A finite-time evasion strategy in pursuit-evasion games using multi-agent reinforcement learning

Yutong Zhu, Ye Zhang, Sihao Sun

***Abstract*— In this paper, we aim to achieve an effective escape strategy for the evader facing cooperative hunting by multiple pursuers in a limited period of time. By introducing a modified reinforcement learning method, the evader's own mobility is exploited to complete the evasion task in an unknown environment. The proposed algorithm is trained and evaluated in a multi-agent pursuit-evasion simulation environment. The results show that the evader is able to complete the escape task efficiently even when the scale of pursuers increases. Finally, the performance of the proposed method is demonstrated by validating it in a semi-physical environment Gazebo and Rviz.**

I. INTRODUCTION

Multi-Agent Systems (MASs) can be non-cooperative, where the agents aim to achieve different goals, such as pursuit-evasion or reach-avoid games. Pursuit-evasion games have been a well-studied topic in many research areas such as game theory, optimal control, behavioural biology [1] and so on for decades. This topic is attractive for a wide range of military and civilian applications, such as missile interception [2], aircraft control [3], search and rescue operations [4], and computer game development [5].

In the current research on pursuit-evasion games, many research have focused on the pursuit conditions and capture strategies of the pursuers [6], [7] in different applications such as formation control [8] and herding strategies [9], [10]. On the other hand, the evasion strategy is equally important for survival when, for example, we need to avoid hostile attackers or prolong capture to complete the mission [1]. Hence, there should a great potential in the research on evaders. However, current research on this aspect is not as much as pursuers. Existing work related to evaders surviving in some ways typically imposes additional constraints, e.g., in aggregation to counteract a single pursuer or group pursuers [11]. Some researchers have only investigated linear or simple path escape strategies of the evader [3]. Therefore, it is essential to explore smarter strategies for the evaders in a pursuit-evasion game.

*This work was supported by the Guangdong Basic and Applied Basic Research Foundation under the Grant No. 2021A1515110569, the National Natural Science Foundation of China under the Grant No. 52202502 and Fundamental Research Funds for the Central Universities under the Grant No. G2021KY05109. (*Corresponding author: Ye Zhang.*)

Yutong Zhu is with the School of Astronautics, Northwestern Polytechnical University, Xi'an 710072, China E-mail: yutong.zhu@mail.nwpu.edu.cn.

Ye Zhang is with the Research & Development Institute of Northwestern Polytechnical University in Shenzhen, Shenzhen 518063, China; the School of Astronautics, Northwestern Polytechnical University, Xi'an 710072, China E-mail: zhang_ye@nwpu.edu.cn.

Sihao Sun is with the Department of Cognitive Robotics, Faculty of Mechanical Engineering, Delft University of Technology, 2629 HS Delft, The Netherlands E-mail: s.sun-2@tudelft.nl.

Furthermore, the complexity of the solution increases with the number of players, making it difficult to obtain an analytical solution with complex and unknown dynamics, which gives rise to learning-based methods, especially reinforcement learning (RL) [12]. In pursuit-evasion games, where at least two players are involved, multi-agent RL (MARL) is applied [13], [14]. An overview of RL for multi-agent pursuit-evasion games can be found in [15]. For example, a concurrent Q-learning method is used in [16] for pursuing and evading agents to learn their optimal policies. In [17], the cooperative pursuit-evasion problem is solved by using communication strategies formulated by RL. Given its advantage in finding optimal solutions in complex and unknown environments, there is a great potential for RL in solving for evasion strategies in a pursuit-evasion game, which is rarely addressed in current studies.

In this paper, a pursuit-evasion game of multiple pursuers chasing one evader is studied. We aim to achieve an effective escape strategy for the evader facing cooperative pursuit by the pursuers in a limited period of time. A novel DQN-based multi-agent pursuit-evasion method (MAPEDQN) is proposed to solve this problem. We first introduce the manoeuvre mechanism to better perform the evasion task in both low and high velocities. Then we utilize a finite-time online learning strategy with the goal of completing evasion task in a complex environment. Meanwhile, we redesign the reward function in order to speed up the training and avoid the problem of untrainability due to high dimensionality. The main contributions of this paper are:

(i) **Manoeuvre mechanism:** based on the DQN algorithm, this paper reconsiders the action space based on the fact that the pursuers adopt a cooperative pursuing strategy and the evader takes its own manoeuvres, which are translated into rewards. Drawing on the iterative process of the DQN algorithm, rewards are resigned to guide the evader to avoid capture by multiple pursuers in a finite time horizon.

(ii) **Potential function:** we introduce the potential function in RL and propose the concepts of position potential and velocity potential. The rewards of the evader are dynamically adjusted from the relative potentials of the pursuers and the evader, which effectively improves the training efficiency.

(iii) **Adaptive composite reward mechanism:** based on the characteristics of dense and sparse rewards, we first set the base action as dense rewards in the pre-training period to enhance the completion rate of the evasion task. In the late stage of the training, the sparse reward is adjusted to enhance the success rate of the evasion task, which strategically avoids the disadvantage of slow training due

to high-dimensional data.

II. Problem Formulation

A. Problem setup

Given a pursuit-evasion game where multiple pursuers try to catch a single evader in a wide, connected area Ω. The positions of the evader and the pursuers are denoted by $g_e(x, y)$, $g_g(x, y)$, which is governed by system dynamics:

$$\begin{aligned} \dot{g}_e &= d,\ g_e(0) = g_e^0 \\ \dot{g}_p^i &= u^i,\ g_p^i(0) = g_p^{i,0},\ i = 1, \ldots, N \end{aligned} \tag{1}$$

where $g_e^0, g_p^{i,0} \in \Omega$ are the initial evader and pursuer positions. The inputs d and u^i is defined in a specific set, which is:

$$\begin{aligned} D &= \{ d | \, \|d\| \leq v_{e,\max} \} \\ U^i &= \left\{ u^i \middle| \, \|u^i\| \leq v_{p,\max} \right\} \end{aligned} \tag{2}$$

where $v_{e,\max}$ and $v_{p,\max}$ are the maximum velocities of the evader and the pursuers, respectively. The evader is considered to have a velocity advantage is when $v_{e,\max} > v_{p,\max}$, and vise versa. The symbol $\|\cdot\|$ denotes the Euclidean norm in $\mathbb{R}^2$. The pursuers' objective is to apprehend the evader by positioning at least one pursuer within a distance $c > 0$ from the evader. The evader can employ a pursuit strategy to elude the pursuers within a finite time t. The environment layout is depicted in Fig.1, with the environment boundaries defined as walls. The blue arrow signifies the relative velocities of the evader and the pursuers, and the angle is denoted by θ_{re}.

Definition 1: Evasion success condition: after escaping for a certain period of time T_e, the evader does not enter the capture range of each pursuer.

$$\begin{aligned} &\forall t \in T_e, \\ &\text{s.t.} \quad \rho\left(g_e, g_p\right) > c \end{aligned} \tag{3}$$

where c denotes the pursuer's capture range and $\rho\left(g_e, g_p\right)$ denotes the evader's Euclidean distance from the pursuer:

$$\rho\left(g_e, g_p\right) = \sqrt{(x_e - x_p)^2 + (y_e - y_p)^2} \tag{4}$$

The goal of RL is to adapt to the action strategy π in accordance with the reward and punishment information during the evader's exploration, so that the agent can receive greater long-term rewards. The long-term reward is the cumulative reward that the learning agent expects to receive in the future after the moment t, which is expressed as

$$R_t = \sum_{k=0}^{T} \omega^k r_{t+k} \tag{5}$$

where $\omega \in [0, 1]$ is the discount factor, which denotes how much the learning subject's decision value the future rewards. When ω is 0, it indicates that the learning subject only considers the rewards that can be obtained in the current step, while $\omega = 1$ indicates that the learning subject values future rewards as much as current rewards. In order to evaluate the strategy π, the state value function is expressed as

$$V^\pi(s) = e\left[R_t | s_t = s, \pi\right] \tag{6}$$

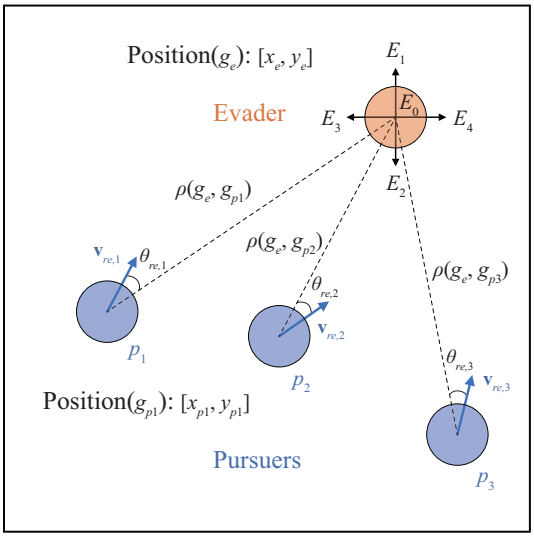


Fig. 1. The simulated environment with three pursuers and one evader. The pursuers are represented by blue circles and the evader is represented by orange circle.

where (6) indicates that the learning subject is in state s_t and following the action strategy π leads to the desired long-term reward. The action value function is expressed as

$$Q^\pi(s, a) = e\left[R_t | \, s_t = s, a_t = a, \pi\right] \tag{7}$$

where $Q^\pi(s, a)$ refers to the expected long-term reward that can be obtained after adopting action a in state s based on strategy π. From the Bellman equation, we have

$$\begin{aligned} V^\pi(s_t) &= e\left[R_t | \, s_t = s\right] \\ &= e\left[r_{t+1} + \omega r_{t+2} + \cdots | \, s_t = s\right] \\ &= e\left[r_{t+1} + \omega \left(r_{t+2} + \omega r_{t+3} + \cdots\right) | \, s_t = s\right] \\ &= e\left[r_{t+1} + \omega \, R_{t+1} | \, s_t = s\right] \\ &= e\left[r_{t+1} + \omega V^\pi \left(s_{t+1}\right) | \, s_t = s\right] \end{aligned} \tag{8}$$

where (8) means that the value function of the current state can be calculated by the value function of the next state. Similarly, the action value function contains the same relation as:

$$Q^\pi(s_t, a_t) = e\left[r_{t+1} + Q^\pi \left(s_{t+1}, a_{t+1}\right) | \, s_t = s, a_t = a\right]. \tag{9}$$

The optimal action value function $Q^*(s, a)$ of the evader is denoted as

$$Q^*(s, a) = \max_\pi e^\pi \left[R_t | \, s_t = s, a_t = a\right]. \tag{10}$$

Therefore, from the Bellman equation, the optimal action value function iteration of the evader can be obtained as

$$Q^*(s, a) = e\left[r_{t+1} + \omega \max_{a_{t+1}} Q^\pi \left(s_{t+1}, a_{t+1}\right) | \, s_t = s, a_t = a\right] \tag{11}$$

By continuously exploring the pursuit-evasion environment and iterating (11) based on the reward values fed back from the environment, the evader can derive the optimal strategy π^* as

$$\pi^* = \arg\max_a Q^*(s, a) \tag{12}$$

namely, the optimal evasion action chosen by the evader based on the optimal policy at a given moment is the one that maximizes the action value function.

In the Q-learning method, the values of $Q(s,a)$ are stored in the Q-table and the Q-table is continuously updated during the exploration process in the following way

$$Q(s,a) \leftarrow Q(s,a)+\varepsilon\left[r+\omega\max_{a_{t+1}}Q(s_{t+1},a_{t+1})-Q(s,a)\right] \tag{13}$$

where ε is the update rate, indicating the magnitude of the update in each iteration.

III. Finite-Time Reinforcement Learning

In this section, we develop a novel DQN-based algorithm for online learning and obtaining the optimal policy in (12) in finite time. Therefore, the proposed MAPEDQN in this paper will redesign a DQN structure with different reward functions reconsidered to enable the evader to escape from multiple pursuers at low and high velocities.

The loss function for MAPEDQN is set to

$$L(\theta)=e\left[Q_{\text{target}}-Q(s_t,a_t;\theta)^2\right] \tag{14}$$

where $Q_{\text{target}} = r+\omega\max_{a_{t+1}}Q(s_{t+1},a_{t+1};\lambda^-)$. θ is the main network parameter in MAPEDQN and λ^- is the target network parameter in MAPEDQN. The main network parameters are updated based on the loss function. The velocity and position of each agent can be determined by:

$$\mathbf{v}(t+1)=\mathbf{v}(t)+\mathbf{a}(t)\cdot\Delta t=\begin{bmatrix} v_x(t)+a_x(t)\cdot\Delta t \\ v_y(t)+a_y(t)\cdot\Delta t \end{bmatrix}^T \tag{15}$$

$$\mathbf{g}(t+1)=\mathbf{g}(t)+\mathbf{v}(t)\cdot\Delta t=\begin{bmatrix} x(t)+v_x(t)\cdot\Delta t \\ y(t)+v_y(t)\cdot\Delta t \end{bmatrix}^T. \tag{16}$$

The state information of the evader and other pursuers is updated at the end of each Δt in the training environment, thus completing the state change of both agents during the training episode. For the escape process of the evader, it is assumed that the position and velocity information of the agents can be observed, such that

$$\mathbf{S}_e=[x_e,y_e,v_{e,x},v_{e,y}] \tag{17}$$

and

$$\mathbf{S}_p^i=\left[x_p^i,y_p^i,v_{p,x}^i,v_{p,y}^i\right] \tag{18}$$

then the state space in the pursuit-evasion learning environment when facing i-th pursuer is designed as:

$$\mathbf{s}=\left[\mathbf{S}_e,\mathbf{S}_p^1,\mathbf{S}_p^2,\ldots,\mathbf{S}_p^n\right] \tag{19}$$

where each agent state information is 4-dimension, the dimension of the state space to be observed by all agents in the presence of n pursuers is $4(n+1)$. We define the action space of the evader as

$$E=\{E_0,E_1,E_2,E_3,E_4\} \tag{20}$$

where E_0,E_1,E_2,E_3,E_4 represent the five types of actions in the evader's escape flight action library: no-drive state, forward-drive, backward-drive, left-drive and right-drive, respectively. In the pursuit-evasion environment, the evader carries out the escape movement through the five basic flight actions. The improved sparse reward function of the evader in the pursuit-evasion environment is designed as

$$r_c=\{-c_1,c_2,0\} \tag{21}$$

where c_1,c_2 is a positive constant value.

A repulsive field is generated by the pursuer, and the change in potential energy of the evader under the potential field of the pursuer is taken as the reward value for each training time step. According to the traditional repulsive field function, the potential field strength at the evader is small when the evader is far away from the pursuer, and it is difficult to give the evader a reasonable reward feedback value based on the relative position change information. To solve this problem, we define the improved position potential field function as

$$U_g(p)=\begin{cases}\delta\left(\dfrac{1}{\rho(g_e,g_p)}-\dfrac{1}{\rho_0}\right), & \rho(g_e,g_p)\le\rho_0 \\ 0, & \rho(g_e,g_p)>\rho_0\end{cases} \tag{22}$$

where δ is a constant of the potential field coefficients with a value greater than zero. ρ_0 denotes the radius of influence of this pursuer beyond which there will be no repulsion of the pursuer against the evader. During the training, if the relative distance between the evader and that pursuer decreases, then the potential energy of the evader under the repulsive field generated by that pursuer will increase. The smaller the relative distance between the evader and the pursuer, the faster the potential energy of the evader changes. Then the action that causes the relative distance between the evader and the pursuer to decrease at this point should be given a greater negative reward. Therefore, the reward function for relative position change is designed as follows

$$r_g=-\sum_{i=1}^{n}\Delta U_g^i(p) \tag{23}$$

where $\Delta U_g(p)$ is the change in potential energy of the evader under the action of the repulsive field generated by this pursuer before and after performing a training time step.

The relative velocity change between the evader and pursuer before and after each time step should also be taken into account in the reward function design. When the pursuer's velocity direction points to the evader, its threat increases with the velocity, which gives rise to a velocity potential field defined as

$$U_v(p)=\begin{cases}\beta\left|\mathbf{v}_{re}\right|\cos\theta_{re}, & \rho(g_e,g_p)\le\rho_0 \\ 0, & \rho(g_e,g_p)>\rho_0\end{cases} \tag{24}$$

where β is the coefficient constant, $\mathbf{v}_{re}$ denotes the relative velocity of the evader and the pursuer. $\theta_{re}\in(-\theta,\theta)$ means the angle between the relative distance and the relative velocity of the pursuer and the evader, which is

$$\cos\theta_{re}=\frac{(g_e-g_p)\cdot(\mathbf{v}_p-\mathbf{v}_e)}{|g_e-g_p|\cdot|\mathbf{v}_p-\mathbf{v}_e|}. \tag{25}$$

Algorithm 1 MAPEDQN

Input: Initial air resistance coefficient k_d
Initial mass m and velocity $\mathbf{v}$
Sparse reward constants c_1, c_2
Hyperparameters ε, θ, w
Bias value d

procedure
while p **do**
  for $k = 0, 1, \ldots$ **do**
    $Q_{\text{target}} \leftarrow r + \omega \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}; \lambda^-)$
    $L(\theta) \leftarrow e\left[Q_{\text{target}} - Q(s_t, a_t; \theta)^2\right]$
    $Q(s,a) \leftarrow Q(s,a) +$
    $\varepsilon\left[r + \omega \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s,a)\right]$
    $\pi \leftarrow \arg\max_a Q(s,a)$
    Update optimal strategy with measurement $Q^*(s,a)$
  end for
end while
end procedure

The greater the strength of the potential field at the evader in the relative velocity potential field, the faster that pursuer is approaching the evader and the more critical the evader's situation is. Negative reward feedback should be given to the action that makes the evader's potential energy increase in the relative velocity potential field. Therefore, the reward function for relative velocity change is designed as

$$r_v = -\sum_{i=1}^{n} \Delta U_v^i(p) \tag{26}$$

where $\Delta U_v(p)$ is the change in potential energy of the evader under the action of the relative velocity potential field generated by the pursuer before and after performing a training time step.

In the multiple pursuers problem, relying only on the relative position reward and relative velocity reward cannot give a reasonable reward value for this change in security posture. Therefore, we introduce a new single-step reward factor and define the centre point of the pursuit as

$$g_{ce} = \frac{1}{n}\sum_{i=1}^{n} g_p^i = \frac{1}{n}\sum_{i=1}^{n}\left[x_p^i, y_p^i\right] \tag{27}$$

and we define the pursuit centre change reward as

$$r_{ce} = \Delta\rho\left(g_e, g_{ce}\right) \tag{28}$$

where $\Delta\rho\left(g_e, g_{ce}\right)$ is the change in the evader's distance from the centre point of the pursuit before and after performing a training time step. Flight movements away from the centre of the pursuit are positively rewarded. At each time step during the training process, the evader is considered to be surrounded when the positional relationship between the evader and the pursuer simultaneously satisfies the following conditions

$$\begin{aligned} &\min\left\{x_p^1, x_p^2, \ldots, x_p^n\right\} \le x_e \le \max\left\{x_p^1, x_p^2, \ldots, x_p^n\right\} \\ &\min\left\{y_p^1, y_p^2, \ldots, y_p^n\right\} \le y_e \le \max\left\{y_p^1, y_p^2, \ldots, y_p^n\right\} \end{aligned} \tag{29}$$

Negative or positive rewards are given when the evader is in or out of a surrounded situation, according to the formula

$$r_{su} = \{w, -w\} \tag{30}$$

where w is a hyperparameter. In summary, the reward function of the evader escape process under multiple pursuers is designed as

$$r = \theta_1 r_g + \theta_2 r_v + \theta_3 r_c + \theta_4 r_{ce} + \theta_5 r_{su} + d \tag{31}$$

where $\theta_1, \theta_2, \theta_3, \theta_4, \theta_5$ are constant coefficients, which are used to adjust the degree of influence of each incentive. d is the bias.

The procedure of learning of evasion strategies in an unknown environment is summarized in Algorithm 1.

IV. Simulation

In this section, the simulation is used to verify the effectiveness of the proposed MAPEDQN algorithm. In addition, we discuss the cases where the evader is faster than the pursuers and the evader is slower than or equal to the pursuers. For each agent, we set its mass $m = 3.8$, gravitational acceleration $g = 9.8$, air resistance coefficient $k_d = 1.27e - 1$, bias value $d = 1.5/n$, discount factor $\omega = 0.9$, the radius of influence of the pursuers $\rho_0 = 15$, learning rate 0.1 and exploration factor $\varepsilon = 0.5$. In order to demonstrate the ability of the proposed algorithm exploring unknown environments, we set the initial positions of the pursuers and the evader to be random in a finite square region of $[-1500, 1500] \times [-1500, 1500]$. A time horizon of 400s is set as the maximum evasion time. The evader loses if any pursuer touches the boundary of the evader (denoted as a green circle in Fig.2) before the evasion time runs out.

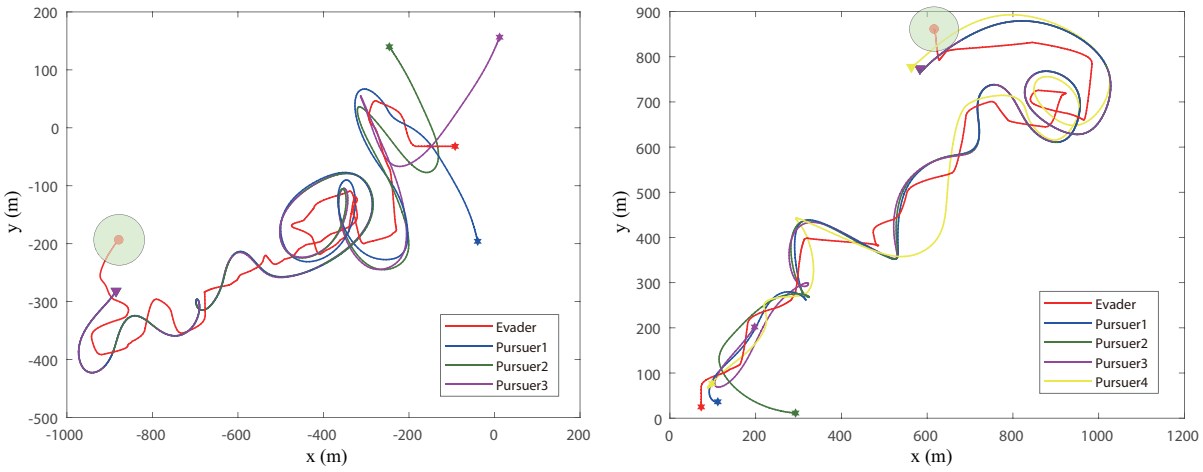
A. The evader is slower than pursuers

Firstly, we perform simulations in slower evader ($v_e < v_p$). Two typical examples of 3 or 4 pursuers v.s. 1 evader are shown in Fig.2. In Fig.2(a), three pursuers fail to reach the boundary of the evader before 400s. After randomly resetting the initial positions of the pursuers and evaders, the evader can still succeed when the number of pursuers increases to 4, as shown in Fig.2(b). Inevitably, we need to further increase the training time so that the evader satisfies the escape success condition at the set time, as shown in Fig.2(b).

B. The evader is faster than pursuers

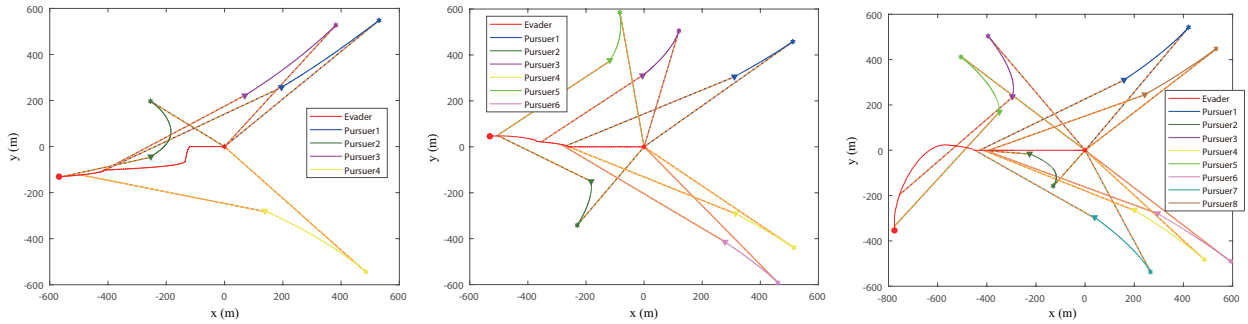
Since the escape task is easier to complete when the evader is faster. Therefore, we adjust the conditions for completing the escape task to increase the difficulty of its escape. A certain initial distance between the evader and the pursuers is given, and the escape task will be considered to be completed only when the distance between the evader and each pursuer is greater than their initial distances after the evader has moved for a certain period of time, otherwise the training continues.

In order to avoid the problem of slow learning or difficulty in effective learning due to sparse rewards, we design the dense rewards of mobility in each direction for training, and



(a) 1 v.s. 3 (b) 1 v.s. 4

Fig. 2. A slower evader with three and four pursuers. Star marker is the start point, triangle mark is the end point, and the green area is the escape range.



(a) 1 v.s. 4 (b) 1 v.s. 6 (c) 1 v.s. 8

Fig. 3. A faster evader with increasing number pf pursuers. The dashed lines connect each pursuer and evader at their initial and ending location to indicate the completion of the evasion task.

the evader still has difficulty in completing the escape task to get rid of the pursuer's capture after a period of time of movement. Based on this, we redesigned rewards as sparse rewards for successful and failed episodes of the escape task after a period of training. The trained evader was able to complete the escape task in Fig.3.

V. Experiment

We demonstrate the effectiveness of the MAPEDQN algorithm applied to multiple ground robots in a semi-physical environment Gazebo and Rviz. All processing is done in realtime, and by using MAPEDQN, the ground robot pursuer team pursues the evader. All experiments were run using Gazebo and Rviz, on an Intel i7-13700K CPU with 32GB of RAM, and an NVIDIA RTX 3070Ti GPU.

The ground robots complete their tasks in a simulated environment with an enclosure 3000m long and 3000m wide. The starting position is random. In contrast to the simulation, this section uses real ground robots in the simulation, considering their size and steering to construct a map of the environment using Rviz.

Fig.4 shows a snapshot of the top-down view of the map, and the trajectory fades with the movement of the ground robot. The red trajectories are the evading robots, and the other colours are the pursuing robots. By using the proposed MAPEDQN algorithm, the evader tries to avoid capture by the pursuer through its own manoeuvrability, which is similar to what we have done in the simulation. However, experimental simulations also show some limitations of MAPEDQN, which should be the basis for future research.

VI. Conclusion

In this paper, we study the problem of a single evader facing multiple pursuers to avoid capture in a multi-agent pursuit-evasion game. In order to cope with the different motion states of the evader, we consider the problem in two cases, slower and faster evader, respectively. By introducing a learning-based algorithm that exploits the mobility of the agents themselves, the escape task in an unknown environment for a finite period of time is achieved. The proposed algorithm has been trained and evaluated in a multi-agent pursuit-evasion simulation environment. The results show that the evader is able to perform the escape task efficiently against multiple pursuers in two motion states. Finally, the task completion was verified under tests with multiple numbers of pursuers to demonstrate the performance of the algorithm. In the future, we will extend the simulation

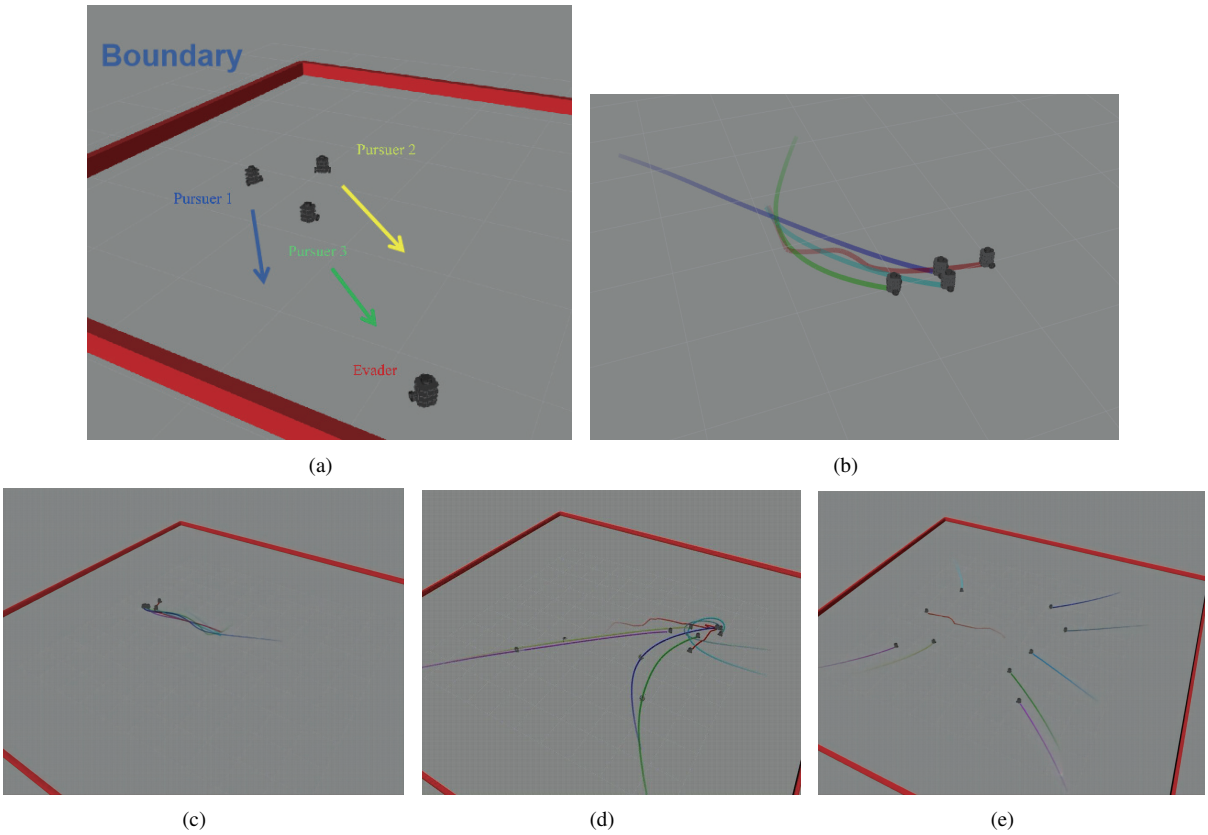


Fig. 4. Snapshots of the map, escape and pursuit trajectories of ground robots. (a) Ground robot model. (b)-(e) Evader avoids the capture by multiple pursuers.

from a 2D to a 3D environment and exploit more efficient algorithms.

REFERENCES

[1] A. Hedenström and M. Rosén, "Predator versus prey: on aerial hunting and escape strategies in birds," *Behavioral Ecology*, vol. 12, no. 2, pp. 150–156, 2001.

[2] V. Turetsky and J. Shinar, "Missile guidance laws based on pursuit–evasion game formulations," *Automatica*, vol. 39, no. 4, pp. 607–618, 2003.

[3] J. M. Eklund, J. Sprinkle, and S. S. Sastry, "Switched and symmetric pursuit/evasion games using online model predictive control with application to autonomous aircraft," *IEEE Transactions on Control Systems Technology*, vol. 20, no. 3, pp. 604–620, 2011.

[4] D. W. Oyler, P. T. Kabamba, and A. R. Girard, "Pursuit–evasion games in the presence of obstacles," *Automatica*, vol. 65, pp. 1–11, 2016.

[5] M. Xu, Z. Pan, H. Lu, Y. Ye, P. Lv, and A. El Rhalibi, "Moving-target pursuit algorithm using improved tracking strategy," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 2, no. 1, pp. 27–39, 2010.

[6] I. E. Weintraub, M. Pachter, and E. Garcia, "An introduction to pursuit-evasion differential games," in *2020 American Control Conference (ACC)*. IEEE, 2020, pp. 1049–1066.

[7] Z. Mu, J. Pan, Z. Zhou, J. Yu, and L. Cao, "A survey of the pursuit–evasion problem in swarm intelligence," *Frontiers of Information Technology & Electronic Engineering*, vol. 24, no. 8, pp. 1093–1116, 2023.

[8] P. Zhou and B. M. Chen, "Distributed optimal solutions for multia-gent pursuit-evasion games for capture and formation control," *IEEE Transactions on Industrial Electronics*, 2023.

[9] R. A. Licitra, Z. I. Bell, E. A. Doucette, and W. E. Dixon, "Single agent indirect herding of multiple targets: A switched adaptive control approach," *IEEE Control Systems Letters*, vol. 2, no. 1, pp. 127–132, 2017.

[10] V. S. Chipade and D. Panagou, "Multiagent planning and control for swarm herding in 2-d obstacle environments under bounded inputs," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1956–1972, 2021.

[11] M. Chen, Z. Zhou, and C. J. Tomlin, "A path defense approach to the multiplayer reach-avoid game," in *53rd IEEE conference on decision and control*. IEEE, 2014, pp. 2420–2426.

[12] N.-M. T. Kokolakis and K. G. Vamvoudakis, "Safety-aware pursuit-evasion games in unknown environments using gaussian processes and finite-time convergent reinforcement learning," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[13] Z. Sui, Z. Pu, J. Yi, and X. Tan, "Path planning of multiagent constrained formation through deep reinforcement learning," in *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2018, pp. 1–8.

[14] Y. Wang, H. He, and C. Sun, "Learning to navigate through complex dynamic environment with modular deep reinforcement learning," *IEEE Transactions on Games*, vol. 10, no. 4, pp. 400–412, 2018.

[15] E. Yang and D. Gu, "Multiagent reinforcement learning for multi-robot systems: A survey," tech. rep, Tech. Rep., 2004.

[16] A. T. Bilgin and E. Kadioglu-Urtis, "An approach to multi-agent pursuit evasion games using reinforcement learning," in *2015 International Conference on Advanced Robotics (ICAR)*. IEEE, 2015, pp. 164–169.

[17] Y. Wang, L. Dong, and C. Sun, "Cooperative control for multi-player pursuit-evasion games with reinforcement learning," *Neurocomputing*, vol. 412, pp. 101–114, 2020.