Hybrid Formation Control for Multi-Robot Hunters Based on Multi-Agent Deep Deterministic Policy Gradient

Article *in* MENDEL · December 2021

DOI: 10.13164/mendel.2021.2.023

2 authors:

Oussama Hamed
Aix-Marseille University
8 PUBLICATIONS **75** CITATIONS

SEE PROFILE

Mohamed Hamlich
Université Hassan II Mohammedia
52 PUBLICATIONS **358** CITATIONS

SEE PROFILE

Hybrid Formation Control for Multi-Robot Hunters Based on Multi-Agent Deep Deterministic Policy Gradient

Oussama Hamed[✉], **Mohamed Hamlich**

Complex Cyber Physical Systems Laboratory,ENSAM, Casablanca, University of Hassan II, Morocco

oussamahamed11@gmail.com[✉]

Abstract

The cooperation between mobile robots is one of the most important topics of interest to researchers, especially in the many areas in which it can be applied. Hunting a moving target with random behavior is an application that requires robust cooperation between several robots in the multi-robot system. This paper proposed a hybrid formation control for hunting a dynamic target which is based on wolves' hunting behavior in order to search and capture the prey quickly and avoid its escape and Multi Agent Deep Deterministic Policy Gradient (MADDPG) to plan an optimal accessible path to the desired position. The validity and the effectiveness of the proposed formation control are demonstrated with simulation results.

Keywords: *Reinforcement Learning, Multi-Robot System, Cooperative Hunting, Path Planning, Mobile Robot, Collaborative Robots.*

Received: 10 November 2021
Accepted: 14 December 2021
Published: 21 December 2021

1 Introduction

Research on formation control of mobile robot hunters has been studied in industry and academia and robustly applied to a variety of complex tasks such search and rescue, military, agriculture, industry [9]; where this formation control is for chasing and trapping a dynamic target which could be another robot or any kind of targets. In order to perform this hunting process, a team of cooperated and coordinated mobile robots is widely used [5]. For example, Hamed recently introduced a hunting strategy using a multi-robot system [5]. Many researches have been done on hunting with multi-robot systems problems. Among them, there are many methods for hunting a target by means of several mobile robots that are based on generative adversarial network [1], dynamic prediction [2] and Deep Reinforcement Learning (DRL) [11, 13]. The nature inspired methods [4] are effective in chasing a dynamic target with random behavior in real time in unexpected environments. Moreover, in the hunting process with multi-robot systems, the mobile robots must cooperate and coordinate in order to catch the target as quickly as possible and prevent it from escaping while avoiding obstacles in the environment. Therefore, a multi agent nature inspired method should be used with a combination of a path planning algorithm to reach the desired hunting position rapidly without colliding with obstacles. In this paper, we present a new formation control for chasing and trapping a dynamic target by means of multi mobile robots based on Wolf Swarm Algorithm [5] and Multi-Agent Deep Deterministic Policy Gradient (MADDPG) [10]. The use of this multi-robot cooperation strategy for hunting a dynamic target [5] is due to its high performance in the hunting process. In this strategy, distributed robots cooperate and take turns by analyzing information in real time in order to encircle the target and reach it quickly. The mobile robot hunters reach the hunting point rapidly and without colliding with the obstacles in the environment through MADDPG. Therefore, this algorithm is suitable for the path planning problems in real time under complex environments for multi agents. The simulation results show that the proposed algorithm helps to hunt a target which behaves with uncertain behavior in an obstacle-filled environment. The main contributions of this paper are: First, proposing a robust cooperation and coordination of multi-robot systems to plan an optimal, accurate, and accessible path to hunt dynamic targets in the presence of obstacles. Second, reducing the resources required for the hunting process by means of a set of robots in real time and quickly adapting to changing conditions during hunting. The rest of this paper is organized as follows: Section 2, presents an overview of related works. In Section 3, we summarize the necessary methods needed to create our work and describe the proposed method. Results evaluation and analysis are given in Section 4, followed by the future work and conclusion in Section 5.

2 Related Work

As related work, an improvised multi-robot cooperation strategy for hunting a dynamic target is introduced in [6]. This hunting strategy aims to hunt dynamic targets in search spaces that do not contain obstacles, while in this paper obstacles have been taken into account in addition to developing a hunting behavior of the robots. A multi-agent collaborative hunting algorithm based on game theory and Q-learning for a single escaper is proposed in [14]. The process of this algorithm is presented as follows: First, a cooperative hunting team is formed, and a cooperative hunting game model is developed. Second, the trajectory of the escaper's restricted T-step cumulative reward is formed by learning of the escaper's strategy choice, and the trajectory is modified to the hunter's strategy set. Finally, by solving the cooperative hunt game, the Nash equilibrium solution is discovered, and each hunter performs the equilibrium strategy to finish the hunt. In [3], Xiang proposed a fuzzy-based potential field hierarchical reinforcement learning approach for target hunting by multi-AUV in 3-D underwater environments to improve the efficiency of target hunting and the smoothness of AUV's trajectory. Kangrui proposed a scenario for multi-UAV in an urban environment for hunting a target using Deep Reinforcement Learning [15]. First, they create a multi-UAV hunting target game model and construct the reward mechanism. Second, they provide an enhanced Multi-Agent Deep Deterministic Policy Gradient (MADDPG) to compete against a game AI target. Finally, they create a virtual platform based on the Unity3D gaming engine to replicate the urban battle scenario.

3 Method

3.1 Improvised Multi-Robot Cooperation Strategy for Hunting a Dynamic Target

This chapter describes the multi-robot cooperation strategy for hunting a dynamic target [6]. This method is inspired from wolf hunting behavior. The robots in the swarm are classified into three classes; the leader, the followers, and the antagonists. N robots switch roles based on fitness function $\varnothing(R,T)$ where R is the coordinate of the robot and T is the coordinate of the target. The closer the robot is to the target the value of the function becomes larger. The robot becomes a leader if it has the highest value of Ø. The antagonist is a robot that follows its own way in order to search for prey. In the swarm the number of antagonists is an integer from the interval $[(N-1)/(\lambda+1),(N-1)/\lambda]$ where $\lambda = [1, N/2]$ is the antagonism proportion factor. In order to reduce the search area for the target, the angle between the leader and the reference x axis is calculated by the equation (1)-(2).

$$\Theta = 2\arctan\frac{R^1_y - R^a_{ky}}{\sqrt{(R^1_x - R^a_{kx}+1)^2+(R^1_y - R^a_{ky})^2}+R^1_x - R^a_{kx}+1} \tag{1}$$

$$\Theta' = \begin{cases}\Theta, & \Theta \in [0,\pi]\\ \Theta + 2\pi, & \Theta \notin [0,\pi]\end{cases} \tag{2}$$

Where R^a_{kx} and R^a_{ky} are the coordinates of the antagonist robot R^a_k. Using this angle, the seeking area is limited (equation (3)-(4)). The antagonist robot chooses its next position by examining all the possible temporary positions $\left(\widetilde{R^a_{kx}}, \widetilde{R^a_{ky}}\right)^{\mathrm{T}}_{\varphi}$ according to equation 3.

$$\begin{pmatrix}\widetilde{R^a_{kx}}\\ \widetilde{R^a_{ky}}\end{pmatrix}_{\varphi} = \begin{pmatrix}R^a_{kx}\\ R^a_{ky}\end{pmatrix} + \begin{pmatrix}\beta_1\\ \beta_2\end{pmatrix} \odot \begin{pmatrix}\cos(\frac{2\pi\phi}{\psi})\\ \sin(\frac{2\pi\phi}{\psi})\end{pmatrix} \tag{3}$$

where β_1 and β_2 are the seeking factors that are expressed in the equation (8). The factor ψ is the amount of the global directions whither the antagonist R^a_k searches for the prey, and φ is the advancing direction factor where:

$$\Phi = \left[\frac{\psi \times \Theta'}{2\pi} - l_1, \frac{\psi \times \Theta'}{2\pi} + l_2\right] \tag{4}$$

the antagonist chooses the temporary position with the highest function value, then this value is compared to that of the leader. The antagonist becomes the leader if

$$\varnothing(R^a_k, T) > \varnothing(R_1, T)$$

The rest of the robots are the followers. The follower has two behaviors and it switches them according to the distance between it and the leader Dconv. The distance of convergence Dconv is described as follows:

$$\text{Dconv}{=}\left(\frac{1}{\max(xs){-}\min(xs)}+\frac{1}{\max(ys){-}\min(ys)}\right)^{-1} \times \sigma \tag{5}$$

Where $\sigma =]0,1]$ is the convergence factor. (xs, ys) are the boundaries of the search space. The follower wolf behaves with the summoning behavior when the distance between it and the leader is greater than Dconv. The follower wolf advances to the leader, according the equation (6):

$$\begin{pmatrix}R^f_{jx}(i+1)\\ R^f_{jy}(i+1)\end{pmatrix} = \begin{pmatrix}R^f_{jx}(i)\\ R^f_{jy}(i)\end{pmatrix} + \begin{pmatrix}\alpha_1\\ \alpha_2\end{pmatrix} \odot \begin{pmatrix}R^l_x - R^f_{jx}(i)\\ R^l_y - R^f_{jy}(i)\end{pmatrix} \odot \begin{pmatrix}\varepsilon_1\\ \varepsilon_2\end{pmatrix} \odot \begin{pmatrix}|R^l_x - R^f_{jx}(i)|^{-1}\\ |R^l_y - R^f_{jy}(i)|^{-1}\end{pmatrix} \tag{6}$$

The deviation factors ε_1 and ε_2 are random numbers where $0 < (\varepsilon_1, \varepsilon_2) \leq 1$, and i is the iteration index. α is the advancing vector which is expressed in the equation (8).

When the distance between the follower robot and the leader is less than Dconv, this robot encircles the target. This behavior is called preying behavior. The update of follower's position is expressed by the expression:

$$\binom{R^f_{jx}(i{+}1)}{R^f_{jy}(i{+}1)} = \binom{R^f_{jx}(i)}{R^f_{jy}(i)} + \binom{\gamma_1}{\gamma_2} \odot \binom{R^l_x{-}R^f_{jx}(i)}{R^l_y{-}R^f_{jy}(i)} \odot \binom{\tau_1}{\tau_2} \tag{7}$$

τ_1 and τ_2 are the encirclement factors, where $0 \leq \tau_1, \tau_2 \leq 2\text{–}i/max\_Iteration$, and γ is the preying vector which is expressed in the equation (8). The vectors α, β and γ are calculated as follows:

$$\binom{\beta_1}{\beta_2} = \frac{4}{3} \cdot \binom{\gamma_1}{\gamma_2} = \frac{1}{2} \cdot \binom{\alpha_1}{\alpha_2} = \binom{\max(xs){-}\min(xs)}{\max(ys){-}\min(ys)} \cdot S \tag{8}$$

The scalar S is the step factor where $S \in]0,1[$.

3.2 Multi-Agent DDPG

Reinforcement learning (RL) is a branch of machine learning that focuses on making progressive decisions. The fundamental goal of this strategy is to figure out how agents should conduct actions in a given environment in order to maximize cumulative rewards as shown in Fig. 1. Markov decision processes (MDP) are an excellent mathematical model for RL problems, and a direct learning mechanism proposed to achieve the goal. In the following learning stage, the agent decides to obtain not only the current remuneration, but also cumulative remuneration [12]. The agent choses an action a at each state s which makes finite pairs of state and action $\{(s_0, a_0), (s_1, a_1), \dots\}$.

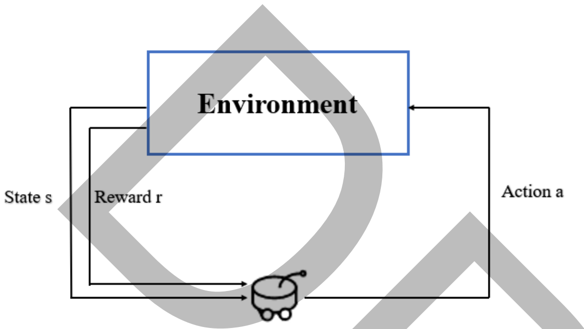

Figure 1: Structure of agent-environment interaction in MDP.

The return $G_t t$ is the total discounted reward for the sequence $\{(s_0, a_0), (s_1, a_1), \dots\}$ from time-step t:

$$G_t((s_0, a_0), (s_1, a_1), \dots) = \sum_{k=0}^{\infty} \gamma^k R_{t+k} \tag{9}$$

Where R_t is the reward at time-step t and γ is the discount factor $0 \leq \gamma \leq 1$.

The optimal policy π^* is the policy which maximizes the expected reward.

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\pi}[G_t((s_0, a_0), (s_1, a_1), \dots)] \tag{10}$$

Deep Reinforcement Learning (DRL) refers to the combination of Artificial Neural Network (ANN) and Reinforcement Learning. This combination allows the agent to pick an optimal action in a complex and non-deterministic environment.

MADDPG is an extension of DDPG which is proposed to handle continuous state-action spaces and to use a centralized planning with a decentralized execution for each agent [8]. During learning, all agents are guided by the centralized critic, where there is an entity that oversees the whole system of agents, instructing them on how to update their rules based on local information and how to teach each agent on local policies [7].

The agent i trains its centralized critic Q_i to minimize the following loss:

$$L(\theta^i) = \mathbb{E}_{X,a,x' \sim D}\left[(y^i - Q^i_{\mu}(x, a; \theta^i_Q))^2\right] \tag{11}$$

$$y^i = r^i + \gamma Q^i_{\mu'}(x', a'; \Theta^{i-}_Q)\,|_{a'^j = \mu'^j(o'^j)} \tag{12}$$

where $\mu = \{\mu^1, \dots, \mu^N\}$ is the set of deterministic policies of all agents, $x' = (o'^1, \dots, o'^N), \mu' = \{\mu'^1, \dots, \mu'^N\}$ is the set of target policies, D is the replay memory, and θ^{i-}_Q is the parameter of the target Q network for agent i. To update each agent's actor parameters, the deterministic policy gradient is used to maximize $J(\theta^i_\mu)$ as shown in the equation (13).

$$\nabla_{\theta^i_\mu} J(\mu^i) = \mathbb{E}_{X,a\sim D}\left[\nabla_{\theta^i_\mu} \mu^i(o^i) \nabla_{a'} Q^i_\mu(x, a)\,|_{a'^j = \mu'^j(o'^j)}\right] \tag{13}$$

3.3 Hybrid Formation Control (HFC) for multi hunters

In order to search, track and surround a dynamic target which moves according to a random behavior within a search space that contains obstacles, there are many mobile robots play the role of hunters. Each robot is made up of sensors that can sense how close it is to the target without knowing where is its location. At the beginning of the hunting process, the robots sense the target and communicate with each other in order to exchange sensing values between them, and the robot that senses the most value is the leader, while the others become followers and antagonists. From this step, the follower robots head towards the leader, while the antagonist robots search for prey, and when one of them becomes the closest, it takes the role of the leader. The robots follow the short and unobstructed path while avoiding collision with other robots while heading to their destination. The set of Robots denoted as Rn $\triangleq \{R_1, \dots, R_n\}$, the target T, the NF followers Rf $\triangleq \{R_1, \dots, R_{NF}\}$, and NA antagonist Ra $\triangleq \{R1, \dots, R_{NA}\}$. For the follower in both behaviors, its next destination is calculated according to the given formulas that contain random parameters, allowing the follower robots to go to other options but still maintain his main destination. The computation procedure for each follower can be presented as shown in Algorithm 1.

Algorithm 1 Follower HFC

```
for f=1:NF do
    if $||R_f, R_1|| >$ Dconv do
        Dp = equation (6)
    end if
    if $||R_f, R_1|| <$ Dconv do
        Dp = equation (7)
    end if
    if Dp is not accessible do
        Correct Dp
    end if
    Plan path with MADDPG ($R_f$ to Dp)
    $R_f \leftarrow$ Dp
    $R_l \leftarrow \arg\max_R(\varnothing(R_s, T))$ where $R_s = \{R_l, R_f\}$
end for
```

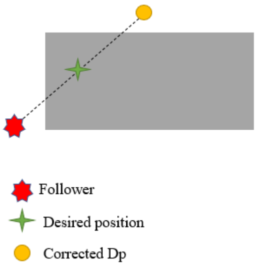

Figure 2: Robot Corrects its desired position.

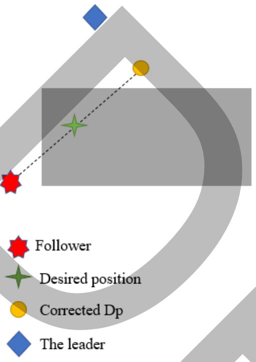

Figure 3: Robot encircles target with the corrected Dp.

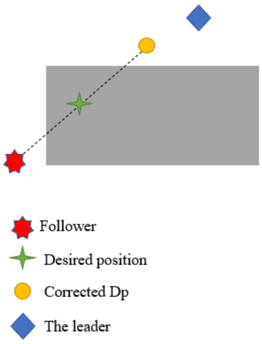

Figure 4: Robot anticipates target position with corrected Dp.

In lines (2, 5), the behavior of the follower is selected, and according to this behavior the desired position Dp is calculated. There are two cases for a desired destination when it is calculated, either the desired destination can be reached or this destination cannot be reached because of an obstacle. In lines (8, 9), if the desired position is not accessible, then a corrector corrects the desired position to make it accessible. The corrector is an algorithm that selects the closest available position away from the obstacle and makes it the new desired destination as shown in Fig. 2. In addition to being accessible, the corrected desired position helps in encircling the target (Fig. 3) or anticipating its position (Fig. 4). After determining the desired position, the MADDPG algorithm is used by the robot to reach this position quickly and without colliding with any obstacle.

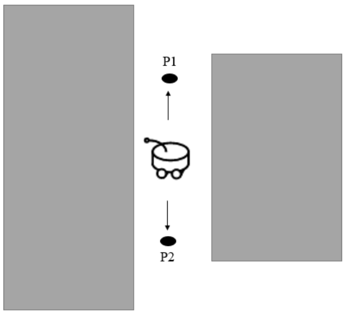

Figure 5: Antagonist robot is stuck.

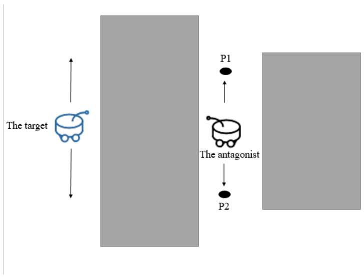

Figure 6: Antagonist robot and target are stuck .

Algorithm 2 Antagonist HFC

```
1: for f=1:NA do
2:     $\Theta$ = equation (1); $\Theta'$ = equation (2)
3:     for $\varphi = \frac{\psi\times\Theta'}{2\pi} - l_1 : \frac{\psi\times\Theta'}{2\pi} + l_2$ do
4:         $\left(\widetilde{R^a_{kx}}, \widetilde{R^a_{ky}}\right)$ = equation (3)
5:         if $\left(\widetilde{R^a_{kx}}, \widetilde{R^a_{ky}}\right)$ not accesible do
6:             continue
7:         end if
8:         Plan the path ($R^a$ to $\widetilde{R^a}$) with MADDPG
9:         if $\varnothing(R^a, T) \leq \varnothing(\widetilde{R^a}, T)$ do
10:            Memorize old $R^a$
11:            $R^a \leftarrow \widetilde{R^a}$
12:            break loop
13:        else
14:            Plan the path ($\widetilde{R^a}$ to $R^a$) with MADDPG
15:        end if
16:    end for
17:    $R_l \leftarrow \arg\max_R(\varnothing(R_s, T))$ where $R_s = \{R_l, R_f\}$
18: end for
```

In lines (5,6), the desired position is not corrected in order to reduce the calculations and execution time. Therefore, the antagonist robot looks for the next accessible position according to equation (3).

In some critical situations, i.g., Fig. 5, the antagonist is stuck in this narrow path and there are only two positions accessible (P1 and P2). To prevent this undesired situation, the antagonist memorizes its previous positions (line 10) and verifies the frequency of its visits to these positions in a short delay. The antagonist chooses another path if it is stuck even though this new path leads it away from the target. The other reason to get a stuck robot is the behavior of the target. If the target is stuck regardless of the reason (narrow path or random velocity) the antagonist would be stuck too (Fig. 6).

In such situations as shown in Fig. 7, the robot can reach its destination by following two paths according to the random parameters used in the control equations. However the path 1 seems to be the shortest to the target, the path 2 can be a better choice for the hunter because this target has unexpected behavior and random velocities.

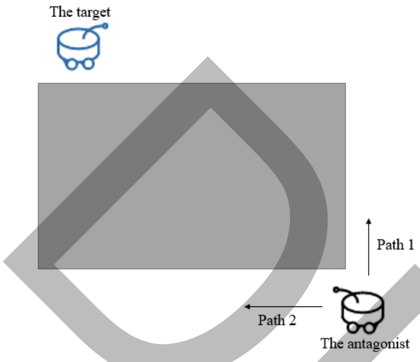

Figure 7: Robot has two paths to follow towards the target.

4 Results and Discussion

To verify our proposed method, we perform a simulation of the hunting process using four mobile robots. We assume the kinematic model of these robots does not have any constraint. The hunting environment is two-dimensional space with a size of 20 x 20 and includes many static obstacles with different sizes that are randomly placed. The initial position of the four hunters and the target are randomly generated as shown in Table 1. Fig. 8 shows the distribution of hunters and the target within the search space.

The goal of these robots is to find and catch the target which is moving with random behavior. In addition to hunting the target, robots have to avoid colliding with static obstacles and other robots (dynamic obstacles).

Table 1: Initial position for the hunters and the target

Robot	Position	Symbol
Hunter 1	[9.2124, 8.1487]	×
Hunter 2	[11.4781, 16.0214]	×
Hunter 3	[19.7411, 5.8412]	×
Hunter 4	[17.5324, 18.3316]	⋄
Target	[14.0012, 10.9896]	$*$

The target's movements are random (see Fig. 9) where T_x and T_y the coordinates of the target are updated: $T_x(i+1) = T_x(i) + \xi_1$ and $T_y(i+1) = T_y(i) + \xi_2$. ξ_1 and ξ_2 are random numbers where $-1.5 \leq (\xi_1, \xi_1) \leq 1.5$. The hunting operation can only be completed with at least three hunters. The goal is to make the hunter robots encircle and hunt the target in this aleatory situation, and make sure that is valid for any other situation. In Fig. 10, hunters plan an accessible and optimal path towards the target while keeping to follow it wherever it goes. Fig. 11 shows the hunters successfully hunt the target at the position [12.2055,12.7532] and the hunting process is well performed.

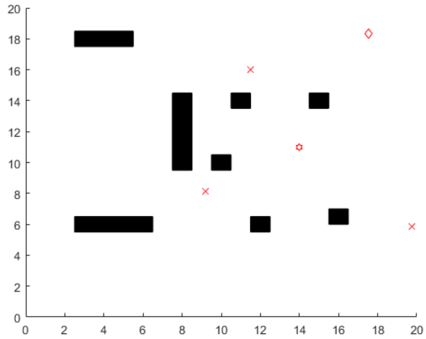

Figure 8: Initial position of the hunters and the target.

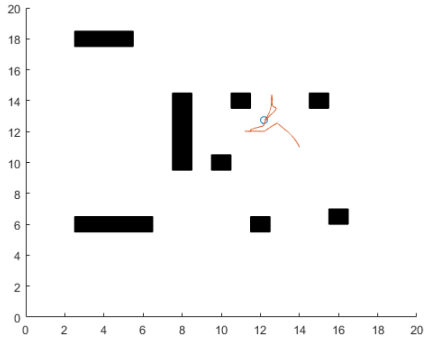

Figure 9: The path of the target during the hunting process.

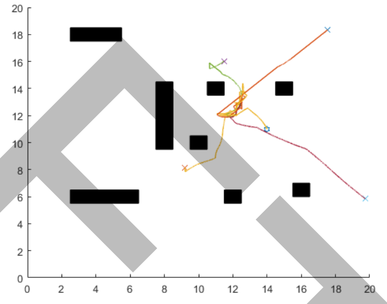

Figure 10: The path of each robot and the target.

Fig. 12 shows the progression of the best cost function which is the cost function of the leader since it is the closest robot to the target. The variations in this function are due to the random behavior of the target. The hunters are always in movement towards the target, and that is the reason for the function decreasing, such as from the 1st to the 6th iteration. Otherwise, the function increases when the target moves away from hunters like the 15th iteration. In the 22th

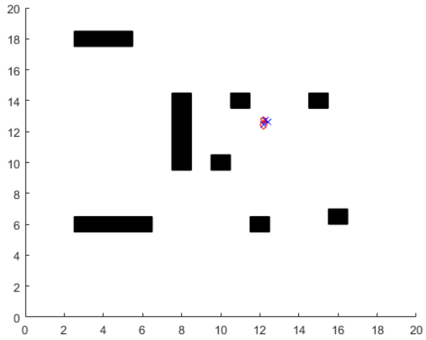

Figure 11: The hunters hunt the target.

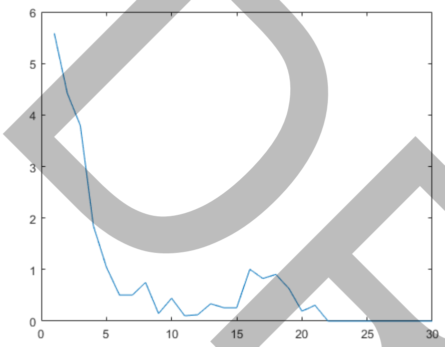

Figure 12: The cost function progression of the leader.

iteration, the function tends towards 0 which means the leader reaches that target with other hunters.

Fig. 13-16 show the trajectory for each robot in addition to the target. Each robot follows an accessible and secure path towards the target. In this experiment, the robots do not depend on deviations factors when they are far from the leader ($||R_f, R_l|| >>$ Dconv), therefore robots go in a straight line and that is introduced in the path of some robots. The random behavior of the robot when they are close to the leader helps to prevent the escape of the target.

The cost function progression of robot 1 is shown in Fig. 17. However, robot 1 is not always the leader, it is constantly getting closer to the target. The cost function progression of robot 1 increases and decreases in the same way as for the leader.

5 Conclusion and Future Work

This paper presents a novel hybrid formation control for multi robots in order to hunt a dynamic target with random behavior without collisions with obstacles into the environment. This formation control adopts a hunting strategy in order to catch the prey quickly and without allowing it to escape. The presence of the antagonist robot contributes to the re-deployment of the robots to the location of the prey and prevents the other robots from gathering in a wrong position. The use of MADDPG algorithm which is one of the well-known MARL algorithms helps to reach the desired

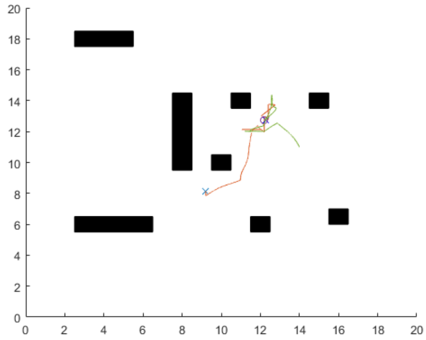

Figure 13: The path of robot 1.

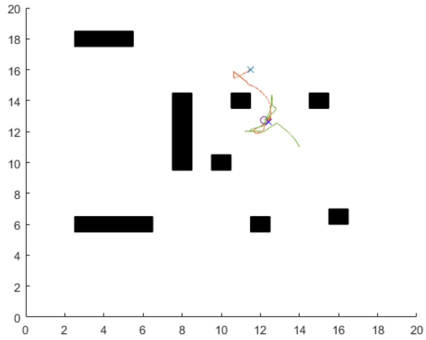

Figure 14: The path of robot 2.

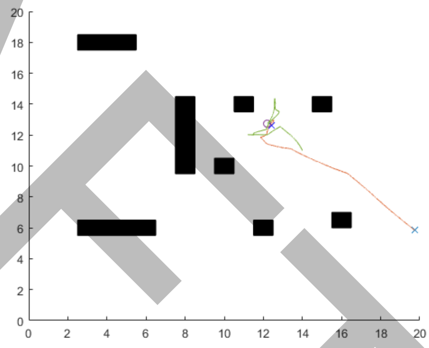

Figure 15: The path of robot 3.

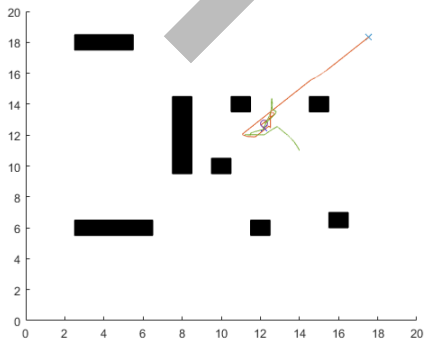

Figure 16: The path of robot 4.

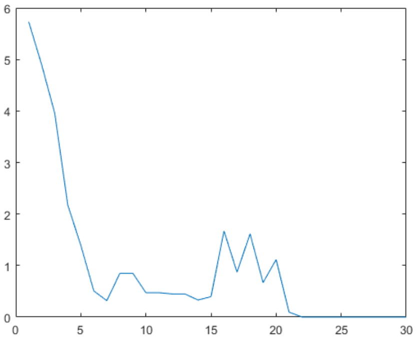

Figure 17: The cost function progression of robot 1.

position quickly and smoothly without collision with obstacles. The results of the simulation shows that the proposed hybrid formation control achieves the desired performance. As future work, multi-dynamic targets should be taken into account with more robot hunters. In addition, the search strategy must be developed to suit prey that moves more quickly than hunters.

References

[1] Cai, L., and Sun, Q. Multiautonomous underwater vehicle consistent collaborative hunting method based on generative adversarial network. *International Journal of Advanced Robotic Systems 17*, 3 (2020), 1729881420925233.

[2] Cao, X., and Xu, X. Hunting algorithm for multi-auv based on dynamic prediction of target trajectory in 3d underwater environment. *IEEE Access 8* (2020), 138529–138538.

[3] Cao, X., and Zuo, F. A fuzzy-based potential field hierarchical reinforcement learning approach for target hunting by multi-auv in 3-d underwater environments. *International Journal of Control 94*, 5 (2021), 1334–1343.

[4] Dewangan, R. K., Shukla, A., and Godfrey, W. W. A solution for priority-based multi-robot path planning problem with obstacles using ant lion optimization. *Modern Physics Letters B 34*, 13 (2020), 2050137.

[5] Hamed, O., and Hamlich, M. Improvised multi-robot cooperation strategy for hunting a dynamic target. *EAI Endorsed Transactions on Internet of Things 6*, 24 (2020), e5.

[6] Hamed, O., and Hamlich, M. Improvised multi-robot cooperation strategy for hunting a dynamic target. *EAI Endorsed Transactions on Internet of Things 6*, 24 (2020), e5.

[7] Kim, W., Cho, M., and Sung, Y. Message-dropout: An efficient training method for multi-agent deep reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence* (2019), vol. 33, pp. 6079–6086.

[8] Kim, W., Park, J., and Sung, Y. Communication in multi-agent reinforcement learning: Intention sharing. In *International Conference on Learning Representations* (2020).

[9] Konda, R., La, H. M., and Zhang, J. Decentralized function approximated q-learning in multi-robot systems for predator avoidance. *IEEE Robotics and Automation Letters 5*, 4 (2020), 6342–6349.

[10] Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, P., and Mordatch, I. Multi-agent actor-critic for mixed cooperative-competitive environments. *arXiv preprint arXiv:1706.02275* (2017).

[11] Ma, J., Lu, H., Xiao, J., Zeng, Z., and Zheng, Z. Multi-robot target encirclement control with collision avoidance via deep reinforcement learning. *Journal of Intelligent & Robotic Systems 99*, 2 (2020), 371–386.

[12] Parak, R., and Matousek, R. Comparison of multiple reinforcement learning and deep reinforcement learning methods for the task aimed at achieving the goal. *MENDEL Journal 27*, 1 (2021), 1–8.

[13] Yun, W. J., Jung, S., Kim, J., and Kim, J.-H. Distributed deep reinforcement learning for autonomous aerial evtol mobility in drone taxi applications. *ICT Express 7*, 1 (2021), 1–4.

[14] Zheng, Y., Fan, W., and Han, M. Research on multi-agent collaborative hunting algorithm based on game theory and q-learning for a single escaper. *Journal of Intelligent & Fuzzy Systems*, Preprint (2021), 1–15.

[15] Zhu, K., Zong, Q., and Zhang, R. Real-time virtual simulation platform for multi-uva hunting target using deep reinforcement learning. In *2021 40th Chinese Control Conference (CCC)* (2021), IEEE, pp. 4978–4983.