 2: In this game, the speed of the evader is higher than the speed of each pursuer. We set the speed of the superior evader as $V_e = 1.1$ m/s and the speed of each pursuer P_o as $V_{p_o} = 1$ m/s. Thus, the number of the pursuers required to capture the superior evader in this game is set based on Eq. (16) as follows,

$$N = \left[\frac{\pi}{\sin^{-1}\left(\frac{V_p}{V_e}\right)}\right]_+ = \left[\frac{\pi}{\sin^{-1}\left(\frac{1}{1.2}\right)}\right]_+ = \Big[2.75\Big]_+ = 3 \tag{49}$$

The initial positions and orientations of the superior evader and the pursuers are set as in Game 1.

Game3: In this game, the speed of the evader is also higher than the speed of each pursuer. We set the speed of the superior evader as $V_e = 1.2$ m/s and the speed of each pursuer P_o as $V_{p_o} = 1$ m/s. Thus, the number of the pursuers required to capture the superior evader in this game is set based on Eq. (16) as follows,

$$N = \left[\frac{\pi}{\sin^{-1}\left(\frac{V_p}{V_e}\right)}\right]_+ = \left[\frac{\pi}{\sin^{-1}\left(\frac{1}{1.2}\right)}\right]_+ = \Big[3.19\Big]_+ = 4 \tag{50}$$

In this game, the evader starts its motion from the position $(x_e, y_e) = (0, 0)$ with an initial orientation $\theta_e = 0$ rad. Likewise Game 1 and Game 2, the pursuers in Game 3 start their motions from random positions defined as follows, $(x_{p_1}, y_{p_1}) = \big(d_1 \cos\theta_{d_1}, d_1 \sin\theta_{d_1}\big)$, $(x_{p_2}, y_{p_2}) = \big(d_2 \cos(\theta_{d_1} + \pi/2), d_2 \sin(\theta_{d_1} + \pi/2)\big)$, $(x_{p_3}, y_{p_3}) = \big(d_3 \cos(\theta_{d_1} + \pi), d_3 \sin(\theta_{d_1} + \pi)\big)$, and $(x_{p_4}, y_{p_4}) = \big(d_4 \cos(\theta_{d_1} + 3\pi/2), d_4 \sin(\theta_{d_1} + 3\pi/2)\big)$.

5.2 Results

Fig. (5) to Fig. (8) show the paths of the pursuers of Game 1 when each pursuer learns its control strategy by the proposed algorithm. Fig. (5) to Fig. (8) also show the path of the evader when the evader in Game 1 learns its control strategy by the RGFACL algorithm. The initial positions of the pursuers are set as $(x_{p_1}, y_{p_1}) = 40\angle 0.5$, $(x_{p_2}, y_{p_2}) = 50\angle(0.5 + 2\pi/3)$ and $(x_{p_3}, y_{p_3}) = 30\angle(0.5 + 4\pi/3)$. As shown in Fig. (5) to Fig. (8), the superior evader is always captured by one or some of the pursuers learning their control strategies by the proposed algorithm. In each figure, the evader has a specific target to go to. However, when the distance between the evader and the nearest pursuer is less than the tolerance distance d_{tol}, the evader's priority becomes to escape from that pursuer. However, because of the formation control mechanism used by the proposed learning algorithm, the evader is always enclosed by the pursuers until it is eventually captured by one or some of the pursuers.

Fig. (9) to Fig. (12) show the paths of the pursuers of Game 2 when each pursuer learns its control strategy by the proposed algorithm. Fig. (9) to Fig. (12) also show the path of the evader of Game 2 when the evader learns its control strategy by the RGFACL algorithm. The initial positions of the pursuers are set as in Game 1. The figures show that the pursuers learning their control strategy by the proposed algorithm always succeed to capture the superior evader although the evader moves with a speed that is higher than the speed of the pursuers.

Table 1 shows the simulation results we perform on Game 3. The simulation of Game 3 is conducted 10 times, and the capture time of the evader is averaged over the number of captures. The evader's target in Game 3 is defined as the position $(x_e^T, y_e^T) = (500, 500)$. If the distance between the nearest pursuer is less than the tolerance distance ($d_{tol} = 10$ m), the evader's priority becomes to escape from that pursuer. As can be seen in Table 1, the superior evader is always captured by one or some of the learning pursuers. Hence, the simulation we conduct on Game 1, Game 2, and Game 3 show the effectiveness of the proposed algorithm.

Table 1 Simulation results to capture the superior evader of Game 3

V_e	V_{p_o}	N	Number of captures of 10 trials	average of capture time
1.2	1.0	$[3.19]_+ = 4$	10	60.73s

6 Conclusion

In this paper, we propose a new fuzzy reinforcement learning algorithm that tunes the pursuers in a multi-pursuer single-superior-evader pursuit-evasion differential game so that the superior evader is captured by one or some of the pursuers. We mean by the "superior evader" that the evader that has a speed that is similar to or higher than the speed of each pursuer in the game. The proposed algorithm uses the well-known Apollonius circle mechanism to define the capture region of each pursuer based on

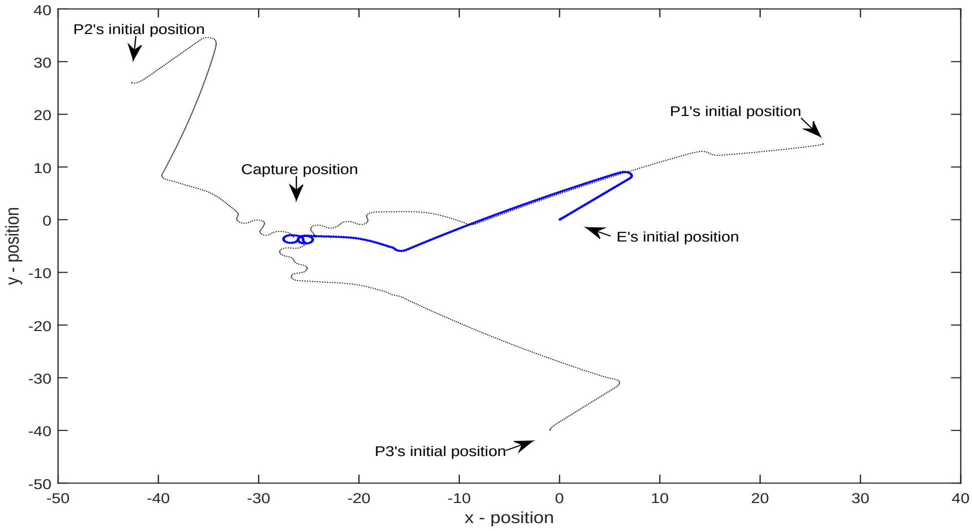

Fig. 5 The paths of the pursuers of Game 1 (thin-line paths) when each pursuer P_o learns its control strategy by the proposed algorithm; the path of the evader (thick-line path) when the evader learns its control strategy by the RGFACL algorithm. The target of the evader here is the position $(x_e^T, y_e^T) = (500, 500)$.

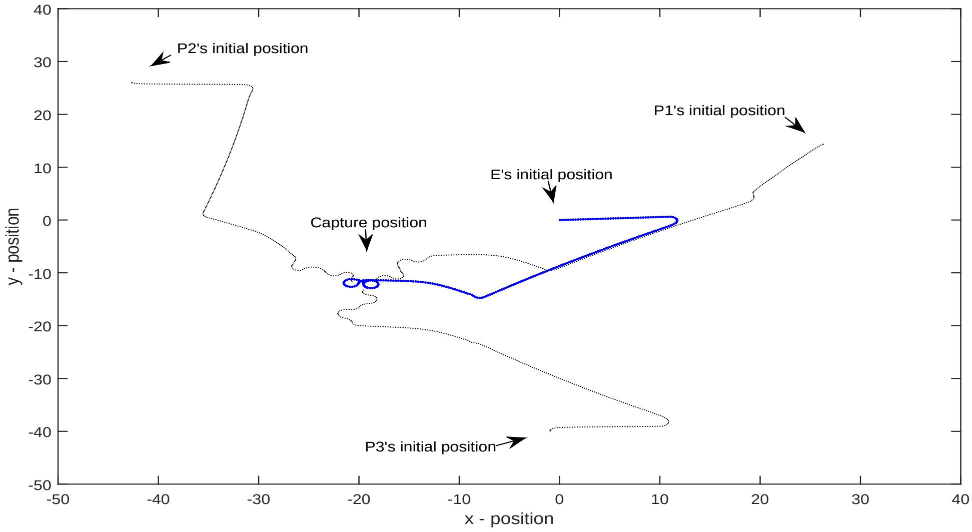

Fig. 6 The paths of the pursuers of Game 1 (thin-line paths) when each pursuer P_o learns its control strategy by the proposed algorithm; the path of the evader (thick-line path) when the evader learns its control strategy by the RGFACL algorithm. The target of the evader here is the position $(x_e^T, y_e^T) = (500, 0)$.

its location and the location of the superior evader. The proposed algorithm uses the RGFACL algorithm to tune the FLC of each pursuer so that the pursuers learn their control strategies to capture the superior evader. A new formation control mechanism is proposed in this work and is used with the Apollonius circle mechanism to construct the reward function of each learning pursuer. The formation control mechanism used by the proposed algorithm guarantees that the pursuers are distributed around the superior evader in order to avoid collision between pursuers. The formation control mechanism used by the proposed algorithm also makes the Apollonius circles of each two adjacent pursuers intersect or are at least tangent to each other so that the capture of the superior evader can occur. The proposed algorithm is a decentralized algorithm as no communication among the pursuers is required. The only information the proposed algorithm requires is the position and the speed of the superior evader.

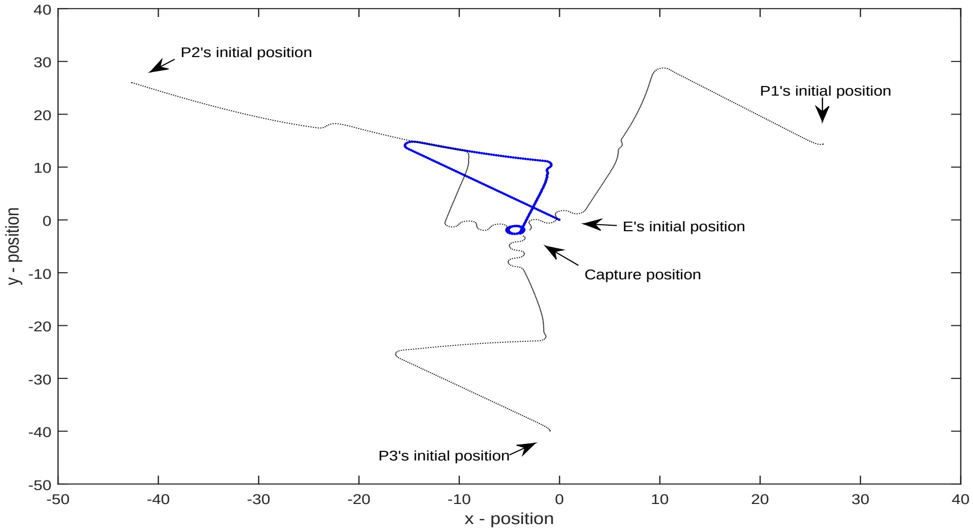

Fig. 7 The paths of the pursuers of Game 1 (thin-line paths) when each pursuer P_o learns its control strategy by the proposed algorithm; the path of the evader (thick-line path) when the evader learns its control strategy by the RGFACL algorithm. The target of the evader here is the position $(x_e^T, y_e^T) = (-500, 500)$.

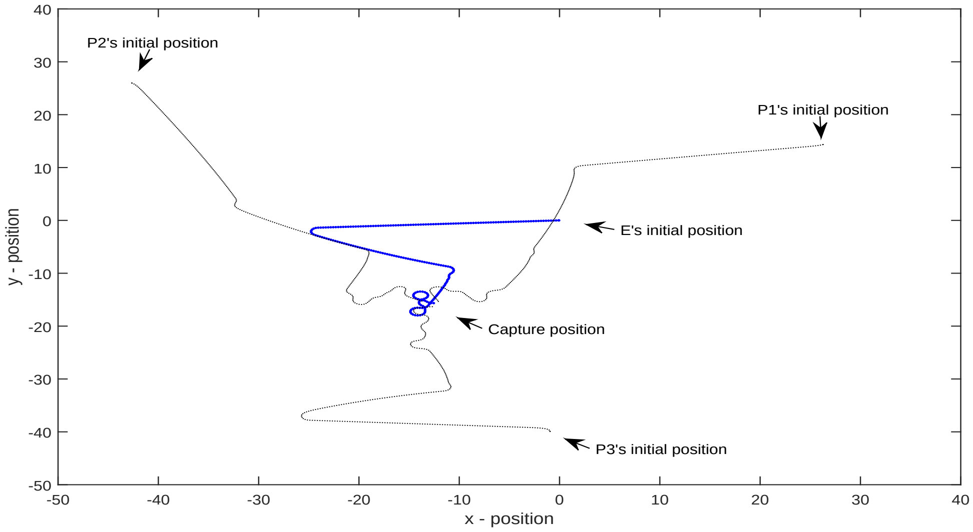

Fig. 8 The paths of the pursuers of Game 1 (thin-line paths) when each pursuer P_o learns its control strategy by the proposed algorithm; the path of the evader (thick-line path) when the evader learns its control strategy by the RGFACL algorithm. The target of the evader here is the position $(x_e^T, y_e^T) = (-500, 0)$.

The proposed algorithm is used to learn different multi-pursuer single-superior-evader pursuit-evasion differential games. The simulation results show the effectiveness of the proposed algorithm.

References

1. L. X. Wang. *A Course in Fuzzy Systems and Control*, Upper Saddle River, NJ: Prentice Hall, 1997.
2. H. M. Schwartz, *Multi-agent Machine Learning: A Reinforcement Approach*, John Wiley & Sons, 2014.
3. L. Baird, *Residual algorithms: Reinforcement learning with function approximation*, In ICML, pp. 30-37, 1995.
4. M. D. Awheda, and H. M. Schwartz, *The Residual Gradient FACL Algorithm for Differential Games*, Proceedings of the 28th IEEE Canadian Conference on Electrical and Computer Engineering (CCECE 2015), Halifax, Nova Scotia, Canada, May 3-6, 2015.

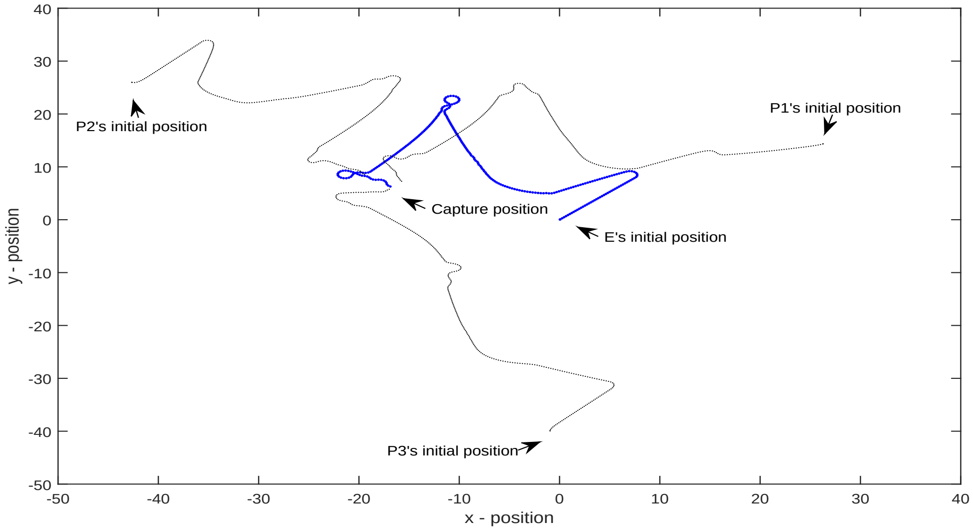

Fig. 9 The paths of the pursuers of Game 2 (thin-line paths) when each pursuer P_o learns its control strategy by the proposed algorithm; the path of the evader (thick-line path) when the evader learns its control strategy by the RGFACL algorithm. The target of the evader here is the position $(x_e^T, y_e^T) = (500, 500)$.

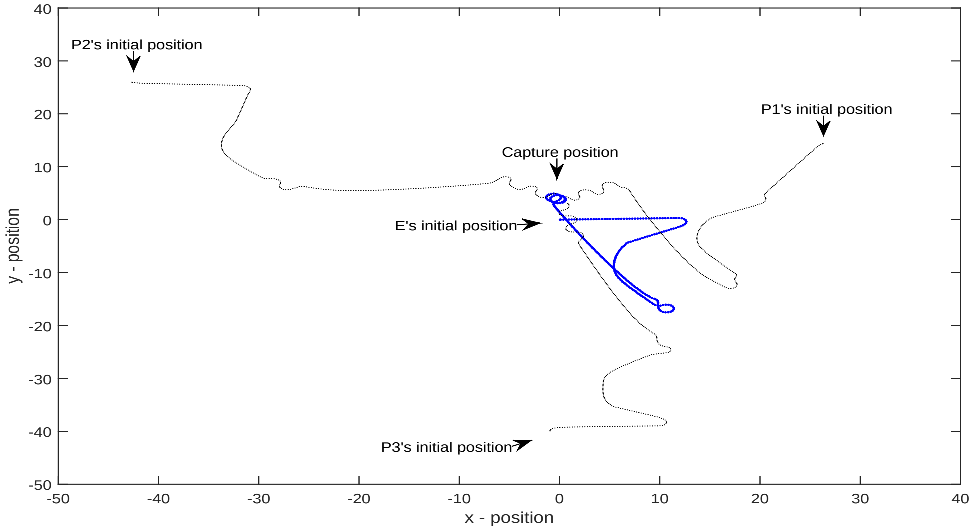

Fig. 10 The paths of the pursuers of Game 2 (thin-line paths) when each pursuer P_o learns its control strategy by the proposed algorithm; the path of the evader (thick-line path) when the evader learns its control strategy by the RGFACL algorithm. The target of the evader here is the position $(x_e^T, y_e^T) = (500, 0)$.

5. S. F. Desouky and H. M. Schwartz, *Q (λ)-learning adaptive fuzzy logic controllers for pursuit-evasion differential games*, International Journal of Adaptive Control and Signal Processing 25.10 910-927, 2011.
6. L. Busoniu, D. Ernst, R. Babuska, and B. D. Schutter, *Fuzzy partition optimization for approximate fuzzy Q-iteration*, In Proceedings of the 17th IFAC World Congress (IFAC-08), 2008.
7. W. Hinojosa, S. Nefti, and U. Kaymak, *Systems control with generalized probabilistic fuzzy-reinforcement learning*, Fuzzy Systems, IEEE Transactions on 19.1: 51-64, 2011.
8. L. Jouffe, *Fuzzy inference system learning by reinforcement methods*, IEEE Trans. Syst., Man, Cybern. C, vol. 28, no. 3, 338-355, 1998.
9. R. Abielmona, E. Petriu, M. Harb, and S. Wesolkowski, *Mission-driven robotic intelligent sensor agents for territorial security*, Computational Intelligence Magazine, IEEE, vol. 6, no. 1, pp. 55-67, 2011.
10. N. M. Stiffler and J. M. O'Kane, *A complete algorithm for visibility-based pursuit-evasion with multiple pursuers*, In Robotics and Automation (ICRA), IEEE International Conference on, pp. 1660-1667, 2014.
11. C. Jun, S. Bhattacharya, and R. Ghrist, *Pursuit-evasion game for normal distributions*, In Intelligent Robots and Systems (IROS 2014), IEEE/RSJ International Conference on, pp. 83-88, 2014.

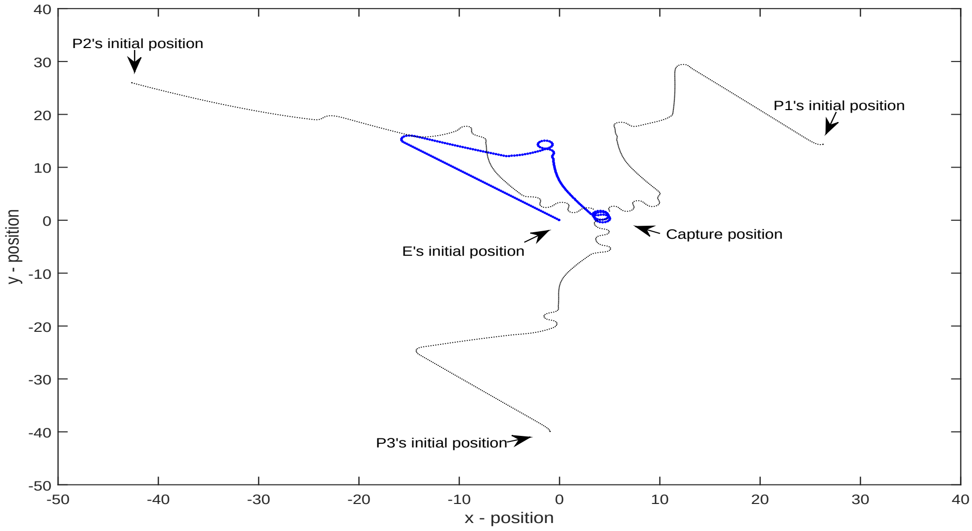

Fig. 11 The paths of the pursuers of Game 2 (thin-line paths) when each pursuer P_o learns its control strategy by the proposed algorithm; the path of the evader (thick-line path) when the evader learns its control strategy by the RGFACL algorithm. The target of the evader here is the position $(x_e^T, y_e^T) = (-500, 500)$.

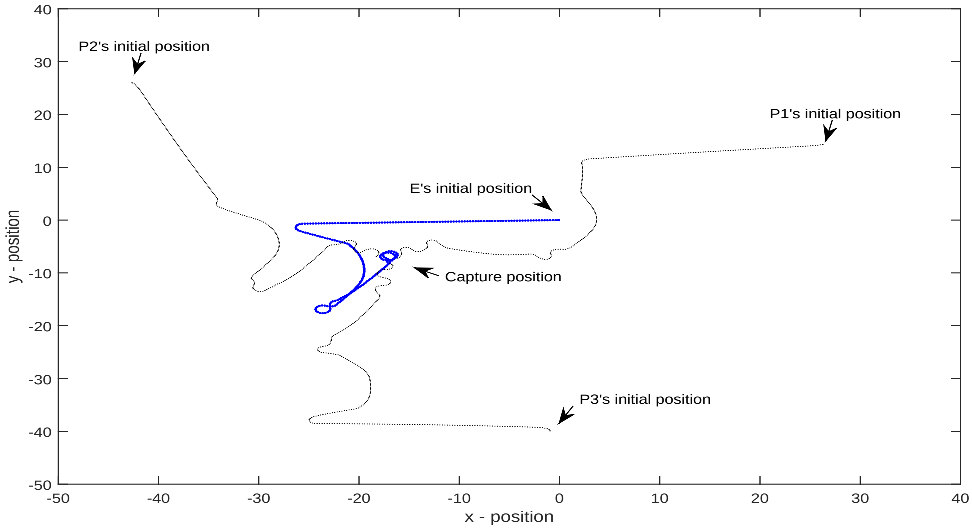

Fig. 12 The paths of the pursuers of Game 2 (thin-line paths) when each pursuer P_o learns its control strategy by the proposed algorithm; the path of the evader (thick-line path) when the evader learns its control strategy by the RGFACL algorithm. The target of the evader here is the position $(x_e^T, y_e^T) = (-500, 0)$.

12. A. Festa and R. B. Vinter, *A decomposition technique for pursuit evasion games with many pursuers*, In Decision and Control (CDC), IEEE 52nd Annual Conference on, pp. 5797-5802, 2013.
13. E. Bakolas, *Evasion from a group of pursuers with double integrator kinematics*, In Decision and Control (CDC), IEEE 52nd Annual Conference on, pp. 1472-1477, 2013.
14. M. Pachter, E. Garcia, and D. W. Casbeer, *Active target defense differential game*, In Communication, Control, and Computing (Allerton), 52nd Annual Allerton Conference on, pp. 46-53, 2014.
15. S. Bhattacharya, T. Basar, and M. Falcone, *Numerical approximation for a visibility based pursuit-evasion game*, In Intelligent Robots and Systems (IROS), IEEE/RSJ International Conference on, pp. 68-75, 2014.
16. N. M. Stiffler and J. M. O'Kane, *A sampling-based algorithm for multi-robot visibility-based pursuit-evasion*, In Intelligent Robots and Systems (IROS), IEEE/RSJ International Conference on, pp. 1782-1789, 2014.
17. D. W. Oyler, P. T. Kabamba, and A. R. Girard, *Pursuit-evasion games in the presence of a line segment obstacle*, In Decision and Control (CDC), IEEE 53rd Annual Conference on, pp. 1149-1154, 2014.
18. I. Exarchos and P. Tsiotras, *An asymmetric version of the two car pursuit-evasion game*, In Decision and Control (CDC), IEEE 53rd Annual Conference on, pp. 4272-4277, 2014.

19. I. Becerra, V. Macias, and R. Murrieta-Cid, *On the value of information in a differential pursuit-evasion game*, In Robotics and Automation (ICRA), IEEE International Conference on, pp. 4768-4774, 2015.
20. W. Lin, Z. Qu, and M. Simaan, *Nash strategies for pursuit-evasion differential games involving limited observations*, Aerospace and Electronic Systems, IEEE Transactions on 51, no. 2, 1347-1356, 2015.
21. Q. Wang and M. Liu, *Learning in hide-and-seek*, IEEE/ACM Transactions on Networking, 2015.
22. W. Scott and N. E. Leonard, *Dynamics of pursuit and evasion in a heterogeneous herd*, In Decision and Control (CDC), IEEE 53rd Annual Conference on, pp. 2920-2925, IEEE, 2014.
23. A. Kumar and A. Ojha, *An evader-centric strategy against fast pursuer in an unknown environment with static obstacles*, In Control, Automation, Robotics and Embedded Systems (CARE), International Conference on, pp. 1-6, IEEE, 2013.
24. J. Dong, X. Zhang, and X. Jia, *Strategies of Pursuit-Evasion Game Based on Improved Potential Field and Differential Game Theory for Mobile Robots*, In Instrumentation, Measurement, Computer, Communication and Control (IMCCC), Second International Conference on, pp. 1452-1456, IEEE, 2012.
25. X, Wang, J. B. Cruz Jr, G. Chen, K. Pham, and E. Blasch, *Formation control in multi-player pursuit evasion game with superior evaders*, In Defense and Security Symposium, International Society for Optics and Photonics, 2007.
26. M. Wei, G. Chen, J. B. Cruz, L. S. Haynes, M. H. Chang and E. Blasch, *A decentralized approach to pursuer-evader games with multiple superior evaders in noisy environments*, In Aerospace Conference, 2007.
27. S. Jin and Z. Qu, *Pursuit-evasion games with multi-pursuer vs. one fast evader*, In Intelligent Control and Automation (WCICA), 8th World Congress, 2010.
28. M. Wei, G. Chen, J. B. Cruz, L. Hayes and M. H. Chang, *A decentralized approach to pursuer-evader games with multiple superior evaders*, In Intelligent Transportation Systems Conference, pp. 1586-1591, 2006.
29. R. Liu and C. Ze-Su, *A novel approach based on evolutionary game theoretic model for multi-player pursuit evasion*, In Computer, Mechatronics, Control and Electronic Engineering (CMCE), International Conference on, Vol. 1, pp. 107-110, 2010.
30. S. Givigi and H. M. Schwartz, *Decentralized learning in multiple pursuer-evader Markov games*, In Control & Automation (MED), 19th Mediterranean Conference on, pp. 1379-1385, 2011.
31. D. Li and J. B. Cruz, *Better cooperative control with limited look-ahead*, In American Control Conference, 2006.
32. D. Li, J. B. Cruz, G. Chen, C. Kwan and M. H. Chang, *A hierarchical approach to multi-player pursuit-evasion differential games*, In Decision and Control, European Control Conference, CDC-ECC'05, 44th IEEE Conference on, pp. 5674-5679, 2005.
33. M. Wei, G. Chen, J. B. Cruz, L. Haynes, K. Pham, K. and E. Blasch, *Multi-pursuer multi-evader pursuit-evasion games with jamming confrontation*, Journal of Aerospace Computing, Information, and Communication, 4(3), 693-706, 2007.
34. Z S. Cai, L. N. Sun and H. B. Gao, *A novel hierarchical decomposition for multi-player pursuit evasion differential game with superior evaders*, In Proceedings of the first ACM/SIGEVO Summit on Genetic and Evolutionary Computation, pp. 795-798, ACM, 2009.
35. F. Bao-Fu, P. Qi-Shu, H. Bing-Rong, D. Lei, Z. Qiu-Bo and Z. Zhaosheng, *Research on high speed evader vs. multi lower speed pursuers in multi pursuit-evasion games*, Information Technology Journal, 11(8), 2012.
36. H. Wang, Q. Yue, and J. Liu, *Research on Pursuit-evasion games with multiple heterogeneous pursuers and a high speed evader*, Control and Decision Conference (CCDC), 27th Chinese, IEEE, 2015.
37. S. Jin and Z. Qu, *A heuristic task scheduling for multi-pursuer multi-evader games*, Information and Automation (ICIA), IEEE International Conference on, 2011.
38. M. Kothari, J. G. Manathara, and I. Postlethwaite, *A Cooperative Pursuit-Evasion Game for Non-holonomic Systems*, World Congress, Vol. 19, No. 1, 2014.
39. H. V. Hasselt and M. Wiering, *Reinforcement learning in continuous action spaces*, Approximate Dynamic Programming and Reinforcement Learning, ADPRL 2007, IEEE International Symposium on, 2007.
40. K. Doya, *Reinforcement learning in continuous time and space*, Neural computation 12.1, 219-245, 2000.
41. W. D. Smart and L. P. Kaelbling, *Practical reinforcement learning in continuous spaces*, ICML, 2000.
42. A. Lazaric, M. Restelli, and A. Bonarini, *Reinforcement learning in continuous action spaces through sequential Monte Carlo methods*, Advances in neural information processing systems, 2007.
43. S. F. Desouky and H. M. Schwartz, *Self-learning fuzzy logic controllers for pursuit-evasion differential games*, Robotics and Autonomous Systems, vol. 59, 22-33, 2011.
44. T. Takagi and M. Sugeno, *Fuzzy identification of systems and its applications to modelling and control*, IEEE Transactions on Systems, Man and Cybernetics SMC-15, 116-132, 1985.
45. R. Isaacs, *Differential Game*, John Wiley and Sons, 1965.
46. S. M. LaValle, *Planning Algorithms*, Cambridge University Press, 2006.
47. S. H. Lim, T. Furukawa, G. Dissanayake and H.F.D. Whyte, *A time-optimal control strategy for pursuit-evasion games problems*, In International Conference on Robotics and Automation, New Orleans, LA, 2004.