2-22-2022

Twin Delayed Deep Deterministic Policy Gradient-Based Target Tracking for Unmanned Aerial Vehicle with Achievement Rewarding and Multistage Training

Najmaddin Abo Mosali
Tun Hussein Onn University of Malaysia

Syariful Syafiq Shamsudin
Tun Hussein Onn University of Malaysia

Omar Alfandi
Zayed University

Rosli Omar
Tun Hussein Onn University of Malaysia

Najib Al-fadhali
Tun Hussein Onn University of Malaysia

IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.Doi Number

Twin Delayed Deep Deterministic Policy Gradient-Based Target Tracking for Unmanned Aerial Vehicle with Achievement Rewarding and Multistage Training

Najmaddin Abo Mosali[1], Syariful Syafiq Shamsudin[2], Omar Alfandi[3] ,Rosli Omar[4] and Najib Al-fadhali[5]

[1, 2]Research Center for Unmanned Vehicles,Faculty of Mechanical and Manufacturing Engineering,Universiti Tun Hussein Onn Malaysia, 86400 Parit Raja, Batu Pahat, Johor, Malaysia
[3]Technological Innovation at Zayed University, United Arab Emirates
[4,5]Faculty of Electrical and Electronic Engineering, Universiti Tun Hussein Onn Malaysia UTHM, 86400 Parit Raja, Johor, Malaysia

Corresponding author: Najmaddin Abo Mosali (najmabumosally@gmail.com), Syariful Syafiq Shamsudin (syafiq@uthm.edu.my)

ABSTRACT Target tracking using an unmanned aerial vehicle (UAV) is a challenging robotic problem. It requires handling a high level of nonlinearity and dynamics. Model-free control effectively handles the uncertain nature of the problem, and reinforcement learning (RL)-based approaches are a good candidate for solving this problem. In this article, the Twin Delayed Deep Deterministic Policy Gradient Algorithm (TD3), as recent and composite architecture of RL, was explored as a tracking agent for the UAV-based target tracking problem. Several improvements on the original TD3 were also performed. First, the proportional-differential controller was used to boost the exploration of the TD3 in training. Second, a novel reward formulation for the UAV-based target tracking enabled a careful combination of the various dynamic variables in the reward functions. This was accomplished by incorporating two exponential functions to limit the effect of velocity and acceleration to prevent the deformation in the policy function approximation. In addition, the concept of multistage training based on the dynamic variables was proposed as an opposing concept to one-stage combinatory training. Third, an enhancement of the rewarding function by including piecewise decomposition was used to enable more stable learning behaviour of the policy and move out from the linear reward to the achievement formula. The training was conducted based on fixed target tracking followed by moving target tracking. The flight testing was conducted based on three types of target trajectories: fixed, square, and blinking. The multistage training achieved the best performance with both exponential and achievement rewarding for the fixed trained agent with the fixed and square moving target and for the combined agent with both exponential and achievement rewarding for a fixed trained agent in the case of a blinking target. With respect to the traditional proportional differential controller, the maximum error reduction rate is 86%. The developed achievement rewarding and the multistage training opens the door to various applications of RL in target tracking.

INDEX TERMS Navigation, Reinforcement Learning, Target Tracking, Twin Delayed Deep Deterministic Policy Gradient, Unmanned Aerial Vehicles.

I. INTRODUCTION

Unmanned aerial vehicle (UAV) applications are increasing day by day, and aerial vehicles are being used as part of many recent technological applications. Some examples are in shipping [1], surveillance [2], [3], [4], battlefield [5], rescuing applications [6], [7], and inspection [8], [9]. Aerial vehicles are now divided into three categories: teleoperated [10], [11], semi-autonomous [12], [13], and full autonomous [14]. Enabling aerial vehicle applications requires essential autonomous features with regard to autonomy within the system.

Vehicles that can be autonomous must be able to decide on and react to events without direct intervention by humans. Some fundamental aspects are common to all autonomous vehicles. These aspects include sensing and perceiving the environment, analysing the gained information,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3154388, IEEE Access

communicating, planning and making decisions, and acting using control algorithms and actuators. For example, in the autonomous tracking feature of a UAV to a target, a camera is used for sensing the environment. Next, the gained information is analysed to detect the target. The detection is sent to the decision-making algorithm that enables the mobility of the UAV autonomously. Once this feature is shown to be working in a stable and robust way, it is deployed to UAVs as an autonomous feature that assists in operating UAVs and human–vehicle interaction.

Operating unmanned flying vehicles is useful; however, it can be challenging when the vehicle interacts with the environment. This interaction could be, for instance, in the form of landing on the ground or landing pads, docking into a station, approaching terrain for inspection, or approaching another aircraft for refueling purposes. Such tasks can often be solved when the vehicle is remotely piloted, especially when the pilot has a first-person view of the environment. However, human control may not always be possible. For instance, the unavailability of a suitable data link or the precision and/or speed required for the maneuver may be outside human capabilities. Thus, it is important to find effective and flexible strategies to enable vehicles to perform such tasks autonomously.

Well-developed features of autonomous UAV control include stability enhancement and waypoint flight, autonomous tracking, and autonomous landing. However, new developments in the design of UAVs, as well as the emergence of new application areas, demand robust and adaptive control techniques for different flight conditions, such as aggressive maneuvering flight [15], robust disturbance rejection [16], obstacle avoidance [17], fault tolerance [18], formation flying [19], and the use of new sensing and perception paradigms such as computer vision [20]. Even when the vehicle performs tasks autonomously, the efficiency and reliability of the communication link to the ground station or other aerial vehicles are important. This is because when the autonomous UAV sends information about itself or its environment to the ground station or other vehicles, it may also need to receive updated mission parameters from the ground station or information from other vehicles. These ambitious requirements of autonomous operation require systematic and innovative methods for planning, navigation, decision-making, control, sensing, and communications [21].

In dynamic and nonlinear control, building a mathematical function of the plant is needed to assure a stable controller. The stability of the controller is analyzed based on complicated mathematical methods and techniques. In many real-world applications, the accuracy of the plant's mathematical model is questionable. Furthermore, engineers perform mathematical approximations to simplify the model development. These approximations are based on some assumptions that limit the generalizability of the controller. The assumption can lead to stability and reliability issues, such as violating the simplification assumptions considered in the approximation when the controller operates in real-world scenarios. Hence, to avoid such approximations and nonvalid assumptions, the concept of free model control is used. However, instead of using it based on repeated trial and error for tuning a simplified controller, it can be used to develop an accurate controller that embeds sufficient gained knowledge from the plant [22].

Reinforcement learning (RL) is one type of model-free control based on artificial intelligence (AI). It has proven itself an effective and practical approach to controlling nonlinear and complex dynamic systems, especially when accurate modeling is difficult. Furthermore, integrating RL with a deep-neural network for scene analysis from video and decision-making based on extensive training has found its niche valuable in AI products in the automotive industry and driverless cars [23] and the control of aerial vehicles [24]. The reason for this is the ability to train the RL model based on an extensive number of driving scenarios and then to use the learned knowledge in operation. Hence, RL is considered a type of model-free control as it does not require a model for control application. Among the RL models, the Deep Deterministic Policy Gradient (DDPG) has been developed [25]. It is considered the first deterministic actor–critic that employs deep neural networks for learning in the actor and critic. It is a model-free, off-policy algorithm that extends both the Deep Q Network (DQN) and the DDPG because it uses some insight from DQN, such as replay buffer and target network, to make the DPG work with deep networks. However, it has a problem of sensitivity to hyperparameters. Recently, one algorithm has replaced the DDPG: the Twin Delayed Deep Deterministic Policy Gradient (TD3) [26]. It is being considered a replacement because it is a continuation of the DDPG algorithm, with some ingredients that make it more stable with better performance, such as reducing the over-estimation bias because of the delayed training architecture and the learning speed.

This article aims to develop a target tracking by a UAV using TD3-based RL. The developed algorithm contains a proportional differential (PD) controller for boosting the exploration and handling the control on one axis, whereas TD3 controls the UAV on the other two axes. The article includes several contributions as follows:

1) To the best of the authors' knowledge, this study is the first to apply TD3 for the UAV-based target tracking problem with PD for boosting the exploration of the TD3 in training. Previously, the work of [27] has applied TD3 combined with meta-learning. However, it was based on a simple simulation model in XY only without addressing the stabilization of the third dimension. In this work, TD3 was adopted instead of the DDPG. This is because it has an architecture that solves several problems in the DDPG.
2) It proposes a novel reward formulation for UAV-based target tracking that enables a careful combination of the various dynamic variables in the reward functions. The novel rewarding function incorporates two exponential functions to limit the effect of velocity and acceleration

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3154388, IEEE Access

to prevent the deformation in the policy function approximation.

3) It proposes an enhancement of the rewarding function by including piecewise decomposition to enable the policy's more stable learning behaviour and move away from the linear reward toward achievement formula.
4) A thorough evaluation is conducted to evaluate the developed models and compare them with standard evaluation metrics.

The remainder of the article is organized as follows. The literature survey is given in Section II. Next, the methodology for target tracking implementation by UAV based on TD3 and reinforcement learning is presented in Section III. The experimental evaluation and results are provided in Section IV. Finally, the conclusion and direction for future studies are given in Section V.

III. LITERATURE SURVEY

The UAV-based tracking problem can be categorized into trajectory tracking and target tracking. Several approaches based on RL are found for trajectory tracking. In [28], RL created quadrotor controllers for hovering at a fixed point and circular trajectory tracking. Policy gradient-based actor–critic architectures that use neural networks as the function approximator have been used for both the value and policy functions. For target tracking, RL-based UAV was used to track both the stand-alone UAV and cooperative UAVs. In [29], multiagent reinforcement learning (MARL) for target tracking was proposed. It includes local and global observation definition, action, dedicated reward functions, and the learning method with a joint state and action tracker for a stable strategy training procedure. Curriculum learning and sequencing the intractable pursuit process into four statuses is adopted. Each status corresponds to a more trackable subtask, and all statuses are organized into a curriculum that characterizes the order of solving the subtasks. Based on the four predefined statuses, a status-oriented cooperative pursuit reward is developed to guide pursuers in learning complex cooperative pursuit strategies by addressing the tractable subtasks sequentially.

The literature includes numerous works for developing target tracking based on RL. In the work of [30], RL-based coordination of a swarm of drones for target searching and monitoring was proposed. The problem addressed was the trajectories planning in cooperative patrolling and tracking missions. The environment was split into several grids, and the grid represented the location of the UAV. A stationary station for refueling the UAV was deployed. The actions of RL were formulated at the upper management level of the UAV. In other studies, deep RL was used to assist the UAV in target detection. In the work of [31], a coarse-to-fine deep scheme was used to address the aspect ratio variation in UAV tracking. The coarse tracker first produced an initial estimate for the target object. Then, a sequence of actions was learned to fine-tune the four boundaries of the bounding box. The coarse-tracker and the fine-tracker were designed to have different action spaces and operating targets. The former dominates the entire bounding box, and the latter focuses on the refinement of each boundary. They are trained jointly by sharing the perception network with an end-to-end RL architecture. However, in other research works, RL was utilized for commanding the UAV at lower levels. For the autonomous landing of an aerial vehicle on a moving target, tracking is a vital functionality. Deep Q learning was the most used for a single drone [32]. Other approaches have adopted deep reinforcement learning to handle the continuous nature of control. In the work of [33], tracking was used with landing based on decomposition into two separate tasks, namely, marker alignment and vertical descent.

In addition, the divide-and-conquer paradigm was used for splitting the tasks into two subsequent tasks in which each one was assigned to a DQN. In the work of [34], the DDPG was integrated with the RL framework. The approach considered the tracking in X, Y as part of the reinforcement control, whereas Z was separated. In addition, the work proposed a rewarding function that does not consider adequate dynamics, making the approach applicable only in simple maneuvers in landing. In the work of [35], a sequential DQN was trained in a simulator before it was deployed in the real world, handling noisy conditions. In the work of [36], an autonomous landing based on RL solved by the least-square policy iteration was performed. The target was stationary, and the rewarding functions used two terms, one for the position error and the other for the velocity error with adaptive weighting. The weights were considered to be exponentially changing with respect to the error so that the position error gained more weight when the error was large, and the velocity error gained more weight when the error was small. The authors have not discussed the quantization of the velocity and the position in their work. In the work of [37], image-based visual serving has been proposed using Kalman filtering and RL. Their work has shown the importance of using velocity error in the reward function and the effectiveness of asymmetric rewards. Considering that the reward plays an essential role in the controller's performance, some researchers have attempted to design an inverse RL for reward optimization. In the work of [38], the hidden reward function of a quadratic form from the demonstrated flights was learned using inverse RL. Next, the optimal reward function that minimizes the trajectory tracking error was found, and a reinforcement learning-based controller using this reward function was proposed. In the work of [39], Target Following DQN (TF-DQN), a deep reinforcement learning technique based on DQNs was proposed with a curriculum training framework for the UAV to persistently track the target in the presence of obstacles and target motion uncertainty. For the reward function, a piecewise reward was proposed to enable different rewards according to the status of the collision compared with the noncollision. In the work of [40], the constrained Markov decision process (CMDP) was formulated based on the flight decision process with the goal of optimizing the redundant UAV flight path. The target continuously broadcasts radio frequency signals to all UAVs in their work.

The goal is to realize the target within a given time threshold. The Q-learning was formulated based on coordinated constraint action-based multi-agent Q learning.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3154388, IEEE Access

They aimed to improve the tracking performance based on the addition of a constraint on the rewarding.

In the work of [41], a DDPG-based control framework was used to provide learning and autonomous decision-making capability for UAVs. In addition, an improved method, named mixture noise DDPG (MN-DDPG), for introducing a type of mixed noises to assist UAV by exploring stochastic strategies for optimal online planning was proposed. Finally, an algorithm of task-decomposition and pretraining for efficient transfer learning to improve the generalization capability of the UAV's control model was built based on the MN-DDPG. In the work of [27], metalearning has been incorporated in the training of the TD3 to enable more generalization and faster convergence. For metalearning, the authors have created a metabuffer. The algorithm samples from this buffer were based on the metalearning rate for updating the hyperparameters.

In the work of [42], UAV tracking and landing tasks based on a randomly moving platform have been handled using the DDPG. The algorithm uses three coordinates for relative position and velocity as distance and velocity change as action. The reward is the relative distance with a threshold penalty. In the work of [35], the DQN was used for landing. The approach was based on a divide-and-conquer paradigm that split a task into sequential subtasks, each one assigned to a DQN. Random sampling was used to improve the generalization. In the work of [43], the problem of search and rescue based on multiple UAVs was tackled in a 3D environment. Cramér–Rao Lower Bound (CRLB) of the joint measurement likelihood function was used to select the action. The actions in their formulation are discrete, which is helpful in simplification but affects fine tracking. In addition, the state definition does not include the dynamic information of the target, which also does not make the algorithm perform well in highly dynamic conditions. Table I includes an overview of the various RL-based models developed in the literature for UAV tracking application, reviewing their developed RL basics and attributes. As observed in the table, none of them has used the TD3 as an agent. Hence, this confirms that implementing TD3-based tracking has not yet been accomplished in the literature, making it one of the novelties provided in the current article, as stated earlier.

TABLE I
OVERVIEW OF RL-BASED APPROACHES FOR UAV TRACKING APPLICATION

Author	Multi-UAV	State	Action	Reward	Agent
[30]	√	A node within Upper Confidence Tree (UCT)	Moving UAV from one grid to one of its four adjacent grids within the searching area	the fuel status + the sum of the probability of whom the grids are located inside the fleet's horizon	Q-learning
[31]	×	The appearance information + the action history information	Stop-action + expand outward and move inward depending on the relative direction.	Binary function based on the intersection-over-union (IoU)	Q-learning
[32]	×	Extracted features from the raw camera image.	To the speed in the quadrotor in x and y directions. Speed in the z-direction is not considered	The position information of the marker and the agent are used to construct the reward function. The maximal reward changes when the altitude differs. The lower the height, the less the maximal reward.	Deep Q Reinforcement Learning
[44, 45]	×	Relative position on x, y, and z, Velocity on x, y, and altitude	Acceleration on x and y	Relative position, velocity and acceleration based on x and y	Deep Q Reinforcement Learning
[35]	×	The image acquired by a downward-looking camera mounted on the UAV	Backwards, right, forward, left, stop, descent, land	-	Sequential Deep Q-Network (SDQN)
[36]	×	Instantaneous error in position and velocity	Control velocities	Two-term reward function: one uses error with respect to position and second uses error with respect to velocity	Least Square Policy Iteration (LSPI) based RL
[46]	×	Position, angle, velocity, and angular rate	-	quadratic reward function	Inverse Reinforcement Learning Algorithm
[40]	√	Consists of the received signal strength (RSS) information obtained by the UAV	The flight direction of the UAV	The improvement in the RSS is only considered when the action changes significantly from the previous action	Q-learning
[41]	×	Distance, velocity azimuth and surrounding obstacles	Acceleration and angular rate, Mixture noise has been added to action for generalization	Four types of reward: track, course, safety, steady	DDPG
[27]	×	Position and angle	Acceleration and angular rate	Normalized distance and penalty	Metalearning
[42]	×	Position and velocity of UAV and target concerning x, y and z	The velocity of UAV concerning x, y and z	The relative distance between UAV and target	DDPG

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3154388, IEEE Access

[43]	√	The absolute position of UAVs and the relative position between UAVs and targets	Discrete actions of changing the position of UAVs	Two Global and one local reward	Deep reinforcement learning, Deep Dueling Q-network

VI. METHODOLOGY

This section provides the developed methodology to accomplish target tracking by a UAV based on the TD3 and RL. The methodology consists of problem formulation. Next, the general framework is presented, followed by the observation and state. Next, the definition of the action and the rewarding model are provided and, finally, the episode completion logic.

A. Problem Formulation

Assume that a target exists within the field of view of a UAV and is moving with an unknown trajectory. The problem is to control the UAV to maintain the target in the center of the image of the UAV's frame. Without loss of generality, it is assumed that the target is moving in the plane yz and the UAV and the TD3-based RL are responsible for controlling the UAV to perform its tracking in yz. For dimension x, a PD controller is responsible for controlling the UAV to maintain the same distance with respect to the target. The target was detected based on the AprilTag detection algorithm. In addition, the low levels command of changing the acceleration of the UAV with respect to the axes x, y and z were performed based on the internal proportional integral differential (PID) control embedded in the UAV controller, which exists in most commercial UAVs nowadays.

The article focuses on the upper-level TD3-based RL training to provide the required tracking within different scenarios of target mobility. A conceptual diagram of target tracking using the UAV is presented in Figure 1.

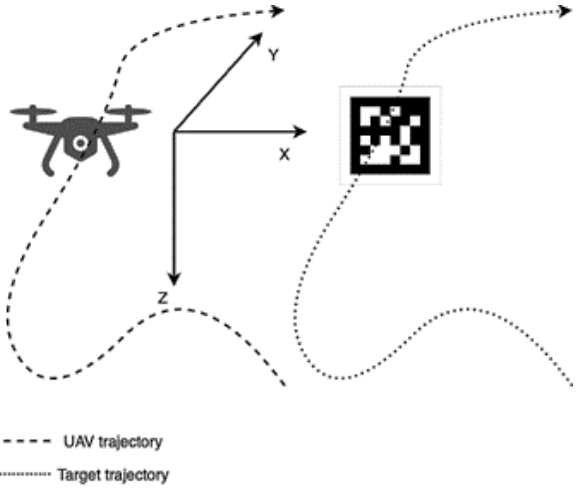

Figure 1 The conceptual diagram of target tracking based on UAV.

B. The general framework

The general framework of establishing UAV tracking of the target using TD3-based RL is presented in Algorithm 1, and a conceptual block diagram for it is depicted in Figure 2. As shown in the figure, the state estimation provides the needed information to the two controllers, namely, the PD and the RL agents. Next, a block of inverse Kinematic was enabled for outputting the low-level control signals that are affecting the environment. After that, the camera and inertial sensing were used to update the state of the environment.

As observed in algorithm 1, the initialization starts by initiating the PD controller and the TD3 networks in line number 2.

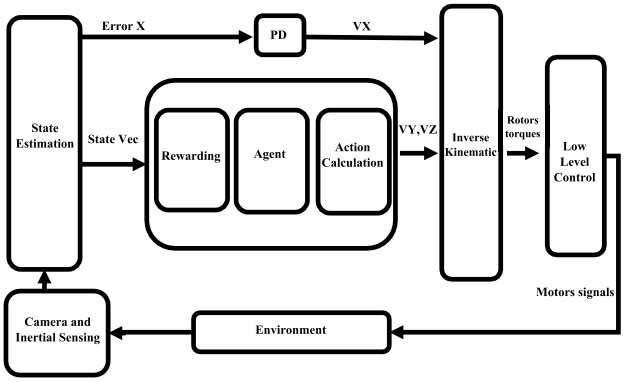

Figure 2 The conceptual diagram of the developed RL based tracking.

In addition, the initialization includes defining the number of episodes to train the TrainingEpsdsNum; the PD exploration steps PDExplrStps, the desired relative position DsrdRelPos based on the GetDesiredRelativePosition(), and the initial UAV position DsrdDronePos. The role of the TrainingEpsdsNum was to determine how many episodes were needed to finish the training. Increasing the value of the TrainingEpsdsNum does not mean a more mature agent because of over-fitting. Hence, it is important to enabling agent selection based on the validation phase to decide which agent is the best among the generated episodes. The role of the DsrdRelPos is to define the range of accepted errors in this control problem. The role of the DsrdDronePos is to enable training from different locations of the initial drone position. The PDExplrStps role is to control the boosting phase when the PD is used to guide the UAV instead of the TD3 until enough maturity is reached by the buffer experience to change to the TD3 mode.

The algorithm starts by launching the simulation at line 7 using LaunchSimulation(). Next, it uses GetBufferExperiencesNum()to update the size of BufExpcNum, which shows the index of the current last update of the experience buffer. It is important to note that this variable is updated upon each control step, as is shown in the pseudocode in line number 29. Afterwards, ConstructStateVector() was performed to build the state vector, respectively. The main loop in the algorithm is located between lines 14 and 41, and it is the loop of episodes. Inside the loop, there is another loop for each episode separately, placed in lines 16 to 34. In this loop,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3154388, IEEE Access

there are two branches: the first one is where the PD controller is consulted for generating actions for y, z and z and angular rotation around z, and the second one is where the PD controller is consulted to select actions for only x and the angular rotation around z while the TD3 handles y and z control, which represents the core tracking part. Upon the control, there is a step of updating the buffer using the command AddExperienceToBuffer() in line 27. In addition, it can be seen that when the buffer gets sufficient data and the PD exploration phase finishes, there is a repeated step of updating the TD3 knowledge in line 32.

Algorithm 1 Pseudocode Training Main

```
1: Initialization
2: Initialize ()
3: TrainingEpsdsNum
4: PDExplrStps
   DsrdRelPos
5: end initialization
6: Start Algorithm
7:  LaunchSimulation()
8:  BufExpcNum ← GetBufferExperiencesNum()
9:  PrevStaVec ← ConstructStateVector()
14: for EpsdNum ←0, TrainingEpsdsNum do
15:     EpsdCmplt ← False
16:     While EpsdCmplt =False do
17:         if BufExpcNum < PDExplrStps then
18:             [yzActn, xwzActn]← GenerateActionUsingPD()
19:         elseif BufExpcNum >= PDExplrStps then
20:             xwzActn ←GenerateActionUsingPD()
21:             yzActn ←GenerateActionUsingTD3(PrevStaVec)
22:         end if
23:         Advance DroneMotion(xwzActn, yzActn)
25:         NextStaVec← ConstructStateVector()
27:         AddExperienceToBuffer()
29:         BufExpcNum←GetBufferExperiencesNumber()
30:         if BufExpcNum > PDExplrStps then
30:             ExtractRandomMinibatchFromBuffer()
32:             UpdateTD3PolicyNetwork()
33:         end if
34:     end while
35:     if EpsdCmplt =True then
36:         RelaunchDroneSimulation()
39:         PrevStaVec← ConstructStateVector()
40: end if
41: end for
42: End Algorithm
```

C. Observation and State

The observation updated at each moment, t, includes nine variables, calculated based on the position of the drone at the moment t $(x_{drone,t}, y_{drone,t}, z_{drone,t})$ and the position of the target at the moment t $(x_{target,t}, y_{target,t}, z_{target,t})$.

The observation is given in the vector

$O_t(x_{rel,t}, y_{y,rel,t}, z_{rel,t}, v_{x,rel,t}, v_{y,rel,t}, v_{z,rel,t}, a_{x,rel,t}, a_{y,rel,t}, a_{z,rel,t})$

The state is given based on the part of the observation or

$s_t = (y_{y,rel,t}, z_{rel,t}, v_{y,rel,t}, v_{z,rel,t})$

D. Action

The action vector consists of two elements, $a_t = (c_{y,t}, c_{z,t})$ where $c_{y,t}$ denotes the action of changing the acceleration of y, $c_{z,t}$ denotes the action of changing the acceleration of z. It pointed out that this part is under the mission of the TD3, whereas the action of changing the acceleration or x or the angular rate around z is given as $act_{PD} = (c_{x,t}, c_{wz,t})$ and it is under the mission of the PD controller that is integrated with the TD3.

E. Rewarding Model

The reward is the essential part for guaranteeing a good performance of the RL convergence toward the optimal policy. It should enable optimal action selection given a certain state and provide more stable convergence. The previous researchers [45] include the error concerning the distance, velocity, and acceleration in the reward. In addition, they try to make the reward normalized to make the learning more stable. The classical rewarding model is given in Equation (1):

$$r = -w_p\bar{r_p} - w_v\bar{r_v} - w_a\bar{r_a} \quad (1)$$

where w_p denotes the weight of the position rewarding term, w_v denotes the weight of the velocity rewarding term, w_a denotes the weight of the acceleration rewarding term, and $\bar{r_p}$ denotes the normalized relative distance between the drone and the target and it is calculated based on Equation (2):

$$\bar{r_p} = \frac{y_{rel,t}^2 + z_{rel,t}^2}{\|R_p\|} \quad (2)$$

where R_p denotes the maximum magnitude of $y_{rel,t}^2 + z_{rel,t}^2$ and it is used for normalization, $\bar{r_v}$ denotes the normalized relative distance between the drone and the target and it is calculated based on Equation (3):

$$\bar{r_v} = \frac{vy_{rel,t}^2 + vz_{rel,t}^2}{\|R_v\|} \quad (3)$$

where R_a denotes the maximum magnitude of $vy_{rel,t}^2 + vz_{rel,t}^2$ and it is used for normalization, $\bar{r_a}$ denotes the normalized relative acceleration between the drone and the target and it is calculated based on Equation (4):

$$\bar{r_a} = \frac{ay_{rel,t}^2 + az_{rel,t}^2}{\|R_a\|} \quad (4)$$

R_a denotes the maximum magnitude of $ay_{rel,t}^2 + az_{rel,t}^2$ and it is used for normalization.

The modification in the reward is carried out based on the following:

1) A novel approach for rewarding is developed where the reward is not given at one time based on the three terms of position, velocity, and acceleration. However, it is given progressively throughout the training, where the entire set of episodes is decomposed into three stages. The rewarding based on the position term is given in the first stage, the rewarding based on the velocity term is given in the second stage, and the rewarding based on the acceleration term is given in the last term. This approach is called multistage rewarding. The pseudocode of multistage rewarding is given in Algorithm 2. As observed in the code, from lines 1 to 4, the first stage of position-based rewarding is executed. From lines 5 to 7, the second stage of velocity-based rewarding is given, and from lines 8 to 10, the stage of acceleration-based rewarding is given.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3154388, IEEE Access

Algorithm 2 Pseudocode Multi Stage Shaping Function

Input:
(1)EpsdNum: Episode Number.
PosEpsdsNum
VelEpsdsNum
AcelEpsdsNum
Output:
Shaping
1: **Start Algorithm**
2: **if** EpsdNum < PosEpsdsNum then
3: Shaping=CalPositionTerm()
4: **else if** (EpsdNum > PosEpsdsNum) and
5: (EpsdNum <VelEpsdsNum)then
6: Shaping =CalVelTerm()
7: **else if** (EpsdNum > VelEpsdsNum) and 9:(EpsdNum < 8:AcelEpsdsNum) then
9: Shaping =CalAccTerm()
10: **end if**
11: **End Algorithm**

2) An exponential factor for weighting the velocity and acceleration terms in the reward is incorporated. They are given in Equation (5-6):

$$w_v = w_{0,v} e^{-v} \quad (5)$$

$$w_a = w_{0,a} e^{-a} \quad (6)$$

The role of these terms is to assure that the rewarding of the dynamics will not exceed its safe level of affecting the policy surface.

3) An achievement concept of rewarding was developed where the reward formula changes according to entering or exiting a surrounding square frame around the target. To elaborate this concept, it was assumed that the target is surrounded with K frames, presented in the set $F = \{f_1, f_2, \dots f_K\}$. The reward is modified in Equation (7):

$$r_{pw}(t) = \begin{cases} r(t) + c_1 & if(target\ is\ within\ f_1) \\ r(t) + c_2 & if(target\ is\ within\ f_2) \\ . & \vdots \\ r(t) + c_K & if(target\ is\ within\ f_K) \end{cases} \quad (7)$$

where f_1 is surrounding f_2, f_2 is surrounding f_3, and so on until the last frame f_K. $c_1 < c_2 < \cdots c_K$. The model is called an achievement-based rewarding because the constants c_i are given at each frame as an extra reward because of the agent's achievement.

F. Episode Completion Logic

The episodes consist of the fixed target set of episodes and the moving target set of episodes. The completion of one episode and the starting of a new episode is based on combinatory logic. More specifically, the episode ends with the availability of one of three conditions in the fixed target, namely entering the inner area of a square surrounding the target, exceeding the area of simulation, or exceeding the allocated steps for the episodes.

On the other side, the episode ends with the availability of one of two conditions in the case of the moving target, namely exceeding the area of simulation or exceeding the allocated steps for the episodes. The algorithm that shows the logic of episode completion is given in Algorithm 3. The part from line 6 enables the terminal state successfulness flag in the case of the fixed target. Lines 7 to 11 enables the flag of failure to reach the terminal state due to exceeding the area in the case of the moving target.

Algorithm 3 Episode Completion Status

Input:
(1) StaVec: State Vector.
(2) TrmnlStaThrshld: Terminal State Threshold. Index 1 for position and 2 for velocity
(3) MaxRelPos: Maximum Relative Position.
(4) EpsdStpNum: Episode Step Number.
(5) MaxEpsdStps: Maximum Episode Steps.
(6) TagTrajType: Tag Trajectory Type.
Output:
EpsdCmplt: Episode Completion.
1: **Start Algorithm**
2: EpsdCmplt ←False
3: TrmnlStaStatus ←False
4: FlgAreaExcd← False
5: MaxEpsdStpsStatus ←False
6: **if** absolute(StaVec['yAxisLinearPos']) < TrmnlStaThrshld(1) and absolute(StaVec['zAxisLinearPos']) < TrmnlStaThrshld(1) and absolute(StaVec['yAxisLinearVelocity']) < TrmnlStaThrshld(2) and absolute(StaV ec['zAxisLinearVelocity']) < TrmnlStaThrshld(2) then
7: TrmnlStaStatus← True
8: **end if**
9: **if** absolute(StaV ec['yAxisLinearPos']) > MaxRelPos['yAxis'] or absolute(StaV ec['zAxisLinearPos'])>MaxRelPos['zAxis'] then
10: FlgAreaExcd ←True
11: **end if**
12: **if** EpsdStpNum = MaxEpsdStps then
13: MaxEpsdStpsStatus ←True
14: **end if**
15: **if** TagTrajType ='fixed' then
16: **if** TrmnlStaStatus =True or FlgAreaExcd =True or MaxEpsdStpsStatus =True then
17: EpsdCmplt ←True
18: **end if**
19: **end if**
20: **if** TagTrajType ='moving' then
21: **if** FlgAreaExcd =True or MaxEpsdStpsStatus =True then
22: EpsdCmplt ←True
23: **end if**
24: **end if**
25: **End Algorithm**

VII. EXPERIMENTAL EVALUATION AND RESULTS

For simulation, the Gazebo simulator was used. It is a three-dimensional dynamic simulator that can correctly and effectively model UAVs and robots. For training, the set of the initial random positions was selected with $N = 9$, and it is given as:

$RP =$
$\{(0,0.15,0.5), (0,0.15,1.15), (0,0.15,1.5), (0,0.5,0.5),$
$(0,0.5,1.15), (0,0.5,1.5),\ (0,-0.5,0.5),$
$(0,-0.5,1.15),\ (0,-0.5,1.5)\}.$

For the multistage rewarding, $K = 5$, $c_1 = 20, c_2 = 40, c_3 = 60, c_4 = 80$ and $c_5 = 100$ werre used. The parameters of the experiments are presented in Table II. In addition, the TD3 parameters in Table III are presented. As given in the table, the number of hidden layers is 2, and the number of hidden neurons in each layer is 256. Other parameters are the standards used by researchers for TD3 implementation.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3154388, IEEE Access

TABLE II
PARAMETERS OF THE REWARDING MODEL

Parameter Name	Value
PosEpsdsNumForFixedTag	1000
VelEpsdsNumForFixedTag	500
AcelEpsdsNumForFixedTag	100
PosEpsdsNumForMovingTag	150
VelEpsdsNumForMovingTag	75
AcelEpsdsNumForMovingTag	15
c_1	20
c_2	40
c_3	60
c_4	80
c_5	100
K	5

TABLE III
PARAMETERS OF TD3 ALGORITHM

Parameter Name	Value
Hidden layers number	2
Hidden layer nodes number	256
Discount factor	0.99
Optimizer	Adam
Learning rate for Actor networks	0.0003
Learning rate for Q-networks	0.0003
Buffer size	10800000
Batch size	256
PD exploration steps	10000
Episodes number	1840
Maximum episode steps	4500 (150 2nds at 30Hz frequency)
Soft update coefficient	0.005
Policy delay	2
Action noise	$N(0,0.1^2)$
Target noise	$N(0,0.2^2)$
Noise clip	0.5

The evaluation results were reported under boxplot visualization to characterize the random behavior of the performance for each model. The labelling coding presented in Table IV was used for the various models evaluated. The type of evaluated agent from the models was added as a title for each figure. The original TD3 model does not include achievement reward or exponential weighting. In addition, it was based on the combined training of position, velocity and acceleration, named as combined (C). Two types exist agents: agents trained by fixed target only (F) and agents trained by fixed and moving target (FM). For FM agents, the training was based on the first stage of training on a fixed target and the second stage of training on moving targets within the square path with a diameter of 0.5, 1 and 1.5 meters. It is pointed out that the C agent of FM can be called metalearning TD3 because it used the same concept of [27]. Two evaluation metrics are presented for each agent type, namely the accumulated error on the y axis, which is named as E_y and the accumulated error on the z axis, which is named as E_z. They both indicate the accumulated root mean square error.

TABLE IV
LABELLING CODING FOR THE MODELS USED IN THE EVALUATION

Model name	Label code	Achievement reward	Exponential weighting
Combined	C	No	No
Combined-Achievement	CA	Yes	No
Combined-Exponential	CE	No	Yes
Combined- Achievement-Exponential	CAE	Yes	Yes
Multilevel	ML	No	No
Multilevel-Achievement	MLA	Yes	No
Multilevel -Exponential	MLE	No	Yes
Multilevel - Achievement-Exponential	MLAE	Yes	Yes
Proportional Differential	PD	No	No

A. Fixed Target

The developed TD3-based tracking was evaluated based on two types of analysis. The first one is the analysis of the statistical results of the errors in both Y and Z, given in Subsection 1. The second one is the evaluation of the time series of the relative distance between the UAV and the target in both Y and Z throughout the experiment, given in Subsection 2. For both analyses, a boxplot was selected to capture the random behavior in the experiments and incorporate it in the evaluation.

1) STATISTICAL RESULTS

It was observed in Figures 3, 4, 5 and 6 that the accumulated error on Y and Z, for the F agent axis, shows that the best achieving agent was Multilevel - Achievement-Exponential (MLAE) with an accumulated error of less than 50. The worst performance was observed for Combined-Exponential (CE), which has reached an error of close to 350 for Y and 400 for Z. This provides that incorporating the exponential weighting in the combined rewarding is not useful in improving the latter.

In addition, it was observed that all Multilevel-Achievement (MLA), Combined- Achievement-Exponential (CAE) and Combined-Achievement (CA) have provided much better performances than both proportional differential (PD) and Combined (C), which are just classical TD3-based models with no modifications. The ranges of errors provide that adding an achievement term to the TD3 is useful for improving the tracking performance and reducing the error. Furthermore, combining both the achievement rewarding formula and the exponential weighting terms provides better performance than using the achievement rewarding alone. Another observation is that the width of the boxplot is reduced for the achievement-based agents, namely CA, CAE, MLA and MLAE, which means more stability in the performance when they are trained on a fixed target, i.e., F-agent. More specifically, as is observed from Table V of the summary of the errors in Y and Z that F-agent MLAE with the error of Y of 39.53 in Y axis has increased to 125 in Fixed then Moving trained (FM), and the error of Z has increased from 51 in the F training case to 122 in the FM agent. The stability generated from achievement rewarding is interpreted by the piecewise formula that makes the agent aware of its progress in the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3154388, IEEE Access

tracking and its motivation when it passes from one region to another closer to the target. Also, it was observed that the best agent in the FM training was MLA, with an error of 64 on Y and 113 on Z.

The time series is presented in Figure. 5, showing good tracking performance by maintaining the location of the target in both Y and Z despite the frequent sensor failure cases that are shown at the bottom graph.

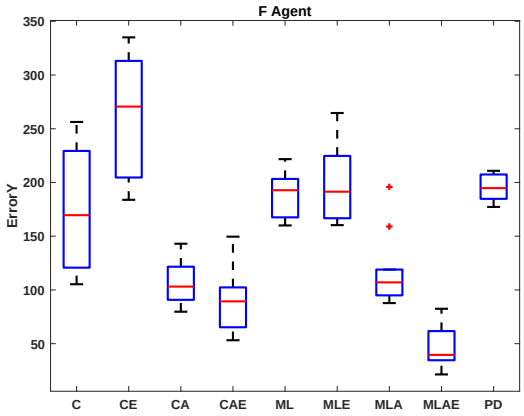

Figure. 3 Boxplot of an error on Y-axis for various agent types trained on fixed target and tested on fixed trajectory scenario.

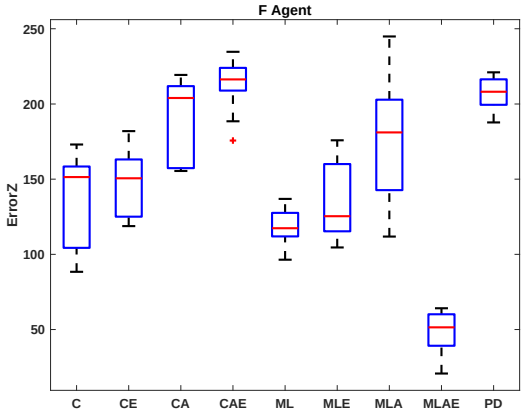

Figure. 4 Boxplot of an error on Z-axis for various agent types trained on fixed target and tested on fixed trajectory scenario.

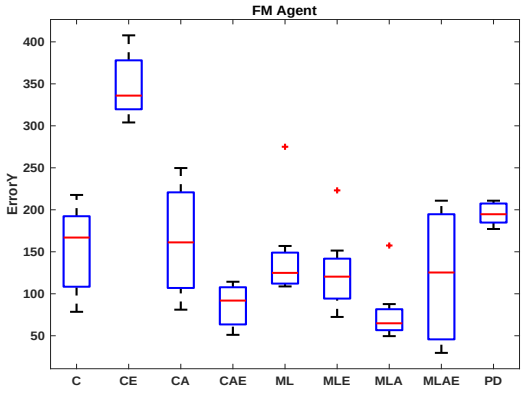

Figure 5 Boxplot of an error on the Y-axis for various agent types trained on fixed and moving target and tested on fixed trajectory scenario.

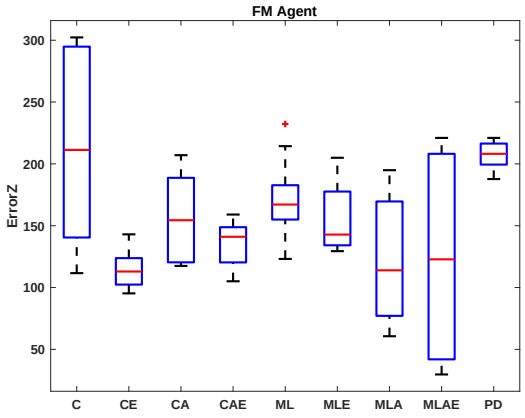

Figure 6 Boxplot of an error on Z-axis for various agent types trained on fixed and moving target and tested on fixed trajectory scenario.

TABLE V
SUMMARY OF THE ERRORS IN Y AND Z FOR F AND FM TRAINING TYPES AND THE DIFFERENT TYPES OF THE AGENTS FOR FIXED TARGET TESTING

Training Type	F		FM	
Agent Type	ErrorY	ErrorZ	ErrorY	ErrorZ
C	169.601	151.403	166.922	211.281
CE	270.580	150.576	336.036	112.968
CA	103.095	203.972	161.246	154.457
CAE	89.3708	216.347	91.9356	141.056
ML	192.755	117.360	124.874	167.104
MLE	191.440	125.371	120.427	142.800
MLA	106.995	181.061	64.8968	113.958
MLAE	39.536	51.4315	125.353	122.805
PD	194.773767	208.118	194.774	208.118
Error Reduction Rate	70%	75%	67%	42%

2) TIME SERIES RESULTS

The visualization of the dynamic performance is given by presenting the time series of the unit step response. As depicted in Figure 7, the tracking shows good performance for both Y and Z despite the cases of sensor failures caused by the nondetection of the tag. Hence, the model shows good robustness of the UAV tracking.

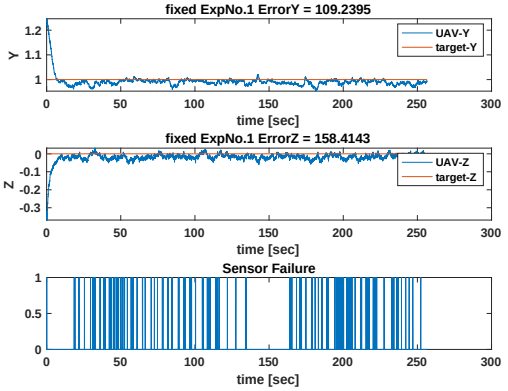

Figure 7 Time response of the scenario of one example fixed target scenario for the best agent.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3154388, IEEE Access

B. Moving with Square Trajectory

The evaluation of the target that moves along a square trajectory was decomposed into two subsections. The first is the statistical evaluation, presented in (1), and the second is the time series evaluation, presented in (2).

1) STATISTICAL RESULTS

Similarly, the tracking performance of the square trajectory scenarios conducted by the object observed from Figures 8, 9, 10, and 11 show that the best-achieved tracking performance was accomplished by MLAE for the F-agent, with an accumulated error in Y and Z close to 50. On the other side, the maximum error has occurred by the PD, showing an error of approximately 300 in Y and Z. Additionally, a decline in the performance for the FM agents with the well-accomplished performance of MLA and the least performance of CE was observed. The median values of the errors are shown in Table VI, demonstrating that MLAE has generated an error of 43 and 54 in Y and Z, respectively. In addition, good tracking performance in the time-series graph in the table for the MLAE model is visualized.

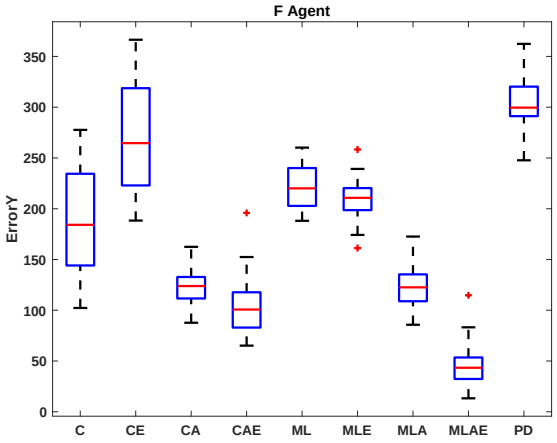

Figure 8 Boxplot of an error on Y-axis for various agent types trained on fixed target and tested on square trajectory scenario.

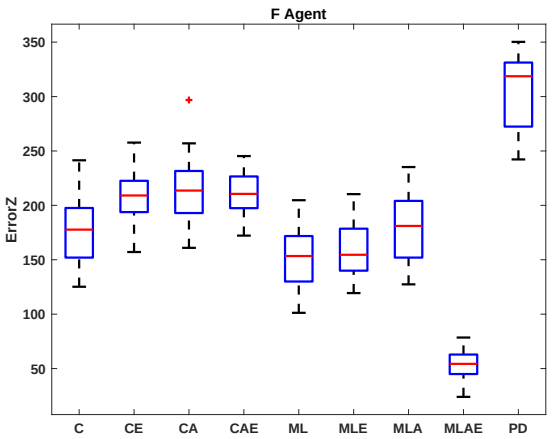

Figure 9 Boxplot of an error on Z-axis for various agent types trained on fixed target and tested on square trajectory scenario.

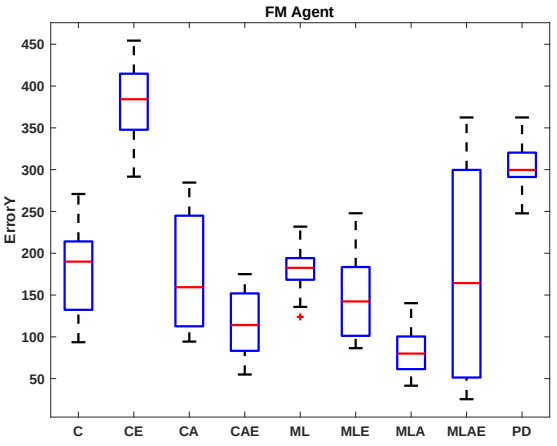

Figure 10 Boxplot of an error on Y-axis for various agent types trained on fixed followed by moving target and tested on square trajectory.

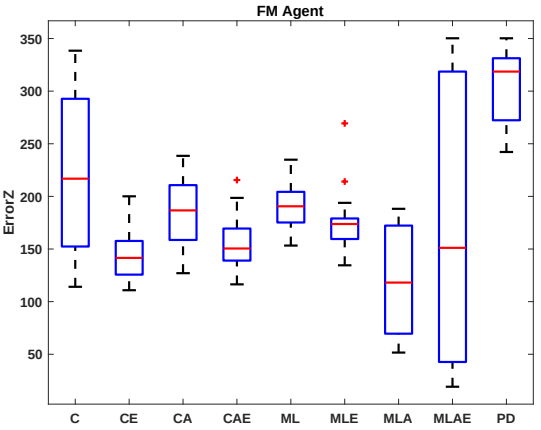

Figure 11 Boxplot of an error on Z-axis for various agent types trained on fixed followed by moving target and tested on square trajectory scenario.

TABLE VI
SUMMARY OF THE ERRORS IN Y AND Z FOR F AND FM TRAINING TYPES AND THE DIFFERENT TYPES OF THE AGENTS FOR SQUARE TARGET TESTING

Training Type	F		FM	
Agent Type	Error Y	Error Z	Error Y	Error Z
C	184.166	177.681	189.982	216.855
CE	264.580	209.041	384.108	141.549
CA	123.832	213.582	159.404	186.677
CAE	100.680	210.517	114.026	150.481
ML	220.008	153.393	182.467	190.611
MLE	210.726	154.562	142.353	173.775
MLA	122.576	181.085	79.970	118.095
MLAE	43.3472	54.3656	164.230	151.103
PD	299.611	318.629	299.611	318.629
Error Reduction Rate	86%	83%	73%	63%

2) TIME SERIES RESULTS

For visualizing the dynamic behavior of the tracking, the time series of the UAV compared with the target in Y and Z is provided in Figure 12. The tracking shows less deviation

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3154388, IEEE Access

between the two-time series, showing good tracking performance despite the cases of sensor failures in detecting the tag, which is represented by pulses in the bottom graph.

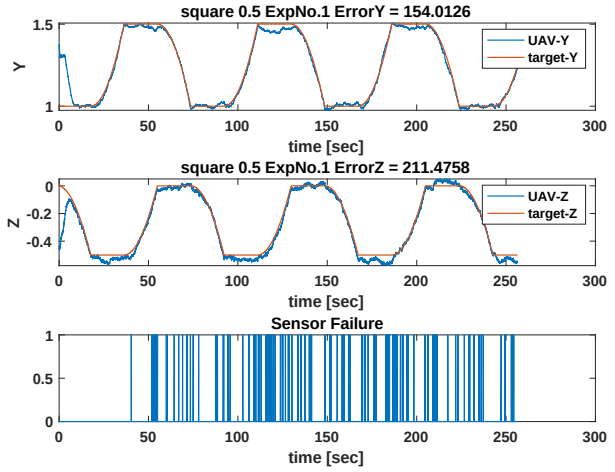

Figure 12 Time response of the scenario of one example square target scenario for the best agent.

c. Blinking Target

The final testing scenario was conducted on the blinking target, which explores the dynamic aspect of the tracking performance when the target moves in a disconnected way.

1) STATISTICAL RESULTS

The statistical results of the simulation experiments were also conducted for the blinking target. As observed in Figures 13, 14, 15, and 16, the least generated error on Y was 51 for the CAE agent, and the least generated error on Z was 52 for the MLE agent in the case of the F-trained agent.

On the other side, the least generated error on Y was 60 for the CAE agent, and on Z, it was 56 for the CE agent in the case of the FM trained agent. This indicates the superiority of the CAE performance at blinking-targets tracking. In addition, observing the behavior of the boxplot, the testing of the FM trained agents has resulted in a longer box, which shows less stability than the case of testing on the F-trained agents. The median values of the errors are shown in Table VII.

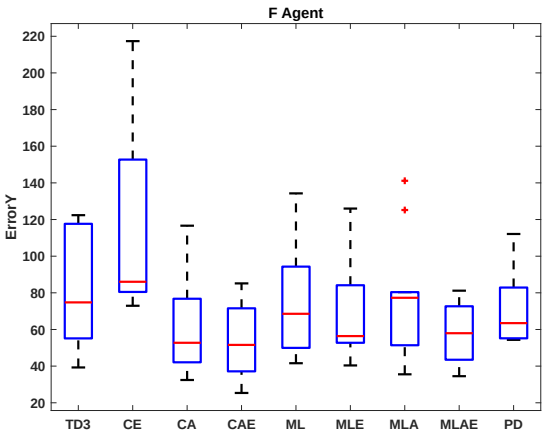

Figure 13 Boxplot of an error on Y-axis for various agent types trained on fixed target and tested on blinking target scenario.

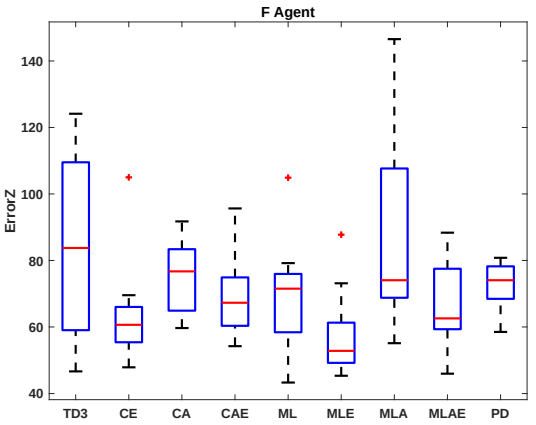

Figure 14 Boxplot of an error on Z-axis for various agent types trained on fixed target and tested on blinking target scenario.

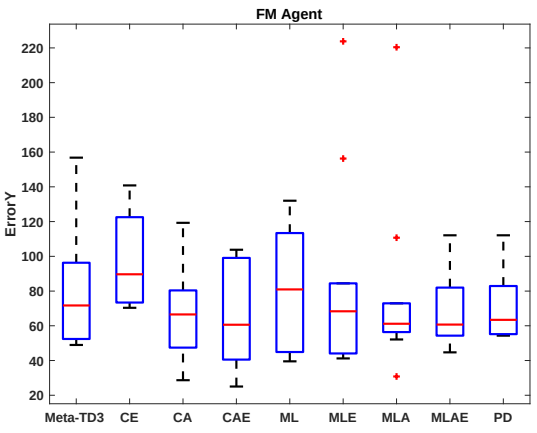

Figure 15 Boxplot of an error on Y-axis for various agent types trained on fixed followed by moving target and tested on blinking target scenario.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3154388, IEEE Access

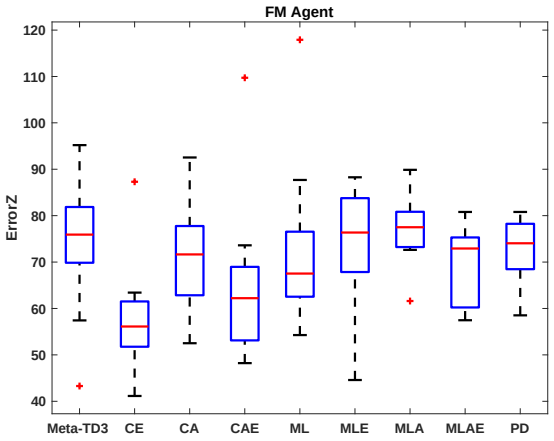

Figure 16 **Boxplot of an error on Z-axis for various agent types trained on fixed followed by moving target and tested on blinking target scenario.**

TABLE VII
SUMMARY OF THE ERRORS IN Y AND Z FOR F AND FM TRAINING TYPES AND THE DIFFERENT TYPES OF THE AGENTS FOR BLINKING TARGET TESTING

Training Type	F		FM	
Agent Type	Error Y	Error Z	Error Y	Error Z
C	74.770	83.772	71.722	75.915
CE	86.037	60.663	89.645	56.110
CA	52.727	76.731	66.527	71.640
CAE	51.554	67.295	60.585	62.225
ML	68.559	71.521	80.934	67.519
MLE	56.393	52.823	68.363	76.365
MLA	77.294	74.054	61.153	77.516
MLAE	57.952	62.598	60.699	72.917
PD	63.443	74.056	63.443	74.056

2) TIME SERIES RESULTS

The tracking response of one scenario from the experiments of the best accomplishing agent with respect to both Y and Z signals is visualized in Figure 17. The results show that within 5 seconds, the UAV was capable of maintaining minimum error on both Y and Z with respect to the target. In addition, the UAV was not affected by the frequent sensor failure that occurs because of the reduced quality of the UAV camera as it is considered as a cheap sensor.

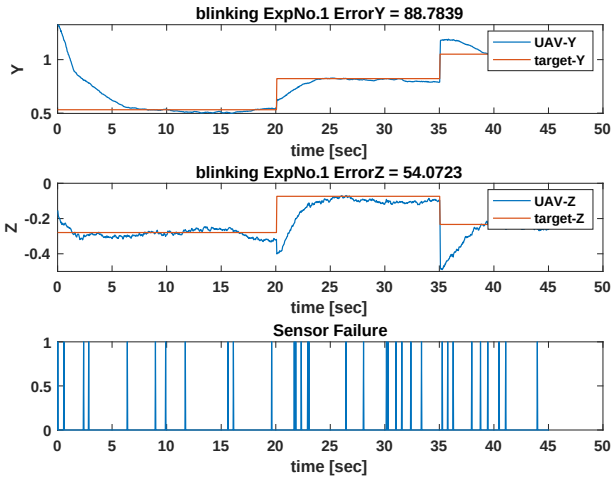

Figure 17. **Time response of the scenario of one example blinking target scenario for the best agent**

D- Cross Analysis

Comparing the various models based on both F agent and FM agent for the fixed scenario, it is found that MLAE has accomplished the least errors for F agent, 39 and 51 for *Y* and *Z* respectively, while MLA has accomplished the least error for FM agent in Y which is 65 and the second least in Z which is 114. The same was observed for the moving scenario. However, the superiority of MLAE and MLA was not found for the blinking scenario. This is interpreted by the difference between training an agent using standard fixed or moving scenarios on one side and training on random movement (blinking) on the other side. The latter is more challenging in providing represented knowledge to the agent.

E- Learned Lessons

It was observed from the three sets of scenarios that the developed RL based tracking improves the performance of the moving scenarios. This improvement is accomplished by minimizing the distance between the target and UAV, considering the dynamical variables such as velocity and acceleration, and capturing the behavior of target mobility. Additionally, the multi-level rewarding based training (MLA) based on position, followed by velocity and acceleration, is more beneficial for improving the learning of the dynamical behavior based on RL than combining the three variables in one rewarding function. Also, it was observed that the piecewise rewarding function or achievement rewaeding (CA) is useful for increasing learning effectiveness for dynamical behavior such as tracking than the simple continuous rewarding function. Lastly, the agent selection algorithm helps avoid overfitting, resulting from a higher allocated number of episodes for training.

VIII. CONCLUSION AND FUTURE WORKS

In this article, a novel algorithm for target tracking using the UAV is presented. The algorithm uses a recently developed agent architecture of RL, named TD3. The agent is

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3154388, IEEE Access

responsible for Y and Z control, whereas the third dimension, x, is controlled by the PID controller. This is by considering that the target only moves within y and z dimensions. The state contains the relative position and velocity between the UAV and the target. The actions are responsible for changing the acceleration of y and z. The reward was formulated based on three terms: position, velocity, and acceleration rewarding. The training was carried out based on two concepts: single-stage and combinatory rewarding of the three terms and multistage rewarding based on position, velocity, and acceleration one after the other. In addition, two methods were used for training: 1-fixed target training to produce the F-agent 2-fixed, followed by moving target training to produce the FM agent.

Two developments were added: (1) exponential factor was added to the velocity and acceleration terms to limit their effect on the policy surface, and (2) achievement rewarding to add more stability to the performance. The evaluation was based on three testing scenarios: fixed target, square trajectory target, and blinking target. The results showed that the best-accomplished performance was achieved by the multistage concept with both exponential and achievement rewarding for the fixed trained agent in the case of the fixed and square moving target and for a combined agent with both exponential and achievement rewarding for fixed trained agent in the case of the blinking target. This reveals that both combinatory and multistage training with both exponential and achievement when conducting the training on a fixed target is more effective for learning. Furthermore, the role of the exponential term in limiting the effect on the dynamic target, which is secondary in the learning and the role of achievement in boosting the training and stabilizing it, are promising concepts for developing more complicated models of tracking. Future work should extend the model to 3D-based RL tracking and explore its applicability to specific real-world applications such as target following in the military.

REFERENCES

[1] P. Grippa, D. A. Behrens, F. Wall, and C. Bettstetter, "Drone delivery systems: Job assignment and dimensioning," Autonomous Robots, vol. 43, no. 2, pp. 261-274, 2019.
[2] B. Mishra, D. Garg, P. Narang, and V. Mishra, "Drone-surveillance for search and rescue in natural disaster," Computer Communications, vol. 156, pp. 1-10, 2020.
[3] A. Abdallah, M. Z. Ali, J. Mišić, and V. B. Mišić, "Efficient security scheme for disaster surveillance UAV communication networks," Information, vol. 10, no. 2, p. 43, 2019.
[4] R. S. De Moraes and E. P. De Freitas, "Multi-UAV based crowd monitoring system," IEEE Transactions on Aerospace Electronic Systems, vol. 56, no. 2, pp. 1332-1345, 2019.
[5] H. You, "Mission-driven autonomous p0erception and fusion based on UAV swarm," Chinese Journal of Aeronautics, 2020.
[6] E. Lygouras, N. Santavas, A. Taitzoglou, K. Tarchanidis, A. Mitropoulos, and A. Gasteratos, "Unsupervised human detection with an embedded vision system on a fully autonomous uav for search and rescue operations," Sensors, vol. 19, no. 16, p. 3542, 2019.
[7] G. Joshi, B. Pal, I. Zafar, S. Bharadwaj, and S. Biswas, "Developing Intelligent Fire Alarm System and Need of UAV," in International Conference on Unmanned Aerial System in Geomatics, 2019, pp. 403-414: Springer.
[8] B. B. Kocer, T. Tjahjowidodo, M. Pratama, and G. G. L. Seet, "Inspection-while-flying: An autonomous contact-based nondestructive test using UAV-tools," Automation in Construction, vol. 106, p. 102895, 2019.
[9] Y. Zhang, X. Yuan, W. Li, and S. Chen, "Automatic power line inspection using UAV images," Remote Sensing, vol. 9, no. 8, p. 824, 2017.
[10] D. Bareiss, J. R. Bourne, and K. K. Leang, "On-board model-based automatic collision avoidance: application in remotely-piloted unmanned aerial vehicles," Autonomous Robots, vol. 41, no. 7, pp. 1539-1554, 2017.
[11] J. Aleotti et al., "Detection of nuclear sources by UAV teleoperation using a visuo-haptic augmented reality interface," Sensors, vol. 17, no. 10, p. 2234, 2017.
[12] A. Khadka, B. Fick, A. Afshar, M. Tavakoli, and J. Baqersad, "Non-contact vibration monitoring of rotating wind turbines using a semi-autonomous UAV," Mechanical Systems and Signal Processing, vol. 138, p. 106446, 2020.
[13] D. Zhang and R. P. Khurshid, "Variable-Scaling Rate Control for Collision-Free Teleoperation of an Unmanned Aerial Vehicle," arXiv preprint arXiv:1911.04466, 2019.
[14] A. Uryasheva, M. Kulbeda, N. Rodichenko, and D. Tsetserukou, "DroneGraffiti: autonomous multi-UAV spray painting," in ACM SIGGRAPH 2019 Studio, 2019, pp. 1-2.
[15] A. Castillo, R. Sanz, P. Garcia, W. Qiu, H. Wang, and C. Xu, "Disturbance observer-based quadrotor attitude tracking control for aggressive maneuvers," Control Engineering Practice, vol. 82, pp. 14-23, 2019.
[16] L.-X. Xu, H.-J. Ma, D. Guo, A.-H. Xie, and D.-L. Song, "Backstepping sliding-mode and cascade active disturbance rejection control for a quadrotor UAV," IEEE/ASME Transactions on Mechatronics, vol. 25, no. 6, pp. 2743-2753, 2020.
[17] A. Singla, S. Padakandla, and S. Bhatnagar, "Memory-based deep reinforcement learning for obstacle avoidance in UAV with limited environment knowledge," IEEE Transactions on Intelligent Transportation Systems, 2019.
[18] M. De Benedetti, F. D'Urso, G. Fortino, F. Messina, G. Pappalardo, and C. Santoro, "A fault-tolerant self-organizing flocking approach for UAV aerial survey," Journal of Network Computer Applications, vol. 96, pp. 14-30, 2017.
[19] J. Wang, Z. Zhou, C. Wang, and Z. Ding, "Cascade structure predictive observer design for consensus control with applications to UAVs formation flying," Automatica, vol. 121, p. 109200, 2020.
[20] R. Gabriele and L. Marco, "A Computer Vision Line-Tracking Algorithm for Automatic UAV Photovoltaic Plants Monitoring Applications [J]," Energies, vol. 13, no. 4, 2020.
[21] V. M. Becerra, "Autonomous control of unmanned aerial vehicles," ed: Multidisciplinary Digital Publishing Institute, 2019.
[22] M. Fliess, "Model-free control and intelligent PID controllers: towards a possible trivialization of nonlinear control?," IFAC Proceedings Volumes, vol. 42, no. 10, pp. 1531-1550, 2009.
[23] A. E. Sallab, M. Abdou, E. Perot, and S. Yogamani, "Deep reinforcement learning framework for autonomous driving," Electronic Imaging, vol. 2017, no. 19, pp. 70-76, 2017.
[24] K. Kersandt, "Deep reinforcement learning as control method for autonomous uavs," Universitat Politècnica de Catalunya, 2018.
[25] M. Pina Navarro, "Control mediante aprendizaje por refuerzo del robot Pepper," 2021.
[26] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in International Conference on Machine Learning, 2018, pp. 1587-1596: PMLR.
[27] B. Li, Z. Gan, D. Chen, and D. Sergey Aleksandrovich, "UAV Maneuvering Target Tracking in Uncertain Environments Based on Deep Reinforcement Learning and Meta-Learning," Remote Sensing, vol. 12, no. 22, p. 3789, 2020.
[28] C.-H. Pi, K.-C. Hu, S. Cheng, and I.-C. Wu, "Low-level autonomous control and tracking of quadrotor using reinforcement learning," Control Engineering Practice, vol. 95, p. 104222, 2020.
[29] W. Du, T. Guo, J. Chen, B. Li, G. Zhu, and X. Cao, "Cooperative pursuit of unauthorized UAVs in urban airspace via Multi-agent

reinforcement learning," Transportation Research Part C: Emerging Technologies, vol. 128, p. 103122, 2021.
[30] T. Wang, R. Qin, Y. Chen, H. Snoussi, and C. Choi, "A reinforcement learning approach for UAV target searching and tracking," Multimedia Tools Applications, vol. 78, no. 4, pp. 4347-4364, 2019.
[31] W. Zhang, K. Song, X. Rong, and Y. Li, "Coarse-to-fine uav target tracking with deep reinforcement learning," IEEE Transactions on Automation Science Engineering, vol. 16, no. 4, pp. 1522-1530, 2018.
[32] Y. Xu, Z. Liu, and X. Wang, "Monocular Vision based Autonomous Landing of Quadrotor through Deep Reinforcement Learning," in 2018 37th Chinese Control Conference (CCC), 2018, pp. 10014-10019: IEEE.
[33] R. Polvara, S. Sharma, J. Wan, A. Manning, and R. Sutton, "Autonomous Vehicular Landings on the Deck of an Unmanned Surface Vehicle using Deep Reinforcement Learning," 2019.
[34] A. Rodriguez-Ramos, C. Sampedro, H. Bavle, P. De La Puente, and P. Campoy, "A deep reinforcement learning strategy for UAV autonomous landing on a moving platform," Journal of Intelligent & Robotic Systems, vol. 93, no. 1-2, pp. 351-366, 2019.
[35] R. Polvara, M. Patacchiola, M. Hanheide, and G. Neumann, "Sim-to-Real quadrotor landing via sequential deep Q-Networks and domain randomization," Robotics, vol. 9, no. 1, p. 8, 2020.
[36] M. B. Vankadari, K. Das, C. Shinde, and S. Kumar, "A reinforcement learning approach for autonomous control and landing of a quadrotor," in 2018 International Conference on Unmanned Aircraft Systems (ICUAS), 2018, pp. 676-683: IEEE.
[37] R. Srivastava, R. Lima, K. Das, and A. Maity, "Least square policy iteration for ibvs based dynamic target tracking," in 2019 International Conference on Unmanned Aircraft Systems (ICUAS), 2019, pp. 1089-1098: IEEE.
[38] S. Choi, S. Kim, and H. J. Kim, "Inverse reinforcement learning control for trajectory tracking of a multirotor UAV," International Journal of Control, Automation Systems, vol. 15, no. 4, pp. 1826-1834, 2017.
[39] S. Bhagat and P. Sujit, "UAV Target Tracking in Urban Environments Using Deep Reinforcement Learning," in 2020 International Conference on Unmanned Aircraft Systems (ICUAS), 2020, pp. 694-701: IEEE.
[40] Y.-J. Chen, D.-K. Chang, and C. Zhang, "Autonomous tracking using a swarm of UAVs: A constrained multi-agent reinforcement learning approach," IEEE Transactions on Vehicular Technology, vol. 69, no. 11, pp. 13702-13717, 2020.
[41] B. Li, Z.-p. Yang, D.-q. Chen, S.-y. Liang, and H. Ma, "Maneuvering target tracking of UAV based on MN-DDPG and transfer learning," Defence Technology, vol. 17, no. 2, pp. 457-466, 2021.
[42] J. Xie, X. Peng, H. Wang, W. Niu, and X. Zheng, "UAV Autonomous Tracking and Landing Based on Deep Reinforcement Learning Strategy," Sensors, vol. 20, no. 19, p. 5630, 2020.
[43] J. Moon, S. Papaioannou, C. Laoudias, P. Kolios, and S. Kim, "Deep Reinforcement Learning Multi-UAV Trajectory Control for Target Tracking," IEEE Internet of Things Journal, 2021.
[44] S. Lee, T. Shim, S. Kim, J. Park, K. Hong, and H. Bang, "Vision-based autonomous landing of a multi-copter unmanned aerial vehicle using reinforcement learning," in 2018 International Conference on Unmanned Aircraft Systems (ICUAS), 2018, pp. 108-114: IEEE.
[45] A. Rodriguez-Ramos, C. Sampedro, H. Bavle, P. De La Puente, and P. Campoy, "A deep reinforcement learning strategy for UAV autonomous landing on a moving platform," Journal of Intelligent Robotic Systems, vol. 93, no. 1-2, pp. 351-366, 2019.
[46] S. Choi, S. Kim, and H. J. Kim, "Inverse reinforcement learning control for trajectory tracking of a multirotor UAV," International Journal of Control, Automation Systems, vol. 15, no. 4, pp. 1826-1834, 2017.