Preparation of Papers for AIAA Technical Journals

First A. Author * and Second B. Author Jr.[†]
Business or Academic Affiliation 1, City, State, Zip Code

Third C. Author[‡]
Business or Academic Affiliation 2, City, Province, Zip Code, Country

Fourth D. Author[§]
Business or Academic Affiliation 2, City, State, Zip Code

In this paper, a whole novel method, called manoeuvrability enhanced reinforcement learning via gaussian process (MERL-GP), is proposed to deal with problems including escape strategy, local optima, and uncertainty for multi-robot high-dimensional data. MERL-GP contains manoeuvrability action, composite reward mechanism, and gaussian process. Specifically, manoeuvrability action provides more escape strategies. Composite reward mechanism overcomes the sparse reward and local optima problems. Gaussian process approximation solves the Q-function and allows an accurate online update of the parameters of the posterior mean and covariance. Simulation and experiment results on escape tasks for ground and aerial robots demonstrate the effectiveness and robustness of our method.

Nomenclature

g_{pi}, g_e	=	positions of the pursuer i and evader
r_d	=	detection range
r_c	=	communication range
N	=	number of the pursuers
v_{pi}, v_e	=	velocities of the pursuer i and evader
m_{pi}, m_e	=	mass of the pursuer i and evader
H_{pi}, H_e	=	driving force of the pursuer i and evader
$\epsilon_{pi}, \epsilon_e$	=	noise of the pursuer i and evader
S^t	=	environment state
A_p^t	=	action space of the pursuer

*Insert Job Title, Department Name, Address/Mail Stop, and AIAA Member Grade (if any) for first author.
†Insert Job Title, Department Name, Address/Mail Stop, and AIAA Member Grade (if any) for second author.
‡Insert Job Title, Department Name, Address/Mail Stop, and AIAA Member Grade (if any) for third author.
§Insert Job Title, Department Name, Address/Mail Stop, and AIAA Member Grade (if any) for fourth author (etc.).

o_{pi}^t	=	observation received by pursuer i
π	=	training policy
$J(\pi_p)$	=	expected discounted returns of the pursuer
R_p, R_e	=	long-term rewards of the pursuer and evader
γ	=	discount factor
$Q(s, a_e), Q(s, a_e)^*$	=	value function and optimal value function
$\bar{Q}, \Delta Q$	=	mean and random zero-mean residual
$V(s^t), V(a_e^t)$	=	state and action value function
ε	=	update rate
$\mathcal{N}$	=	normal distribution
θ	=	main network parameter
λ	=	objective network parameter
E	=	manoeuvrability action
δ	=	potential field coefficient
ρ	=	euclidean distance between the pursuer i and evader

I. Introduction

Interaction in multi-robot systems is widely known in nature [1] and various human activities, including missile guidance [2], autonomous aircraft [3], and emergency rescue [4]. To better understand and predict potential outcomes of interactions among multiple robots, each robot has its own goals and preferences. Thus it is essential to have potential decision-making mechanisms among them. Uncertain interactions among robots can be studied within the framework of dynamic non-zero-sum multi-player games. A special class of such problems is pursuit-evasion games, where multiple players seek to capture or evade each other. In this paper, we consider steering the evader to avoid capture by multiple pursuers while the pursuer applies multiple pursuit strategies. Finding exact solutions to such pursuit-evasion games can be a complex task due to the high dimensionality of the problem. Specifically, there are difficulties in learning complex strategies in high-dimensional state and action spaces as the number of robots and spatial dimensions increases [5–7].

Multi-robot pursuit-evasion games are usually considered as an optimization problem with the objective of minimizing time and energy costs. Time-optimal geometric forms of pursuit-evasion strategies are studied in Refs. [8, 9], and optimal solutions are provided. In Ref. [10], a method is proposed to deal with the pursuit-evasion differential game in the presence of disturbance in dynamic flow-field with one evader and multiple pursuers. The special case where the pursuer is faster than the agile evader is studied from a bionic perspective in Ref. [11].

Moreover, in the current research on pursuit-evasion games, most of the studies have focused on the pursuit conditions

and capture strategies of the pursuers, such as Refs. [12–16]. Whereas in studies that focus on the evader, such as Refs. [11, 17]. The optimal escape strategy for two pursuers faced by an evader in two scenarios where the pursuers can have access to the position-velocity information of the evader and only to the position information in Ref. [17].

Recently, multi-agent reinforcement learning (MARL) has made great progress in pursuit-evasion scenarios. A number of MARL methods are proposed, but they have their own disadvantages. Multi agent deep deterministic policy gradient (MADDPG) [18], which employs a centralized training framework with decentralized execution to enhance the cooperative behaviour of agents in a hybrid cooperative competitive environment. Similarly, the cooperative pursuit problem is addressed by formulating a communication strategy using an improved RL-based method (MARL-ring and MARL-line) in Ref. [19]. Although MADDPG, MARL-ring and MARL-line can improve the cooperative ability of robots, these methods do not apply to environments with a large number of robots as they require the use of all the robots' states or observations in constructing their critic networks. As the number of robots increases, these methods will become increasingly difficult to be trained.

To address these problems, Q-learning has been taken into consideration and applied to pursuit-evasion scenarios, especially for the evader escape strategy [20]. Inspired by its potential, some Q-learning-based methods are applied to game problems in Refs. [21, 22]. In addition, the design of manoeuvrability action [23] and reward allocation [24] in a game strategy is critical to the effective implementation of the strategy. For stochasticity and uncertainty in the presence of disturbances and noise, learning-based frameworks via gaussian processes have been widely used for safety-guaranteed control systems in Refs. [25–27]. However, these research methods are not satisfactory for solving problems with high dimensionality.

To address the limitation of the above problems, a manoeuvrability enhanced reinforcement learning (MERL) is proposed and additional Gaussian Process (MERL-GP) is used to enhance the efficiency of high dimensionality, which includes manoeuvrability action, composite reward mechanism, and gaussian process. The main contributions of this paper are summarized as follows:

1) MERL provides high probability escape strategies for the evader facing the pursuers with kinds of strategies in a pursuit-evasion scenario and solves the problem of slow learning efficiency. Manoeuvrability action provides more escape strategies potential and composite reward mechanism overcomes the sparse reward and local optimum problems.
2) MERL-GP simultaneously solves the problem of difficulty in solving high-dimensional pursuit-evasion games and difficulty in training with high-dimensional data by solving the Q-function through the approximation via GP, which implies that it reduces the difficulty of training due to the increase in the number of robots and can be scaled up with a large number of robots in training.
3) For stochasticity and uncertainty of the system, the value function is learnt by GP regression using Monte Carlo samples with discounted reward as our target, which allows for an accurate online update of the parameters of the

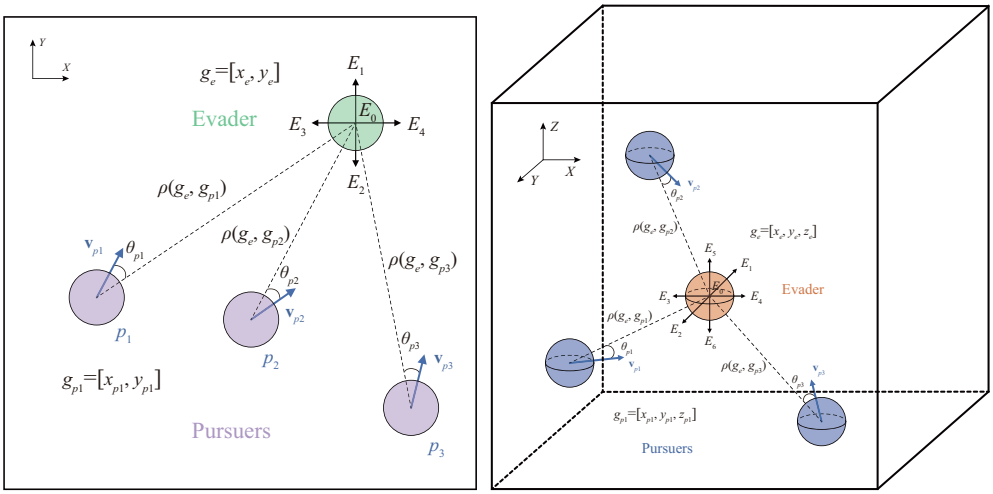
posterior mean and covariance.

The paper is organized as follows. Section II gives problem formulation. The preliminaries are described in Sec. III. The methodology of MERL and MERL-GP to use manoeuvrability action, composite reward mechanism, and GP is presented in Sec. IV. The method is demonstrated and validated through simulation shown in Sec. V. In Sec. VI, semi-physical experiment is carried out in several scenarios. Section VII concludes the paper.

II. Problem Formulation

In a general pursuit-evasion game, ground robots in two-dimensional space are considered in this paper to evaluate our method. Subsequently, our method can be naturally extended to aerial robots in three-dimensional space as well.

As shown in Fig. 1, robots represented by circles or balls with colours move in a rectangular or cuboid map. The boundary condition for the environment is the each side of the map. The boundary is considered as an obstacle. Each robot will be penalised for crossing the boundary. In addition, all robots are randomly generated in a map where the detection range of a robot is denoted as r_d and the communication range is denoted as r_c.



(a) Ground robots (b) Aerial robots

Fig. 1 Illustration of ground robots and aerial robots in a pursuit-evasion game.

In this paper, each robot can only obtain the state of other robots within the detection range r_d. The environment of this setting is considered partially observable [28]. If a robot wants to obtain the states of other robots beyond the detection range, it needs to communicate with the other robots. For more details on the communication setup refer to Ref. [29]. Communication is only allowed to occur with pursuers or evaders in the same team.

In Fig. 1(a), we consider a pursuit-evasion 2D environment with N pursuers and an evader. Let $g_e = [x_e, y_e]$ and $g_{pi} = [x_{pi}, y_{pi}]$ denote the positions of the evader and pursuer i, and $v_e = [v_e^x, v_e^y]$ and $v_{pi} = [v_{pi}^x, v_{pi}^y]$ denote the velocities of the evader and pursuer i, respectively. θ_{pi} denotes the angle between the line-of-sight and velocity of the

pursuer i and evader. In addition, S^t denotes the state of the environment at time t, including the velocity and position information of all robots. The representation of the 3D environment is similar and its details are shown in Fig. 1(b).

Besides, the dynamics model of each robot in this paper is a double-integrator model

$$\begin{aligned} \dot{g}_{pi} &= v_{pi}, \quad \dot{g}_e = v_e \\ \dot{v}_{pi} &= \frac{H_{pi}}{m_{pi}} + \epsilon_{pi}, \quad \dot{v}_e = \frac{H_e}{m_e} + \epsilon_e \end{aligned} \tag{1}$$

where m_{pi} and m_e are the mass of pursuer i and evader, $H_{pi} = (H_{pi}^x, H_{pi}^y)$ or $H_{pi} = (H_{pi}^x, H_{pi}^y, H_{pi}^z)$ and $H_e = (H_e^x, H_e^y)$ or $H_e = (H_e^x, H_e^y, H_e^z)$ denote the driving force applied to the pursuer i and evader in the 2D $x-y$ axis, ϵ_{pi} and ϵ_e denote the possible noise, respectively. In addition, (H^x, H^y) or (H^x, H^y, H^z) is also the action of robots. The direction and the magnitude of the force are discrete. The stacking of acceleration is caused by force in multiple directions that make the velocities of robots continuous.

III. Preliminaries

A. Partially Observable Markov Decision Process

Since each pursuer can only observe its own state and environment information within the pursuer detection range, the problem in this paper can be regarded as partially observable Markov decision process (POMDP), which is an extension of the Markov process. It is defined by the environment state S^t, the action space $A_p^t = \left[a_{p1}^t, \ldots, a_{pN}^t\right]$, where a_{pi}^t is the action H_{pi} of the pursuer. The observation received by pursuer i at time t is defined as o_{pi}^t. Each pursuer i learns a policy $\pi_p : o_{pi}^t \rightarrow A_p^t$ which maps each pursuer's observation to a distribution over its set of actions. Subsequently, the next states are obtained by the transition function $T : \left[S^t \times a_{p1}^t \times \ldots \times a_{pN}^t\right] \rightarrow S^t$. Each pursuer i can obtain rewards R_p as a function of state and action spaces. The aim of the pursuers is to maximize the expected discounted returns

$$J(\pi_p) = \mathbb{E}_{a_p \sim \pi_p, S \sim T}\left[\sum_{t=0}^{\infty} \gamma^t r_{pi}^t \left(S^t, a_{pi}^t, \ldots, a_{pN}^t\right)\right] \tag{2}$$

where r_{pi}^t denotes the reward that the pursuer i obtains at time t and $\gamma \in [0, 1]$ represents the discount factor acting on the importance of future rewards.

B. Reinforcement Learning for Pursuit-Evasion Games

In our game, we set up a larger number of pursuers, which means that pursuers are a greater threat to the evader. Then the evader is lighter than the pursuers, which implies that the evader has better potential mobility optimization. The pursuers are driven by two capture strategies, one is a random strategy and the other is a pre-trained learning strategy. The evader does not know the pursuit strategy. The pursuer has only partial observation of the environment.

The Q-learning algorithm [30], a data-based RL technique, is novelly used as a method to generate better evasion strategies [20] by identifying possible game patterns in multi-robot games. In this paper, Q-learning is used as the basic training method. The goal of the learning is to dynamically adjust the action policy π_e based on the evader's reward and penalty information during exploration, so that the evader can obtain greater long-term rewards

$$R_e^t = \sum_{k=0}^{\infty} \gamma^k r^{t+k} \tag{3}$$

At time t, the evader obtains different long-term rewards according to each action policy. In order to evaluate the policy π_e, a value function $Q(s^t, a_e^t)$ is introduced, including state value function $V(s^t) = \mathbb{E}_\pi \left[R_e^t | s^t\right]$ and action value function $V(a_e^t) = \mathbb{E}_\pi \left[R_e^t | a_e^t\right]$, where $V(s^t)$ indicates that the learning subject is in the state of S^t and follows the action policy π_e and $V(a_e^t)$ refers to the expected long term rewards that can be gained by taking the action a_e^t based on the state of the policy π_e. In addition, the optimal state-action value function of the evader is denoted as

$$\begin{aligned} Q^*(s, a_e) &= \max_{\pi_e} \mathbb{E}_{\pi_e} \left[R_e^t\right] \\ &= \mathbb{E}_{\pi_e} \left[r^t + \gamma^t \max_{a_e^t} Q(s^t, a_e^t)\right] \end{aligned} \tag{4}$$

By continuously exploring the pursuit-evasion environment and iterating Eq. (4) based on the reward values fed back from the environment, the evader can derive the optimal strategy $\pi^* = \arg\max_{a_e^t} Q^*(s^t, a_e^t)$. Therefore, the optimal evasion action chosen by the evader based on the optimal policy at a given time is the one that maximizes the value function. Moreover, the Q-value is continuously updated during the exploration process in the following way

$$Q(s, a_e) \leftarrow Q(s, a_e) + \varepsilon^t \left[r^t + \gamma^t \max_{a_e^t} Q(s^t, a_e^t) - Q(s, a_e)\right] \tag{5}$$

where ε denotes the update rate.

In this study, we focus on an escape strategy rather than a pursuit strategy. The goal is to develop a manoeuvrability escape strategy that is applicable to a multi-robot high-dimensional data environment. Escape means that the evader will not be collided by the pursuers or will not be confined to a small area from which it cannot escape. For the setup dealing with manoeuvring actions and high-dimensional data, details will be presented in Sec. IV.

C. Gaussian Process

As a typical model-free approach, RL allows agents to learn and improve its performance based on observation and reward [31]. However, the high dimensionality of the robot data information makes learning slow and uncertainty increases. To achieve effective multi-robot RL training, GP for stochasticity and uncertainty is introduced.

GP is the extension of a multivariate Gaussian distribution to infinite number of random variables [32]. It

can be regarded as a distribution over random functions, with any finite sub-collection of random variables in GP following a multivariate Gaussian distribution. Specifically, if any subset of random variables $\{g_{pi}, \ldots, g_{pN}\}$ and their corresponding $\{h(g_{p1}), \ldots, h(g_{pN})\}$ follow the distribution

$$\begin{bmatrix} h(g_{p1}) \\ \vdots \\ h(g_{pN}) \end{bmatrix} \sim \mathcal{N}\left[\begin{pmatrix} m(g_{p1}) \\ \vdots \\ m(g_{pN}) \end{pmatrix}, \begin{pmatrix} k(g_{p1}, g_{p1}) & \ldots & k(g_{p1}, g_{pN}) \\ \vdots & \ddots & \vdots \\ k(g_{pN}, g_{p1}) & \ldots & k(g_{pN}, g_{pN}) \end{pmatrix}\right] \tag{6}$$

follows GP, which can be denoted as

$$h(\cdot) \sim \mathcal{GP}\left(m(\cdot), k(\cdot, \cdot)\right) \tag{7}$$

where $m(\cdot)$ and $k(\cdot, \cdot)$ are the mean function and covariance function, respectively.

In general, any real-valued function can serve as the mean function for GP. However, a covariance function, also known as kernel matrix, is usually required to be positive semi-definite. If such requirement is met, GP can be regarded as a kernel-based probability distribution over function $h(\cdot)$, where the variance $k(\cdot, \cdot)$ encodes the noise of the environment from uncertainty. Furthermore, the bivariate normal distribution [33] which will be exploited in Sec. IV is introduced

Lemma 1 *Bivariate Normal Distribution: Let $\mathcal{N}$ denote the normal distribution, and let X and Y be random vectors distributed as*

$$\begin{bmatrix} X \\ Y \end{bmatrix} \sim \mathcal{N}\left[\begin{pmatrix} \mathcal{A}_x \\ \mathcal{A}_y \end{pmatrix}, \begin{pmatrix} k_{ii} & k_{ij} \\ k_{ji} & k_{jj} \end{pmatrix}\right] \tag{8}$$

then we have $X|Y \sim \mathcal{N}(\bar{X}, P)$ with

$$\begin{aligned} \bar{X} &= \mathcal{A}_x + k_{ij} k_{jj}^{-1} (Y - \mathcal{A}_y) \\ P &= k_{ii} - k_{ij} k_{jj}^{-1} k_{ji} \end{aligned} \tag{9}$$

where P is the schur complement of k_{ii} in the partitioned matrix $\begin{pmatrix} k_{ii} & k_{ij} \\ k_{ji} & k_{jj} \end{pmatrix}$.

IV. Methodology

In this section, the overall structure of MERL-GP is first described. Then, the details of the components of MERL-GP are given, including the manoeuvrability action, the composite reward mechanism, and the high-dimensional data processing including Gaussian process for stochasticity and uncertainty.

A. Overall Structure

The overall structure of our method is given in Fig. 2. The historical time series data of all robots are considered as inputs. Inputs from unknown environments are read by the MERL controller and GP, where they are directly used as observation in the case of low-dimensional data, and additionally in the case of high-dimensional data, the value function is updated by passing it through the GP's state estimator and approximating it. Meanwhile, the rewards from the GP and MERL controller are embedded in the value function through the composite reward mechanism to obtain a policy. The escape strategy generated by the manoeuvrability action returns to the robot in the environment. The whole process is trained in an end-to-end manner with backpropagation. We now discuss the components in MERL-GP.

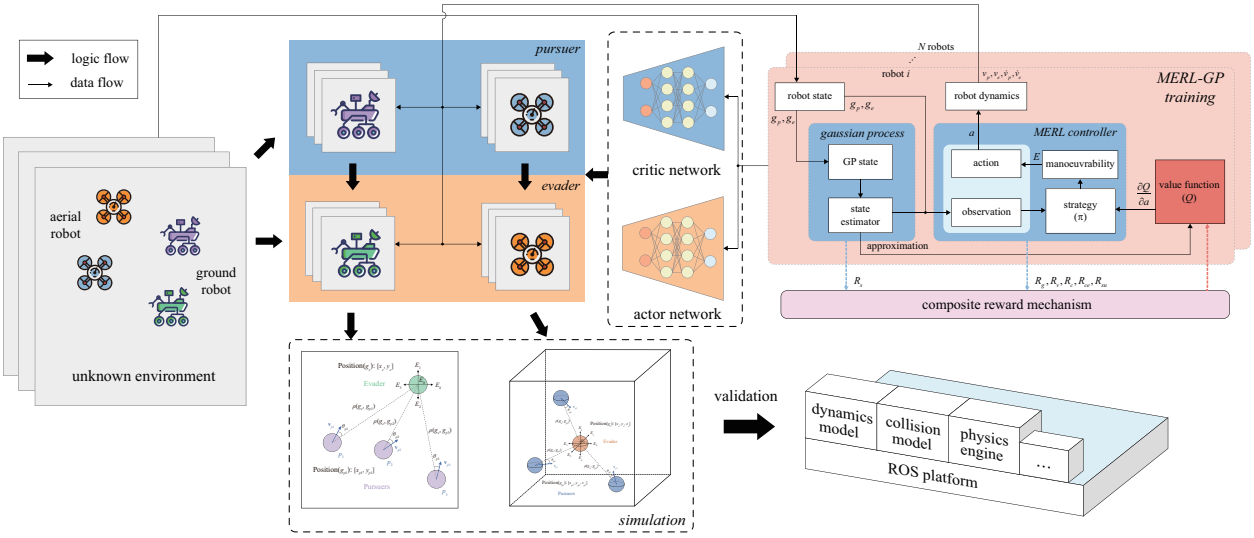


Fig. 2 An overall overview of the framework, including the architecture of MERL-GP and the application of multi-robot pursuit-evasion games.

B. Manoeuvrability Action

As described in Sec. III, according to Eq. (5) the loss function is set as

$$L(\theta) = \mathbb{E}\left[Q_{\text{obj}} - Q(s^t, a^t, \theta)\right] \tag{10}$$

where θ is a main network parameter. The policy π_e is parameterised by the parameter θ of the neural network. By directly adjusting the parameter θ, it maximizes the objective function

$$Q_{\text{obj}} = r^t + \gamma^t \max_{a_e^t} Q(s^t, a_e^t, \lambda) \tag{11}$$

where λ is an objective network parameter. The main network parameter is updated based on the loss function.

In the training environment, the reward r^t obtained by the evader at time t contains manoeuvrability action

information. Each robot obtains the velocity v^t of the next step using the velocity v^{t+1} of the current step and the acceleration a^t. Similarly, each robot obtains the position g^t of the next step using the position g^{t+1} of the current step and the velocity v^t. Thus we have the following equation

$$\begin{aligned} v^{t+1} &= v^t + a^t \Delta t \\ g^{t+1} &= g^t + v^t \Delta t \end{aligned} \tag{12}$$

when Δt is small, the robot's velocity can be approximated as remaining constant throughout Δt. The robot's velocity can be considered to remain unchanged.

After each Δt in the training environment, the state information of the evader and all pursuers is updated as shown in Fig. 2. The POMDP and communication of the pursuers make the position and velocity information of the evader easily observable by the pursuers during the escape process. Specifically, the state information of the evader being observed is denoted as $S_e = \left[x_e, y_e, v_e^x, v_e^y\right]$ or $S_e = \left[x_e, y_e, z_e, v_e^x, v_e^y, v_e^z\right]$ and the state information of the pursuers observed by the evader is expressed as $S_{pi} = \left[x_{pi}, y_{pi}, v_{pi}^x, v_{pi}^y\right]$ or $S_{pi} = \left[x_{pi}, y_{pi}, z_{pi}, v_{pi}^x, v_{pi}^y, v_{pi}^z\right]$. Therefore, when the evader faces N pursuers, the state space is designed as $s = \left[S_e, S_{p1}, S_{p2}, \ldots, S_{pN}\right]$, where the state information of each ground robot and air robot is 4-dimension and 6-dimension, respectively. The total number of dimensions of the state space observed by the ground and aerial robots when one of the N pursuers exists within the detection range is $4(N+1)$ and $6(N+1)$, respectively. In addition, the manoeuvrability action space of the evader is defined as

$$\begin{aligned} E_{2D} &= \{E_0, E_1, E_2, E_3, E_4\} \\ E_{3D} &= \{E_0, E_1, E_2, E_3, E_4, E_5, E_6\} \end{aligned} \tag{13}$$

where $E_0, E_1, E_2, E_3, E_4, E_5, E_6$ represents the 7 types of actions in the evader's manoeuvrability action library: no-drive, forward-drive, backward-drive, left-drive, right-drive, up-drive, and down-drive, respectively. In pursuit-evasion games, the manoeuvrability action is represented by sparse rewards as $r_m^t = \{-c_1, c_2, 0\}$, where c_1, c_2 are two positive values. More details of the reward allocation will be described in Composite Reward Mechanism.

C. Composite Reward Mechanism

In this paper, we convert the problem of changing the relative position of the evader and pursuers into a problem of changing the potential power under the potential field generated by the pursuers. The repulsive field is generated by the pursuers, and the change in potential power of the evader under the potential field of the pursuers is taken as the reward value for each step. Based on the repulsive potential function in our previous work [34], the repulsive potential function

generated by the pursuer in this paper is defined as

$$U_g(\rho) = \begin{cases} \delta\left(\dfrac{1}{\rho\left(g_e, g_{pi}\right)} - \dfrac{1}{\rho_0}\right), & \rho\left(g_e, g_{pi}\right) \le \rho_0 \\ 0, & \rho\left(g_e, g_{pi}\right) > \rho_0 \end{cases} \tag{14}$$

where δ is a positive potential field coefficient and ρ_0 denotes the radius of influence of the pursuer i. The Euclidean distance between the pursuer i and evader is denoted as $\rho\left(g_e, g_{pi}\right)$.

After a training time step in the learning environment, if the relative distance between the evader and pursuer i decreases, then the potential power of the evader under the repulsive field generated by the pursuer i will increase. The smaller the relative distance between the evader and pursuer i, the faster the potential power of the evader changes. Therefore, after a training time step, if the relative distance between the evader and pursuer decreases, a negative reward should be given. When the relative distance is smaller, it means that the evader's state is more critical. Then the case that causes the relative distance to decrease should be given a greater penalty. Therefore, the reward function for relative position change is designed as

$$r_g^t = -\sum_{i=1}^{N} \Delta U_g(\rho) \tag{15}$$

where $\Delta U_g(\rho)$ denotes the change in potential power of the evader under the repulsive field generated by the pursuer i before and after performing a training time step.

In addition, the relative velocity change between the pursuer i and evader at each time step should be considered in the reward function design. When the relative velocity angle θ_{re} of the pursuer i in the detection range is smaller and the relative velocity v_{re} is larger, its threat to the evader is higher. To describe this threat, we propose the concept of velocity potential field as follows

$$U_v(g_e, g_{pi}) = \begin{cases} \eta\,|v_{re}|\cos\theta_{re}, & \rho\left(g_e, g_{pi}\right) \le \rho_0 \\ 0, & \rho\left(g_e, g_{pi}\right) > \rho_0 \end{cases} \tag{16}$$

where η denotes a coefficient constant and $v_{re} = v_e - v_{pi}$ represents the relative velocity of the pursuer i and evader. The relative velocity angle is expressed as

$$\theta_{re} = \arccos \frac{\left(g_e - g_{pi}\right)\left(v_{pi} - v_e\right)}{\left|g_e - g_{pi}\right|\left|v_{pi} - v_e\right|} \tag{17}$$

The larger the potential field of the evader, the faster the pursuer approaches the evader. Similarly, a penalty is given

to the evader for increased potential power in the relative velocity potential field. This reward function is set as

$$r_v^t = -\sum_{i=1}^{N} \Delta U_v(g_e, g_{pi}) \tag{18}$$

where $\Delta U_v(g_e, g_{pi})$ denotes the change in potential power of the evader under the relative velocity potential field generated by the pursuer i before and after performing a training time step.

Moreover, relying only on relative position reward and relative velocity reward does not give a reasonable reward value for the change in the evader's security state. Usually, when the pursuers are applied to learned pursuit strategy, the pursuers' formation influences the evader's escape strategy. The geometric centre of the pursuit formation is $g_{ce} = \frac{1}{N}\sum_{i=1}^{N} g_{pi}$ and the pursuit centre change reward is defined as

$$r_{ce}^t = \Delta\rho(g_e, g_{ce}) \tag{19}$$

where $\Delta\rho(g_e, g_{ce})$ denotes the change in the evader's distance from the center point of the pursuit formation before and after performing a training time step. Manoeuvrability actions away from the centre of the pursuit formation are positively rewarded.

Besides, the evader is considered to be surrounded when the positional relationship between the evader and pursuers simultaneously satisfies

$$\begin{aligned} x_e &\in \left[\min\left\{x_{p1}, \ldots, x_{pN}\right\}, \max\left\{x_{p1}, \ldots, x_{pN}\right\}\right] \\ y_e &\in \left[\min\left\{y_{p1}, \ldots, y_{pN}\right\}, \max\left\{y_{p1}, \ldots, y_{pN}\right\}\right] \\ z_e &\in \left[\min\left\{z_{p1}, \ldots, z_{pN}\right\}, \max\left\{z_{p1}, \ldots, z_{pN}\right\}\right] \end{aligned} \tag{20}$$

instead, when the evader is not under the surrounding, it is given a reward $r_{su}^t = \{w, -w\}$, where w is a hyperparameter. Therefore, the reward function of escape process for the evader is designed as

$$r^t = \omega_1 r_g^t + \omega_2 r_v^t + \omega_3 r_m^t + \omega_4 r_{ce}^t + \omega_5 r_{su}^t + d \tag{21}$$

where ω is the weight coefficient and d denotes the bias.

Remark 1: Manoeuvrability action and composite reward mechanism are mainly designed for the evader, while the design of rewards for the pursuers including learned pursuit strategy and random pursuit strategy will be presented in Sec. V.

D. Gaussian Process for Stochasticity and Uncertainty

A solution that utilizes the characteristics of GP to deal with multi-robot and uncertainty problems is presented in this section. Based on the preliminary GP functions provided in Sec. III, we introduce an approximation for the value function and estimation with Monte Carlo.

1. Approximation via Gaussian Process

In general, the discounted reward accumulated from time t given a policy π_p is a random process, and can be written as

$$R_p^t = \gamma R_p^{t+1} + r_p^{t+1} \tag{22}$$

Due to the stochasticity in state transitions, for the state-action pair (s^t, a^t), the accumulated discounted reward can be decomposed into two parts: its mean $Q(s^t, a_{pi}^t)$, and random zero-mean residual $\Delta Q(s^t, a_{pi}^t)$. Let us assume

$$R_p^t = Q(s^t, a_{pi}^t) + \Delta Q(s^t, a_{pi}^t) \tag{23}$$

which employs a Bayesian methodology that allows us to consider the value $Q(\cdot)$ as a stochastic entity. Due to our subjective uncertainty about the transition distribution, we define $Q(s^t, a_{pi}^t)$ and $\Delta Q(s^t, a_{pi}^t)$ as the extrinsic and intrinsic uncertainty in the stochastic process, respectively. For a more detailed discussion of intrinsic and extrinsic uncertainty refer to Ref. [35]. Substituting Eq. (22) into (23), we have

$$r_p^{t+1} = Q(s^t, a_{pi}^t) - \gamma Q(s^{t+1}, a_{pi}^{t+1}) + \mathcal{M}\left((s^t, a_{pi}^t), (s^{t+1}, a_{pi}^{t+1})\right) \tag{24}$$

with

$$\mathcal{M}\left((s^t, a_{pi}^t), (s^{t+1}, a_{pi}^{t+1})\right) = \Delta Q(s^t, a_{pi}^t) - \gamma \Delta Q(s^{t+1}, a_{pi}^{t+1}) \tag{25}$$

The finite-dimensional random process $\mathbf{r}_p^t$, $\mathbf{Q}^t$, $\mathcal{M}^t$ and the $t \times (t+1)$ matrix $\mathcal{K}^t$ can be expressed as

$$
\begin{aligned}
\mathbf{r}_p^t &= \left[r_p^0, \ldots, r_p^t\right]^T \\
\mathbf{Q}^t &= \left[Q(s^0, a^0), \ldots, Q(s^t, a^t)\right]^T \\
\mathcal{M}^t &= \left[\mathcal{M}\left((s^0, a^0), (s^1, a^1)\right), \ldots, \mathcal{M}\left((s^{t-1}, a^{t-1}), (s^t, a^t)\right)\right]^T \\
\mathcal{K}^t &= \begin{bmatrix} 1 & -\gamma & 0 & \ldots & 0 \\ 0 & 1 & -\gamma & \ldots & 0 \\ \vdots & & & \ldots & \vdots \\ 0 & 0 & \ldots & 1 & -\gamma \end{bmatrix}
\end{aligned} \tag{26}
$$

therefore, Eq. (24) can be written concisely as

$$\mathbf{r}_p^t = \mathcal{K}^{t+1}\mathbf{Q}^t + \mathcal{M}^t \tag{27}$$

Moreover, we impose a gaussian prior over every element in $\mathbf{Q}^t$, i.e., a prior distribution $\mathbf{Q}^t \sim \mathcal{GP}(\mathbf{0}, k_{ij})$, where $\mathbf{0}$ is a vector of zero and $k_{ij} = k\left((s^i, a^i), (s^j, a^j)\right)$. Similarly, its corresponding random process for $\Delta Q(s^t, a^t)$ is defined as

$$\Delta\mathbf{Q}^t = \left[\Delta Q(s^0, a^0), \ldots, \Delta Q(s^t, a^t)\right]^T \sim \mathcal{GP}(\mathbf{0}, \sigma^2\mathbf{I}) \tag{28}$$

where σ is the covariance parameter and $\mathbf{I}$ is the unit matrix. Given that Q and ΔQ are both gaussian, and the discounted accumulated reward R_p^t should follow GP. Using the standard results for joint gaussian distributed random variables [36], we obtain the joint distribution of $\mathbf{r}_p^t$ and $Q(s^t, a^t)$

$$
\begin{bmatrix} \mathbf{r}_p^t \\ Q(s^t, a^t) \end{bmatrix} \sim \mathcal{N}\left[\begin{pmatrix} \mathbf{0} \\ 0 \end{pmatrix}, \begin{pmatrix} \mathcal{K}^t k (\mathcal{K}^t)^T + \sigma^2 \mathcal{K}^t (\mathcal{K}^t)^T & \mathcal{K}^t \mathcal{L}^t(s^t, a^t) \\ \mathcal{L}^t(s^t, a^t)^T (\mathcal{K}^t)^T & k\left((s^t, a^t), (s^t, a^t)\right) \end{pmatrix}\right] \tag{29}
$$

with

$$\mathcal{L}^t(s^t, a^t) = \left[k\left((s^0, a^0), (s^t, a^t)\right), \ldots, k\left((s^t, a^t), (s^t, a^t)\right)\right] \tag{30}$$

In order to derive the posterior distribution of $Q(s^t, a^t)$ conditioned on the observed sequence of reward $\mathbf{r}_p^t$, based on Lemma 1 we have

$$Q(s^t, a^t)|\mathbf{r}_p^t \sim \mathcal{N}\left(\bar{Q}(s^t, a^t), cov\left((s^t, a^t), (s^t, a^t)\right)\right) \tag{31}$$

with the mean and covariance as

$$\begin{aligned}\bar{Q}(s^t, a^t) &= (\mathcal{L}^t)^T \left[(\mathcal{K}^t)^T \left(\mathcal{K}^t k (\mathcal{K}^t)^T + \sigma^2 \mathcal{K}^t (\mathcal{K}^t)^T \right)^{-1} \mathbf{r}_p^t \right] \\ cov\left((s^t, a^t), (s^t, a^t)\right) &= k_{tt} - (\mathcal{L}^t)^T \left[(\mathcal{K}^t)^T \left(\mathcal{K}^t k (\mathcal{K}^t)^T + \sigma^2 \mathcal{K}^t (\mathcal{K}^t)^T \right)^{-1} \mathcal{K}^t \right] \mathcal{L}^t \end{aligned} \tag{32}$$

where $\bar{Q}(s^t, a^t)$ denotes the desired trajectory at time t if the action a^t is chosen.

Remark 2: Given the current state information s^t, $cov\left((s^t, a^t), (s^t, a^t)\right)$ reflects the pursuers' confidence in estimating the Q-value. Notably, Eqs. (31) and (32) present the state information in an implicit way. Obviously, s^t contains state information that is potentially relevant among the pursuers. Moreover, Eqs. (31) and (32) are the key to update the estimated Q-value based on state, action and reward. Therefore, GP also establishes an important link between state information and decision quality.

2. Estimation with Monte Carlo

The validity of our model can be confirmed by performing a whitening transformation on Eq. (27). By Eq. (23), $\mathbb{E}_{\pi_p}[\Delta Q] = 0$, so we have $\mathbb{E}_{\pi_p}\left[\mathcal{M}\left((s^t, a^t), (s^{t+1}, a^{t+1})\right)\right] = 0$. Since $\mathcal{M}^t = \mathcal{K}^t \Delta \mathbf{Q}^t$, we have $\mathcal{M}^t \sim \mathcal{N}(\mathbf{0}, \Sigma^t)$ with

$$\begin{aligned}\Sigma^t &= \sigma^2 \mathcal{K}^t (\mathcal{K}^t)^T \\ &= \sigma^2 \begin{bmatrix} 1+\gamma^2 & -\gamma & 0 & \dots & 0 \\ -\gamma & 1+\gamma^2 & -\gamma & \dots & 0 \\ \vdots & & & \dots & \vdots \\ 0 & 0 & \dots & -\gamma & 1+\gamma^2 \end{bmatrix}\end{aligned} \tag{33}$$

Since the noise covariance matrix Σ^t is positive definite, there exists a square matrix $\mathcal{H}^t$ satisfying $(\mathcal{H}^t)^T \mathcal{H}^t = (\Sigma^t)^{-1}$. From Eq. (27), we have $\mathcal{H}^t \mathbf{r}_p^t = \mathcal{H}^t \mathcal{K}^{t+1} \mathbf{Q}^t + \mathcal{H}^t \mathcal{M}^t$. The transformed noise term $\mathcal{H}^t \mathcal{M}^t$ has a covariance matrix with $\mathcal{H}^t \Sigma^t (\mathcal{H}^t)^T = \mathcal{H}^t \left[(\mathcal{H}^t)^T \mathcal{H}^t\right]^{-1} (\mathcal{H}^t)^T = \mathbf{I}$. Therefore, the transformation $\mathcal{H}^t$ whitens the noise, where it is written by

$$\mathcal{H}^t = (\mathcal{K}^{t+1})^{-1} = \begin{bmatrix} 1 & \gamma & \gamma^2 & \dots & \gamma^t \\ 0 & 1 & \gamma & \dots & \gamma^{t-1} \\ \vdots & & & \dots & \vdots \\ 0 & 0 & \dots & 0 & 1 \end{bmatrix} \tag{34}$$

The transformed model is $\mathcal{H}^t \mathbf{r}_p^t = \mathbf{Q}^t + \mathcal{H}^t \mathcal{M}^t$ with white gaussian noise $\mathcal{H}^t \mathcal{M}^t \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I})$. In addition, we use the Monte Carlo sample of discounted reward as the target for GP regression to learn the value function, which is the

generative model. This has the advantage of allowing us to update the parameters of the posterior mean and covariance online.

V. Simulation

A. Simulation Setting

Multi-robot pursuit-evasion game tasks in two different space including aerial and ground robots are designed to validate the effectiveness of MERL and MERL-GP. Random pursuit strategy and learned pursuit strategy imposed on the pursuers are used to evaluate the evader's escape strategy in the face of general and complex situations.

For all the tasks, the ideal situation is that each robot obtains observation and action information from all other robots to complete the plan better. However, due to communication distance limitations, robots cannot obtain information from all other robots in a realistic environment. Moreover, the communication limitations become more drastic as one moves from a small number of robots to a swarm of robots. Therefore, the method for getting more information from other robots is through communication enhanced network [29]. The map is set in a limited four sides area or cubic area of $[-1500, 1500] \times [-1500, 1500]$ or $[-1500, 1500] \times [-1500, 1500] \times [-1500, 1500]$ and the initial positions of pursuers and evader are randomly generated. Besides, the action space is discrete, and each robot can control unit acceleration or deceleration in the x and y axis. These scenarios are implemented based on the dynamics in Sec. II, where the robots are able to move around with a double-integrator dynamics model.

In the simulations, MERL is compared with MADDPG [18], Transfer [37], Min-Max Q-learning [20], and MERL-GP. The states of all robots are required by the first algorithm MADDPG during training to construct its critic network. The second algorithm Transfer ignores the temporal relationship among the robots. The third algorithm Min-Max Q-learning transforms the high-dimensional state space into a low-dimensional manifold that is applied to the pursuit-evasion games. The fourth algorithm MERL-GP is another version of MERL with GP that is used to validate its effectiveness in the face of high-dimensional data from more robots. The parameters of the training process, the network, and MERL are given in Table 1.

B. Pursuit-Evasion Games for Ground Robots

1. Task Setting

In this task, a number of pursuit strategies are pre-defined to serve as the baseline method for initial training, evaluation, and comparison. The basic strategies include a random pursuit strategy and a learned pursuit strategy, as explained below

- Random pursuit strategy: Each pursuer is independently controlled by a high-gain proportional pursuit controller targeting the evader's location. The cooperation among pursuers is random and fuzzy.

Table 1 Parameters of environment and training

Parameters	Value	Parameters	Value
Robot mass (pursuer/evader)	3.8/1.9 kg	Max control force	1 N
Gravitational acceleration	9.8 m/s^2	Time step	0.5 s
Side length	3000 m	Learning rate	0.0001
Robot radius	0.15 m	Max gradient normalization	2
Detection range	750 m	Discount factor	0.99
Communication range	1500 m	Max episode	10000
Escape range	50 m	Batch size	64
Evader air resistance coefficient	0.127 N/m/s	Exploration factor range	0.05-0.5
Pursuer air resistance coefficient	0.1 N/m/s	Exploration factor decay	0.004

- Learned pursuit strategy: In training, the evader is driven by the MERL strategy. The composite reward function for this task is defined as follows:

$$
\begin{aligned}
R_{pi}^{t} &= \sum_{i=1}^{N}\left(R_{dist}^{t}+R_{coll}^{t}+\alpha R_{cap}^{t}\right)+\beta R_{succ} \\
R_{dist}^{t} &= -\rho\left(g_{pi}, g_{e}\right) \\
R_{coll}^{t} &= \begin{cases} 0, & \rho\left(g_{pi}, g_{pj}\right) > r_{pi}+r_{pj} \\ -2, & \rho\left(g_{pi}, g_{pj}\right) \leq r_{pi}+r_{pj} \end{cases} \\
R_{cap}^{t} &= \begin{cases} 0, & \rho\left(g_{pi}, g_{e}\right) > c \\ 10, & \rho\left(g_{pi}, g_{e}\right) \leq c \end{cases} \\
R_{succ} &= \begin{cases} 20, & \mathrm{num}_{e}=0 \\ 0, & \mathrm{num}_{e}\neq 0 \end{cases}
\end{aligned}
\tag{35}
$$

  where R_{dist} is the distance reward, R_{coll} is the collision reward, r_p and r_e represent the radius of the pursuer and evader, respectively, R_{cap} is the reward obtained by pursuer if it catches the evader, and R_{succ} denotes the reward obtained by all pursuers if all evaders are captured. num_e represents the number of uncaptured evaders. α and β are the coefficients, that the larger α, the more the robot pays attention to individual rewards and the larger β, the more the robot pays attention to swarm rewards.

2. Simulation Results

The learning curves of all the methods in terms of mean rewards are shown in Fig. 3. All five methods are trained with the pursuers driven by the aforementioned random pursuit strategy. When the number of robots is three, there is little difference in performance among all the methods. The reason for this is that the relationships among the robots are

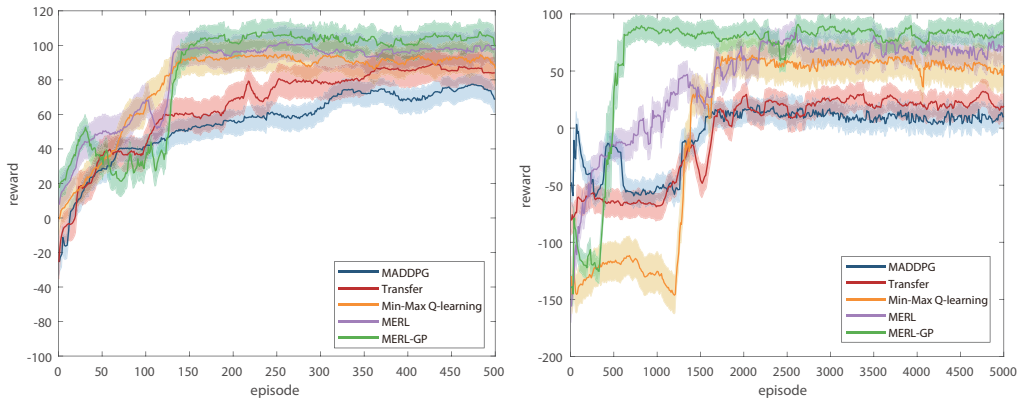
simpler, and all the methods can learn a satisfactory strategy regardless of whether they have a graph convolutional layer or contain the GP. Nevertheless, it is interesting to note that MERL-GP can be rewarded more than all baseline methods when training is complete. However, MERL-GP converges slower than MERL and Min-Max Q-learning without GP. It can be concluded that MERL-GP is more difficult to be trained because it has a more complex mechanism. It is obviously not suitable for when the amount of data is small.

When the number of robots increases, MERL-GP complex interaction and manoeuvrability processing is better. As shown in Fig. 3(b), the performance of MADDPG and Transfer is severely degraded, while MERL-GP performs significantly better than the other methods. Compared with other methods, MERL-GP has faster convergence and more stable performance. Specifically, MERL-GP converges to a steady state after 1000 update episodes, while the other methods converge to a steady state after 2000 update episodes. The results show that MERL-GP can handle complex interactions among a large number of robots. In addition, it can use GP to cope with high-dimensional state spaces. In addition, the negative impact of MERL-GP on training difficulty is much smaller than the positive impact of GP in high-dimensional data processing. In contrast, without the help of composite reward mechanism and GP, other easy-to-train methods perform poorly in complex environments where the number of robots increases. In addition to these basic analyses, there are a number of phenomena worth discussing:

- Except for MERL-GP, MADDPG converges the fastest but converges to a minimum value. This phenomenon may be due to the limitations of MADDPG. Specifically, MADDPG does not possess a recurrent network and cannot handle partially observable environments and history-dependent decisions. As a result, in partially observable environments with a large number of robots, complex interactions among robots cannot be handled and MADDPG is at a local optimum. In contrast, Transfer's results perform slightly better than MADDPG both with a small number of robots and with a much larger number.
- The results of MERL perform better Min-Max Q-learning and have a smaller variance of the curve, and the combined effect is better for a small number of robots, because the MERL composite reward mechanism helps the evader to better find the optimal escape strategy.
- The results of MERL-GP show better performance when the number of robots increases compared to the other methods, which means that GP is effective for high dimensional data.

The learned models are saved and evaluated after each trial. We test our escape strategy with both random pursuit strategy and the learned pursuit strategy. Each model is evaluated for 100 evaluation episodes. The evaluation episodes use the same environment as the training episodes. The difference is that detection noise on the action is excluded. Escape performance is evaluated by success rate and average step. We define that an event is successful in the escape task if the escape is completed before 400 steps, when the condition is

(i) when the maximum speed of the evader is less than the maximum speed of the pursuer, the distance between the evader and each pursuer is more than 150 m or the distance is more than 50 m at 400 steps.



(a) 3 robots (b) 12 robots

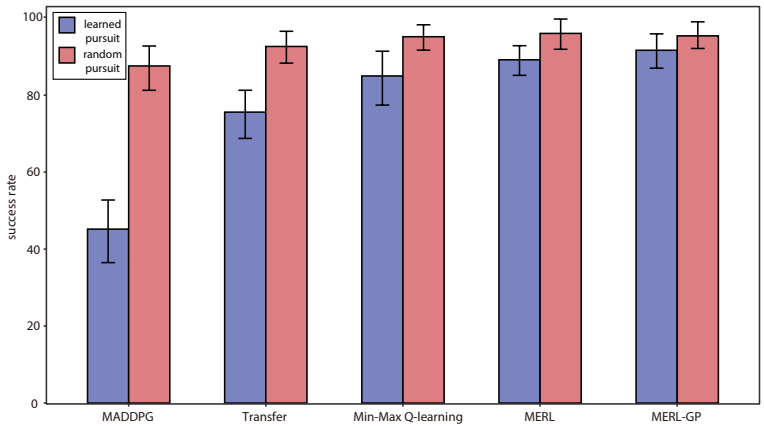
Fig. 3 Training rewards in pursuit-evasion games for ground robots. The solid lines and the shaded areas show the average values and standard deviations in each episode over 10 independent trails.

(ii) when the maximum speed of the evader is greater than or equal to the maximum speed of the pursuer, the real-time distance between the evader and each pursuer is more than the initial distance.

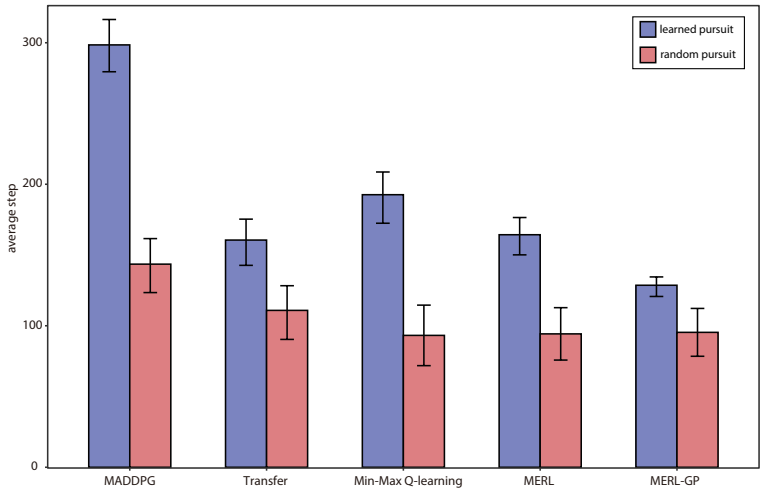
The success rate is the percentage of successful episodes in evaluation episodes. The average step is the average number of steps before the escape completes in an episode. The success rate in this study primarily reflects the effectiveness of the escape strategy. The lower average step indicates a more efficient escape strategy. The average success rate and average step of the learning model over the 100 evaluated episodes are shown in Figs. 4(a) and 4(b), respectively.

From the low success rate shown in Fig. 4(a), we find that MADDPG is unable to learn a better strategy for the escape task. The robot that learns based on local observation only is prone to fall into local optimum. Transfer, Min-Max Q-learning, and our methods succeed in obtaining good escape strategies such that the success rate of completing the task is high. Specifically, due to the ability of the first two methods to observe the full extent of the environment, a high success rate and a low average number of steps are obtained despite the fact that the pursuer is allowed to communicate with the information of other pursuers. However, their learning rate is still low compared to our methods. In addition, we also observe lower performance when confronted with a learned pursuit strategy compared to a random pursuit strategy. This is due to the fact that the learned pursuit strategy is trained to actively capture evaders. However, although our escape strategy is only trained to handle the random pursuit strategy, it still obtains better results when confronted with the learned pursuit strategy, which implies that our escape strategy has good generalization ability.

In order to better illustrate the pursuit-evasion game process for ground robots, the dynamic evolution of the two escape behaviours in conditions (i) and (ii) is shown in Fig. 5. As shown in Fig. 5, the cases where the evader is captured contain different numbers of ground robots in the initial training stage. When the training is completed, the cases where the evader successfully escapes are shown in Fig. 6. In Figs. 6(a) and 6(b), the learned pursuit strategy is



(a) Average success rate



(b) Average step

Fig. 4 Average success rate and step of the random and learned pursuit strategy under five methods.

applied and the evader satisfies escape condition (i). Although the maximum speed of the evader is less than that of the pursuers, the evader escapes from the pursuer through its manoeuvring actions. In Figs. 6(c) and 6(d), the random pursuit strategy is applied and the evader satisfies escape condition (ii). In this case, the evader faces the threat of more pursuers and completes the escape task by finding the optimal escape route.

C. Pursuit-Evasion Games for Aerial Robots

1. Task Setting

In this task, similar to that of the ground robots, learned pursuit strategy or random pursuit strategy remains embedded in the pursuers. The difference is that the 3D space significantly increases the difficulty of training the aerial robot's strategy due to the growth in data dimensions. The conditions for success in the escape task are the same as those set when testing the ground robots.

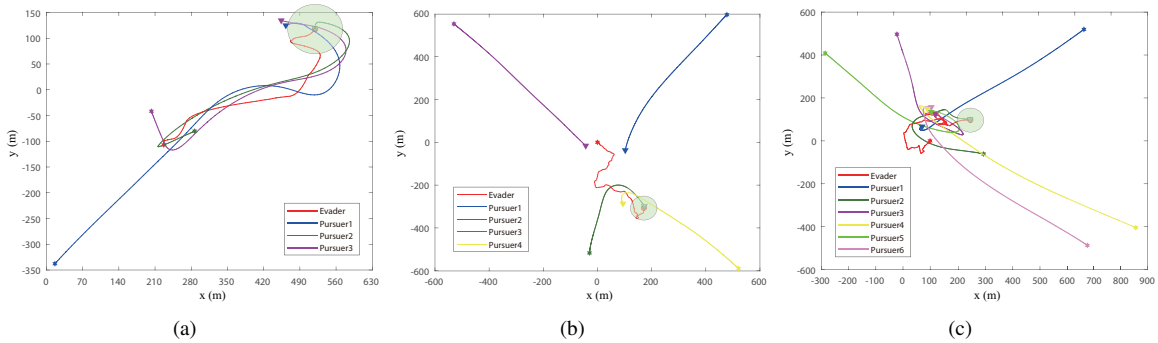


Fig. 5 Illustration of the initial training failure cases in pursuit-evasion games for ground robots: a) learned pursuit strategy, and b,c) random pursuit strategy.

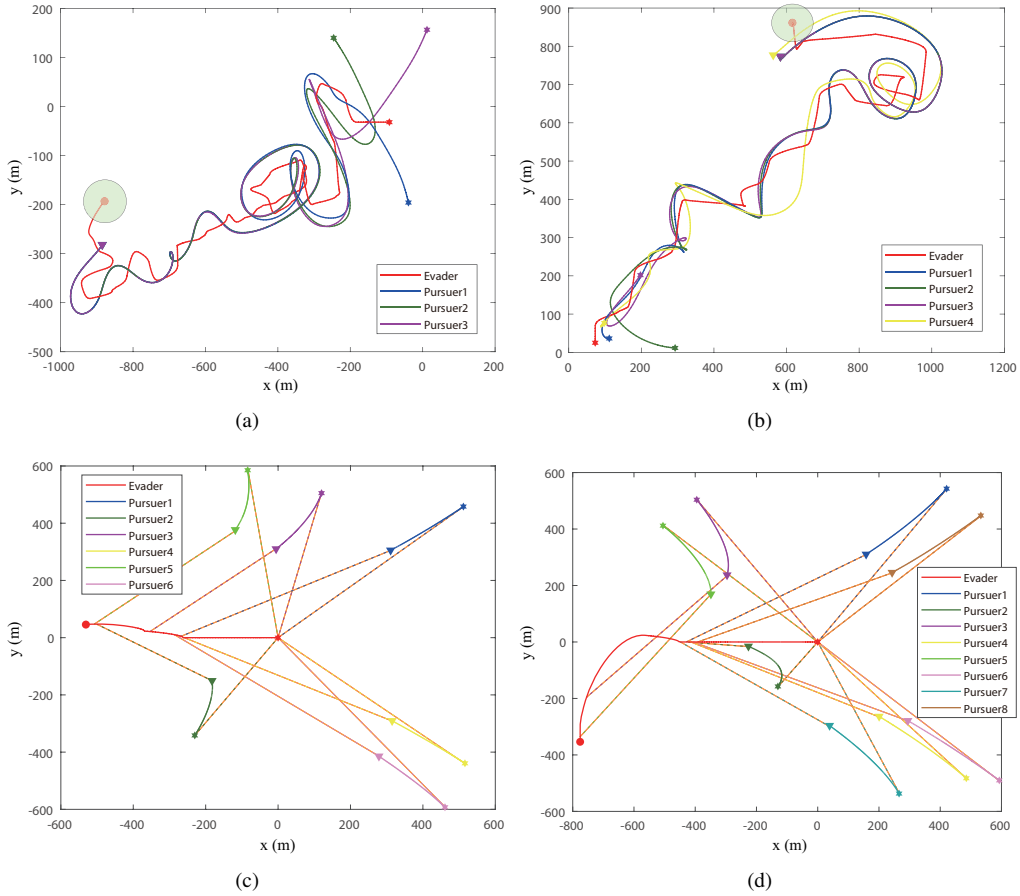


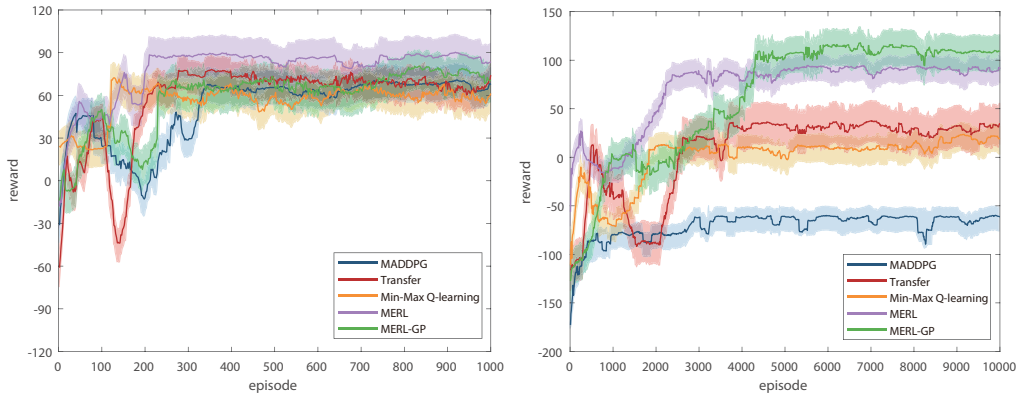
Fig. 6 Illustration of the training end cases in pursuit-evasion games for ground robots: a,b) learned pursuit strategy and satisfying condition (i), and c,d) random pursuit strategy and satisfying condition (ii).

2. Simulation Results

The learning curves of all methods for the average reward are shown in Fig. 7(a). All five methods are trained with the pursuers driven by the random pursuit strategy. When the number of robots is three, all methods equally have similar performance. The reason for this is that the low dimensional computation due to the small number of robots in both 2D

and 3D space makes the relationship among robots relatively simple. Interestingly, MERL can be rewarded more than all baseline methods when training is complete. The reason for this is the simpler structure of MERL-GP compared to MERL-GP when faced with a small number of robots.

When the number of robots increases, MERL-GP complex interactions and GP are better at handling high dimensional data. As shown in Fig. 7(b), the performance of the other methods degrades severely, while the performance of MERL-GP is significantly due to the other methods. MERL-GP reaches stability after 5000 update episodes. The results show that MERL-GP can handle complex interactions among a large number of robots. Besides, it can use manoeuvrability enhanced to complete the evader escape task facing multiple pursuers with the random pursuit strategy. In addition, the negative impact of the complex structure of MERL-GP on training difficulty is much smaller than the positive impact of GP in handling high-dimensional data. In contrast, without the help of manoeuvrability enhanced strategy and GP, the other methods that are easy to train have difficulty in completing the evader escape task or falling into local optima in complex environments with an increasing number of robots.



(a) 3 robots (b) 20 robots

Fig. 7 Training rewards in pursuit-evasion games for ground robots. The solid lines and the shaded areas show the average values and standard deviations in each episode over 10 independent trails.

In addition to the data during training, the evaluation results of 100 independent simulations in Table 2 show similar results as in Fig. 7. The values of three metrics, including success rate, rewards, and steps, for the five comparison methods in the four cases in Table 2. The mean values are the statistical results of running 100 times of the same method on the same test scenario.

Moreover, t-test [38] is used to statistically evaluate the validity of the method. Based on the mean value, standard deviation, and the sample data for 100 tests, we calculated the t-test values for MERL-GP and the other methods. '+', '=' and '-' indicate that the index values obtained by the method of this paper are better than the results of the other methods in the two-tailed t-test with a significance level of 5%, respectively.

As shown in Table 2, our method obtains 39 optimal measurements in all test scenarios. There is no significant

Table 2 Evaluation results of pursuit-evasion games for aerial robots

Method	Metrics	n=3			n=6		
		success rate	reward	step	success rate	reward	step
MADDPG	mean	96.2	91.2683	105.4622	94.6	24.0226	115.5886
	t-test	4.4397 (+)	14.1612 (+)	-11.4851 (+)	5.3371 (+)	20.9971 (+)	-18.3032 (+)
Transfer	mean	97	101.1851	110.8635	96.4	39.3404	104.5791
	t-test	3.9285 (+)	13.306 (+)	-14.3362 (+)	4.3168 (+)	8.2048 (+)	-8.5796 (+)
Min-Max Q-learning	mean	100	113.2219	93.1925	99.4	36.1168	101.4617
	t-test	ϕ (=)	8.0942 (+)	-2.4896 (=)	1.7355 (=)	10.3761 (+)	-8.9537 (+)
MERL	mean	100	118.7360	94.2656	99.6	48.9934	91.5230
	t-test	ϕ (=)	1.2723 (=)	-1.3531 (=)	1.4156 (=)	3.3614 (+)	-4.0268 (+)
MERL-GP	mean	100	117.9058	95.3385	100	64.8348	86.2669
	t-test	-	-	-	-	-	-
		n=12			n=20		
		success rate	reward	step	success rate	reward	step
MADDPG	mean	87.6	-0.2695	129.9101	45.2	-127.9634	298.4788
	t-test	11.3032 (+)	24.6172 (+)	-26.5581 (+)	24.5968 (+)	75.8487 (+)	-61.2081 (+)
Transfer	mean	93.8	12.9636	114.1817	75.6	-4.1059	138.9427
	t-test	7.3541 (+)	19.4651 (+)	-20.8732 (+)	12.6911 (+)	23.6260 (+)	-3.9305 (+)
Min-Max Q-learning	mean	99	19.5638	99.6901	78.4	-11.4854	192.6153
	t-test	1.6094 (+)	13.4344 (+)	-24.1560 (+)	11.5598 (+)	17.5810 (+)	-18.8610 (+)
MERL	mean	99.1	47.6252	72.8084	96	43.7897	164.3268
	t-test	1.3913 (=)	1.9328 (+)	2.3601 (=)	4.5598 (+)	7.3414 (+)	-12.8101 (+)
MERL-GP	mean	100	52.4208	75.0396	99.6	49.2294	128.6292
	t-test	-	-	-	-	-	-

difference among the performance of other methods and our methods in a small number of robot scenarios. With a larger number of robots, MERL-GP can obtain more rewards and higher efficiency than other methods, which means that MERL-GP is better at handling scenarios with a large number of robots. In addition, the performance gap between MERL and MERL-GP suggests that using MERL is more suitable for a small number of robots, whereas MERL-GP is more suitable for a larger number of robots.

In order to better describe the pursuit-evasion game process for aerial robots, the dynamic evolution of the two escape behaviours in conditions (i) and (ii) is shown in Fig. 8. As shown in Fig. 8, the cases where the evader is captured contain different numbers of aerial robots in the initial training stage. When the training is completed, the cases where the evader successfully escapes are shown in Fig. 9. In Figs. 9(a) and 9(b), the learned pursuit strategy is applied and the evader satisfies escape condition (i). Although the maximum speed of the evader is less than that of the pursuers, the evader escapes from the pursuer through its manoeuvring actions. In Figs. 9(c) and 9(d), the random pursuit strategy is applied and the evader satisfies escape condition (ii).

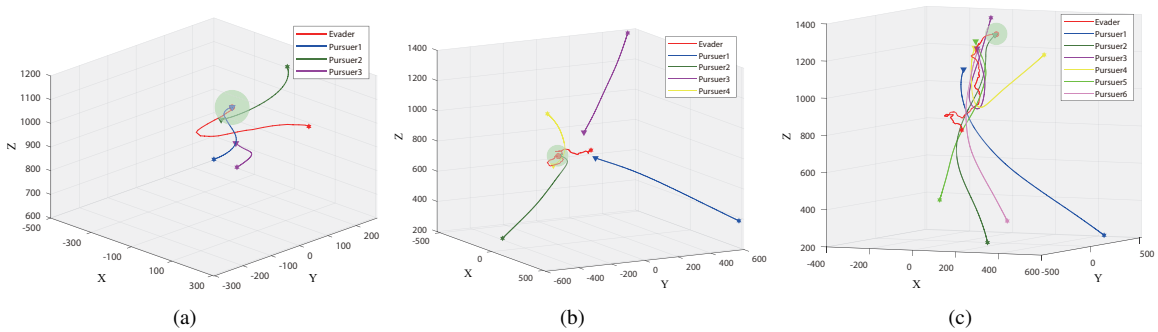


Fig. 8 Illustration of the initial training failure cases in pursuit-evasion games for aerial robots: a) learned pursuit strategy, and b,c) random pursuit strategy.

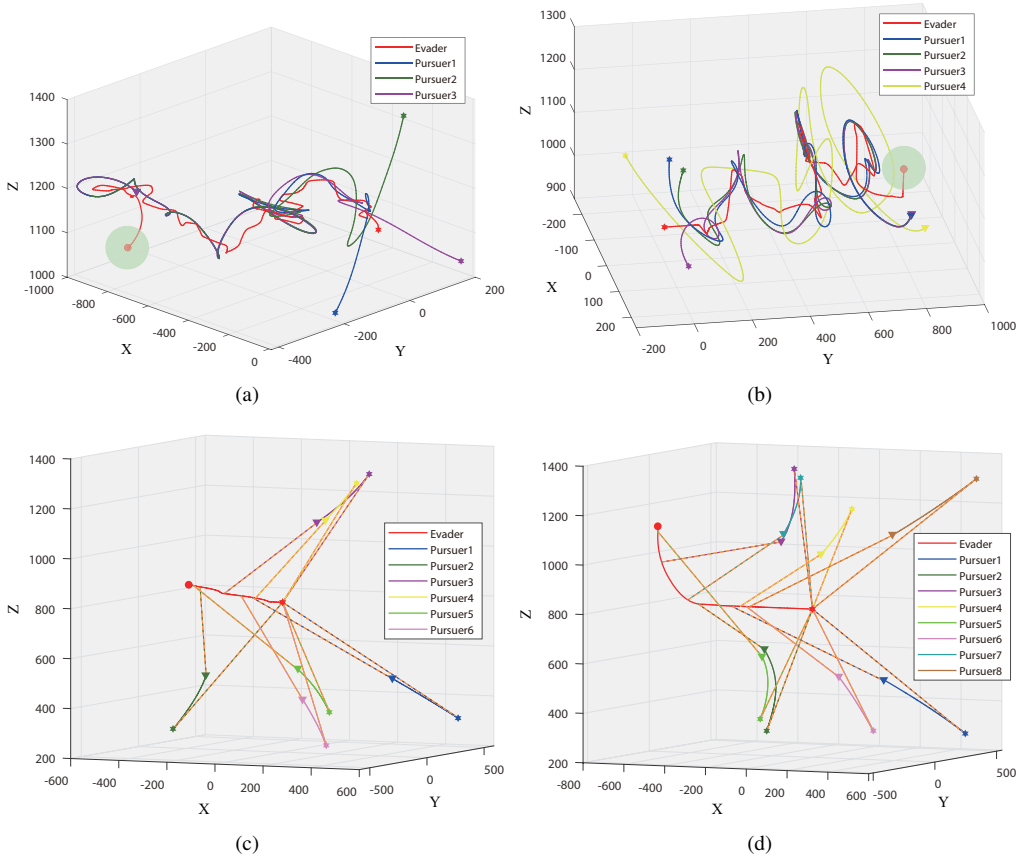
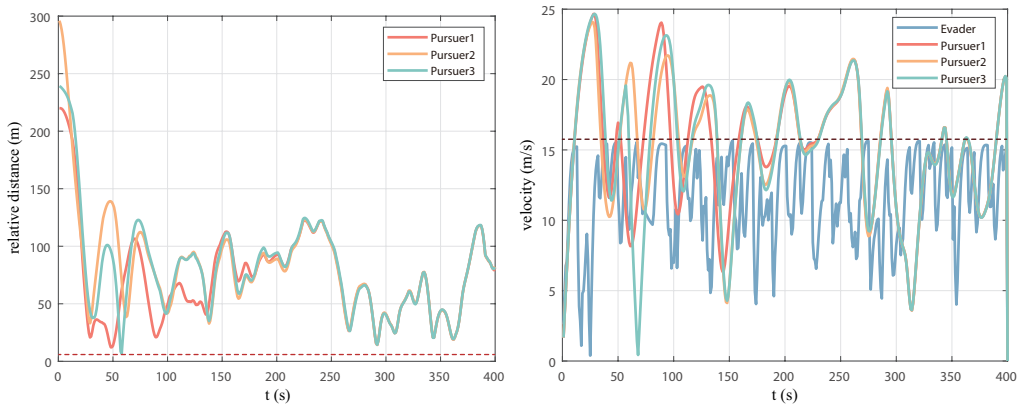


Fig. 9 Illustration of the training end cases in pursuit-evasion games for aerial robots: a,b) learned pursuit strategy and satisfying condition (i), and c,d) random pursuit strategy and satisfying condition (ii).

In addition, for example, the time-varying relative distances and time-varying velocities of the evader and the three pursuers in scenario a) of Fig. 9 are shown in Fig. 10. In Fig. 10(a), the red dashed line indicates the minimum distance between the evader and each pursuer. The maximum velocity bound of the evader is indicated by the dark dashed line in Fig. 10(b), which shows that the maximum velocity of the evader is always smaller than the maximum velocity of the

pursuers to satisfy the escape condition (i).



(a) Relative distance (b) Velocity

Fig. 10 Time-varying relative distances and time-varying velocities of the evader to the three pursuers in scenario a) of Fig. 9.

D. Ablation Models

In this section, we briefly analyze the ablation models, including MERL and MERL-GP. The results for the ground robot escape task in pursuit-evasion games are shown in the previous section (refer to Figs. 3 and 4). The results for the aerial robot escape task are shown in the previous section (refer to Fig. 7 and Table 2). Furthermore, we also perform additional ablation simulations of the pursuers applied by learned pursuit strategy in the aerial robot escape task to assess the effectiveness of our methods (refer to Fig. 11).

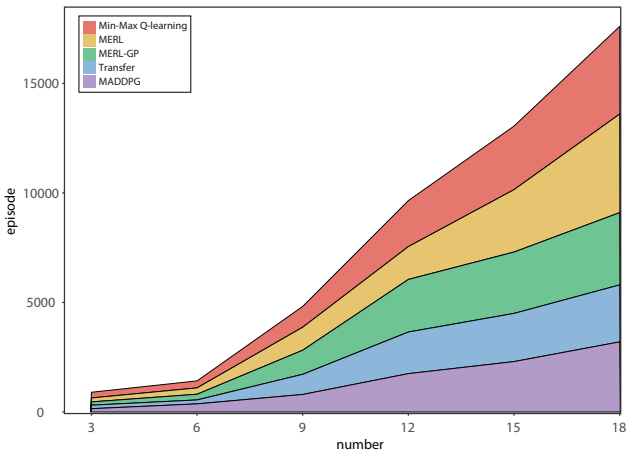


Fig. 11 Convergence episodes of escape task in pursuit-evasion games.

It can be observed that MERL performs when the scenario contains a small number of robots due to MERL-GP both for ground and aerial robots. However, as the number of robots increases, the performance of MERL-GP is superior compared to MERL. Moreover, we note that MERL-GP improves the performance of the aerial robot escape task better

than the ground robot escape task. This phenomenon suggests that the complex structure of the GP reduces processing efficiency in the face of low-dimensional data. However, the efficiency improvement in the face of high-dimensional data is much higher than the reduction of processing efficiency by the complex structure. Therefore, it can be concluded that GP helps to improve the performance of our method.

As shown in Fig. 11, the number of episodes required for convergence of MERL, MERL-GP and the other methods under three robots is basically the same. As the number of robots increases, the increase in the number of episodes is greater for MERL than for MERL-GP. MERL-GP converges faster than MERL. While MADDPG and Transfer fall into local optimums although the number of rounds to converge is smaller. The results show that GP can be designed to accelerate convergence effectively.

Table 3 Model evaluation indicators

Method	Number	MAE	RMSE
MADDPG	n=3	4.28	7.32
	n=6	5.63	7.94
	n=12	6.37	8.87
	n=20	7.98	10.02
Transfer	n=3	1.04	1.36
	n=6	1.27	1.69
	n=12	1.56	2.12
	n=20	2.35	3.83
Min-Max Q-learning	n=3	0.79	1.03
	n=6	0.87	1.25
	n=12	0.91	1.34
	n=20	1.55	2.84
MERL	n=3	**0.54**	**0.86**
	n=6	**0.57**	**0.92**
	n=12	**0.63**	**0.98**
	n=20	1.02	1.43
MERL-GP	n=3	0.62	0.96
	n=6	0.64	0.98
	n=12	0.72	1.01
	n=20	**0.92**	**1.36**

We compare the performance of different methods using the Root Mean Squared Error (RMSE) and the Mean Absolute Error (MAE). The RMSE and MAE values of the different methods with different number of robots are compared and the better model is the one that produce the lowest RMSE and MAE values. The results show that MERL and MERL-GP have lower values compared to the other three methods in Table 3. In MERL, MAE and RMSE are smaller when the number of robots is less. Whereas, the MERL-GP has smaller values with more number of robots.

Most importantly, all the results show that MERL-GP can further improve the performance of more robots in escape task. These advantages manifest as a considerable increase in rewards (refer to Figs. 3 and 7 and Table 2) or faster

convergence (refer to Figs. 3(b) and 11). This means that GP is necessary for better and more stable results when dealing with high dimensional data.

E. Robustness of MERL and MERL-GP

In order to validate the robustness and generalization of MERL and MERL-GP, all methods are applied to different scenarios with a number of aerial robots ranging from 3 to 20. Scenarios with each number of robots are tested for 100 escape tasks to get the average escape rate. Specifically, the robot initial positions of each task are randomly generated and the error bars reflect the accuracy of the test.

As shown in Fig. 12, the initial escape rates of each method is high, but the escape rates of MADDPG, Transfer, and Min-Max Q-learning in the escape task decay significantly as the number of robots increases. MERL starts to decay significantly after the number of robots is increased to 15, and the escape rate of the tasks for MERL-GP decays more slowly compared to the other methods.

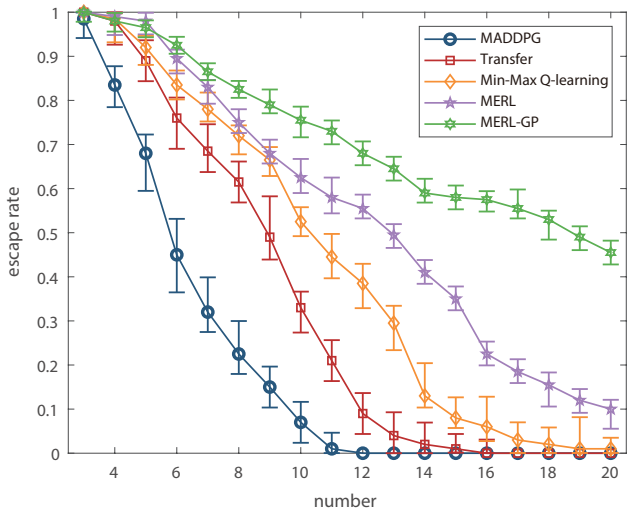


Fig. 12 Escape rate of the aerial robot escape task when the pursuers are applied to the learned pursuit strategy.

VI. Experiment

A. Experiment Setting

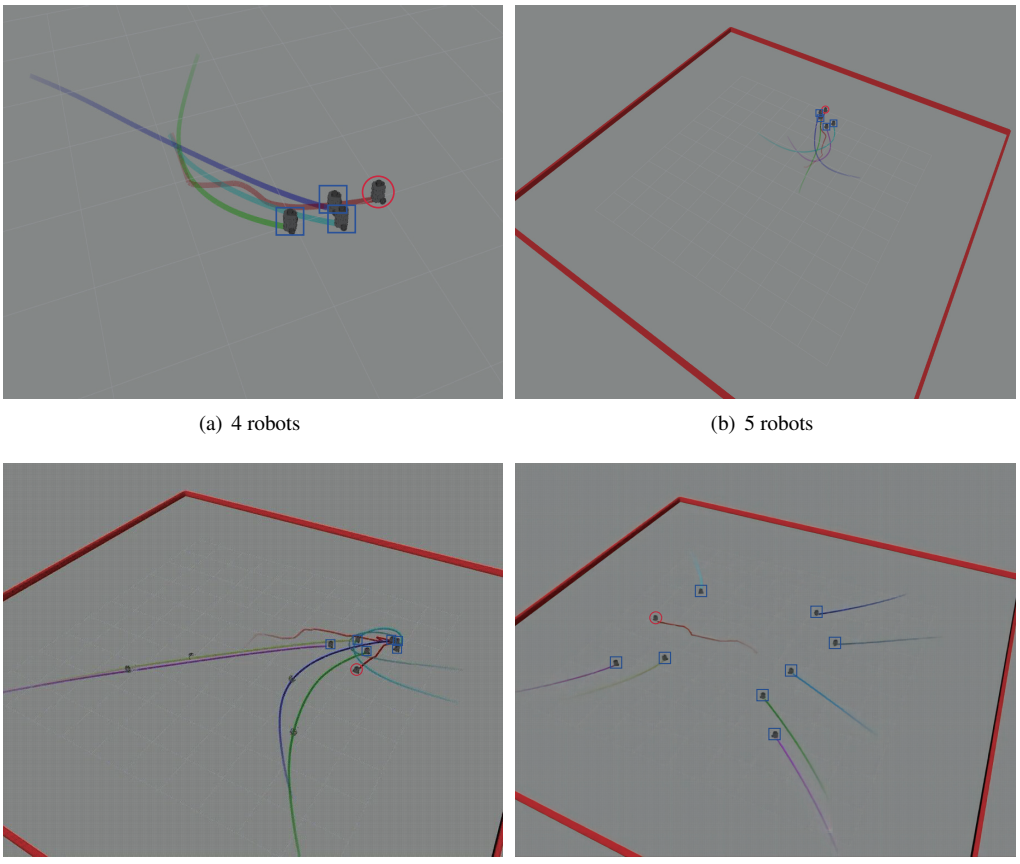
In addition, we also demonstrate the effectiveness of the MERL and MERL-GP algorithm applied to multiple ground robots in a semi-physical environment Gazebo and Rviz. All processing is done in realtime, and by using MERL or MERL-GP, the ground robot pursuer team pursues the evader. All experiments are 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 3000×3000 m. 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. The dynamics model of the real ground robot can be found in Ref. [39].

B. Experiment Results

As shown in Fig. 13, the motion trajectories of the ground robots fade with time varying through the overhead view, where the red circle is the evader and the blue square is the pursuer. MERL is applied to the number of robots 4 and 5 as shown in Figs. 13(a) and 13(b). MERL-GP is applied to the number of robots 7 and 9 as shown in Figs. 13(c) and 13(d). The applicability of our methods in the physical world is verified through several successful experiments. Although our simulations also include gravity, air friction, and the complete dynamics of the robot, some behaviours are difficult to quantify in the simulations including robot torque, lossy drive, and ground friction.



(a) 4 robots

(b) 5 robots

(c) 7 robots

(d) 9 robots

Fig. 13 Video stills illustrating cases when the number of ground robots is 4, 5, 7, and 9, respectively.

C. Robustness to Noise

In this section, we investigate the robustness of the system to noise in a semi-physical experiment. We observe two main sources of noise: the noise added to the evader, and the noise generated by the pursuer when measuring the

position of the evader. For each case, we added different levels of noise (refer to Ref. [40]) to test the effect of noise on the escape task.

The results at 4 robots in the random initial condition are shown in Fig. 14. The additional noise on the trajectories comes from unmodelled dynamics and communication delays. The results show that the ground robot remains stable relative to the desired trajectory despite being subjected to persistent noise, demonstrating the robustness of our method in semi-physical experiments.

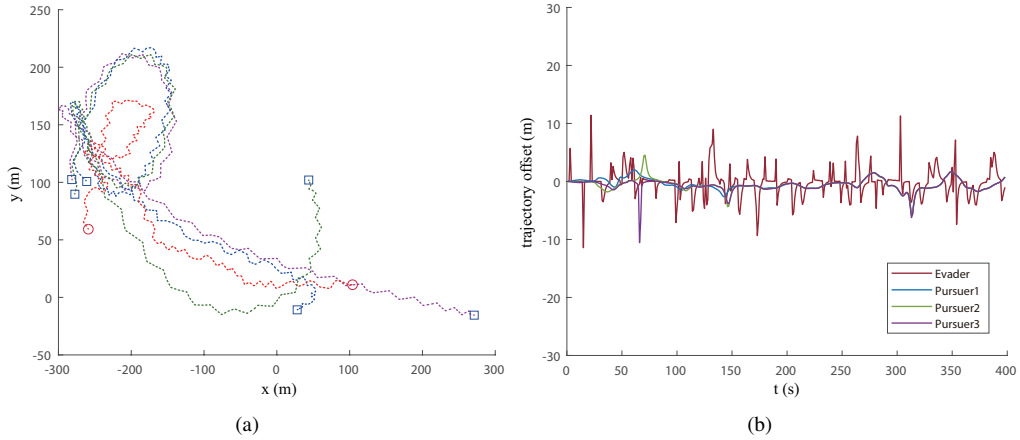


Fig. 14 An example of trajectory offsets with added noise: a) the motion trajectories of the evader and pursuers, and b) the offset distance relative to the desired trajectory.

VII. Conclusion

In this paper, MERL and MERL-GP are proposed to obtain high probability escape strategies for the evader in pursuit-evasion scenarios, overcoming the sparse reward problem and local optima, and addressing the difficulty of solving and training on high-dimensional data in a system containing stochasticity and uncertainty. Specifically, manoeuvrability action provides more escape strategies. Composite reward mechanism overcomes the sparse reward and local optima problems. Gaussian process approximation solves the Q-function and allows an accurate online update of the parameters of the posterior mean and covariance. These methods improve the training efficiency and speed up the convergence process. Our methods are evaluated by simulations and experiments in tasks for ground and aerial robots. The results show that our methods outperform several popular methods and have better adaptability.

Funding Sources

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 the Practice and Innovation Funds for Graduate Students of Northwestern Polytechnical University under the Grant No. PF2024040.

Acknowledgments

This work is 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 the Practice and Innovation Funds for Graduate Students of Northwestern Polytechnical University under the Grant No. PF2024040.

References

[1] Hedenström, A., and Rosén, M., "Predator versus prey: on aerial hunting and escape strategies in birds," *Behavioral Ecology*, Vol. 12, No. 2, 2001, pp. 150–156. https://doi.org/10.1093/beheco/12.2.150.

[2] Turetsky, V., and Shinar, J., "Missile guidance laws based on pursuit–evasion game formulations," *Automatica*, Vol. 39, No. 4, 2003, pp. 607–618. https://doi.org/10.1016/S0005-1098(02)00273-X.

[3] Eklund, J. M., Sprinkle, J., and Sastry, S. S., "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, 2011, pp. 604–620. https://doi.org/10.1109/TCST.2011.2136435.

[4] Oyler, D. W., Kabamba, P. T., and Girard, A. R., "Pursuit–evasion games in the presence of obstacles," *Automatica*, Vol. 65, 2016, pp. 1–11. https://doi.org/10.1016/j.automatica.2015.11.018.

[5] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al., "Human-level control through deep reinforcement learning," *nature*, Vol. 518, No. 7540, 2015, pp. 529–533. https://doi.org/10.1038/nature14236.

[6] Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al., "Mastering the game of go without human knowledge," *nature*, Vol. 550, No. 7676, 2017, pp. 354–359. https://doi.org/10.1038/nature24270.

[7] Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al., "Grandmaster level in StarCraft II using multi-agent reinforcement learning," *nature*, Vol. 575, No. 7782, 2019, pp. 350–354. https://doi.org/10.1038/s41586-019-1724-z.

[8] Zhou, Z., Zhang, W., Ding, J., Huang, H., Stipanović, D. M., and Tomlin, C. J., "Cooperative pursuit with Voronoi partitions," *Automatica*, Vol. 72, 2016, pp. 64–72. https://doi.org/10.1016/j.automatica.2016.05.007.

[9] Von Moll, A., Casbeer, D. W., Garcia, E., and Milutinović, D., "Pursuit-evasion of an evader by multiple pursuers," *2018 International Conference on Unmanned Aircraft Systems (ICUAS)*, IEEE, 2018, pp. 133–142. https://doi.org/10.1109/icuas.2018.8453470.

[10] Sun, W., Tsiotras, P., Lolla, T., Subramani, D. N., and Lermusiaux, P. F., "Multiple-pursuer/one-evader pursuit–evasion game in dynamic flowfields," *Journal of guidance, control, and dynamics*, Vol. 40, No. 7, 2017, pp. 1627–1637. https://doi.org/10.2514/1.G002125.

[11] Li, W., "A dynamics perspective of pursuit-evasion: Capturing and escaping when the pursuer runs faster than the agile evader," *IEEE Transactions on Automatic Control*, Vol. 62, No. 1, 2016, pp. 451–457. https://doi.org/10.1109/tac.2016.2575008.

[12] Ramana, M., and Kothari, M., "Pursuit strategy to capture high-speed evaders using multiple pursuers," *Journal of Guidance, Control, and Dynamics*, Vol. 40, No. 1, 2017, pp. 139–149. https://doi.org/10.2514/1.G000584.

[13] Garcia, E., Fuchs, Z. E., Milutinovic, D., Casbeer, D. W., and Pachter, M., "A geometric approach for the cooperative two-pursuer one-evader differential game," *IFAC-PapersOnLine*, Vol. 50, No. 1, 2017, pp. 15209–15214. https://doi.org/10.1016/j.ifacol.2017.08.2366.

[14] Von Moll, A., Pachter, M., Shishika, D., and Fuchs, Z., "Circular target defense differential games," *IEEE Transactions on Automatic Control*, Vol. 68, No. 7, 2022, pp. 4065–4078. https://doi.org/10.1109/tac.2022.3203357.

[15] Pachter, M., Moll, A. V., Garcia, E., Casbeer, D., and Milutinović, D., "Cooperative pursuit by multiple pursuers of a single evader," *Journal of Aerospace Information Systems*, Vol. 17, No. 8, 2020, pp. 371–389. https://doi.org/10.2514/1.I010739.

[16] Selvakumar, J., and Bakolas, E., "Feedback strategies for a reach-avoid game with a single evader and multiple pursuers," *IEEE transactions on cybernetics*, Vol. 51, No. 2, 2019, pp. 696–707. https://doi.org/10.1109/tcyb.2019.2914869.

[17] Makkapati, V. R., Sun, W., and Tsiotras, P., "Optimal evading strategies for two-pursuer/one-evader problems," *Journal of guidance, control, and dynamics*, Vol. 41, No. 4, 2018, pp. 851–862. https://doi.org/10.2514/1.G003070.

[18] Lowe, R., Wu, Y. I., Tamar, A., Harb, J., Pieter Abbeel, O., and Mordatch, I., "Multi-agent actor-critic for mixed cooperative-competitive environments," *Advances in neural information processing systems*, Vol. 30, 2017. https://doi.org/10.48550/arXiv.1706.02275.

[19] Wang, Y., Dong, L., and Sun, C., "Cooperative control for multi-player pursuit-evasion games with reinforcement learning," *Neurocomputing*, Vol. 412, 2020, pp. 101–114. https://doi.org/10.1016/j.neucom.2020.06.031.

[20] Selvakumar, J., and Bakolas, E., "Min–max Q-learning for multi-player pursuit-evasion games," *Neurocomputing*, Vol. 475, 2022, pp. 1–14. https://doi.org/10.1016/j.neucom.2021.12.025.

[21] Selvakumar, J., and Bakolas, E., "Evasion with terminal constraints from a group of pursuers using a matrix game formulation," *2017 American Control Conference (ACC)*, IEEE, 2017, pp. 1604–1609. https://doi.org/10.23919/ACC.2017.7963182.

[22] Liu, D., Wei, Q., Wang, D., Yang, X., and Li, H., *Adaptive dynamic programming with applications in optimal control*, Springer, 2017. https://doi.org/10.1007/978-3-319-50815-3.

[23] Rodriguez-Ramos, A., Sampedro, C., Bavle, H., Moreno, I. G., and Campoy, P., "A deep reinforcement learning technique for vision-based autonomous multirotor landing on a moving platform," *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2018, pp. 1010–1017. https://doi.org/10.1109/iros.2018.8594472.

[24] Eteke, C., Kebüde, D., and Akgün, B., "Reward learning from very few demonstrations," *IEEE Transactions on Robotics*, Vol. 37, No. 3, 2020, pp. 893–904. https://doi.org/10.1109/tro.2020.3038698.

[25] Berkenkamp, F., Moriconi, R., Schoellig, A. P., and Krause, A., "Safe learning of regions of attraction for uncertain, nonlinear systems with gaussian processes," *2016 IEEE 55th Conference on Decision and Control (CDC)*, IEEE, 2016, pp. 4661–4666. https://doi.org/10.1109/cdc.2016.7798979.

[26] Berkenkamp, F., and Schoellig, A. P., "Safe and robust learning control with Gaussian processes," *2015 European Control Conference (ECC)*, IEEE, 2015, pp. 2496–2501. https://doi.org/10.1109/ecc.2015.7330913.

[27] Kokolakis, N.-M. T., and Vamvoudakis, K. G., "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*, Vol. 35, No. 3, 2022, pp. 3130–3143. https://doi.org/10.1109/TNNLS.2022.3203977.

[28] Kaelbling, L. P., Littman, M. L., and Cassandra, A. R., "Planning and acting in partially observable stochastic domains," *Artificial intelligence*, Vol. 101, No. 1-2, 1998, pp. 99–134. https://doi.org/10.1016/S0004-3702(98)00023-X.

[29] Pu, Z., Wang, H., Liu, Z., Yi, J., and Wu, S., "Attention enhanced reinforcement learning for multi agent cooperation," *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 34, No. 11, 2022, pp. 8235–8249. https://doi.org/10.1109/TNNLS.2022.3146858.

[30] Watkins, C. J., and Dayan, P., "Q-learning," *Machine learning*, Vol. 8, 1992, pp. 279–292. https://doi.org/10.1007/bf00992698.

[31] Gavrilovska, L., Atanasovski, V., Macaluso, I., and DaSilva, L. A., "Learning and reasoning in cognitive radio networks," *IEEE Communications Surveys & Tutorials*, Vol. 15, No. 4, 2013, pp. 1761–1777. https://doi.org/10.1109/surv.2013.030713.00113.

[32] Engel, Y., Mannor, S., and Meir, R., "Reinforcement learning with Gaussian processes," *Proceedings of the 22nd international conference on Machine learning*, 2005, pp. 201–208. https://doi.org/10.1145/1102351.1102377.

[33] Rasmussen, C. E., and Williams, C. K. I., *Gaussian Processes for Machine Learning*, The MIT Press, 2005. https://doi.org/10.7551/mitpress/3206.001.0001.

[34] Zhang, Y., Zhu, Y., Li, H., and Wang, J., "A hybrid optimization algorithm for multi-agent dynamic planning with guaranteed convergence in probability," *Neurocomputing*, Vol. 592, 2024, p. 127764. https://doi.org/10.1016/j.neucom.2024.127764.

[35] Mannor, S., Simester, D., Sun, P., and Tsitsiklis, J. N., "Bias and variance in value function estimation," *Proceedings of the twenty-first international conference on Machine learning*, 2004, p. 72. https://doi.org/10.1145/1015330.1015402.

[36] Doucet, A., and Wang, X., "Monte Carlo methods for signal processing: a review in the statistical signal processing context," *IEEE Signal Processing Magazine*, Vol. 22, No. 6, 2005, pp. 152–170. https://doi.org/10.1109/msp.2005.1550195.

[37] Agarwal, A., Kumar, S., and Sycara, K., "Learning transferable cooperative behavior in multi-agent teams," *arXiv preprint arXiv:1906.01202*, 2019. https://doi.org/10.48550/arXiv.1906.01202.

[38] Judge, G., "Applied multivariate analysis," *Journal of Econometrics*, Vol. 3, No. 3, 1975, p. 320. https://doi.org/10.1016/0304-4076(75)90039-1.

[39] Michael, N., and Kumar, V., "Planning and control of ensembles of robots with non-holonomic constraints," *The International Journal of Robotics Research*, Vol. 28, No. 8, 2009, pp. 962–975. https://doi.org/10.1177/0278364909340280.

[40] Pierson, A., and Schwager, M., "Controlling noncooperative herds with robotic herders," *IEEE Transactions on Robotics*, Vol. 34, No. 2, 2017, pp. 517–525. https://doi.org/10.1109/tro.2017.2776308.