

Tactical Planning Interception Enhancement using Expert Learning - Twin Delayed Deep Deterministic Policy Gradient

Nicolas Lucotte, Adolfo Perrusquía, *Member, IEEE*, Antonios Tsourdos, *Senior Member, IEEE*, and Weisi Guo, *Senior Member, IEEE*, and Hyo-Sang Shin, *Senior Member, IEEE*

Abstract—The accurate interception of adversarial unmanned aerial vehicles (UAVs) is paramount for the protection of people and national facilities. Urban cities pose several challenges for target interception algorithms due to the presence of buildings and flying constraints that limit the manoeuvrability of UAVs for target interception. Deep Reinforcement Learning (DRL) algorithms have been deployed to solve the task effectively. However, the design of its inner elements such as the reward function and action distribution limits its generalisation to different environments. To solve this issue, this paper proposes a novel twin-delayed deep deterministic policy gradient (TD3) based expert learning algorithm that combines previous expert experiences with on-line learning to regularise and improve the policy learning effectively. This is done by following an action distribution algorithm that allows a learner agent to mix its own actions with expert ones for learning improvement and fast convergence. Extensive simulation studies are carried out under diverse urban cities configurations to show the robustness and high-accuracy of the proposed approach compared with traditional DRL baseline algorithms.

Index Terms—Expert learning; Artificial Potential Field; Twin-Delayed Deep deterministic policy gradient; interception; collision avoidance; urban city

I. INTRODUCTION

Incursions of adversarial unmanned aerial vehicles (UAVs) into cities pose severe challenges both in security and safety of people and key assets due to their highly diverse and heterogeneous applications. Counter systems are usually applied to intercept the adversarial UAV before they achieve their unknown mission. However, the interception task becomes challenging in cities due to the presence of buildings, flying regulatory constraints, and the risk of damaging people. Therefore, it is crucial to design novel interception methodologies that considers the emerging challenges in urban cities, whilst maintaining safety regulation standards. Traditional interception approaches have been mainly focused on missiles [?] which follow a specific trajectory profile; such that the interception task is reduced in solving a geometric optimisation problem. These approaches are applied in open-sky environments such that there are no obstacles that prevent the UAV interceptor to move freely. However, this does not apply in contested environments such as urban cities [?].

There is a flourishing research community that uses UAV-catchers to capture adversarial UAVs and tow them into a safe

area instead of neutralize them [?], [?]. In this case, the UAV-catcher (interceptor) must adapt its trajectories and anticipate the movements of the UAV target. This allows to maximise the interception likelihood in a reduced time. In this paper, we are inspired in the design of a trajectory interception algorithms for UAV-catchers that permits the adaptation of the mission profile for a fast and accurate interception of the UAV target.

A. Related work

The research for UAV target interception has been focused on two main topics: pose estimation algorithms for trajectory prediction and guiding algorithms for path planning.

1) *State estimation algorithms*: Which are the most common approaches used to update the trajectory of an UAV for either landing, avoiding obstacles or to intercept a target goal [?], [?]. These approaches use vision techniques to estimate the location of the UAV from simple geometric relations combined with state estimation algorithms such as state observers or physics-informed algorithms [?]. The detection of obstacles has been achieved by data fusion techniques using a variety of sensors including LiDARs [?] and cameras [?]. The detection of moving obstacles or intruders have been addressed by combining sensing technologies with deep neural detectors such as YOLO [?]. This allows to accurately detect the position of the intruder within an image and estimating the states using either variants of Kalman filter [?] or trajectory prediction algorithms, e.g., reservoir computing, recurrent networks, or Gaussian processes [?], [?].

2) *Guidance algorithms*: The most simple guidance algorithm for missile interception in open-sky environments is the proportional navigation (PN) algorithm [?]. To consider congested and contested environments, switching algorithms are used to consider different dynamic environments [?]. Artificial potential fields (APFs) is a low computational cost algorithm that has been used in several multi-task missions [?], [?]. Their key idea is to design virtual repulsive and attractive forces to guide the UAV towards an objective, whilst avoiding collision with moving or static obstacles.

Reinforcement learning (RL) algorithms have been extensively used in multi-task and path planning algorithms [?], [?]. In this setting, deep deterministic policy gradient (DDPG) algorithm has been implemented for UAV interception under contested environments [?], [?]. This is achieved by designing an adequate reward function. Nevertheless, the algorithm takes

N. Lucotte, A. Perrusquía, A. Tsourdos, Weisi Guo, and H.S. Shin are with the Faculty of Engineering and Applied Sciences, Cranfield University, Bedford, UK. (e-mail: adolfo.perrusquia-guzman@cranfield.ac.uk).

a large amount of time. Twin-delayed DDPG (TD3) has been implemented to improve the learning time of DDPG by using a multi-experience pool mechanism [?]. Here, it is highlighted that learning from one single experience pool is not optimal, since the pool is filled with optimal and non-optimal experiences during the exploration. So, three experience pools are designed to store trajectories that i) fail, ii) success, and iii) a temporal one that distributed the trajectories in the success and fail pools. By using success and fail pools allow the TD3 algorithm to overcome the local minima issue of DDPG.

Previous approaches use one single reward function to execute a multi-task problem [?], [?]. Novel approaches consider to use multiple policies to solve specific tasks [?], e.g., one for obstacle avoidance and other for UAV target tracking, and then fuse them. Here two teacher networks based on the Double Dueling Deep Q-Networks (D3QN) are used. Policy distillation is used to fuse the two policies. This solution offers an improved tracking accuracy and duration, and exhibits good generalization. Also, this technique facilitates the design of the reward function.

All the aforementioned works focused mainly on ideal environments without considering disturbances such as noise in either the observations or the outputs, external forces (e.g., wind), or observability blockage due to the presence of both moving and static obstacles. This traduces into policies that are not robust in presence of these factors. Therefore, there is a flourish community interested in enhancing the robustness of RL algorithms. To do so, the concept of adversarial learning is incorporated into the design of the RL algorithm [?], [?]. The key idea, is to find policies that are robust by solving a game theory problem, i.e., either zero sum or non-zero sum games [?], [?]. In addition, the concept can be expanded into imitation learning (IL) or inverse reinforcement learning (IRL) approaches, whose main goal is to imitate and/or improve the policy of a learner by exploiting previous experiences of an expert policy [?], [?], [?].

B. Contributions

In view of the above, this paper proposes a target interception algorithm that adopts the concept of adversarial learning in the form of expert learning with a twin-delayed DDPG (TD3) algorithm. Here, the TD3 algorithm is modelled as a learner that combines its own experiences with the trajectories of an expert model to improve the robustness of the final policy and its generalisation to diverse environment configurations. In this paper we use APF and a mixed TD3-APF algorithm as expert models that solve the UAV interception task in presence of obstacles in an urban city. Extensive simulation studies are carried out to verify the robustness and effectiveness of the approach under different obstacles density, obstacles shape, level of noise, and targets with different velocity.

The contributions of the approach are fourfold:

- 1) A novel methodology named Expert Learning TD3 (EL-TD3) enabling faster and better training by combining intelligently expert and learner trajectories.
- 2) The interception radius is reduced from 10 meters into 1 meter which is less conservative than previous interception approaches [?].

- 3) The approach avoids local minima and get stuck when the UAV interceptor is close to the UAV target, or when an obstacle is between the both UAVs.
- 4) The interception time is notably reduced, and does not require to re-design the reward function or action space.

II. PROBLEM STATEMENT

The proposed methodology is depicted in Fig.???. The diagram shows an adversarial UAV target that is detected in an restricted urban area. It is assumed that this scenario happens in an intelligent city equipped with multi-frequencies LiDARs able to detect the position of the UAV target accurately [?], [?]. An UAV interceptor is deployed to intercept the UAV target. The UAV target will try to escape from the UAV interceptor by applying evasion manoeuvres. The main goal of the UAV interceptor is to intercept safely (without colliding with obstacles) the UAV target as quick as possible to prevent its advance in the restricted zone.

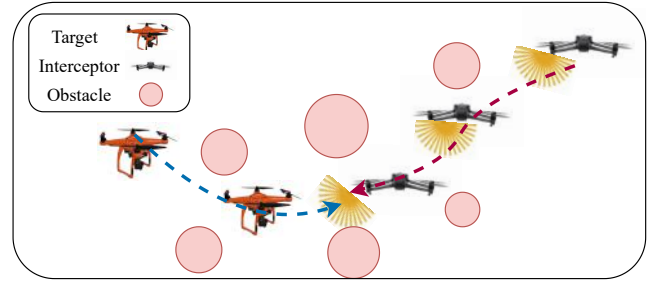


Fig. 1. High-level diagram of the tactical interception problem.

A. UAVs modelling

Consider a simple kinematic model to describe the movement of the UAVs in a 2D-plane. In this paper, we assume a fixed altitude and the pitch and roll movements are neglected. We assume that the yaw rotation angle is equivalent to the orientation angle of the velocity vector $\mathbf{v}$. The kinematic model is given by

$$\begin{bmatrix} \mathbf{p} \\ \mathbf{v} \\ \mathbf{a} \end{bmatrix}_k = \begin{bmatrix} \mathbf{p} \\ \mathbf{v} \\ \mathbf{a} \end{bmatrix}_{k-1} + \begin{bmatrix} \mathbf{v} \\ \mathbf{a} \\ \mathbf{0}_2 \end{bmatrix} \Delta t, \quad (1)$$

where $\mathbf{p}_k = [p_x, p_y]_k^\top$ denotes the position vector in the x and y -coordinates of the UAV, $\mathbf{v}_k = [v_x, v_y]_k^\top$ is the velocity vector of the UAV in each axis, $\mathbf{a}_k = [a_x, a_y]_k^\top$ is the acceleration vector in each axis, and Δt defines the sampling time between two independent measurements.

We used the acceleration components as virtual control inputs that control the movement of the UAV. The acceleration components are described as

$$\mathbf{a} = \begin{bmatrix} a_x \\ a_y \end{bmatrix} = a \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \end{bmatrix}, \quad (2)$$

where a is a constant acceleration value that will be used to vary the movement of the UAV according to the orientation θ . We consider the following constraints for both the target and

interceptor UAV: maximum velocity of the interceptor $v_{max}^I = 20$ m/s, maximum velocity of the target $v_{max}^T = 18$ m/s, the maximum acceleration is $a_{max} = 5$ m/s² for both UAVs. The sampling time is set to $\Delta t = 0.05$ seconds. So, we consider the following acceleration values $a \in [-5, 5]$ m/s² and orientations of $\theta \in [-180, 180]$ degrees ($^\circ$).

It is assumed that the UAV target has an evasion strategy based in an artificial potential field (APF) which will be discussed in the next section.

B. 2D Environment design

A square environment of $\mathcal{E}_x \times \mathcal{E}_y$ m² is designed to model an urban environment (see Fig. ??). Here, we set $\mathcal{E}_x = \mathcal{E}_y = 200$ m. Circle-shape obstacles are considered in this implementation which is common in different state-of-the-art implementations [?], [?], [?]. In each training episode, we randomly set the location of the obstacles within the square area of 50-150 meters and their radius interval within 3-10 meters. This allows the RL agents to have better generalisation across diverse environment scenarios. For training, we consider $N = 10$ obstacles and both the positions of the target and interceptor are randomly set in the center of the 2D environment.

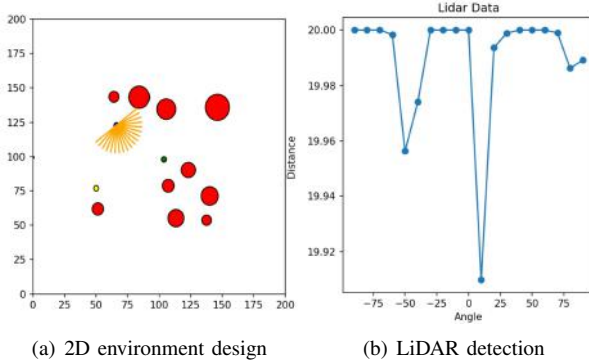


Fig. 2. Environment model and LiDAR measurements

Remark 1: The proposed approach can be extended into 3D environments [?]. However, it requires a different treatment due to the regulation constraints of UAVs in urban areas. This requires to encode safety requirements imposed by aerial authorities which is out of the scope of this work, but topic of our future research.

1) *Observations:* In this work, the UAV interceptor is endowed with 19 linear LiDARs. Each LiDAR has a range of 20 meters of detection. These LiDARs have a coverage of 180 $^\circ$ in the direction of the velocity vector. The LiDARs are uniformly distributed with an interval of 10 $^\circ$ (see Fig. ??).

The LiDARs detection process works as follows. If the UAV interceptor is close to an obstacle or the target, then the lines of the LiDAR will be shorter due to the intersection. Each line defines an observation of the environment which will be further used by the RL agent for the tactical planning interception. Measurement noise is modelled by an exponential distribution such as unique data samples are obtained in each time step. The LiDAR data are stored in the vector $\mathbf{l} = [l_1, l_2, \dots, l_{19}]^T$.

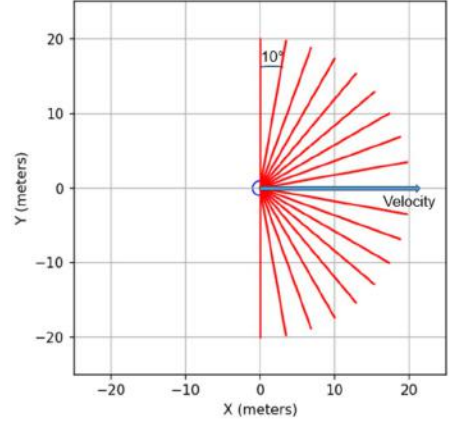


Fig. 3. LiDARs uniform distribution based on the velocity vector

III. TD3 BASELINE ALGORITHM

In this paper we focus on a modified version of the deep deterministic policy gradient (DDPG) known as twin-delayed deep deterministic policy gradient (TD3). This algorithm is commonly used in the literature [?], [?] for target interception due to the deterministic and robust policy nature. Three key improvement points are incorporated in the TD3 algorithm: (i) *Clipped double Q-learning*: TD3 is composed of two critic networks. (ii) *Delayed policy updates*: the policy is updated at a lower frequency compared with the critic network to enhance the stability of the network; and (iii) *Target policy smoothing*: noise is added in the target action to smooth the Q-function when the actions change.

Each critic network is parametrized by a set of weights denoted as $\phi_i, i = 1, 2$ that aims to minimise their respective loss function

$$\mathcal{L}(\phi_i) = \mathbb{E}_{\mathbf{h}}[(Q(\mathbf{x}_k, \mathbf{u}_k | \phi_i) - y_k^i)^2], \quad (3)$$

with $y_k^i = r_k + \gamma Q^{\mathbf{h}}(\mathbf{x}_{k+1}, \mathbf{h}(\mathbf{x}_{k+1}) | \phi_i)$, where r_k is the reward function and γ is the discount factor. The actor network is parametrized by the weights θ and learns the optimal policy $\mathbf{h}(\mathbf{x} | \theta)$ that maximises the Q-function $Q(\mathbf{x}, \mathbf{u} | \phi_i)$ as

$$\max_{\theta} \mathbb{E}_{\mathbf{x} \sim \mathcal{D}}[Q_{\phi_i}(\mathbf{x}, \mathbf{h}(\mathbf{x} | \theta))], \quad (4)$$

where $\mathcal{D}$ is a replay buffer that stores the trajectory sequences $(\mathbf{x}_k, \mathbf{u}_k, r_k, \mathbf{x}_{k+1})$ in each iteration k . Both the actor and critic networks use a mini-batch sampled from the buffer. When the replay buffer is full, then the oldest tuples are deleted and replaced by new ones.

The target network is a copy of the main network and is composed of an actor network $\mathbf{h}(\mathbf{x} | \theta')$ and a critic Q network $Q(\mathbf{x}, \mathbf{u} | \phi'_i)$. A soft-update is used to maintain a stable performance while learning,

$$\theta' = \tau \theta + (1 - \tau) \theta', \quad \phi'_i = \tau \phi_i + (1 - \tau) \phi'_i, \quad (5)$$

where $\tau \in [0, 1]$ is the soft update rate. A random noise is added to the chosen action as $\mathbf{u}' = \mathbf{u} + \sigma$, where σ is a Gaussian distributed noise, to improve the generalisation capabilities of the learned policy.

A. State and Action Space Modelling

The state space considered in this implementation is based on the data from: the position of the interceptor, the estimated position of the target, and the LiDARs' measurements. These data are used to estimate other observations such as the velocity of the UAVs. In addition, we consider as additional information the orientation of both UAVs. Fig. ?? shows the proposed geometric relation between both UAVs, where B, B' denote the angles of the velocity vectors of the UAV interceptor and UAV target, respectively.

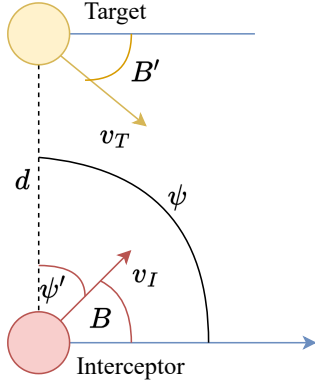


Fig. 4. State variables observations

The state space is composed of 30 measurements denoted as $\mathbf{x} = [x_1, x_2, \dots, x_{30}]^T$. Each measurement is described as follows,

- The coordinates of the UAV interceptor defined by $x_1 = \frac{I_x}{\mathcal{E}_x}$ and $x_2 = \frac{I_y}{\mathcal{E}_y}$, where $x_1, x_2 \in [0, 1]$.
- The velocity vector orientation of the interceptor on the absolute frame $x_3 = \frac{B}{180} \in [-1, 1]$.
- The ratio velocity $x_4 = \frac{v}{v_{\max I}} \in [0, 1]$.
- The coordinates of the UAV target $x_5 = \frac{T_x}{\mathcal{E}_x}$ and $x_6 = \frac{T_y}{\mathcal{E}_y}$ with $x_5, x_6 \in [0, 1]$.
- The velocity vector orientation of the target on absolute frame $x_7 = \frac{B'}{180} \in [-1, 1]$.
- The ratio velocity of the target $x_8 = \frac{v'}{v_{\max T}} \in [0, 1]$.
- The 19 measurements from the LiDARs $x_j^n = \frac{I_n}{l_{\max}} \in [0, 1]$ where $j = 9, 10, \dots, 27$ and $n = 1, 2, \dots, 19$.
- The direction of the target in the absolute frame $x_{28} = \frac{\psi}{180} \in [-1, 1]$.
- The distance between the target and the interceptor $x_{29} = \frac{d}{\sqrt{\mathcal{E}_x^2 + \mathcal{E}_y^2}} \in [0, 1]$.
- The direction of the target in the interceptor frame $x_{30} = \frac{\psi'}{180} \in [-1, 1]$.

The acceleration values $a \in [-5, 5]$ m/s² and the orientation of the acceleration $\theta \in [-180^\circ, 180^\circ]$ are used as main actions for the interceptor. The action space is as well normalised to maintain stability of the overall training. Hence, the action space is described by $a_{RL} = [-1, 1]$ and $\theta_{RL} \in [-1, 1]$, and hence $\mathbf{u} = [a_{RL}, \theta_{RL}]^T$.

B. Reward function design

We design a reward function as a linear combination of seven different sub-reward terms with unitary weights. Each sub-reward has an associated positive scalar denoted by $k_t, k_d, k_o, k_l, k_v, k_f, k_s > 0$. The sub-reward terms are summarized as follows

- Terminal sub-reward r_e : this sub-reward can have one of the following three values per episode (1) $r_e = k_t$ if the target is intercepted, (2) $r_e = -k_t$ if the interceptor collides with an obstacle, and (3) $r_e = 0$ if the maximum number of steps in an episode is reached.
- Distance sub-reward $r_d = k_d(d_{k-1} - d_k)$: this sub-reward encourages the UAV interceptor to approach to the UAV target in each step.
- Orientation sub-reward $r_o = k_o \min\left(\frac{90^\circ - \psi'}{180^\circ}, 0\right)$: this sub-reward is used to motivate the UAV interceptor to align and move in the same direction as the UAV target.
- LiDAR sub-reward $r_l = k_l \sum_{n=1}^N \min\left(\frac{1}{l_n} - \frac{1}{l_{\max}}, 1.5\right)$: this sub-reward is used to penalise the UAV interceptor is close to an obstacle.
- Speed sub-reward $r_v = k_v(v_{\max} - v)$: this sub-reward is used to encourage the UAV interceptor to approximate the UAV target as fast as possible.
- Safety sub-reward $r_f = \min\left(\frac{k_f(l_{10} - d_s)}{d_s}, 0\right)$: the purpose of this sub-reward is to prevent the UAV interceptor to navigate towards an obstacle in a straight line. For this particular sub-reward we used the LiDAR measurement l_{10} that is aligned to the velocity direction. Low measurement values imply high collision risk. We define a safety distance d_s that permits to punish the UAV interceptor when the likelihood of collision is high.
- Time sub-reward $r_s = -k_s$: this is a simple negative value that is given in each step to encourage the RL agent to converge faster to an optimal policy.

The final reward function is then given by

$$r_k = r_e + r_d + r_o + r_l + r_v + r_f + r_s.$$

IV. TD3-APF ALGORITHM

RL algorithms faces two particular challenges in planning interception tasks:

- (1) The UAV interceptor gets stuck when it is close to the UAV target. In this case, the reward values are not meaningful such that the UAV interceptor remains in the same location without intercepting the UAV target.
- (2) The UAV interceptor gets stuck when both UAVs are separated by two or more aligned obstacles. In this case, the distance between both UAVs is small but the interceptor is not capable to learn how to get around the obstacles to avoid the stuck point as in (1).

These issues are usually alleviated by action shaping mechanisms in the form of hard-coding rules that moves the interceptor outside the stuck region to surround the obstacles for an effective interception. Other approach is to integrate other algorithms to support the RL agent in specific scenarios, e.g., in (1) or (2) cases. Artificial Potential Field (APF) is

widely used in planning problems due to its simplicity and effectiveness. The key idea is to use attractive and repulsive forces that allows the UAV navigate in unknown environments with obstacle avoidance capabilities.

Therefore, to improve the RL agent performances in the aforementioned cases, we design an APF algorithm that allows the UAV interceptor to reach the target without getting stuck. In addition, we consider that the UAV target uses an independent APF algorithm for evasion purposes.

A. UAV Target APF

We consider the following attractive force for the UAV target

$$F_a^T = k_a^T \cdot \overline{TP}, \quad (6)$$

where F_a^T is the attractive force applied on the UAV target, k_a^T is the respective attractive coefficient, and $\overline{TP}$ is the displacement from the UAV target to the platform. The repulsive force is defined as

$$F_r^T = \left(\sum_{n=1}^N k_{r_n}^T \left(\frac{1}{d_n} - \frac{1}{r_t^T} \right) \frac{\overline{TO}_n}{d_n^3} \right) + k_{r_c}^T \left(\frac{1}{d} - \frac{1}{r_t^T} \right) \frac{\overline{TI}}{d^3}, \quad (7)$$

where N is the number of obstacles, $k_{r_n}^T$ is the repulsive coefficient of the obstacle n , $k_{r_c}^T$ is the repulsive coefficient of the interceptor, and d_n is the distance between the target and the closest point of the obstacle n , d is the distance between the interceptor and the target, r_t^T is the repulsive threshold of the target, $\overline{TO}_n$ is the displacement from the target to the closest point of the obstacle n , and $\overline{TI}$ is the displacement vector from the target to the interceptor. The final force is $F_T = F_a^T + F_r^T$ and $v_T = F_T$, where v_T is the velocity of the target.

B. UAV Interceptor APF

The UAV interceptor is designed to reach the target with obstacle avoidance capabilities. It is assumed that the interceptor has noisy and limited observations given by the data provided by the 19 linear LiDARs, the position of the target and its own position.

The attractive force of the UAV interceptor is,

$$F_a^I = k_a^I \cdot \overline{IT}, \quad (8)$$

where F_a^I is the attractive force of the interceptor, k_a^I is the respective attractive coefficient, $\overline{IT}$ is the displacement vector from the UAV interceptor to the UAV target.

The location of the obstacles are not available for the UAV interceptor, so we design a repulsive force for each intersection point of the LiDARs measurements. Here, an intersection is considered when the data obtained by the LiDARs is lower than 19 meters. The repulsive force is designed as

$$F_r^I = \sum_{m=1}^M k_{r_m}^I \left(\frac{1}{l_m} - \frac{1}{r_t^I} \right) \cdot \frac{\overline{IL}_m}{l_m^2}, \quad (9)$$

where M is the number of LiDARs with intersection points, $k_{r_m}^I$ is the repulsive coefficient of the intersection point m , l_m is the data of the LiDAR m , r_t^I is the repulsive threshold of the

UAV interceptor, and $\overline{IL}_m$ is the displacement vector from the UAV interceptor to the intersection point m . The total force of the UAV interceptor is $F_I = F_a^I + F_r^I$ and $v_I = F_I$, where v_I is the velocity of the UAV interceptor. This APF mechanism solves the challenge (1).

To deal with the challenge (2), we design a simple condition in the UAV target APF to improve the interception capabilities. The proposed condition consists in if the UAV target is within a radius of 19 meters from the UAV interceptor and within the detection range of the LiDARs, then the UAV interceptors detect if any obstacle is located between the UAV interceptor and the UAV target. If there are no obstacles, then a straight velocity vector will be applied in the direction of the UAV target. We can summarize this mechanism as follows: if the following conditions are satisfied $\|\overline{IT}\| < 19$ and $\|B\| < 90^\circ$, then the following indices are computed

$$I_{up} = \left\lceil \frac{B}{10^\circ} \right\rceil + 10^\circ,$$

where $\lceil \cdot \rceil$ and $\lfloor \cdot \rfloor$ denote the ceiling and floor functions, respectively. Subsequently, if the conditions $l_{I_{up}} < \|\overline{IT}\|$ and $l_{I_{down}} < \|\overline{IT}\|$ are satisfied, we set the velocity of the UAV interceptor as

$$v_I = F_I = F_a^I. \quad (10)$$

The forces of both the target and interceptor UAVs are restricted based on the maximum velocity and acceleration values. To do so, we consider the following inequality $\|v_k - v_{k-1}\| \leq a_{max} \Delta t$ as adaptation mechanism. When this inequality is not satisfied, then both the velocity and acceleration vectors are slightly changed by

$$\begin{aligned} \begin{bmatrix} b_x \\ b_y \end{bmatrix} &= \frac{v_k - v_{k-1}}{\Delta t}, \quad \alpha = \arctan\left(\frac{b_y}{b_x}\right) \\ \mathbf{a} &= a_{max} \begin{bmatrix} \cos(\alpha) \\ \sin(\alpha) \end{bmatrix}, \quad v_k = v_{k-1} + \mathbf{a} \Delta t. \end{aligned} \quad (11)$$

The new velocity and acceleration vectors satisfy the adaptation mechanism and hence, both the maximum velocity and acceleration constraints. On the other hand, if $\|v\| \geq v_{max}$ then we calculate a new angle as follows

$$B = \arctan\left(\frac{v_y}{v_x}\right), \quad \begin{bmatrix} v_x \\ v_y \end{bmatrix} = v_{max} \begin{bmatrix} \cos(B) \\ \sin(B) \end{bmatrix}. \quad (12)$$

The same process applies for both the UAV interceptor and UAV target. Then, the next position of the UAV (either target or interceptor) is computed by

$$p_k = p_{k-1} + v_k \Delta t. \quad (13)$$

The TD3-APF algorithm works as follows: the TD3 algorithm is used to get close to the target in a threshold distance and in the detection range of the LiDARs. Then the algorithm switch to the APF for UAV target interception. In this paper, we used a threshold of 10 meters.

Remark 2: The proper design of the reward function and both repulsive and attractive forces play a critical role in the overall performance of both RL algorithms and APF approaches. Here, the design process is time-consuming and require prior expert knowledge to ensure that the desired performance is effectively obtained. This paper aims to solve this

issue by using an expert learning approach that is discussed in the next section.

V. EXPERT LEARNING TD3 (EL-TD3)

An expert learning algorithm is proposed to deal with the challenge of Remark ???. This is done by designing a complete data-driven RL algorithm that adopts the behaviour of a previous expert policy as guidance to improve its current control policy and achieve competitive performances.

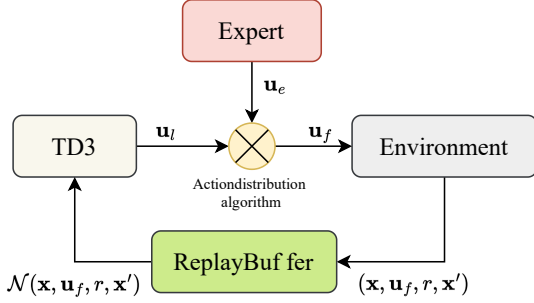


Fig. 5. High-level diagram of the Expert Learning TD3 (EL-TD3)

The proposed approach is shown in Fig. ???. The diagram consists in five main elements: a standard TD3 algorithm that acts as a learner, an expert model of previous trained RL algorithms or APF, a replay buffer of combined learner and expert trajectories, the environment, and an action distribution algorithm that selects the action that will be adopted by the UAV interceptor. At each time step, both the TD3 learner and the expert model propose an action u_l and u_e , respectively. Then, the action distribution algorithm is applied to select one of those actions as a final action u_f . This action will be sent to the environment and the resulting reward and state will be stored in the replay buffer as a tuple (x, u_f, r, x') . This creates a distribution of valid actions that has to be defined in function of the situation. Here, the critics and actor networks can be initialised with random parameters, or previous pre-trained weights. This expert learning technique has shown better convergence results compared with the TD3 baseline algorithm.

A. Action distribution algorithm

The proposed action distribution algorithm is given in Algorithm ??. The basic idea is to train the TD3 learner with expert actions and its own actions in specific time steps. This allows the learner to collect its own experiences about how to solve the interception task and, at specific time steps, use the expert data to regularise its learning towards a desired performance.

In this paper, the action switching between the learner and the expert are given in only the first 2,000 time steps. Here, in the first 500 steps of an episode e we use only the actions of the expert to guide the learner towards a desired performance, then after 500 time steps the action distribution algorithm starts to switch between the expert and the learner actions. In the next 500 time steps, we want that the replay buffer contains

a diverse pool of valid actions gathered from the expert and the learner, then we use a simple condition $e\text{mod}(2) = 0$ to ensure that the replay buffer is balanced between the expert and learner actions. This allows the agent to shape its policy based in its own experiences and expert data. The next 1,000 time steps of an episode we apply a similar mechanism, but in this case we want that the learner use more its experiences rather than the expert data. This emulates how the learner starts to gain confidence and requires less expert guidance. To do so, we use the conditions $e\text{mod}(4) = 0$ and $e\text{mod}(8) = 0$ to start decreasing the appearance of the expert actions in the replay buffer. After 2,000 time steps of an episode we use only the actions of the learner, which means the learner is completely confident of its knowledge and does not require more expert guidance. This process is repeated for each episode.

Algorithm 1 Action distribution algorithm

```

1: for  $t = 1$  to  $T$  do
2:   for  $e = 1$  to  $\max_{\text{episodes}}$  do
3:     Propose TD3 learner action with exploration noise
        $u_l \sim h(x_t|\theta) + \epsilon$ .
4:     Propose expert action  $u_e$ 
5:     if  $t < 500$  then
6:        $u_f \leftarrow u_e$ 
7:     else if  $t \geq 500$  and  $t < 1,000$  then
8:       if  $e\text{mod}(2) = 0$  then
9:          $u_f \leftarrow u_e$ .
10:      else
11:         $u_f \leftarrow u_l$ 
12:      end if
13:    else if  $t \geq 1,000$  and  $t < 1,500$  then
14:      if  $e\text{mod}(4) = 0$  then
15:         $u_f \leftarrow u_e$ .
16:      else
17:         $u_f \leftarrow u_l$ 
18:      end if
19:    else if  $t \geq 1,500$  and  $t < 2,000$  then
20:      if  $e\text{mod}(8) = 0$  then
21:         $u_f \leftarrow u_e$ .
22:      else
23:         $u_f \leftarrow u_l$ 
24:      end if
25:    else
26:       $u_f \leftarrow u_l$ 
27:    end if
28:  end for
29: end for

```

VI. SIMULATION STUDIES

The proposed approach is tested and compared against different algorithms including: APF, DDPG, TD3, TD3-APF, EL-TD3 using APF as expert, and EL-TD3 using TD3-APF as expert. In each episode, a random environment is generated to obtain a policy that has good generalisation. Each random environment differs on the number of obstacles, radius, and the locations of the target and interceptor. An episode consists of 500 time steps. We use a total of 1,000 episodes to train the

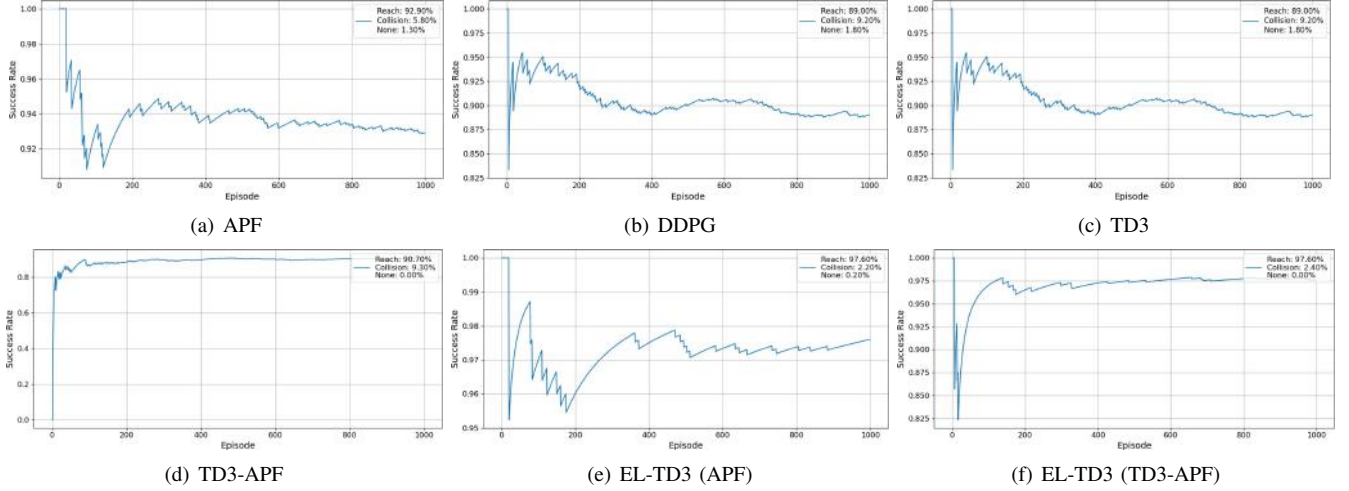


Fig. 6. Success interception rate across all agents

algorithms. This means that we trained the algorithms under 1,000 random environments. The UAV target uses an APF algorithm for evasion of the UAV interceptor. The algorithms are compared based on: the average time of interception and the success rate.

The hyperparameters of the APF are set to: $k_a^T = 0.5$, $k_{r_n}^T = 1000$, $k_{r_c}^T = 9000$, $r_t^T = 50$, $k_a^I = 6$, $k_{r_n}^I = 1800$ and $r_t^I = 20$. For the reward function we use the following values that achieve the best performance for all models: $k_t = 100$, $k_d = k_o = 0.5$, $k_l = -0.1$, $k_v = -0.8$, $k_f = 0.1$, $d_s = 15$, and $k_k = 0.2$. Both the DDPG and TD3 algorithms use the following hyperparameter values: $\tau = 0.005$, $\gamma = 0.99$, $\sigma = 0.05$, learning rate $lr = 0.0003$, batch size of 256, size of buffer of 1,000,000. Additionally, for the TD3 we set the noise clip as 0.5 and the policy noise to 0.2. Random weights are used for all networks of the RL algorithms for fair comparisons.

A. Model comparisons

Fig. ?? shows the success interception rate results of all the agents in the first 1,000 episodes. For each model we summarize three metrics: (i) success interception rate, (ii) collision rate, and (iii) the non-interception rate (referred as none). The non-interception rate happens when the UAV interceptor is not capable to intercept the UAV target within the time-steps in an episode.

The results highlight that the APF algorithm is capable to solve the interception task more efficiently compared with the DDPG and TD3 baseline algorithms. The collision rate of the APF and not success rate are as well better compared with the RL baseline algorithms. This is attributed to the cases (1) and (2) discussed previously. To deal with this issue the TD3-APF is designed which notably improves the performance of the TD3 baseline algorithm, but its performance is not better than the APF algorithm. Notice that the TD3-APF always intercept the target (or collide in the worst case). This slight performance improvement suggests that the TD3-APF algorithm cannot effectively combine their action distributions. The EL-TD3 algorithms using either the APF or TD3-APF

algorithms as expert model outperform all the previous models in both success, collision, and not-success rates.

Recall, that as the APF algorithm is less optimised, it is more difficult to reach the target when distributing actions. Furthermore, as it is not a RL algorithm, it seems that it is more difficult for the agent to learn from it. However, even by using an expert that is less optimised, the EL-TD3 methodology still results into a faster training and better performances than a basic TD3 algorithm. Notice that once a solution is obtained, it is still possible to refine its parameters by training the EL-TD3 algorithm a second time, and modifying the hyperparameters and/or the reward function.

TABLE I
SUCCESS RATE OF THE AGENTS. BEST RESULTS ARE IN BOLD

Model	Reach the target	Collision	Not success	Average time success case
APF	92.90%	5.80%	1.30%	6.57 s
DDPG	89.00%	9.20%	1.80%	7.31 s
TD3	86.10%	9.50%	4.40%	8.72 s
TD3-APF	90.70%	9.30%	0.00%	6.32 s
EL-TD3 (APF)	97.60%	2.20%	0.20%	6.27 s
EL-TD3 (TD3-APF)	98.60%	1.40%	0.00%	4.06 s

To validate the previous claim, we change the scalar k_k from 0.2 to 0.7, and re-train the EL-TD3 (TD3-APF) algorithm. The success rate results are shown in Fig. ?. Here, the performance of the EL-TD3 is further improved from 97.60% to 98.60% with a low collision rate of 1.40%, and similarly the algorithm always intercept the target (or collide in the worst case). Fig. ? shows an example of the trajectory interception problem using the EL-TD3 (TD3-APF) algorithm. The figure shows smooth trajectories from the UAV interceptor, avoiding the obstacles and reach the UAV target rapidly.

Table ? summarizes the average results obtained for each algorithm across the 1,000 random environments and their respective average interception times. Here, the EL-TD3 (TD3-APF) algorithm outperform all models and intercept the target in less time (4.06 seconds), which is a key factor in the proposed interception task that requires to reach the target as fast as possible to prevent the target UAV achieve its mission.

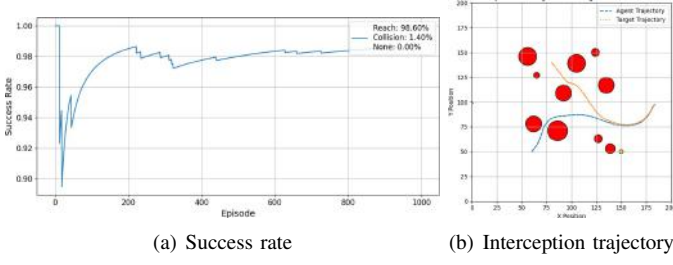


Fig. 7. EL-TD3 (TD3-APF) under reward refinement

B. Robustness and generalisation

We consider different scenarios to test the robustness of the proposed EL-TD3 algorithm. The experiments consist in: 1) changing the obstacles density, 2) obstacles with different shapes, 3) moving obstacles, 4) UAV target with different velocity, and 5) noise measurements.

1) *Obstacles density*: The EL-TD3 will be tested in environments having an higher density of obstacles. So we train the proposed EL-TD3 algorithm under 15 and 20 obstacles similarly to the 10 obstacles case.

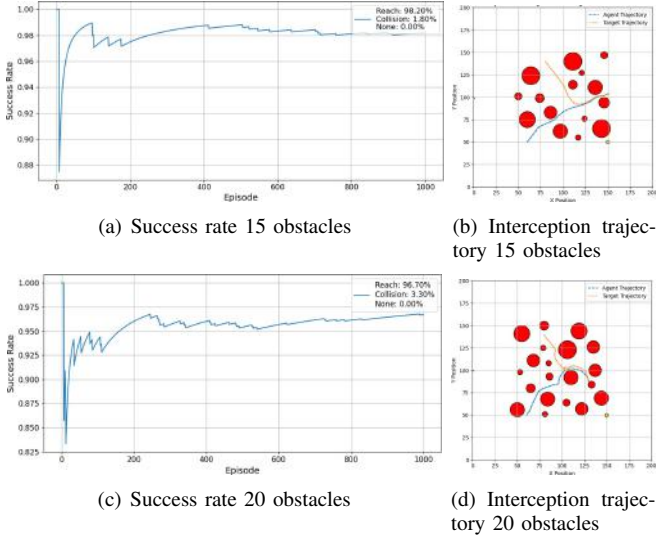


Fig. 8. EL-TD3 performance under different obstacles density

Fig. ?? shows the results of the EL-TD3 algorithm under 15 and 20 obstacles. For both cases the UAV interceptor is capable to reach the target with a relative high success rate. In the 15 obstacles case, the average interception time is of 3.83 seconds, whilst for the 20 obstacles case the interceptor needs in average 3.84 seconds to intercept the UAV target with a minor increase in the collision rate. Here, we observe that the interception time is less than the 10 obstacles case. The reasons behind these performances are twofold: i) the UAV target has more difficulties to move in a dense environment. The APF algorithm used by the UAV target suffers of local minima issues which prevents the UAV to escape from the interceptor. ii) the strategy learnt by the UAV interceptor makes that the UAV target to move close to an obstacle which reduces its manoeuvrability to escape. This technique is certainly effective, but is also risky since it increases the

likelihood of collision which can be observed in the decrease of success rate and the increase of collision rate.

2) *Polygonal obstacles*: We compare the generalisation of the approach by changing the shape of the obstacles to ensure the approach is not biased to just circle-shape obstacles. We change the circle-shape obstacles into convex polygons and test the final policy of the EL-TD3 algorithm. The results are shown in Fig. ??.

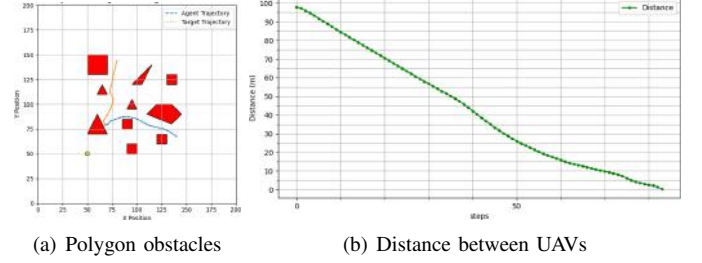


Fig. 9. EL-TD3 performance under polygon obstacles

The results show that the EL-TD3 algorithm has good generalisation capabilities and it can effectively solve the target interception problem even in presence of obstacles of different shape. Here, the distance between the UAV interceptor and the UAV target is reduced in each time step until the target is intercepted, whilst exhibiting a smooth and fast trajectory.

3) *Moving obstacles*: We add more complexity to the interception task by adding moving obstacles or UAV operators within the environment. We consider 3 UAV operators and assumed that they are flying in an operating mode controlled by an APF algorithm without repulsive force on the UAV interceptor. This means that the UAV operators only apply the repulsive force on the fixed obstacles, which means that collisions can occur between the UAV interceptor and them. The maximum speed of each UAV operator is fixed at 50% of the maximum UAV interceptor's speed. Fig. ?? shows the results under this new environment conditions.

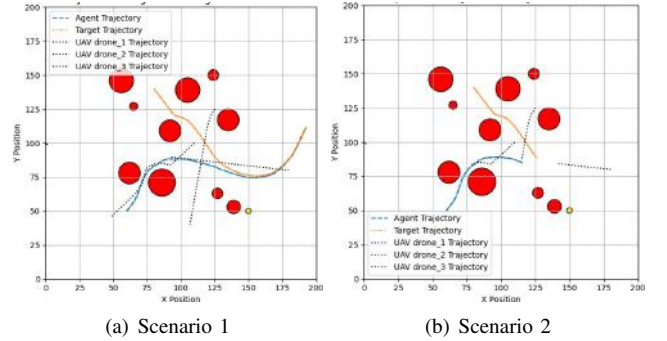


Fig. 10. EL-TD3 performance under moving obstacles

Fig. ?? shows a success scenario, whilst Fig. ?? shows an unsuccessful interception scenario. To obtain these results, the 3 operators were not positioned randomly, instead they were located in such a way that their trajectories would cut the UAV interceptor trajectory. It can be observed that the UAV interceptor try to avoid the collision with the UAV operators but in some cases this does not happen. Here, the EL-TD3

algorithm is not capable to dodge all scenarios with moving obstacles, such that its generalization capabilities are limited to the fixed obstacles case. This can be solved by incorporating UAV operators within the environment in the training stage, such that the final policy is capable to deal with both static and moving obstacles. However, the results demonstrate that the EL-TD3 algorithm shows robustness in some extent.

4) *UAV target with different velocity*: The previous experiments consider an UAV target using a 90% of the maximum velocity of the UAV interceptor. We consider additional scenarios when the UAV target flies with a lower velocity of 75% and 50% of the maximum velocity of the UAV interceptor. This will demonstrate that the trained algorithm can solve the interception task under targets with lower velocity to the one used for training. The results for both cases are shown in Fig. ?? and Fig. ??, respectively.

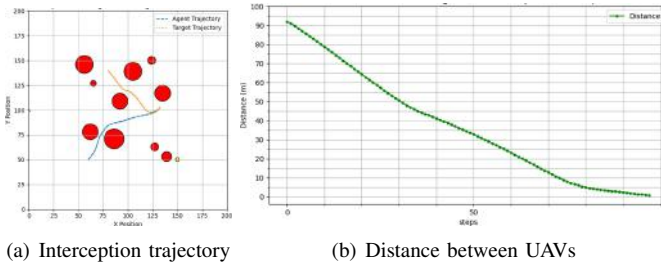


Fig. 11. EL-TD3 performance under UAV target with 75% of target velocity

As it is expected, the interception time is shorter compared with the results of Fig. ?. Here, the UAV interceptor adapts the estimated intersection point based on the velocity of the target. Notice that the interceptor does not take the same direction and intercept the UAV target in the same point. In both scenarios the distance between the UAVs decrease in each time step until the interception occurs. This allows to conclude that the EL-TD3 algorithm is capable to handle UAV targets flying at lower velocities.

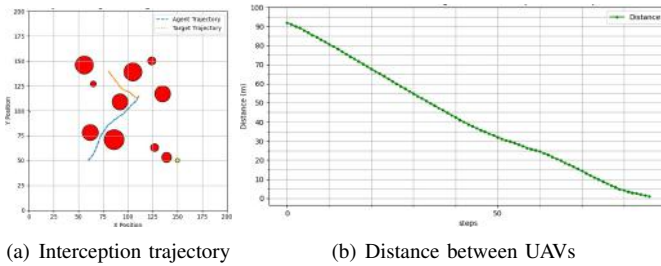


Fig. 12. EL-TD3 performance under UAV target with 50% of target velocity

5) *Noisy measurements*: The previous experiments consider noise in the measurements to model real-world measurements. We further demonstrate the robustness of the approach by increasing the level of noise in the measurements. We consider three cases: 1) measurements without noise, 2) measurements with the double noise, and 3) measurements with the tenfold noise. Fig. ?? shows the results without noise. This is the ideal case which does not affect the EL-TD3 algorithm to effectively solve the interception task with maximum velocity.

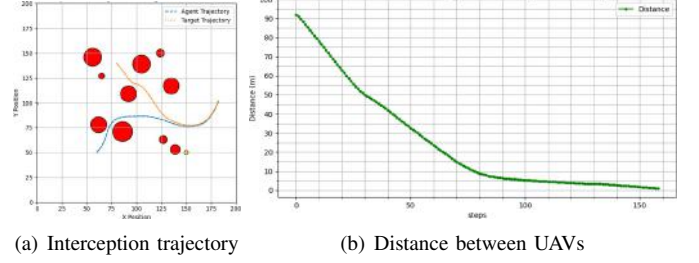


Fig. 13. EL-TD3 performance under measurements without noise

Fig. ?? shows the results with double noise. Minor changes can be observed in terms of interception time which means that the UAV target change velocity in some time steps which slightly increases the interception time.

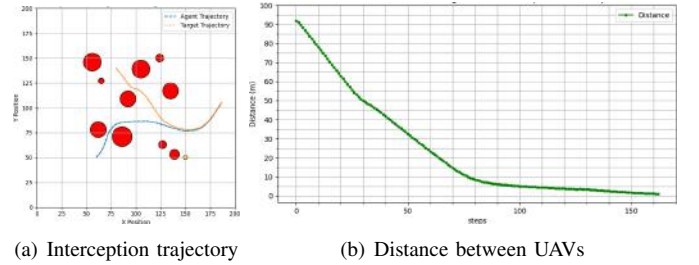


Fig. 14. EL-TD3 performance under measurements with double noise

Fig. ?? shows the results with tenfold noise. In this case, the policy generated by the EL-TD3 has some loss of velocity, and hence the interception time is increased. This can be explained by the fact that if the return data from the LIDAR is shorter than the reality, the agent can consider a risk of collision and slow down. This can lead to loss of velocity and therefore, an increase in the interception time. Nevertheless, the EL-TD3 is still successful to intercept the target smoothly with an acceptable interception time.

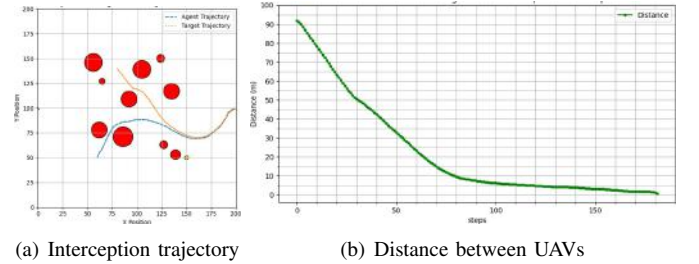


Fig. 15. EL-TD3 performance under measurements with tenfold noise

VII. CONCLUSIONS

This paper reports an interception approach of adversarial drones in urban zones using deep reinforcement learning. The evasion manoeuvres applied by the adversarial drone are given by an artificial potential field (APF), whilst the interceptor is trained under a novel expert learning - twin delayed DDPG (EL-TD3) architecture under an urban zone with obstacles of different densities. The key idea is to use a previous-trained

expert model that regulates the learning manifold of a TD3-learner algorithm. The combined contribution of the expert and the learner helps the UAV interceptor to learn effectively how to intercept the UAV target by adjusting better the distribution of actions. A sparse reward function is designed to ensure target interception in an optimised interception time with collision avoidance capabilities. We used as expert models an APF algorithm and a combined TD3-APF algorithm, where the latter one exhibits better results in the proposed EL-TD3. Extensive simulation studies are conducted to verify the benefits and challenges of the proposed model. Future work, will consider the use of adversarial learning [?] and imitation/inverse learning [?] to model the UAV target as an intelligent agent that modifies the nature of the problem into a game problem. Also the extension into multi-agent interception problem is topic of our future research.



Weisi Guo received the M.Eng. degree in engineering, and the M.A. and Ph.D. degrees in computer science from the University of Cambridge, Cambridge, U.K., in 2005, 2011, and 2011, respectively. He is a Chair Professor of Human Machine Intelligence with Cranfield University, Cranfield, U.K. He has published over 180 papers and is a PI on a number of molecular communication research grants.

Prof. Guo's research has won him several international awards. He was a Turing Fellow at the Alan Turing Institute.



Nicolas Lucotte obtains as part of a double diploma his master's degree in Autonomous Vehicle Dynamics and Control in the School of Aerospace, Transport, and Manufacturing at Cranfield University United Kingdom. He is graduated as general engineer from ECAM Strasbourg-Europe. His areas of interest include reinforcement learning, adversarial learning, and path planning optimisation.



Adolfo Perrusquía received the M.Sc. and Ph.D. degrees, both in Automatic Control from the Automatic Control Department at the CINVESTAV-IPN in 2016 and 2020, respectively. He is currently a lecturer at Cranfield University, and a RAEng alumni. He is a member of the IEEE Computational Intelligence Society and Associate Editor of the IEEE TNNLS. His main research of interest focuses on reinforcement learning, inverse reinforcement learning, data-driven control nonlinear control, machine learning and system identification.



Hyo-Sang Shin received the B.Sc. degree in aerospace engineering from Pusan National University, Busan, South Korea, in 2004, the M.Sc. degree in aerospace engineering with focus on flight dynamics, guidance, and control from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 2006, and the Ph.D. degree in cooperative missile guidance from Cranfield University, Cranfield, U.K., in 2010.

He is currently a Professor with Korea Advanced Institute of Science and Technology and Cranfield University. His research interests include multiple target tracking, probabilistic target detection, and distributed control of multiple agent systems. Dr. Shin is an Associate editor of IEEE Transactions on Aerospace and Electronic Systems and Journal of Intelligent and Robotic Systems.



Antonios Tsourdos obtained a MEng in electronic, control and systems engineering from the University of Sheffield (1995), an MSc in systems engineering from Cardiff University (1996), and a PhD in nonlinear robust missile autopilot design and analysis from Cranfield University (1999). He is a Professor of Control Engineering with Cranfield University, and was appointed Head of the Centre for Cyber-Physical Systems in 2013.

Prof. Tsourdos was a member of the Team Stellar, the winning team for the UK MoD Grand Challenge (2008) and the IET Innovation Award (Category Team, 2009). He was an Associate editor of IEEE Transactions on Aerospace and Electronic Systems, Proceedings of the IMechE Part G: Journal of Aerospace Engineering, Aerospace Science & Technology, and International Journal of Systems Science.



DSPACE

<https://dspace.org/>

Tactical planning interception enhancement using expert learning - twin delayed deep deterministic policy gradient

Lucotte, Nicolas; Perrusquía, Adolfo; Tsourdos, Antonios; Guo, Weisi; Shin, Hyo-Sang

2026-01

Institute of Electrical and Electronics Engineers (IEEE)

<https://doi.org/10.1109/tiv.2025.3620013>

Lucotte N, Perrusquía A, Tsourdos A, et al., (2026) Tactical planning interception enhancement using expert learning - twin delayed deep deterministic policy gradient. IEEE Transactions on Intelligent Vehicles, Volume 11, Issue 1, January 2026, pp. 174-184