2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

Multi-Target Pursuit by a Decentralized Heterogeneous UAV Swarm using Deep Multi-Agent Reinforcement Learning

Maryam Kouzeghar[1], Youngbin Song[1], Malika Meghjani[1], Roland Bouffanais[2]

Abstract— **Multi-agent pursuit-evasion tasks involving intelligent targets are notoriously challenging coordination problems. In this paper, we investigate new ways to learn such coordinated behaviors of unmanned aerial vehicles (UAVs) aimed at keeping track of multiple evasive targets. Within a Multi-Agent Reinforcement Learning (MARL) framework, we specifically propose a variant of the Multi-Agent Deep Deterministic Policy Gradient (MADDPG) method. Our approach addresses multi-target pursuit-evasion scenarios within non-stationary and unknown environments with random obstacles. In addition, given the critical role played by collective exploration in terms of detecting possible targets, we implement heterogeneous roles for the pursuers for enhanced exploratory actions balanced by exploitation (i.e. tracking) of previously identified targets. Our proposed role-based MADDPG algorithm is not only able to track multiple targets, but also is able to explore for possible targets by means of the proposed Voronoi-based rewarding policy. We implemented, tested and validated our approach in a simulation environment prior to deploying a real-world multi-robot system comprising of Crazyflie drones. Our results demonstrate that a multi-agent pursuit team has the ability to learn highly efficient coordinated control policies in terms of target tracking and exploration even when confronted with multiple fast evasive targets in complex environments.**

2023 IEEE International Conference on Robotics and Automation (ICRA) | 979-8-3503-2365-8/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICRA48891.2023.10160919

I. Introduction

In contrast to traditional monolithic systems, large multi-robot systems (MRS) can be operated collectively and dynamically at significantly lower cost and operational complexity [1]. For instance, robot swarms bring about decentralization, scalability, flexibility and fault-tolerance at the group level, while yielding unparalleled effectiveness when operating in unknown, dynamic and unstructured environments [2]. The most critical design component for swarming is the set of behavioral agent's rule leading to effective emergent collective actions. However, as has been widely acknowledged, a key challenge with swarms is the lack of a systematic way of mapping the expected system-level actions with the low-level update rules. Multi-Agent Reinforcement Learning (MARL) is one promising approach towards addressing this challenge. By design, robot swarms operate on the premise of fully decentralized control based on information gathered locally by individual agents [1]. The emergence of swarm intelligence critically depends on the ability to identify an appropriate set of behavioral rules suited for a particular cooperative control strategy aimed at driving the collective response for a particular task at hand. In [3], a range of swarming behaviors are obtained from an ad-hoc combination of "avoidance" and "attraction" behaviors, which are common in the widely used Alignment–Attraction–Avoidance (AAA) model for biological swarms [1]. A more systematic approach to infer the set of behavioral strategies has recently been achieved by means of Deep Reinforcement Learning [4].

In this paper, we propose an approach for learning swarm behaviors dedicated to a class of problems of multi-target pursuit-evasion with partial observability and non-stationarity of environmental features. Specifically, we propose a variant of Multi-Agent Deep Deterministic Policy Gradient (MADDPG) algorithm which is known for its effectiveness in dealing with non-stationarity. In addition, we aim to address scenarios with multiple, evasive, fast and learning targets which further increases the complexity of pursuit-evasion problem. This requires the agents to address the well-known dilemma of exploration and exploitation in multi-agent task allocation problems [5], which is paramount here. To this aim, the MADDPG approach is extended to enable autonomous heterogeneous role assignments in terms of exploring the environment and pursuing identified targets—i.e., exploitation. Specifically, the contributions of this paper are: (a) a role-based MARL model which can track multiple, evading, fast and learning targets while exploring the environment, (b) reward functions that help to develop two heterogeneous agent roles within the cooperative pursuit team enabling them to explore and track (exploit) targets and (c) validation in simulation as well as on physical platforms using Crazyflie micro unmanned aerial vehicles (UAVs).

II. Related Work

Pursuit-evasion is a well-known class of problems in optimization for which a number of analytical solutions have been identified given a known dynamics of the environment [6]–[8]. However, in real-world applications and with decentralized MRS, pursuit-evasion is a challenging problem due to the inevitable uncertainties and complexities in the environment. To address these challenges, reinforcement learning (RL) has been considered to deal with pursuit-evasion in unknown environments [9]. Given the efficiency of many single-agent RL algorithms such as Deep Q Network (DQN) [10], accelerated RL strategies using experience replay [11], self-organizing maps [12], Deep Deterministic Policy Gradient (DDPG) [13], Trust Region Policy Optimization (TRPO) [14] and Proximal Policy Optimization (PPO) [15], there is no doubt about the high potential of MARL in dealing with pursuit-evasion problems since such

[1]Maryam Kouzeghar, Malika Meghjani and Youngbin Song are with Singapore University of Technology and Design, Singapore. {maryam_kouzeghar, malika_meghjani, youngbin_song}@sutd.edu.sg,
[2]Roland Bouffanais is with the University of Ottawa, Canada. roland.bouffanais@uottawa.ca

scenarios involve a multi-agent system (MAS) with at least two groups of agents (pursuer and evader).

The state-of-the-art MARL literature can largely be classified into three distinct thrusts: (1) fully centralized RL among multiple agents, which is the predominant approach [16], (2) centralized training and decentralized execution (CTDE) [4], [17]–[19] applicable to MRS and robot swarms that are designed for decentralization in action, and (3) a fully decentralized approach, both in learning and execution [20]. Further studies and attempts of using Actor-Critic networks have been reported in [18], [21], [22]. Actor-Critic style is basically a combination of value-based and policy-based RL where the actor network decides which action to take (policy-based) and the critic network evaluates the action (value-based). MADDPG follows Actor-critic style [18], and as mentioned above addresses the critical issue of non-stationarity in swarm learning by deploying a centralized critic mechanism that has access to both observations and actions of all agents. This centralized critic on one hand, makes MADDPG a CTDE algorithm which is a suitable approach for MAS and swarms in particular because it yields a decentralized execution with distributed control of swarms, and on the other hand, it is capable of accommodating opposing teams of agents in a competitive style while enabling coordination within a team. In addition, non-stationarity increases the complexity of the problem when dealing with several agents learning concurrently because they are continuously changing policy during training and thus, the environment becomes even more non-stationary from the perspective of individual agents.

In our previous work, we modified the MADDPG algorithm to address the ocean monitoring application using a swarm of autonomous surface vehicles [23]. This work was motivated by a swarm of identical autonomous buoys [3], as well as a heterogeneous swarm [24], meant to carry out dynamic monitoring of large waterbodies. These large-scale physical systems operate autonomously by means of a range of classical swarming behaviors—e.g., aggregation, geofencing, heading consensus [25]. While in [3], adaptive environmental monitoring in dynamic regions is addressed, the framework does not include swarm learning and instead applies biologically inspired, rule-based swarm intelligence models. As mentioned, our ability to identify an appropriate set of behavioral rules suited for a specific cooperative control strategy, is a particular challenge which is at the core of this paper. Specifically, we aim to develop MARL approaches for multi-player pursuit-evasion with enhanced exploration for a heterogeneous swarm of UAVs operating in a fully decentralized manner. While many recent studies on multi-player pursuit-evasion consider a single evading target whether in the framework of MARL [9], [26]–[29] or distributed control [30]–[32], our efforts to address the pursuit of multiple superior evaders, specially in MARL framework, highlights the value of this study. In [33], tackling multiple evaders is attempted in a multi-agent pursuit based on application of self-organizing feature map and reinforcement learning. However, our work enables heterogeneous explorative agents alongside multiple-target-tracking agents in decentralized MARL framework which has better maneuvering capability compared to [33].

III. MADDPG Framework

As mentioned above, MADDPG is a CTDE approach, while our proposed variant of MADDPG, named as Role-based MADDPG, not only provides decentralized execution, but also escalates original MADDPG to semi-decentralized training with individually defined rewarding policies for heterogeneous agents while continuing to have a centralized critic. In this section, we briefly provide the fundamentals of the MADDPG algorithm to later build the Role-based MADDPG upon it. A schematic diagram of the key components of the MADDPG algorithm is shown in Fig. 1 highlighting, the key CTDE concept. The main idea in this Policy Gradient approach is to maximize the objective $J(\theta) = \mathbb{E}_{s\sim p^{\pi}, a\sim\pi_\theta}[\mathcal{R}]$ by taking steps in the direction of the gradient $\nabla_\theta J(\theta) = \mathbb{E}_{s\sim p^{\pi}, a\sim\pi_\theta}[\nabla_\theta \log \pi_\theta(a|s) Q^{\pi}(s,a)]$, and thus directly adjusting the parameter θ of the policy π [34]. By extending the policy gradient framework to deterministic policies $\mu_\theta : S \to A$, one can write the gradient of the objective function as $\nabla_\theta J(\theta) = \mathbb{E}_{s\sim\mathcal{D}}[\nabla_\theta \mu_\theta(a|s) \nabla_a Q^{\mu}(s,a)|_{a=\mu_\theta(s)}]$ [18].

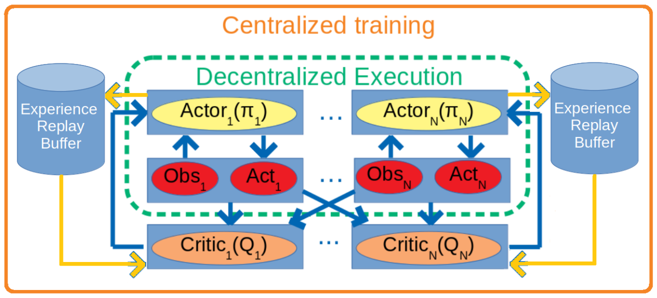


Fig. 1. A schematic diagram of the MADDPG algorithm highlighting the decentralized execution core surrounded by the centralized training

For efficient off-policy training, MADDPG uses an experience replay buffer $\mathcal{D}$ that stores the transition experiences of all agents for each time step t expressed as $e_t = (\mathbf{O_t}, \mathbf{A_t}, r_t, \acute{\mathbf{O}}_\mathbf{t})$ in which the joint observations of all agents is denoted by $\mathbf{O} = (O_1, ..., O_N)$, joint actions $\mathbf{A} = (a_1, ..., a_N)$, the shared reward is r, and $\acute{\mathbf{O}}$ is the new state (observation). To train the agents, a random mini-batch of these transitions are sampled from $\mathcal{D}$ which contains K transitions and is expressed as $(e_1, ..., e_K)$ with $e_j = (\mathbf{O_j}, \mathbf{A_j}, r_j, \acute{\mathbf{O}}_\mathbf{j})$ and $j = 1, 2, ..., K$. To update an agent's centralized critic, a one-step look-ahead TD-error is utilized to minimize the loss function $\mathcal{L}(\theta_i) = \frac{1}{K}\sum_j \left(y^i - Q_i^{\mu}(\mathbf{O}^j, \mathbf{A}^j)\right)^2$ with $y^j = r_i^j + \gamma Q_i^{\acute{\mu}}(\acute{\mathbf{O}}^j, \acute{\mathbf{A}}^j) \mid_{\acute{a}=\acute{\mu}(o^j)}$ where $0 < \gamma \leq 1$ is the discount factor (penalty for uncertainty of future rewards). Furthermore, similar to DDPG, the deterministic policy gradient $\nabla_{\theta_i} J \approx \frac{1}{K}\sum_j \nabla_{\theta_i}\mu_i(o_i^j)\nabla_{a_i} Q_i^{\mu}(\mathbf{O}^j, \mathbf{A}^j) \mid_{a_i=\mu_i(o_i^j)}$ is used to update each agent's actor parameters, and ultimately, the target networks are soft updated at a predefined number of steps using $\acute{\theta}_i \leftarrow \tau\theta_i + (1-\eta)\acute{\theta}_i$ where $0 < \eta \leq 1$ is the learning rate [18].

IV. Problem Formulation

The basic MADDPG algorithm [18] is applied in the Multi-Agent Particle Environment (MPE) from OpenAI [35]. In the MPE environment, the state-space is a continuous two-dimensional (2D) space representing agent positions and the action-space is continuous motion commands also in 2D. Hence, the state-space vector (same as the observation vector) contains the agent's position in two dimensions (x, y), and agent's velocity in the x and y directions. It is worth highlighting that our MAS has to contend with additional environmental complexities mainly associated with the following features: (a) random obstacles and random agent initialization which together help the MRS to deal with unknown arrangements of obstacles in cluttered environments and (b) dealing with multiple faster and evading targets that are equipped with the capability of reinforcement learning for escaping from pursuing agents. Lastly, we consider the following assumptions: (a) there is at least one pursuer per evader and (b) the agents propagate the target location to neighboring agents through a communication network (similar to real-world pursuit-evasion applications).

V. Role-based MADDPG for Multi-target Pursuit-Evasion-Exploration

We introduce a new role-based reward policy for a swarm of heterogeneous pursuers and propose a variant of the MADDPG algorithm. It is worth emphasizing that the heterogeneity in agent's role is reflected in the reward functions, which further highlights the decentralized capability of our proposed role-based MADDPG approach. However, the centralized critic remains in our MARL framework for training phase only. The decentralized reward policies allows us to define any required role for heterogeneous agents. Specifically, we use it for trading-off exploration vs target tracking which is a highly desirable outcome. It is also worth highlighting that considering both pursuers and evaders in a joint space within a competitive framework substantially helps both types of agents to improve their intelligence while training with intelligent opponents. Thus, our proposed role-based reward components are presented below.

Bounding to Space: In order to consistently keep the agents inside a defined region, at each step, all agents are constrained to be within specified boundaries, and in case the agents go beyond the bounded region, they are given a negative reward. This bounding reward overcomes the problem of sparse reward. As an example, for an experimental $6m \times 6m$ environment defined over $[-3, 3]$ in $2D$, the bounding reward is then defined as in Eq. (1) where c_1 and c_2 are positive constant values for thresholds and other constants (i.e., 3 and 9) are to compare agents' location (x_i, y_i) with the space boundaries.

$$B_i = -\min(\exp(3\,|x_i| - 9)\,, c_1) - \min(\exp(3\,|y_i| - 9)\,, c_2) \tag{1}$$

Collision Avoidance: To deal with collision avoidance amongst agents (both pursuers and targets) the reward strategy is given as follows. The distance between any two agents (say a_i and a_j) is checked ($\text{Dist}(i, j)$), and the agents receive a penalty—i.e., a negative reward c_3 when they get too close to one another, thereby violating the mutual physical boundary defined by their radius R. Thus the collision reward is calculated as in Eq. (2).

$$C_i = \begin{cases} -c_3 & \text{if } \text{Dist}(i, j) \leq R(i) + R(j), \\ 0 & \text{else.} \end{cases} \tag{2}$$

Learning Pursuers: During the course of training, the pursuers learn to keep track of the evading target based on the positive reward which it receives upon collision to target defined by CR_p where its value is tuned based on empirical results. Moreover, for handling multiple target tracking, the pursuers are also nurtured with a distance-based reward that is aimed to minimize the distance to target by penalizing the pursuers with the average distance to target DR_p (pursuer's distance reward). Overall, pursuers follow the rewarding policy given in Eq. (3).

$$R_{p_i} = B_i + C_i + CR_{p_i} + DR_{p_i} \tag{3}$$

Learning Evaders: The evading target learns to evade during the course of training where it is punished by a penalty value for being attacked by the pursuers, similar to a realistic pursuit-evasion scenario. A successful attack by a pursuer is defined by a collision with the evader and the penalty value for evader's collision reward CR_e is derived based on empirical results. Finally, evader's reward will be summarized as in Eq. (4).

$$R_{e_i} = B_i + C_i + CR_{e_i} \tag{4}$$

Exploratory Scouts: By applying the above-mentioned components of the rewarding policy, we can successfully represent a realistic multi-target pursuit-evasion scenario. However, beside successful exploitation of swarm for tracking the moving target, exploration is also a vital behavior. We propose an approach to enhance the exploration on top of exploitation (target tracking). This feature will improve safety of the swarm by patrolling the overlooked regions of space while concurrently tracking the identified targets. To this end, we propose Role-based MADDPG. Specifically, we assign two types of roles to the pursuers: (a) pursing the known targets and (b) scouting for the prospect targets. The scouting role allows the pursuers to enhance environment exploration supported by Voronoi-cell-based rewarding policy. A Voronoi diagram is basically partitioning of a plane into regions close to each of a given set of objects (seeds) [36], and for each seed there is a corresponding region, called Voronoi cell, consisting of all points of the plane closer to that seed than to any other. In our case, the moving agents are objects to be considered as seeds, but scouting pursuers are the ones supposed to use the Voronoi-based rewarding such that the maximum Voronoi cell is iteratively minimized while spreading the agents throughout the region and fulfilling the exploration task. The corresponding rewarding algorithm is given in Algorithm 1 resulting in exploratory reward

$E_i = Vor\_Reward$, and finally the overall reward for scouts is given below in Eq. (5).

$$R_{s_i} = B_i + C_i + E_i. \quad (5)$$

Algorithm 1 Voronoi-based Exploration

```
for $Agent\ i = 1, 2, \ldots, (n_p + n_S + n_t)$ do
    Create Voronoi diagram
end for
Compute $Voronoi\_Areas$
Vor_Reward -= $Max(Voronoi\_Areas)$
```

VI. Simulation results

In this section, we compare qualitative and quantitative results for the MADDPG algorithm with our Role-based MADDPG for a multi-target pursuit-evasion scenario in a 6 m $\times$ 6 m space. In this regard, the specifications of our proposed role-based MADDPG follow the original MADDPG provided by OpenAI with a modification of the parameters used for rewards as follows. Moving targets are learning to evade based on receiving a penalty value of -10 upon being collided by pursuers. The targets are set to be 30% faster than the pursuers. Pursuers are governed with actor networks that have been trained with a positive reward (of $+20$) upon colliding with the targets. In addition, the reward for pursuers is shaped based on minimizing their distance to targets. All agents and obstacles are initialized randomly for execution. Moreover, collision avoidance is attained by adopting a penalty of -10 upon collision defined in Section V. The numeric values for penalties and rewards are derived based on empirical results.

In addition, to empirically determine the number of pursuers vs. number of scouts, first we considered a double-target scenario with no scouts and increased the number of pursuers (n_p) incrementally until the average minimum distance to targets fell under the sensory range of the robot (here considered as 50 cm). With the results provided in Table I, $n_p = 5$ is determined as the sufficient number of pursuers to be deployed.

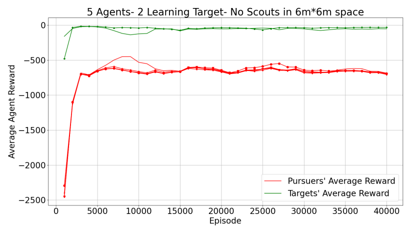
TABLE I: Distance to target analysis for determining n_p (first 2k episodes truncated)

Mean Dist to targets	$n_p = 2$	$n_p = 3$	$n_p = 4$	$n_p = 5$	$n_p = 6$
Min	0.6307	0.54807	0.5029	0.4616	0.4613

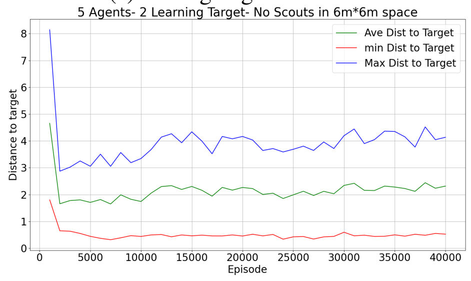
In all the tables reporting distance to target values, the average value is reported for $2,000$ episodes onward to consider the system condition after stable training. Once the number of pursuers were fixed, we incrementally added the number of scouts and measured the percentage of area coverage. It was observed that with $n_s = 5$, the swarm reaches to an average overall sensory coverage of around 35% over the 6 m $\times$ 6 m space. Considering that our MRS is not supposed to deal with area coverage optimization, this amount of coverage is suitable for showing explorative behavior alongside target tracking within the heterogeneous MRS. It is obvious that increasing the number of scouts would benefit the coverage percentage and generalizing this towards the optimal area coverage is straightforward but at the cost of increasing n_s, and higher computational complexity in the learning phase which is accomplished with decentralized reward policies.

A. Results for Multi-Target MADDPG Algorithm

The original MADDPG algorithm is here extended to accommodate multiple targets, specifically with 5 pursuers tracking 2 targets in an environment with 3 randomized obstacles. In order to fulfill the double-target scenario, the pursuit team is split into two sub-teams for tracking targets. From the swarm's behavior, it is observed that the pursuers can successfully keep track of faster targets. However, the whole swarm is dragged towards the current position of targets and other parts of the region lacking any targets, will remain out of sight by the swarm. This is the reason we propose enhancing exploration by adding scout pursuers in our proposed approach. Furthermore, the average reward of agents resulting from the training phase is shown in Fig. 2a, which illustrates how the pursuers (in red) and the evaders (in green) are continuously learning and increasing their reward until the latter plateaus showing stable training. Finally, Fig. 2b depicts the average values for minimum, average, and maximum distance to targets. Given that the pursuers and evaders are respectively designed to have a diameter of 0.15 m and 0.1 m, average minimum distance to targets is observed as 0.4 m (according to Table II) which is an acceptable range to keep track of the two faster targets. For a single target, this metric (i.e., the average minimum distance to target) reaches a value of 0.2 m (not shown here), which provides a confirmation that the pursuing mission (i.e., target tracking) has been successful.



(a) Average agent rewards



(b) Distance to targets

Fig. 2. Rewarding and distance-to-target metric for Multi-Target MADDPG (with 5 pursuers vs. 2 evaders)

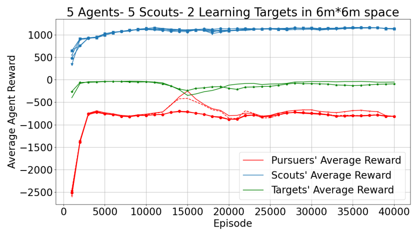
B. Results for Role-based MADDPG Algorithm

In the role-based MADDPG algorithm, we introduce scouts for environment exploration. In our results, evaders,

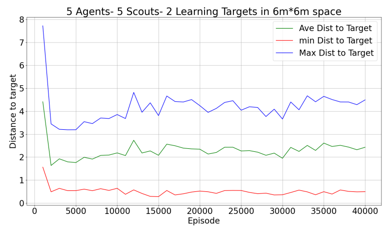
TABLE II: Distance to Targets- Multi-Target MADDPG (first 2k episodes truncated)

mean of min dist	*mean of ave dist*	*mean of max dist*
0.4616	2.1048	3.8639

pursuers and scouts are respectively trained with rewarding policies illustrated earlier in section V. Furthermore, the agents' average rewards resulting from training phase is shown in Fig. 3a where pursuers', evaders' and scouts' rewards are following their individual rewarding levels, and it is obvious that by adding scout agents, the reward level for pursuers and evaders is not effected due to the decentralized rewarding policies for heterogeneous agents. Meanwhile scouts and pursuers are interrelated via the communication network among neighbors (as mentioned in Section IV, assumption (b)). Finally Fig. 3b depicts the minimum, average and maximum distance to targets for Role-based MADDPG, and a measure of distance to intelligent moving targets is summarized in Table. III, and identically, the average minimum distance to the faster targets (observed as 0.4 m) still can confirm a viable tracking and a successful pursuit mission.



(a) Average agent rewards



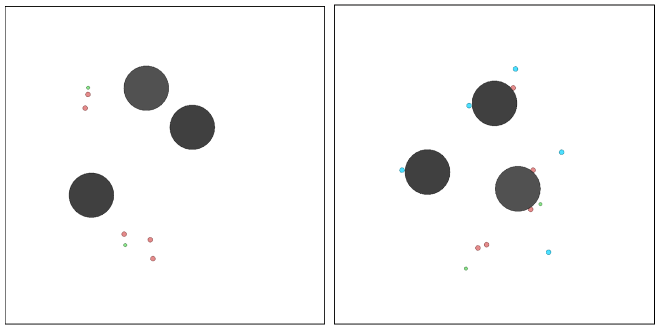
(b) Distance to targets

Fig. 3. Rewarding and distance-to-target metric for Multi-Target Role-based MADDPG (with 5 scouts and 5 pursuers vs. 2 evaders)

TABLE III: Distance to Targets- Role-based MADDPG (first 2k episodes truncated)

mean of min dist	*mean of ave dist*	*mean of max dist*
0.4745	2.2609	4.1233

Lastly, compared to the original MADDPG algorithm, our proposed role-based MADDPG algorithm has not only been able to track multiple targets, but also has been able to explore the overlooked parts of space by means of the the proposed Voronoi-based rewarding policy. To serve a qualitative comparison, Fig. 4a shows the final result of multi-target MADDPG for one execution phase where 2 evading targets are shown in green, 5 pursuing agents are shown in red and randomized static obstacles represented by large circles, and in Fig. 4b a qualitative result of Role-based MADDPG is given incorporating 5 scouts (in blue) into the initial multi-target scenario, and we finally find the pursuers tracking the targets and the scouts exploring the environment (see the demo video: https://youtu.be/aNSCsZv76qY).



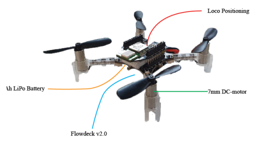
(a) (b)

Fig. 4. Qualitative results of multi-target (a) MADDPG vs. (b) Role-based MADDPG with 2 targets (in green), 5 pursuers (in red) and 5 scouts (in blue) in 6 m × 6 m space

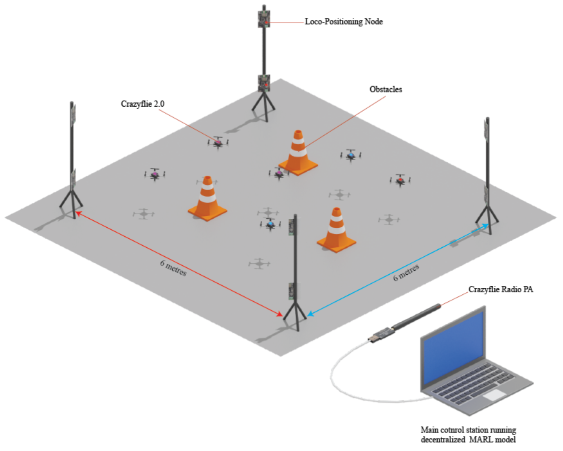
VII. Real-world results with Crazyflie Drones

For the real-world experiments, we used *Crazyflie 2.1* nanocopter developed by Bitcraze to showcase our role-based MARL framework. Given the small form factor of the drones, we used an external computer for computing needs. The drones communicate with the computer, using Bitcraze's *Crazyradio PA*. Additionally, *Loco-Positioning Decks* were attached to each drone along with *Loco-Positioning Nodes* on each corner of the experimental area. The *Loco-Positioning System* works based on Ultra Wide Band (UWB) technology and allows us to send position-based movement commands from the computer using the radio communication channels which can be set using Bitcraze's Crazyflie Client. In order to sense the height of flight for the drones, a *Flowdeck v2* was attached to the bottom of each drone. The decks of a Crazyflie drone mentioned above, are illustrated in Fig. 5a. For performance assessment of above-mentioned MARL models, we set up an experimental environment of 6 m × 6 m that consists of the flight region, a computer as control station, a USB radio antenna and 4 stands on the corners to hold UWB anchors for drones' positioning (two anchors attached to each stand to fulfill the drone localization with totally 8 UWB anchors). Within the experimental environment depicted in Fig. 5b, we are using 3 traffic safety cones as static obstacles and a total of 6 drones (3 scouts, 2 pursuers and 1 faster evading target) have been utilized to validate our proposed Role-based MADDPG algorithm.

At the start of the experiment, we place 6 drones at their initial positions within the experimental environment and let the swarm fly through the region commanded by MARL model. The mission is considered complete when the pursuers, scouts and evaders are showing expected (learned) heterogeneous behaviors of target tracking, exploration and evasion respectively. Screenshot of the demo are illustrated in Fig. 6 where Fig. 6a shows the initial starting points of the

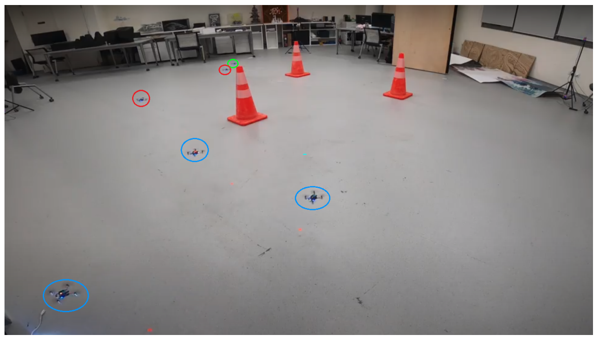


(a)

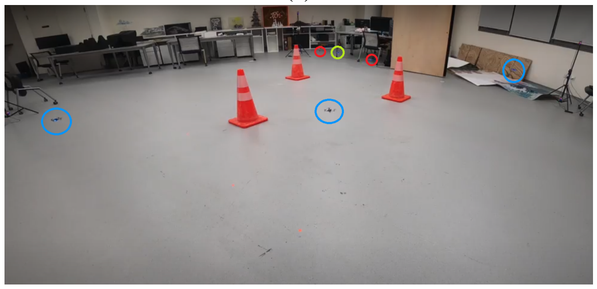


(b)

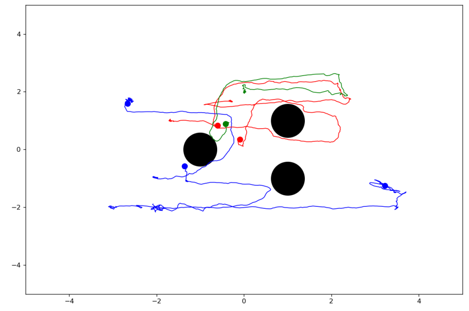
Fig. 5. Real experimental environment. (a) An illustration of Crazyflie decks. (b) An illustration of the experimental test-bed including the UWB anchors, drones, safety cones, and the radio antenna connected to compute.



(a)



(b)



(c)

Fig. 6. Decentralized heterogeneous UAV swarm with Role-based MADDPG- drone demo on multi-player pursuit-evasion with 2 pursuers in red, 1 intelligent evader in green and 3 scouts in blue. (a) Initial frame, (b) Final frame, (c) Complete bird-eye-view trajectory from UWB readings.

agents and Fig. 6b illustrates the final position of the agents after showcasing the learned behavior for corresponding heterogeneous roles. Colored annotations in the provided demo screenshots are following the agents' color code in OpenAI simulation results (red pursuers, green target and blue scouts). In Fig. 6b, we finally observe that the pursuers are continuously keeping track of the target and the scouts are fairly scattered in the environment to handle exploration. In addition, Fig. 6c illustrates the full trajectory of UWB readings from real drone experiment where at the end of the pursuit mission, based on the experimental UWB data, the average distance to target (from pursuers) is calculated to be 0.4983 m. This distance falls under the sensory range of the robot (50 cm) and the pursuit mission is perceived to be successful in this regard. Please see the demo video for further illustrations on the real drone experiment along with OpenAI simulation outputs and real-time UWB readings.

VIII. Conclusion and Future Works

In this paper, based on multi-agent reinforcement learning (MARL), a collaborative/competitive framework is considered for a swarm of UAV agents which are tasked to fulfill pursuit-evasion and also carry out successful exploration. Collaborative task is performed among pursuers and scouts to fulfill the pursuit-exploration mission while pursuers are competing with evaders in terms of target tracking. We have proposed a variant of a MARL algorithm called Role-based Multi-Agent Deep Deterministic Policy Gradient (MADDPG) and apart from introducing rewarding policies for MARL framework to generalize towards a multi-target pursuit-evasion scenario, we have proposed an approach based on Voronoi-cells concept to develop exploration on top of the multi-target pursuit-evasion. The MARL training and simulation results are implemented in MPE environment from OpenAI and the real world experiments are provided on Crazyflie drones. The simulation and real-world results show that the trained models have well adopted the expected behaviors and finally we find the pursuers successfully tracking the faster multiple intelligent targets and also we have some agents involved in exploration task. As a scope for future work, this study can be generalized towards specific applications by designing required swarm behaviors, and considering partial observability among the swarm to rely on emergent communication networks.

Acknowledgement

This research/project is supported by the National Research Foundation Singapore and DSO National Laboratories under the AI Singapore Programme (AISG Award No: AISG2-RP-2020-016).

References

[1] R. Bouffanais, *Design and Control of Swarm Dynamics*, Springer, 2016.

[2] M. Dorigo, G. Théraulaz, and V. Trianni, "Swarm robotics: Past, present, and future," *Proceedings of the IEEE*, vol. 109, no. 7, pp. 1152–1165, 2021.

[3] B. M. Zoss, D. Mateo, Y. K. Kuan, G. Tokić, M. Chamanbaz, L. Goh, F. Vallegra, R. Bouffanais, and D. K. Yue, "Distributed system of autonomous buoys for scalable deployment and monitoring of large waterbodies," *Auton. Robots*, vol. 42, no. 8, pp. 1669–1689, Dec. 2018.

[4] M. Hüttenrauch, A. Šošić, and G. Neumann, "Deep reinforcement learning for swarm systems," *Journal of Machine Learning Research*, vol. 20, no. 54, pp. 1–31, 2019.

[5] H. L. Kwa, J. L. Kit, and R. Bouffanais, "Balancing collective exploration and exploitation in multi-agent and multi-robot systems: A review," *Frontiers in Robotics and AI*, vol. 8, pp. 771520, 2 2022.

[6] M. Pachter, A. Von Moll, E. Garcia, D. Casbeer, and D. Milutinović, "Cooperative pursuit by multiple pursuers of a single evader," *Journal of Aerospace Information Systems*, vol. 17, no. 8, pp. 371–389, Aug. 2020.

[7] A. Von Moll, D. W. Casbeer, E. Garcia, and D. Milutinović, "Pursuit-evasion of an evader by multiple pursuers," in *2018 International Conference on Unmanned Aircraft Systems (ICUAS)*. June 2018, IEEE.

[8] G. Ibragimov and S. Luckraz, "On a characterization of evasion strategies for pursuit-evasion games on graphs," *Journal of Optimization Theory and Applications*, vol. 175, no. 2, pp. 590–596, Aug. 2017.

[9] 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.

[10] V. Mnih, K. Kavukcuoglu, D. Silver, A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb 2015.

[11] T. G. Karimpanal and R. Bouffanais, "Experience replay using transition sequences," *Frontiers in Neurorobotics*, vol. 12, pp. 32, 2018.

[12] T. G. Karimpanal and R. Bouffanais, "Self-organizing maps for storage and transfer of knowledge in reinforcement learning," *Adaptive Behavior*, vol. 27, no. 2, pp. 111–126, 2019.

[13] T. L. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[14] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *Proceedings of the 32nd International Conference on Machine Learning*, Francis Bach and David Blei, Eds., Lille, France, 07–09 Jul 2015, vol. 37 of *Proceedings of Machine Learning Research*, pp. 1889–1897, PMLR.

[15] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms.," *CoRR*, vol. abs/1707.06347, 2017.

[16] J. K. Gupta, M. Egorov, and M. Kochenderfer, "Cooperative multi-agent control using deep reinforcement learning," in *Autonomous Agents and Multiagent Systems*, G. Sukthankar and J. A. Rodriguez-Aguilar, Eds., Cham, 2017, pp. 66–83, Springer International Publishing.

[17] M. Hüttenrauch, A. Šošić, and G. Neumann, "Local communication protocols for learning complex swarm behaviors with deep reinforcement learning," in *International Conference on Swarm Intelligence (ANTS)*. 2018, Springer International Publishing.

[18] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, USA, 2017, NIPS'17, pp. 6382–6393, Curran Associates Inc.

[19] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," in *AAAI Conference on Artificial Intelligence*, 2018.

[20] K. Zhang, Z. Yang, H. Liu, T. Zhang, and T. Basar, "Fully decentralized multi-agent reinforcement learning with networked agents," in *Proceedings of the 35th International Conference on Machine Learning*, Jennifer Dy and Andreas Krause, Eds., Stockholmsmssan, Stockholm Sweden, 10–15 Jul 2018, vol. 80 of *Proceedings of Machine Learning Research*, pp. 5872–5881, PMLR.

[21] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, The MIT Press, second edition, 2018.

[22] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," 2016, cite arxiv:1602.01783.

[23] M. Kouzehgar, M. Meghjani, and R. Bouffanais, "Multi-agent reinforcement learning for dynamic ocean monitoring by a swarm of buoys," in *Global Oceans 2020: Singapore U.S. Gulf Coast*, 2020, pp. 1–8.

[24] F. Vallegra, D. Mateo, G Tokic, R. Bouffanais, and D. K. P. Yue, "Gradual collective upgrade of a swarm of autonomous buoys for dynamic ocean monitoring," *OCEANS 2018 MTS/IEEE Charleston*, pp. 1–7, 2018.

[25] M. Chamanbaz, D. Mateo, B. M. Zoss, G. Toki, E. Wilhelm, R. Bouffanais, and D. K. P. Yue, "Swarm-enabling technology for multi-robot systems," *Frontiers in Robotics and AI*, vol. 4, pp. 12, 2017.

[26] Jhanani Selvakumar and Efstathios Bakolas, "Min-max q-learning for multi-player pursuit-evasion games," 03 2020.

[27] Zhe-Yang Zhu and Cheng-Lin Liu, "Leader-following multi-agent coordination control accompanied with hierarchical q(λ)-learning for pursuit," *Frontiers in Control Engineering*, vol. 2, pp. 721475, 11 2021.

[28] Cristino de Souza, Rhys Newbury, Akansel Cosgun, Pedro Castillo, Boris Vidolov, and Dana Kuli, "Decentralized multi-agent pursuit using deep reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4552–4559, 2021.

[29] Xiaowei Fu, Jindong Zhu, Zhaoying Wei, Hui Wang, and Sili Li, "A uav pursuit-evasion strategy based on ddpg and imitation learning," *International Journal of Aerospace Engineering*, vol. 2022, pp. 3139610, May 2022.

[30] Shuang Ju, Jing Wang, and Liya Dou, "Enclosing control for multiagent systems with a moving target of unknown bounded velocity," *IEEE Transactions on Cybernetics*, pp. 1–10, 2021.

[31] Zhengyuan Zhou, Wei Zhang, Jerry Ding, Haomiao Huang, Duan M. Stipanovi, and Claire J. Tomlin, "Cooperative pursuit with voronoi partitions," *Automatica*, vol. 72, pp. 64–72, 2016.

[32] Yinjie Ni, Shuhua Gao, Sunan Huang, Cheng Xiang, Qinyuan Ren, and Tong Lee, "Multi-agent cooperative pursuit-evasion control using gene expression programming," 10 2021, pp. 1–6.

[33] Muhammad Zuhair Qadir, Songhao Piao, Haiyang Jiang, and Mohammed El Habib Souidi, "A novel approach for multi-agent cooperative pursuit to capture grouped evaders," *The Journal of Supercomputing*, vol. 76, no. 5, pp. 3416–3426, May 2020.

[34] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Advances in Neural Information Processing Systems 12*. 2000, vol. 12, pp. 1057–1063, MIT Press.

[35] I. Mordatch and P. Abbeel, "Emergence of grounded compositional language in multi-agent populations," *arXiv preprint arXiv:1703.04908*, 2017.

[36] *Voronoi Diagrams*, pp. 147–171, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.