International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. , No. (Mon-20..)

http://dx.doi.org/10.12785/ijcds/XXXXXX

Collaborative Multi-Agent Deep Reinforcement Learning Approach for Enhanced Attitude Control in Quadrotors

Taha Yacine Trad[1], Kheireddine Choutri[1], Mohand Lagha[1] and Fouad Khenfri[2]

[1]*Aeronautical Sciences Laboratory, Aeronautical and Spatial Studies Institute, Blida 1 University, Blida, Algeria*
[2]*Energy and Embedded Systems for Transport, ESTACA'Lab, Laval, France*
E-mail address: Trad.tahayacine@etu.univ-blida.dz, choutri.kheireddine@univ-blida.dz, laghamohand@univ-blida.dz, Fouad.KHENFRI@estaca.fr

Received Mon. 20, Revised Mon. 20, Accepted Mon. 20, Published Mon. 20

Abstract: Unmanned Aerial Vehicles (UAVs), particularly quadrotors, have become highly versatile platforms for various applications and missions. In this study, the employment of Multi-Agent Reinforcement Learning (MARL) in quadrotor control systems is investigated, expanding its conventional usage beyond multi-UAV path planning and obstacle avoidance tasks. While traditional single-agent control techniques face limitations in effectively managing the coupled dynamics associated with attitude control, especially when exposed to complex scenarios and trajectories, this paper presents a novel method to enhance the adaptability and generalization capabilities of Reinforcement Learning (RL) low-level control agents in quadrotors. We propose a framework consisting of collaborative MARL to control the *Roll*, *Pitch*, and *Yaw* of the quadrotor, aiming to stabilize the system and efficiently track various predefined trajectories. Along with the overall system architecture of the MARL-based attitude control system, we elucidate the training framework, collaborative interactions among agents, neural network structures, and reward functions implemented. While experimental validation is pending, theoretical analyses and simulations illustrate the envisioned benefits of employing MARL for quadrotor control in terms of stability, responsiveness, and adaptability. Central to our approach is the employment of multiple actor-critic algorithms within the proposed control architecture, and through a comparative study, we evaluate the performance of the advocated technique against a single-agent RL controller and established linear and nonlinear methodologies, including Proportional-Integral-Derivative (PID) and Backstepping control, highlighting the advantages of collaborative intelligence in enhancing quadrotor control in complex environments.

Keywords: Quadrotors, Attitude Control, Multi-Agent Deep Reinforcement Learning, Collaborative Intelligence.

1. INTRODUCTION

Owing to their unparalleled agility and versatility in navigating complex environments, quadrotors have become increasingly an indispensable tools across a myriad of applications. Precise control of a quadrotor's attitude comprising roll, pitch, and yaw is fundamental to achieve stable and responsive flight operations [1]. However, the intricate dynamics of quadrotors, marked by non-linearity, underactuation, and coupling, present notable challenges in control system design. Consequently, researchers have proposed and developed various control theories and techniques to address these challenges.

In the realm of control for quadrotors, various methodologies and techniques have been explored. Linear methods, such as PID [2], LQR [3], and Model Predictive Control [4], offer simplicity and ease of implementation but are limited in their operational scope. To overcome these limitations, more complex nonlinear controllers have been introduced, including Sliding Mode Control [5], Backstepping [6], Adaptive control [7], and H_∞ control [8]. The efficacy of traditional control algorithms often depends on subjective parameter choices, which depend on a comprehensive understanding of the model and experimental context. Balancing accuracy, robustness, and efficiency within a single control function becomes notably challenging in complex scenarios. Deep reinforcement learning (DRL), enabled by advancements in computing power and data accessibility, has emerged as a powerful approach within control theory. It demonstrates remarkable advantages across diverse tasks and applications, often surpassing conventional control methods by autonomously learning control policies directly from interaction with the environment, without relying on explicit models. RL algorithms can adapt to diverse systems by continuously updating their policies based on feedback received from the environment, allowing to handle uncertain and dynamic environments where traditional control methods may struggle. By leveraging reinforcement learning techniques, quadrotors can autonomously learn optimal control strategies through interaction with their environment, thereby improving their performance in tasks such as stabilization, trajectory tracking, and obstacle avoidance.

Building on the success of single-agent RL techniques, Multi-Agent Reinforcement Learning (MARL) has become a promising approach for addressing complex control problems, leveraging the adaptability and learning capabilities of intelligent agents. MARL enables multiple agents to interact and collaborate to achieve common goals, making it particularly suitable for scenarios involving coordination and cooperation. By harnessing the power of MARL in collaborative decision-making processes among UAVs, researchers and control engineers have recently focused on developing innovative and sophisticated control strategies that can enhance the stability, responsiveness, and adaptability of quadrotor systems in various aspects including Multi-UAVs path planing, collision and obstacle avoidance tasks [9].

In this study, we introduce an innovative MARL based approach for quadrotor attitude control, specifically focusing on multiple baseline actor-critic RL algorithms, the Twin Delayed Deep Deterministic Policy Gradient (TD3), the Deep Deterministic Policy Gradient (DDPG), the Soft Actor Critic (SAC) and the Proximal Policy Optimization (PPO) algorithms in a collaborative multi-Agent framework. This approach marks a departure from traditional control paradigms by leveraging collaborative intelligence to optimize control strategies and expand the scope of MARL beyond multi-UAV path planning and obstacle avoidance scenarios, demonstrating its potential to revolutionize quadrotor's low level control system. Through theoretical analyses and simulations, the proposed approach aims to validate the envisioned benefits of employing MARL in UAV systems, paving the way for innovative control capabilities and robustness.

The subsequent sections of this paper delve first into the related works and background of MARL's theoretical foundation, along with the quadrotor's dynamic model and control. We then present the architecture of the Multi-agent based attitude control system, detailing the employed framework, neural network structure and the reward functions. Finally, we present simulation results and performance comparisons against single-agent RL, linear and nonlinear benchmark control techniques including the PID and Backstepping control to validate the effectiveness of the proposed method in achieving stable and precise quadrotor attitude control across various scenarios.

2. BACKGROUND

A. *Related works*

In this section, we explore the extensive body of literature surrounding MARL, starting with existing reviews and recent algorithmic advancements to provide an insightful overview of the methodologies, insights, and trends in MARL research. We then focus on the application of MARL in UAV domains, highlighting novel breakthroughs and promising directions for future exploration.

Firstly, We have selected several reviews that offer comprehensive insights into the MARL current state of research. The study in [10] underscores the significance of MARL in multi-robot systems, emphasizing collaborative learning. It identifies a scarcity of recent surveys in the field and discusses challenges while proposing future applications for enhanced multi-robot systems. In [11], the authors explored recent advances in the algorithms employed in MARL, concentrating on five key approaches for addressing cooperative multi-agent problems. It offers detailed explanations, discusses challenges, and underscores connections among various papers. Additionally, the article covers emerging research areas, real-world applications, and MARL research environments, while in [12], the paper provides an overview divided into three main parts, it first examines training schemes for multiple agents, followed by an analysis of agent behavioral patterns in cooperative, competitive, and blended scenarios. The third part addresses challenges unique to the multi-agent domain and reviews methods used to address them. Moreover, in [13] a thorough analysis of multi-agent reinforcement learning algorithms is discussed, categorizing them based on features and offering a detailed taxonomy. It explores application fields, pros and cons, and compares algorithms in terms of nonstationarity, scalability, and observability, while discussing common benchmark environments.

The landscape of Multi-Agent Reinforcement Learning has witnessed remarkable advancements recently, marked by the introduction of innovative algorithms designed to tackle the complexities of cooperative and competitive multi-agent environments. However, due to the vast array of MARL algorithms available, only a select few will be presented. In [14], this study addresses the challenge of state uncertainty in practical multi-agent reinforcement learning (MARL) implementations, introducing the Markov Game with State Perturbation Adversaries (MG-SPA) model and proposing a robust multi-agent Q-learning (RMAQ) algorithm with convergence guarantees, along with a robust multi-agent actor-critic (RMAAC) algorithm, demonstrating their efficacy in handling high-dimensional state-action spaces and outperforming existing methods in scenarios with state uncertainty. In [15], the authors introduced RACE, a hybrid framework combining Evolutionary Algorithm (EA) and MARL to address challenges in collaboration, low-quality reward signals, and high non-stationarity, achieving improved convergence, robustness, and signal quality in various tasks compared to other algorithms. Also the authors of [16], introduces a distributed zeroth-order policy optimization method for Multi-Agent Reinforcement Learning that enables agents to compute local policy gradients using only partial state and action data, reducing communication overhead and improving learning performance, with numerical experiments demonstrating enhanced sample efficiency against the existing one-point estimators. In [17], this paper addresses the computational inefficiency in Population-based Multi-Agent Reinforcement Learning (PB-MARL) by proposing a solution that employs a stateless central task dispatcher and stateful workers, facilitating parallelism and efficient problem-solving. The proposed

framework, MALib, integrates a task control model, independent data servers, and a streamlined representation of MARL training methods, offering enhanced computational efficiency.

In the UAVs field, MARL approaches have been applied in diverse ways, showcasing their versatility and effectiveness in addressing a wide range of challenges and objectives. The authors in [18] presented a novel method to maximize data offloading efficiency from terrestrial base stations (BSs) using multiple unmanned aerial vehicles (UAVs). By jointly optimizing UAV trajectories and user association indicators under quality-of-service (QoS) constraints, the method aims to enhance user association with UAVs. Leveraging multi-agent reinforcement learning, each UAV operates independently, fostering cooperative behavior among them. Extensive simulations validate the effectiveness of the proposed technique, showing higher performance than both Q-learning and particle swarm optimization. In the same context, another paper [19] also introduces a novel approach to UAV cellular communication using multi-agent learning techniques enabling multiple UAVs to learn from each other through communication and interaction with the environment, providing better coverage compared to conventional terrestrial base station (BS) deployment. Also in [20], the authors introduce a novel graph-attention multi-agent trust region (GA-MATR) reinforcement learning framework to address the multi-UAV assisted communication problem. Utilizing multiple UAVs to maximize data offloading efficiency from terrestrial BSs by jointly optimizing UAV trajectories and user association indicators under QoS constraints, validated through simulations and demonstrating superior performance over benchmark techniques.

Another aspect in the UAVs domain that has been explored using MARL, as described in [21], a paper that introduces a UAV-aided mobile edge computing (MEC) framework employing a multi-agent deep reinforcement learning algorithm, specifically Multi-Agent Deep Deterministic Policy Gradient (MADDPG), to optimize geographical fairness, UAV UE-load fairness, and overall energy consumption for user equipments (UEs), showcasing superior performance compared to traditional algorithms. In [22], the study investigates dynamic resource allocation in UAV communication networks, employing a multi-agent reinforcement learning framework that optimizes long-term rewards without inter-UAV information exchange, demonstrating enhanced performance with balanced exploration and exploitation parameters. The proposed approach achieves a favorable balance between performance improvements and the overhead of exchanging information, as opposed to a scenario where UAVs exchange complete information.

Several papers incorporating MARL approaches have recently emerged on the topics of path following and swarm formations, as in [23], where the authors introduce a decentralized Multi-Agent Deep Reinforcement Learning (MADRL) method using maximum reciprocal reward, leveraging Pointwise Mutual Information (PMI) neural network to capture dependencies among UAVs, and proposing the Reciprocal Reward Multi-Agent Actor-Critic (MAAC-R) algorithm for cooperative tracking policies in UAV swarms, demonstrating enhanced cooperation and scalability in unknown environments compared to baseline algorithms. Also in [24], the authors present a model for cooperative air combat maneuvers involving multiple UAVs based on bidirectional recurrent neural networks (BRNN) and actor-critic architecture under the framework of multi-agent reinforcement learning, demonstrating the model's effectiveness in achieving cooperative tactical maneuver policies that provide UAVs with situational advantages and tactical success in air combat scenarios. Another study [25], that presents an autonomous tracking system for a UAV swarm to localize a radio frequency (RF) mobile target, utilizing omnidirectional received signal strength (RSS) sensors and an enhanced multi-agent reinforcement learning technique to optimize real-time target tracking, demonstrating superior performance in searching time and successful localization probability compared to standard Q-learning and multi-agent Q-learning algorithms.

While most studies have predominantly focused on applying Multi-Agent Reinforcement Learning approaches in swarm intelligence, path planning, collision avoidance, task allocation, communication networks, and target tracking, a gap in the exploration of MARL's potential in low-level control systems has not been investigated, particularly in the domain of quadrotor flight. To the best of our knowledge, MARL has primarily been utilized in high-level control strategies within swarm and multi-UAV contexts, neglecting its potential to revolutionize the control of individual quadrotors. By extending its traditional usage and exploring the application of collaborative intelligent agents in enhancing quadrotor attitude control systems, in this paper we aim to unlock new possibilities to enhance stability, maneuverability, and responsiveness at the fundamental level of quadrotor flight.

B. Quadrotor dynamics

The dynamics of the quadrotor UAV model as shown in Figure 1 are crucial for understanding its behavior and control, and can be effectively described using the Euler-Lagrangian formulation. This formulation provides a systematic approach to set the equations of motion governing the quadrotor's rotational and translational motion [26].

The translational motion of a quadrotor along the x, y and z axes is influenced by the forces generated by its propellers. Considering the quadrotor as a rigid body, the equations of motion can be expressed as:

$$\begin{cases} \ddot{x} = \frac{U_1}{m}(cos(\varphi)sin(\theta)cos(\psi) + sin(\varphi)sin(\psi)) \\ \ddot{y} = \frac{U_1}{m}(cos(\varphi)sin(\theta)sin(\psi) - sin(\varphi)cos(\psi)) \\ \ddot{z} = \frac{U_1}{m}(cos(\varphi)cos(\theta)) - g \end{cases} \quad (1)$$

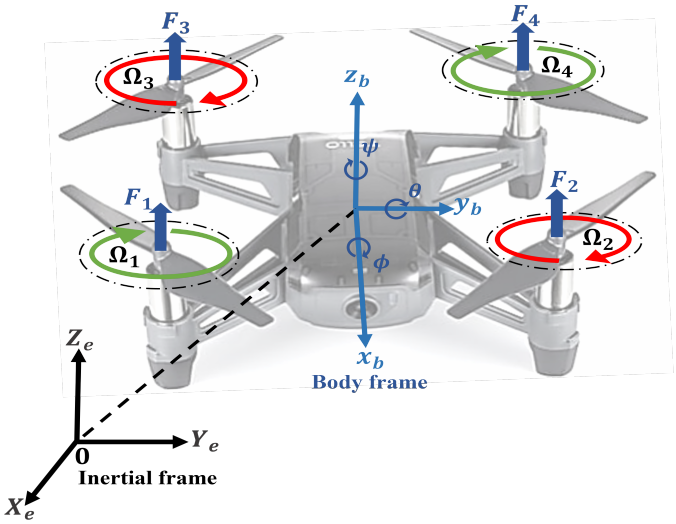

Figure 1. Quadrotor body and inertial frames.

The rotational motion of the quadrotor involves its angular velocities about the body-fixed axes. The moments induced by the propellers induce rotational acceleration, which can be described as:

$$\begin{cases} \ddot{\varphi} = \frac{-(J_{zz}-J_{yy})\dot{\theta}\dot{\psi}+U_2}{J_{xx}} \\ \ddot{\theta} = \frac{-(J_{zz}-J_{xx})\dot{\phi}\dot{\psi}+U_3}{J_{yy}} \\ \ddot{\psi} = \frac{-(J_{yy}-J_{xx})\dot{\phi}\dot{\theta}+U_4}{J_{zz}} \end{cases} \tag{2}$$

By deriving and solving these equations, one can gain insights into the quadrotor's behavior and design control strategies for achieving desired maneuvers and stability across the following control inputs:

$$\begin{cases} U_1 = b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\ U_2 = lb(\Omega_4^2 - \Omega_2^2) \\ U_3 = lb(\Omega_3^2 - \Omega_1^2) \\ U_4 = d(\Omega_1^2 - \Omega_2^2 + \Omega_3^2 - \Omega_4^2) \end{cases} \tag{3}$$

Where the symbols used: m, g, and l represent the quadrotor's mass, gravitational acceleration, and half-length respectively. J_{xx}, I_{yy}, and I_{zz} denote inertial tensor of the symmetric body around x, y and z axis. Additionally, d and b are the drag and thrust factors, Ω_i correspondingly refer to the the speed of each rotor i and U_i represent respectively the lift force, the moments of roll, pitch and yaw.

C. Multi-agent Reinforcement learning Formulation

Multi-Agent Reinforcement Learning (MARL) extends the success of single-agent reinforcement learning to scenarios involving more than one autonomous entity interacting in shared environments. Unlike single-agent RL, MARL introduces challenges related to non-stationarity and the combinatorial nature of interactions among agents. Agents in a multi-agent system must learn policies while adapting to the dynamic behaviors of others, adding complexity to the learning process. This collaborative or competitive aspect necessitates advanced algorithms capable of handling the intricacies and uncertainties inherent in multi-agent environments.

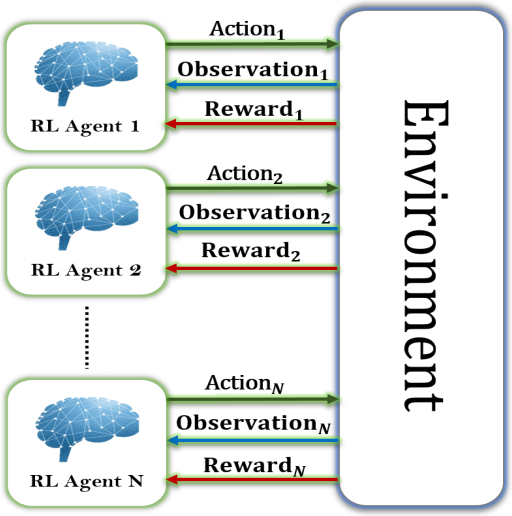

Figure 2. MARL framework.

MARL approach aims to enhance the collective reward achieved by a group of agents and can be represented as a Markov game involving a community of N agents (Figure 2). Markov Decision Processes (MDPs) are characterized by sequential decision-making, where actions impact both the immediate rewards of the agent and the future states of the environment. The state variable, $s \in S$, represents the environment's current state. For each agent $i = \{1, ..., N\}$, the action and observation spaces are presented respectively with $A_i \in A = \{A_1, ..., A_N\}$ and $O_i \in O = \{O_1, ..., O_N\}$. At each time step t, given a state s_t, agent i receives a local observation o_i and subsequently, interacts with the environment following a random policy denoted as π_{ϕ^i}, by taking action $a_t \in A$ and receiving immediate reward R_{t+1}, the environment moves from the current state $s_t \in S$ to the next state $s_{t+1} \in S$. Correlating with the action and the state, the rewards attributed to independent agent i are denoted by r_i. The objective of each agent is to maximize the expected return $R_i = \sum_{t=0}^{T} \gamma^t r_i^t$, where T is the final time step, and γ is a discount factor.

Similar to the case of a single agent, the ability to acquire the optimal stochastic policy or the optimal Q-value. However, because of the dynamic nature of each agent's policies during training, the environment undergoes non-stationarity from the viewpoint of any independent agent. In essence, $P(s_{t+1}|s_t, a_i, \pi_{\phi^1}, ..., \pi_{\phi^N}) \neq P(s_{t+1}|s_t, a_i, \pi'_{\phi^1}, ..., \pi'_{\phi^N})$ with any $\pi_{\phi^i} \neq \pi'_{\phi^i}$, causing the loss of the underlying

assumption of a Markov Decision Process. This suggests that each agent's experience encompasses varied co-player policies, preventing the fixing of these policies for training an agent. Therefore, any endeavor to train such models leads to fluctuations during the training procedure, posing a significant challenge in model training.

To address this challenge, a commonly employed strategy is the utilization of a fully observable critic. This approach involves incorporating the actions and observations of all agents into the critic, ensuring the environment's stability despite alterations in the policies of other agents. In simpler terms, even if $\pi_{\phi^i} \neq \pi'_{\phi^i}$, the probability of transitioning to a new state remains equal when other agents alter their policies. This approach allows for the creation of either one central critic in fully cooperative scenarios or N critic models when each agent observes a local reward. In both cases, the fully observable critic resolves the non-stationarity issue, serving as an effective guide for local actors.

3. MARL CONTROL SYSTEM

In this section, we delve into the intricacies of the quadrotor control system, explore the innovative approach for quadrotor attitude control, and examine the neural network architecture and reward function employed in this work.

A. *MARL attitude control system*

The quadrotor, emblematic of under-actuated systems, intricately balances its movement across six degrees of freedom using only four control inputs. While the high-level controller orchestrates the quadrotor's position and altitude, it is the low-level controller that serves as the backbone of precise maneuvering and stabilization.

At the heart of the low-level controller lies a network of RL agents trained in a full cooperation configuration to maintain a steady orientation, ensuring that the quadrotor's roll (φ), pitch (θ), and yaw (ψ) angles remain consistent amidst dynamic environmental conditions by adeptly interpret input commands (U_2, U_3, U_4) to achieve precise adjustments in the quadrotor's moments. These adjustments are calibrated to synchronize with the high-level controller's directives, allowing for seamless integration between attitude control and overall navigation as illustrated in Figure 3. Applying this approach is pivotal in achieving stability and precision in flight and overcoming the coupled dynamics issue of the quadrotor system.

For this critical task that necessitates policies with high-dimensional and continuous action space capabilities, we have selected the Twin Delayed Deep Deterministic Policy Gradient (TD3), Deep Deterministic Policy Gradient (DDPG), Soft Actor-Critic (SAC), and Proximal Policy Optimization (PPO) algorithms. Each of these algorithms brings unique advantages to the table. TD3 is known for its stability and performance in continuous control tasks, making it well-suited for precise attitude control in dynamic

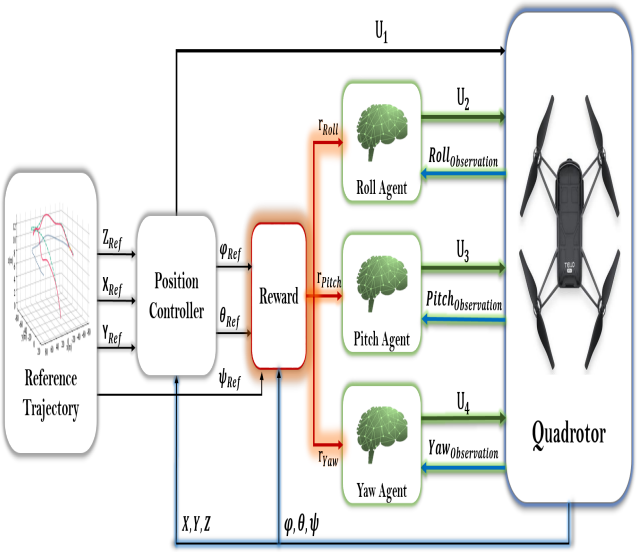

Figure 3. Multi-agent attitude control system.

environments. DDPG, on the other hand, offers efficient exploration capabilities, allowing the agents to learn robust policies even in high-dimensional action spaces. SAC stands out for its ability to handle continuous action spaces with stochastic policies, providing flexibility and adaptability in uncertain conditions. Finally, PPO offers a balance between sample efficiency and simplicity, making it suitable for online learning scenarios where computational resources are limited. By leveraging the strengths of these algorithms in a cooperative formation, the proposed network can effectively maintain a steady orientation under various conditions, ensuring the stability and reliability of the advocated control system.

B. *MATD3 algorithm*

In this section, we will elaborate a detailed explanation of the MATD3 attitude controller. While the overall framework applies to all configurations using the same reward function and neural networks architectures, including MADDPG, MASAC, and MAPPO.

In a multi agent environment, the authors of [27], present a model-free multi-agent reinforcement RL designed to address scenarios where agent i, during the time step t, uses its individual local observation o_i, actions a_i, and rewards r_i. Their approach encompasses competitive, cooperative, and mixed cooperative and competitive games. They introduce the Multi-agent Deep Deterministic Policy Gradient (MADDPG) algorithm, where each agent undergoes training with a DDPG algorithm. In this setup, the actor $\pi_{\phi^i}(o_i; \phi^i)$, characterized by policy weights ϕ^i, processes local observations. Meanwhile, the critic $Q^{\mu}_{\theta^i}$ (with θ^i represent its weights) has access to the actions, observations, and target policies of all agents during the learning phase. Subsequently, each agent's critic concatenates all state-actions as input and, utilizing the local reward, calculates the corresponding Q-value. The

training process involves optimizing for each of the N critics the following loss function:

$$L(\mu_{\theta^i}) = \mathbb{E}_{o^t,a,r,o^{t+1}}[(Q_{\theta^i}(s^t, a_1^t, ..., a_N^t; \mu_{\theta^i}) - y)^2] \quad (4)$$

with:

$$y = r_i^t + \gamma Q_{\theta^i}(o^t, \widetilde{a}_1^{t+1}, ..., \widetilde{a}_N^{t+1}; \widetilde{\mu}_{\theta^i})|_{\widetilde{a}_j^{t+1}=\widetilde{\pi}(o_j^{t+1})} \quad (5)$$

where o^t represents the observation of all agents, $\widetilde{\mu}$ is the target critic and $\widetilde{\pi}$ is the target policy. Consequently, the critic of each agent operates within a stationary environment, requiring access only to local information.

Building upon the strengths of the TD3 algorithm and the need for effective multi-agent reinforcement learning, in [28] authors introduced the Multi-Agent TD3 (MATD3) algorithm, in witch it retains the utilization of twin Q-networks to enhance the deterministic policy in a multi-agent context. MATD3 extends this concept to accommodate the interactions and dependencies between multiple agents, each agent denoted as agent i, maintains its own actor $\pi_{\phi^i}(o_i; \phi^i)$ with policy weights ϕ^i and observing its local environment.

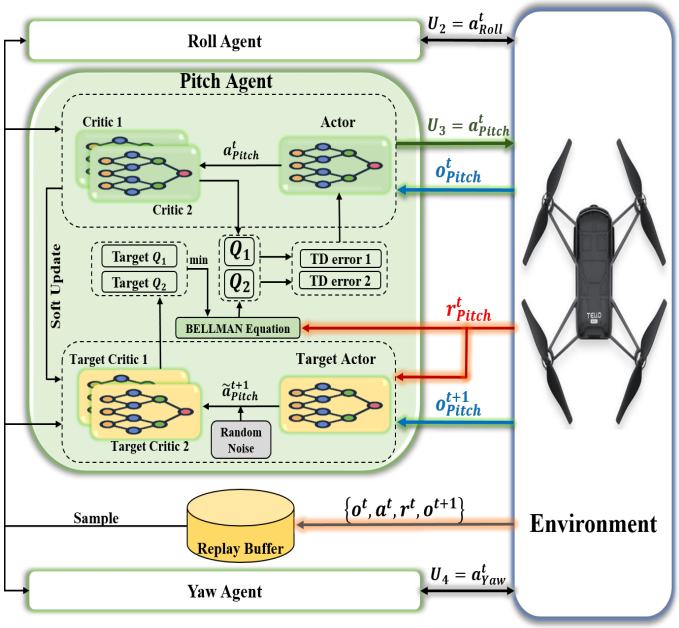

Figure 4. Multi-agent TD3 attitude control system.

The quadrotor's attitude control architecture presented in Figure 4 aims to enable each agent to understand the collective dynamics and decisions of the entire group. The critics of the *Roll*, *Pitch*, and *Yaw* agents collectively consider not only their individual local observations and actions but also integrate insights from the target policies, actions, and observations of all agents in the system. This fosters a cooperative learning process using a Centralized Training, Decentralized Execution (CTDE) mode. It is worth noting that the Centralized Training, Centralized Execution (CTCE) mode also can be employed for this particular MARL application, where the agents are not distributed entities and communication between them can be easily guaranteed. However, the size of the agent and precisely the actor network has to be relatively larger to accommodate centralized execution. This approach essentially converges to a single-agent configuration, where one RL agent generates the three control inputs U_2, U_3 and U_4.

Algorithm 1 presents the pseudo-code of the MATD3 algorithm tailored for addressing the quadrotors attitude control system where:

First, the actor and critic networks for each attitude agent (*Roll*, *Pitch* and *Yaw* agents) along with the replay buffer are initialized. At the start of each episode, a random noise for action exploration ϵ_i is generated, and the initial state is obtained encompassing the concerned Orientations, Orientation rates and Orientation errors with:

$$o = \{\phi, \dot{\phi}, \phi_{error}, \theta, \dot{\theta}, \theta_{error}, \psi, \dot{\psi}, \psi_{error}\} \quad (6)$$

Subsequently, episode iterations proceed, and each agent selects and execute the *Roll*, *Pitch* and *Yaw* moments $[U_2, U_3, U_4]$ according to its policy, and the total rewards are given as:

$$r_i = r_{global} - k sign(e_i.\dot{e_i})|_{i=\phi,\theta,\psi} \quad (7)$$

and:

$$r_{global} = -\alpha \sqrt{e_\phi^2 + e_\theta^2 + e_\psi^2} \quad (8)$$

With k and α positive weights, motivating the agents to minimize both its specific and global tracking errors. All the actions, rewards, and observations are stored as transition sets in the replay buffer R.

During this process, each agent randomly samples a small subset from the replay buffer to set the target value of the Q-function into y_i, using the minimum critic networks Q-value. Thereafter, the parameters θ_1^i and θ_2^i of the critic-networks are updated for the selected samples, along with the policy parameters ϕ^i of the actor-networks by maximizing the gradient ascent. Following the update of the critic and actor networks by each agent, the target-networks parameters $\theta_{n=1,2}^{i_{target}}$ and $\phi^{i_{target}}$ are adjusted to ensure learning stability by restricting the rate of update for the target values.

C. Network structure

All the agents employed are identical, with critic networks that comprises two pathways. The first path, taking the state as input, incorporates three feed-forward hidden layers, each consisting of 128, 128, and 64 neurons, respectively, and the output from this pathway is then merged with the action pathway, which contains a single hidden

Algorithm 1 MATD3 algorithm for the quadrotor attitude control system

for *each agent* i **do**
  Set actor network π_{ϕ^i} *and two critic networks* $Q_{\theta_1^i}, Q_{\theta_2^i}$ *with random weights* $\phi^i, \theta_1^i, \theta_2^i$ *respectively;*
  Set target networks $\phi^{i_{target}} \leftarrow \phi^i$, $\theta_1^{i_{target}} \leftarrow \theta_1^i$, $\theta_2^{i_{target}} \leftarrow \theta_2^i$;
  end for
Initialize the replay buffer R;
for $episode = 1, Max$ **do**
  Generate random noise: $\varepsilon_i \sim clip(\mathcal{N}(0, \widetilde{\sigma}), a_{i_{min}}, a_{i_{max}})$;
  Acquire original observations $o = \{o_{Roll}, o_{Pitch}, o_{Yaw}\}$;
  for $t = 1, max_{timesteps}$ **do**
    Select the control input from the action space of each agent: $a_i^t = \pi_{\phi^i}(o_i^t) + \varepsilon_i$;
    Execute the actions that represent the roll, pitch and yaw moments respectively:
      $a(t) = [a_1^t, a_2^t, a_3^t] = [U_2(t), U_3(t), U_4(t)]$
    Calculate the rewards r_i^t, *and observe* o_i^{t+1};
    Store transition $(o^t, a_1^t, a_2^t, a_3^t, r^t, o^{t+1})$ *in R;*
    Update the state $s \leftarrow s^{t+1}$;
    for *agent* $i = 1$ *to* 3 **do**
      Sample a random mini-batch of N transitions from R;
      Calculate target critic Q value:
        $y_i^t \leftarrow r_i^t + \gamma min[\ Q_{\theta_n^{itarget}}(o_{Roll}^{t+1}, o_{Pitch}^{t+1}, o_{Yaw}^{t+1}, \widetilde{a}_1^{t+1}, \widetilde{a}_2^{t+1}, \widetilde{a}_3^{t+1})]_{n=1,2}$
      Update critics:
        $\theta_1^i \leftarrow min_{\theta_1^i}\ \frac{1}{N} \sum [y_i^t - Q_{\theta 1}^i\ (o_{Roll}^t, o_{Pitch}^t, o_{Yaw}^t, \widetilde{a}_1^t, \widetilde{a}_2^t, \widetilde{a}_3^t)]^2$
        $\theta_2^i \leftarrow min_{\theta_2^i}\ \frac{1}{N} \sum [y_i^t - Q_{\theta 2}^i\ (o_{Roll}^t, o_{Pitch}^t, o_{Yaw}^t, \widetilde{a}_1^t, \widetilde{a}_2^t, \widetilde{a}_3^t)]^2$
      If t *mod* d **then**
        Update ϕ *using the DPG:*
        $\nabla_{\phi^i} J(\phi^i) = \frac{1}{N} \sum \nabla_{\phi_i} \pi_{\phi_i}(o_i^t) \nabla_{a_i} Q_{\theta^i}(o_{Roll}^t, o_{Pitch}^t, o_{Yaw}^t, a_1^t, a_2^t, a_3^t)|_{a_i^t = \pi_{\phi^i}(o_i^t)}$
        Update the target networks:
        $\theta_n^{i_{target}} \leftarrow \tau\theta_n^i + (1-\tau)\theta_n^{i_{target}}|_{n=1,2}$
        $\phi^{i_{target}} \leftarrow \tau\phi^i + (1-\tau)\phi^{i_{target}}$
      end If
    end for
  end for
end for

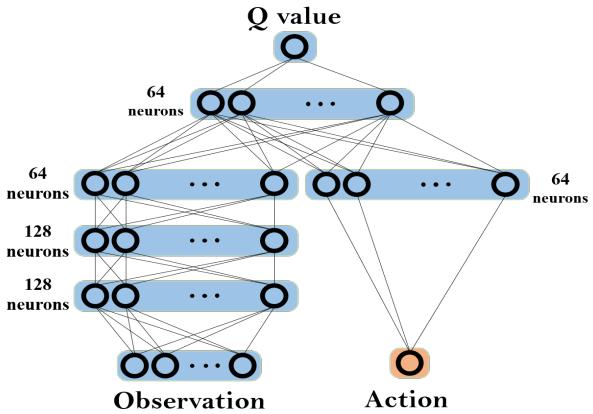

Figure 5. Critic network.

The actor network (Figure 6) for each agent is constructed with four feed-forward hidden layers, each containing 10 neurons.

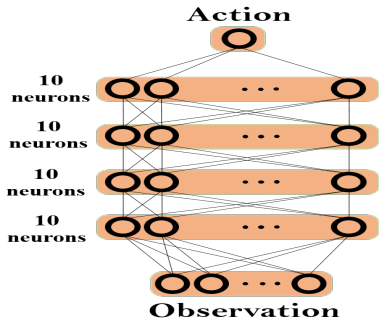

Figure 6. Actor network.

layer comprising 64 neurons. This integrated structure is responsible for generating the Q-value specific to each agent as highlighted in Figure 5.

With the exception of the action output layer that employs a *Sigmoid* activation function, all the layers use the *Rectified Linear Unit*.

4. Results and discussion

Using MATLAB software as a simulation platform, this section presents and analyzes the outcomes of the proposed approach for the quadrotor attitude control system. It assesses the efficacy of the top-performing agents in tracking various predefined trajectories with different levels of complexity and dynamics. Additionally, we conduct a comparative evaluation against conventional single-agent RL, PID, and Backstepping controllers to highlight the adaptability, stability, and precision of our MARL approach.

For the training setup, the objective is to stabilize the quadrotor from any given configuration within the physical limits using the Multi-Agent (MA) based controllers. The desired state is defined using reference Euler angles $\varphi_{Ref}, \theta_{Ref}, \psi_{Ref}$ to be tracked as shown in Figure 3. Three agents, trained through the approach detailed in section 3, ensure the quadrotor's adherence to the orientation commands set by the high-level controller and produce accurate control inputs U_1, U_2, and U_3.

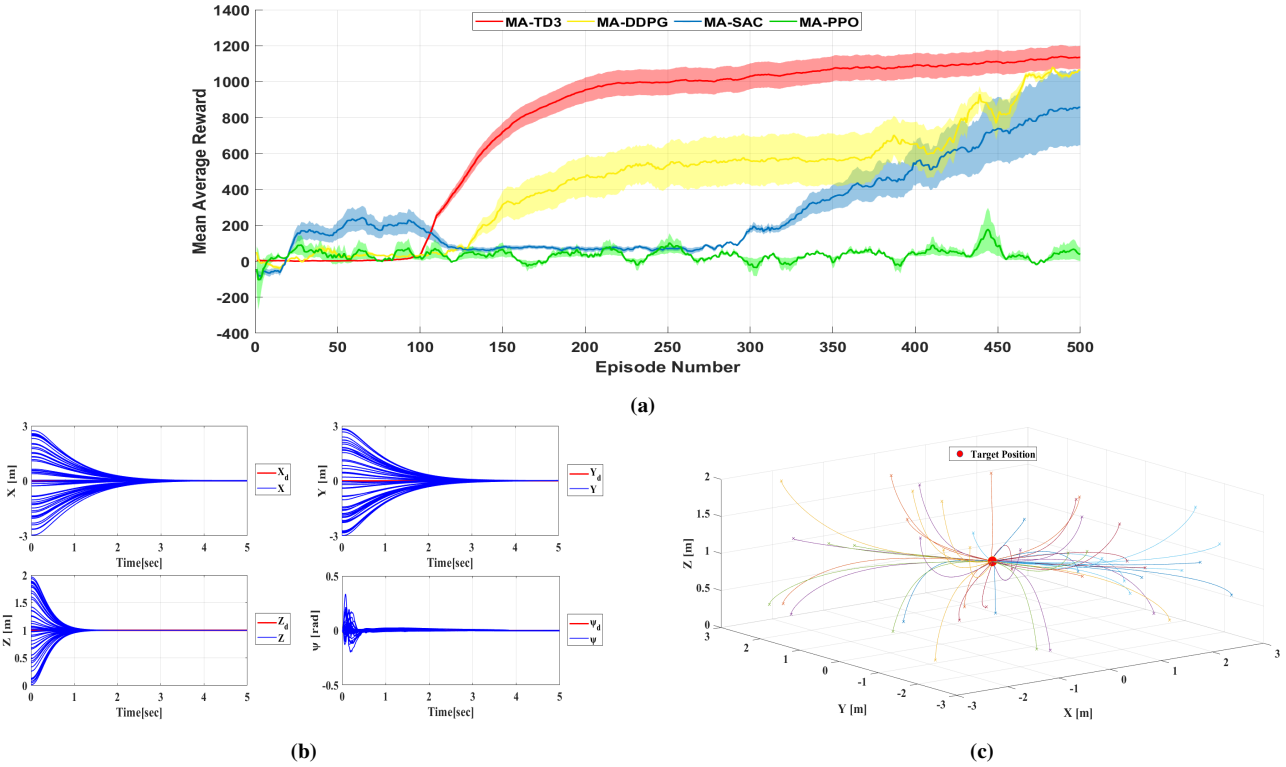

Figure 7. **(a)** Training sessions of the proposed MARL attitude control system, **(b)** MA-TD3 best agents performance for stabilisation task (50 random position and orientation initialisation), **(c)** 3D goal tracking from 50 random initialisation.

The training results summarised in Figure 7.a, shed light on the convergence and learning capabilities of various algorithms using the advocated framework. We conducted a performance comparison of the four algorithms MATD3, MADDPG, MASAC, and MAPPO, aimed to provide a comprehensive analysis of different approaches, showcasing the mean average reward collected by the *Roll*, *Pitch*, and *Yaw* agents over 500 training episodes.

- The MATD3, emerged as the most efficient and effective algorithm, demonstrating both speed and stability in training. The resulting policies exhibited rapid convergence and consistently achieved high rewards throughout the training sessions, the validation test of the obtained policies is shown in Figure 7.b and Figure 7.c, where the trained controllers handle perfectly the stabilisation of the system over 50 random position and orientation initialisation.
- MADDPG exhibited a two-phase training behavior, initially achieving stable learning with low collected rewards before eventually exploring more the action space and approximately attaining the MATD3's rewards level. While MADDPG's performance was inferior to MATD3, the delayed convergence suggests potential for improvement with longer training duration or additional fine-tuning.
- The MASAC demonstrated slower convergence but ultimately achieved higher rewards by the end of the training session. This behaviour may be attributed to its emphasis on entropy regularization, which encourages exploration and prevents premature convergence to suboptimal policies.

- While the other algorithms successfully converged to collaborative policies, MAPPO struggled to find effective cooperation for handling the quadrotor attitude control. Despite efforts to fine-tune hyperparameters and exploration strategies, MAPPO's performance remained subpar compared to the MATD3,MADDPG and MASAC. These challenges underscore the importance of selecting the right algorithm, especially for complex control tasks like quadrotor control systems.

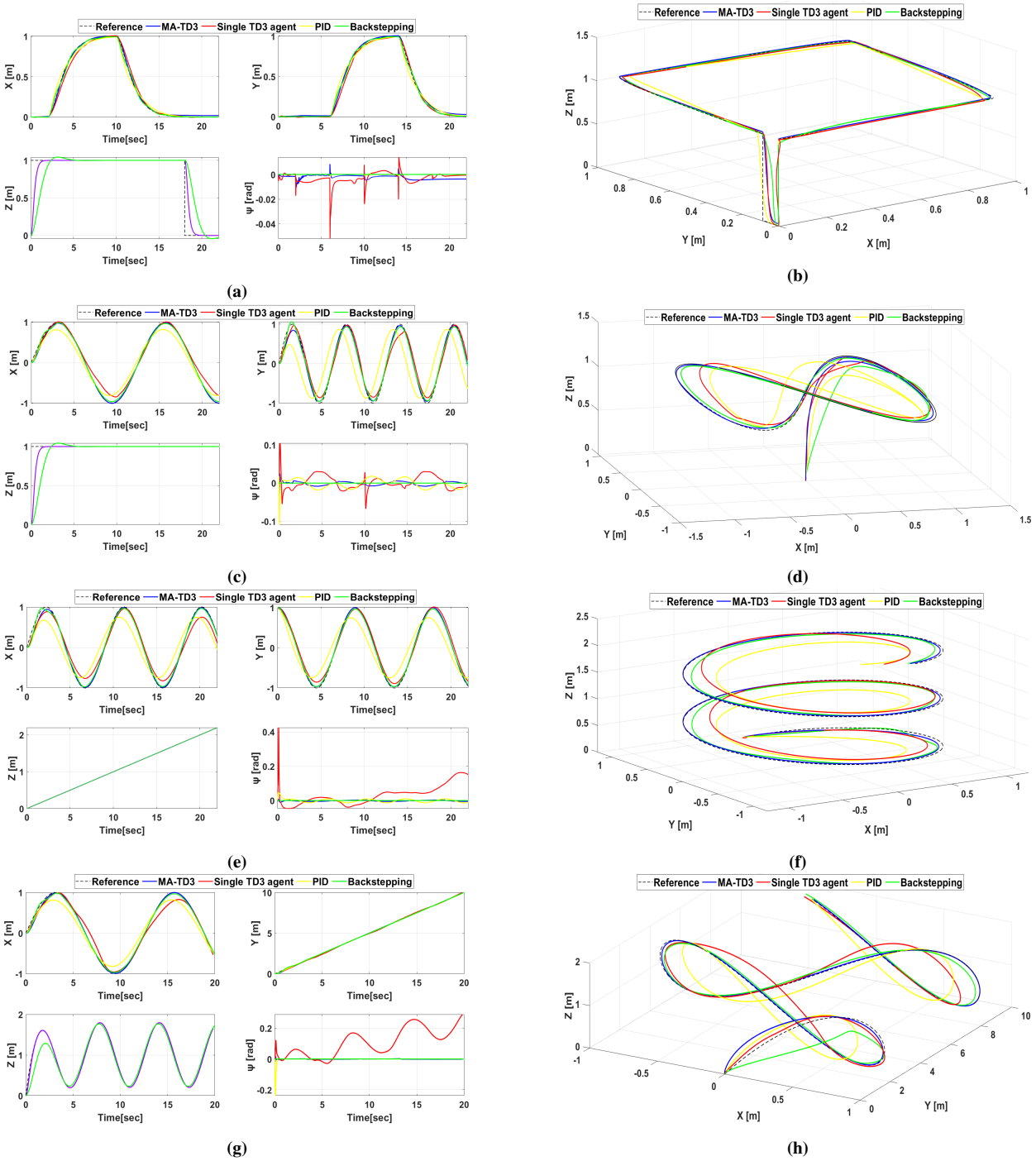

Figure 8. **(a)** The square trajectory position and orientation tracking, **(b)** 3D square trajectory, **(c)** The lemniscate trajectory position and orientation tracking, **(d)** 3D lemniscate trajectory, **(e)** The ellipsoid trajectory position and orientation tracking, **(f)** 3D ellipsoid trajectory,**(g)** An acrobatic trajectory position and orientation tracking, **(h)** 3D acrobatic trajectory.

TABLE I. Mean square error (MSE) on the paths displayed in Figure 8.

Trajectory	Controller	X	Y	Z	ψ
Square	MA-TD3	0.000171	0.0003026	0.02772	0.7982×10^{-5}
	Single TD3 agent	0.001385	0.0007229	0.02772	2.913×10^{-5}
	PID	0.001313	0.001313	0.02772	2.138×10^{-8}
	Backstepping	0.0002831	0.0003845	0.07093	4.418×10^{-40}
Lemniscate	MA-TD3	0.001046	0.004755	0.0134	2.247×10^{-5}
	Single TD3 agent	0.01358	0.01409	0.0134	33.59×10^{-5}
	PID	0.02287	0.2863	0.0134	0.0001551
	Backstepping	0.001209	0.004823	0.0343	2.18×10^{-42}
Ellipsoid	MA-TD3	0.002068	0.0003255	9.226×10^{-9}	1.256×10^{-5}
	Single TD3 agent	0.02165	0.006435	9.226×10^{-9}	0.00444
	PID	0.08643	0.07427	9.226×10^{-9}	0.0001013
	Backstepping	0.002616	0.001351	7.534×10^{-8}	1.111×10^{-42}
Acrobatic	MA-TD3	0.001194	0.001151	0.0005283	3.038×10^{-6}
	Single TD3 agent	0.0142	0.00359	0.0005283	0.01863
	PID	0.02408	0.0003556	0.0005283	0.0002218
	Backstepping	0.001221	0.005915	0.02669	5.986×10^{-42}

The efficiency of the MARL best trained agents is assessed using the Mean Square Error (MSE) metric (displayed in Table I with the lowest MSE recorded for each path is highlighted in green). Where the MSE of the Z position for the Backstepping controller has different values, as it is applied to the overall control system, all the other approaches are applied in low-level control. This comparison involved navigating predefined paths, including square, lamnestic, ellipsoid, and acrobatic trajectories, chosen for their varying complexity and dynamics, as shown in Figure 8.

The results demonstrated that the MATD3 controller consistently outperformed the other methods across all trajectories. Its ability to dynamically adjust policies in response to complex and varying flight conditions resulted in superior tracking precision and robustness. This was particularly evident in the more challenging lamnestic, ellipsoid, and acrobatic trajectories.

The Backstepping controller also showed strong performance especially for stabilising the yaw angle of the quadrotor. While it was effective and reliable in simpler trajectories like the square path, it lacked the dynamic adaptability seen in the MATD3 controller during more complex maneuvers. Nonetheless, the Backstepping approach demonstrated robust control capabilities, making it a viable option for many practical applications.

The single TD3 agent, showing decent performance in simpler trajectories that resemble the learning configuration. Its limited ability to generalize highlighted the need for further training and refinement to improve its adaptability and robustness.

The PID controller, while straightforward and easy to implement, it required meticulous parameter tuning to achieve enhanced performance, particularly in the more dynamic and demanding trajectories.

5. CONCLUSION AND FUTURE WORK

While the majority of recent studies focus on collaborative control strategies among multiple quadrotors using Multi-Agent Reinforcement Learning (MARL) techniques to accomplish tasks like formation flying, path planning, collision and obstacle avoidance, this study introduces a novel MARL approach for enhanced quadrotor attitude control. By employing MARL algorithms, including MATD3, MADDPG, MASAC, and MAPPO, we investigate the adaptability and performance of these methods compared to single-agent RL and various linear and non-linear controllers. Through extensive training and validation phases, we observed that while all controllers performed similarly in scenarios resembling the training configurations, the MARL approach demonstrated superior capabilities and robustness when navigating through complex trajectories such as lamnestic and ellipsoid paths. Specifically, the MATD3 that exhibited rapid and stable learning, outperforming other MARL algorithms and benchmark methodologies in terms of adaptability and tracking performance on paths featuring high-speed maneuvers and rapid altitude changes. These results underscore the potential of MARL techniques for real-world quadrotor applications, where precise and adaptive control in dynamic environments is critical.

In future work, we intend to assess the performance of the proposed approach in dynamic and unpredictable environments, to assess its robustness proprieties. Additionally, we plan to conduct experiments with the developed control technique on real-world quadrotor platforms to

validate their effectiveness and performance in practical applications.

References

[1] T. Y. Trad, K. Choutri, M. Lagha, R. Fareh, and M. Bettayeb, "Simulated annealing-deep deterministic policy gradient algorithm for quadrotor attitude control," in *2023 Advances in Science and Engineering Technology International Conferences (ASET)*. IEEE, 2023, pp. 1–6.

[2] S. Wang, A. Polyakov, and G. Zheng, "Quadrotor stabilization under time and space constraints using implicit pid controller," *Journal of the Franklin Institute*, vol. 359, no. 4, pp. 1505–1530, 2022.

[3] R. Thusoo, S. Jain, and S. Bangia, "Path planning of quadrotor using a* and lqr," in *Recent Developments in Electrical and Electronics Engineering: Select Proceedings of ICRDEEE 2022*. Springer, 2023, pp. 227–238.

[4] M. Okasha, J. Kralev, and M. Islam, "Design and experimental comparison of pid, lqr and mpc stabilizing controllers for parrot mambo mini-drone," *Aerospace*, vol. 9, no. 6, p. 298, 2022.

[5] A.-W. Saif, K. B. Gaufan, S. El-Ferik, and M. Al Dhaifallah, "Fractional order sliding mode control of quadrotor based on fractional order model," *IEEE Access*, 2023.

[6] J. Wang, K. A. Alattas, Y. Bouteraa, O. Mofid, and S. Mobayen, "Adaptive finite-time backstepping control tracker for quadrotor uav with model uncertainty and external disturbance," *Aerospace Science and Technology*, vol. 133, p. 108088, 2023.

[7] M. B. Artuc and I. Bayezit, "Robust adaptive quadrotor position tracking control for uncertain and fault conditions," *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, p. 09544100231181869, 2023.

[8] F. W. Alsaade, H. Jahanshahi, Q. Yao, M. S. Al-zahrani, and A. S. Alzahrani, "A new neural network-based optimal mixed h2/h control for a modified unmanned aerial vehicle subject to control input constraints," *Advances in Space Research*, vol. 71, no. 9, pp. 3631–3643, 2023.

[9] J. Panerati, H. Zheng, S. Zhou, J. Xu, A. Prorok, and A. P. Schoellig, "Learning to fly—a gym environment with pybullet physics for reinforcement learning of multi-agent quadcopter control," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 7512–7519.

[10] J. Orr and A. Dutta, "Multi-agent deep reinforcement learning for multi-robot applications: a survey," *Sensors*, vol. 23, no. 7, p. 3625, 2023.

[11] A. Oroojlooy and D. Hajinezhad, "A review of cooperative multi-agent deep reinforcement learning," *Applied Intelligence*, vol. 53, no. 11, pp. 13 677–13 722, 2023.

[12] S. Gronauer and K. Diepold, "Multi-agent deep reinforcement learning: a survey," *Artificial Intelligence Review*, pp. 1–49, 2022.

[13] L. Canese, G. C. Cardarilli, L. Di Nunzio, R. Fazzolari, D. Giardino, M. Re, and S. Spanò, "Multi-agent reinforcement learning: A review of challenges and applications," *Applied Sciences*, vol. 11, no. 11, p. 4948, 2021.

[14] S. He, S. Han, S. Su, S. Han, S. Zou, and F. Miao, "Robust multi-agent reinforcement learning with state uncertainty," *arXiv preprint arXiv:2307.16212*, 2023.

[15] P. Li, J. Hao, H. Tang, Y. Zheng, and X. Fu, "Race: improve multi-agent reinforcement learning with representation asymmetry and collaborative evolution," in *International Conference on Machine Learning*. PMLR, 2023, pp. 19 490–19 503.

[16] Y. Zhang and M. M. Zavlanos, "Cooperative multi-agent reinforcement learning with partial observations," *IEEE Transactions on Automatic Control*, 2023.

[17] M. Zhou, Z. Wan, H. Wang, M. Wen, R. Wu, Y. Wen, Y. Yang, Y. Yu, J. Wang, and W. Zhang, "Malib: A parallel framework for population-based multi-agent reinforcement learning," *Journal of Machine Learning Research*, vol. 24, no. 150, pp. 1–12, 2023.

[18] A. Mondal, D. Mishra, G. Prasad, G. C. Alexandropoulos, A. Alnahari, and R. Jantti, "Multi-agent reinforcement learning for offloading cellular communications with cooperating uavs," *arXiv preprint arXiv:2402.02957*, 2024.

[19] R. Sabogu-Sumah, K. A. Opare, J. D. D. Gadze, J. J. Kponyo, A.-R. Ahmed, E. Fianko *et al.*, "Optimal coverage enhancement for multiple uavs using multi-agent learning technique," *International Journal of Computing and Digital Systems*, vol. 13, no. 1, pp. 1–1, 2023.

[20] Z. Feng, D. Wu, M. Huang, and C. Yuen, "Graph attention-based reinforcement learning for trajectory design and resource assignment in multi-uav assisted communication," *arXiv preprint arXiv:2401.17880*, 2024.

[21] L. Wang, K. Wang, C. Pan, W. Xu, N. Aslam, and L. Hanzo, "Multi-agent deep reinforcement learning-based trajectory planning for multi-uav assisted mobile edge computing," *IEEE Transactions on Cognitive Communications and Networking*, vol. 7, no. 1, pp. 73–84, 2020.

[22] J. Cui, Y. Liu, and A. Nallanathan, "Multi-agent reinforcement learning-based resource allocation for uav networks," *IEEE Transactions on Wireless Communications*, vol. 19, no. 2, pp. 729–743, 2019.

[23] Z. Wenhong, L. Jie, L. Zhihong, and S. Lincheng, "Improving multi-target cooperative tracking guidance for uav swarms using multi-agent reinforcement learning," *Chinese Journal of Aeronautics*, vol. 35, no. 7, pp. 100–112, 2022.

[24] Z. Jiandong, Y. Qiming, S. Guoqing, L. Yi, and W. Yong, "Uav cooperative air combat maneuver decision based on multi-agent reinforcement learning," *Journal of Systems Engineering and Electronics*, vol. 32, no. 6, pp. 1421–1438, 2021.

[25] 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. 13 702–13 717, 2020.

[26] R. Benotsmane and J. Vásárhelyi, "Towards optimization of energy consumption of tello quad-rotor with mpc model implementation," *Energies*, vol. 15, no. 23, p. 9207, 2022.

[27] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. Pieter Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *Advances in neural information processing systems*, vol. 30, 2017.

[28] F. Zhang, J. Li, and Z. Li, "A td3-based multi-agent deep reinforcement learning method in mixed cooperation-competition environment," *Neurocomputing*, vol. 411, pp. 206–215, 2020.

TAHA YACINE TRAD was born in Setif, Algeria, on 27 September 1992. He received his Master degree in Aeronautical Engineering from the High School of Aeronautical Techniques of Dar-El-Beida, Algeria, in 2018 and started his doctoral studies in the field of Aerial Robots at the Institute of Aeronautics and Spatial Studies of SAAD DAHLAB Blida 1 University - Blida, Algeria in 2019. His current research interests include Autonomous Systems, Intelligent Decision-Making, Navigation and Control , Multi agents system.

Kheireddine CHOUTRI was born in Constantine, Algeria, on 26 February 1992. He received the Master degree in Aeronautical Engineering from the Aeronautics Institute of Blida, Algeria, in 2015. In this same year, he started his doctoral studies in the field of Aerial Robots in the Aeronautics and spatial studies Institute of SAAD DAHLAB Blida 1 University - Blida, Algeria. His current research activities include UAV Guidance, Navigation and Control , Multi agents system.

Mohand LAGHA was born in Tizi-ouzou, Algeria, on 30 June 1976. He received the Engineer Diploma in Aeronautical Engineering from the Aeronautics Institute of Blida, Algeria, in 2000, the M.Sc. in Aeronautics Sciences from the SAAD DAHLAB University of Blida, Algeria, in 2003. He received the Ph.D. degree (with honors) in Aeronautical Engineering at the Aeronautics Department of SAAD DAHLAB Blida University on 3rd July 2008, and the habilitation (HDR) on September 2010. At present he is Full Professor in Aeronautics and spatial studies Institute of SAAD DAHLAB Blida 1 University - Blida, Algeria. His current research activities include estimation theory, radar signal processing, weather radar signal analysis and UAVapplications.

Fouad Khenfri received an engineering degree from the University of Biskra, Biskra, Algeria, in 2008, and a M.Sc. degree in electrical and computer engineering from the Polytechnic School of Algiers, Algeria, in 2011, and a Ph.D. degree in computer science and automation from Nantes Central School (Ecole Centrale de Nantes), France, in 2016. He is currently an Assistant Professor at ESTACA (Ecole Superieure des Techniques Aeronautiques et de Construction Automobile), France, since 2016. His current research interests include system control, embedded software design, and embedded system performance analysis and optimization.