Kufa Journal of Engineering
Vol. 17, No. 3, July 2026, P.P. 467-483
Article history: Received 14 June 2025, last revised 12 October 2025,
accepted 28 October 2025

MEMORY-AUGMENTED REINFORCEMENT LEARNING FOR UAV NAVIGATION USING PPO-LSTM

Maryam Allawi Haddad[1] and Dhayaa Raissan Khudher[2]

[1] M.Sc. Student, Department of Computer Engineering, University of Basrah, Iraq, Email: pgs.maryam.allawi@uobasrah.edu.iq.

[2] Department of Computer Engineering, University of Basrah, Basrah, Iraq, Email: dhayaa.khudher@uobasrah.edu.iq.

https://doi.org/10.30572/2018/KJE/170327

ABSTRACT

The problems of partial observability and sensor shortage pose a significant challenge for autonomous Unmanned Aerial Vehicles (UAVs) as they prove to be challenging for conventional Deep Reinforcement Learning (DRL) methods to undertake well under such conditions. In this paper, a memory-augmented Proximal Policy Optimization (PPO) model extended using a Long Short-Term Memory (LSTM) network is proposed as a solution to such challenges. The observation space is constructed from 2D LiDAR and Inertial Measurement Unit (IMU) data to sense simultaneously external observation and internal state of motion, whereas the action space consists of continuous velocity commands. A shaped reward function is optimized for encouraging safe target approaching, obstacle avoidance, and convergence speed. Experimental outcomes show that the PPO-LSTM described herein achieves smoother paths, more robust reward convergence, and a much lower rate of collision than regular PPO. It also generalizes to new environments with movable obstacles. Qualitatively, the success rate increased from 64.5% to 83.9%, collision frequency reduced by over 70%, and path efficiency increased from 0.60 to 0.85, without suffering from unstable training behavior.

KEYWORDS

Autonomous drone navigation, obstacle avoidance, PPO, LSTM, and Partial observability.

1. INTRODUCTION

Recently, UAVs have gained significant attention in many applications such as mapping, surveillance, delivery, and infrastructure inspection. Consequently, developing a complete autonomous drone has become a pressing need. The current methods in this area utilize two primary approaches: model-based and model-free. The first approach considers Model Predictive Control (MPC), Potential Fields (Liu et al., 2024), and Vector Field Guidance (Wang et al., 2023),(Al-Hsnawy and Al-Ghanimi, 2024), which suffer from modelling difficulties and high computational demands. In contrast, model-free approaches, particularly Deep Reinforcement Learning (DRL), offer adaptability, scalability and are suited for real-world deployment(Al-Hsnawy and Al-Ghanimi, 2024). To overcome the challenges of conventional RL, recent research has combined deep learning with RL. (Ismael et al., 2025).

Among many DRL algorithms, PPO has emerged as a stable (Schulman et al., 2017), robust, and effective solution for continuous control tasks. However, its performance significantly depends on full observability of the environmental state. This requirement is often unattainable in real-world scenarios due to partial sensor coverage, occlusion, and temporal uncertainty. To overcome this challenge, LSTM and the PPO algorithms are combined to enhance decision-making in the presence of partial observability.

This work develops an end-to-end PPO with an LSTM framework for UAVs using LiDAR-based observations. The reward function is customized to promote smooth, safe, and efficient navigation behaviors. Further, our approach is generalized to an unseen environment with dynamic obstacles.

2. RELATED WORKS

Designing fine-grained UAV control strategies that support continuous action spaces, using DRL algorithms, has been widely adopted for UAV navigation RL. The synergy enhances the ability of autonomous agent to handle Current studies have integrated deep learning with RL to overcome the limitations of traditional high-dimensional sensor data (e.g., camera, lidar, and inertial measurements) and effectively interpreting complex environments. The principal role of the deep neural networks is replacing the tabular of RL, by approximating policy function, value function or other models related to the environment. This enables agents to learn policies directly from sensory inputs, which is an essential capability for autonomous navigation in unknown environments.

In the context of autonomous navigation, an agent must perceive its surroundings using various types of sensors (e.g., camera, Lidar, and IMU). Enhancing the perceptual capabilities of the agent can increase the dimensionality of the observation space. High-dimensional observation

spaces may negatively impact learning efficiency and policy convergence, particularly in dynamic environments.

Recent studies have utilized a camera to construct the observation space of the DRL. The performance of vision-based system methods can be affected by low-light conditions and motion blur. Conversely, the data captured by the LiDAR sensor from the environment is more structured and easier to use with DRL algorithms; its data is not affected by lighting conditions. Consequently, LiDAR is well-suited for navigation tasks in an environment with obstacles. This vision-based methods can be affected by issues such as low-light conditions, motion blur, and the significant computational cost. Moreover, they often require convolutional neural networks (CNNs) with large model sizes (Mansour et al., 2025). Consequently, the training stability is negatively affected and significantly increase convergence time. Using LiDAR as a sensory input to the DRL algorithm is effective to achieve faster convergence and success rates.

According to these trade-offs, this work adopts PPO as the core learning algorithm. PPO is used in many applications, such as corridor navigation, autonomous landing (Rodriguez-Ramos et al., 2019a), and dynamic obstacle avoidance. However, despite its empirical success, PPO has several known limitations. Due to its on-policy nature, PPO suffer from sampling inefficiency, a fixed clipping threshold in its surrogate loss, which may hinder optimal convergence, and a lack of temporal memory, which may lead to low effectiveness in partially observable environments. To address these limitations associated with the PPO algorithm, several augmentations have been introduced. The RE-PPO (Reward Engineering PPO) algorithm is introduced to improve sampling efficiency. PPO integrates LSTM networks into the policy architecture to allow the agent to better handle partial observability.

This work addresses these challenges by developing an end-to-end PPO with an LSTM framework for UAVs using LiDAR-based observations. We designed a customized reward function to promote smooth, safe, and efficient navigation behaviors.

3. BACKGROUND

3.1. Proximal Policy Optimization (PPO)

PPO is an RL technique that belongs to the policy gradient algorithm class of on-policy and actor-critic methods. It is widely recognized for its strong performance, training stability, and ease of implementation compared to other policy optimization techniques. In this algorithm, the policy (actor) is optimized using gradients estimated from data collected by interaction with the environment and the value function (critic) is used to reduce the variance.

In the traditional policy gradient methods, the advantage function $A^{\pi}(s, a)$ is defined in Eq.1, where s and a are the state and action, respectively.

$$A^{\pi}(s,a) = Q^{\pi}(s,a) - V^{\pi}(s) \quad (1)$$

Q^{π} is the action value function and V^{π} Is the state value function. This form of advantage function suffers from high variance and instability. The Generalized Advantage Estimation (GAE) method is introduced to reduce the variance in the advantage calculation using a weighted combination of residuals. δ_t , as in Eq.2.

$$\delta_t = r_t + \gamma V^{\pi}(s_{t+1}) - V^{\pi}(s_t) \quad (2)$$

Where r_t Is the received reward after executing an action, and γ is the discount factor. The GAE estimate is demonstrated in Eq.3, where λ is the smoothing parameter of GAE.

$$A_t^{GAE(\gamma,\lambda)} = \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_{t+l} \quad (3)$$

Eq.4 depicts the standard policy gradient objective (unclipped objective).

$$J(\theta) = \mathbb{E}_{\tau\sim\pi\theta}\left[\sum_{t=0}^{T} A^{\pi}(s_t,a_t)\log\pi\theta(a_t|s_t)\right] \quad (4)$$

The expectation $\mathbb{E}_{\tau\sim\pi\theta}$ Indicates that this process is repeated multiple times on sampled trajectories $\tau\sim\pi\theta$. Later, Trust Region Policy Optimization (TRPO) (Schulman et al., 2015) was introduced to constrain the policy update by limiting the change in policy distribution according to Eq.5.

$$J_{TRPO}(\theta) = \mathbb{E}_{s\sim\rho\theta\pi_{old},a\sim\pi\theta_{old}}\left[\frac{\pi\theta(a|s)}{\pi\theta_{old}(a|s)} A^{\pi\theta_{old}}(s,a)\right] \quad (5)$$

Subject to the constraint in Eq.6.

$$\mathbb{E}_{s\sim\rho\theta\pi_{old}}[D_{KL}(\pi\theta_{old}(.|s)||\pi\theta(.|s))] \le \delta \quad (6)$$

Where $\rho_{\theta_{old}}$ Is the state visitation distribution under the old policy and D_{KL} Kullback-Leibler divergence. This saves the new policy $\pi\theta$ from drastic change.

Although TRPO has significantly improved the policy stability, it involves complex second-order optimization and matrix inversions, leading to difficult implementation in practice.

PPO was developed to retain TRPO's policy stability and simplify the problem using first-order optimization. It uses the clipped surrogate objective:

$$J_{PPO}(\theta) = \mathbb{E}_t[\min(r_t(\theta)A_t, clip(r_t(\theta),1-\epsilon,1+\epsilon)A_t)] \quad (7)$$

Where:

$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ Is the sampling ratio.

A_t Is the estimated advantage function.

ϵ is a hyperparameter (typically 0.1–0.3) that controls the update range by clipping the ratio to ensure a small update.

Using the min function to keep the policy updates within a trusted region by clipping the objective in case the new policy moves too far from the old one. This retains both policy improvement and stability of TRPO without its complexity.

PPO also promote exploration and prevents premature convergence by incorporating value function loss and an entropy. The total PPO loss function becomes:

$$L_{total}(\theta) = L^{CLIP}(\theta) - c1 * L^{VF}(\theta) + c2 * H[\pi_\theta] \quad (8)$$

Where:

$L^{CLIP}(\theta)$ Is the clipped surrogate policy loss (to be minimized).

$L^{VF}(\theta)$ The value function loss.

$H[\pi_\theta]$ The entropy of the policy.

PPO can update both policy and value networks simultaneously to ensure stable and efficient learning.

The simplicity and robustness of PPO have made it one of the most commonly used DRL algorithms in robotics, control, and navigation tasks.

3.2. PPO and LSTM augmentation

Learning policy in any policy gradient algorithm depends mainly on the observation space provided by the environment. However, in many tasks particularly in UAV navigation—achieving full observability is challenging due to sensor limitations, occlusion, or processing delays (Hanover et al., 2023). Partial observation has a direct impact on optimal decision-making by the agent. The robustness of the PPO as an on-policy algorithm heavily depends on the assumption of near-full observability. In the real world, this is hard to meet; therefore, PPO may struggle because its policy update relies on precise state representations. This limitation highlights the need to use the past policy information to update the new policy. To mitigate this limitation, PPO integrate memory-based architecture, such as LSTM networks. This network enables PPO to infer missing information by maintaining a hidden state over time.

LSTM is a Recurrent Neural Network (RNN) that maintains an internal memory state to give the ability to process sequential information over time. This allows the agent to infer temporal dependencies and reconstruct latent information from past observations.

Integrating PPO and LSTM extends the policy from $\pi_\theta(s_t)$ to $\pi_\theta(s_t, h_t)$, where h_t It is the hidden state of the LSTM that summarizes the past. Similarly, the value function becomes $V_\theta(s_t, h_t)$. This modification allows the agent to learn temporally extended strategies and improve performance. These memory-driven control policies are essential for robust navigation in partially observable or sensor-limited environments.

4. METHODOLOGY

This section presents the methodological foundation for our proposed autonomous drone navigation system. We begin by formulating the problem as a Markov Decision Process (MDP), followed by an overview of the observation, action space, and reward function design. Then extend the discussion to PPO-LSTM to accommodate partial observability in dynamic environments and the neural network architecture.

4.1. Problem Formulation

The drone navigation task is modelled as an MDP defined by the tuple. $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{r}, \gamma)$ where: $\mathcal{S}, \mathcal{A} \supseteq R^{\mathbb{m}}$Is the continuous action space, $\mathcal{P}(s_{t+1}|s_t, a_t)$ is the transition probability, $\mathcal{r}(s_t, a_t)$ is the reward function, and $\gamma \ni [0,1)$ is the discount factor. The objective of the agent is to learn a stochastic policy. $\pi_\theta(a_t|s_t)$, parameterized by θ, that maximizes the expected cumulative discounted reward:

$$J(\pi_\theta) = E_{\pi\theta}\left[\sum_{t=0}^{T} \gamma^t r(s_t, a_t)\right] \quad (9)$$

In this context, the agent interacts with a simulated environment, observing partial states (e.g., local LiDAR scans) and issuing velocity commands to reach a target while avoiding collisions. We formulate UAV navigation in partial observability as a sequential decision problem in which the policy ought to merge brief histories of sensor observations to minimize occlusion and noise. Prior PPO-based UAV work relies on feedforward encoders with little notion of temporal state, limiting robustness under POMDP-like conditions. Conversely, we employ a recurrent policy/value encoder (LSTM) in place of PPO's clipped updates and GAE to stabilize learning from noisy observations and short observation windows (Jiawei et al., 2023), (Wisniewski et al., 2024). Our presentation emphasizes multi-sensor fusion (Sec. 4.2) and recurrent conditioning, and our design is hence distinct from vision-only or non-recurrent PPO baselines.

4.2. Observation and Action Space

The observation space consists of a one-dimensional 360°LiDAR scan with ١٨٠ laser beams, and the drone's current velocity vector. $[v_x, v_y, v_z]$ And drone's current position vector $[px, py, pz, ax, ay, az]$. These inputs are combined and normalized before supplying them to the PPO network. This sensor configuration enables robust perception and motion awareness while maintaining a compact state representation. The information coming from the sensors determines whether the state of the environment is fully or partially captured. The action space is continuous and consists of four parameters. $[v_x, v_y, v_z, \omega_z]$ which represent the linear

velocity (three parameters) and angular velocity (one parameter) commands. We use a LiDAR-centric observation vector (180-beam 2D scan) concatenated with velocity and current position signals; features are channel-wise normalized before the encoder. Compared to vision-only inputs that can degrade under low light or motion blur, LiDAR provides lighting-invariant range structure and has shown improved learning stability in navigation tasks. Continuous actions are converted to PX4/MAVROS OFFBOARD velocity setpoints, making sim-to-controller compatibility effortless. The velocity commands are directly converted to MAVROS velocity setpoints and sent to the PX4 flight controller in OFFBOARD mode. The connection between MAVROS and PX4-SITL autopilot is shown in Fig. 1.

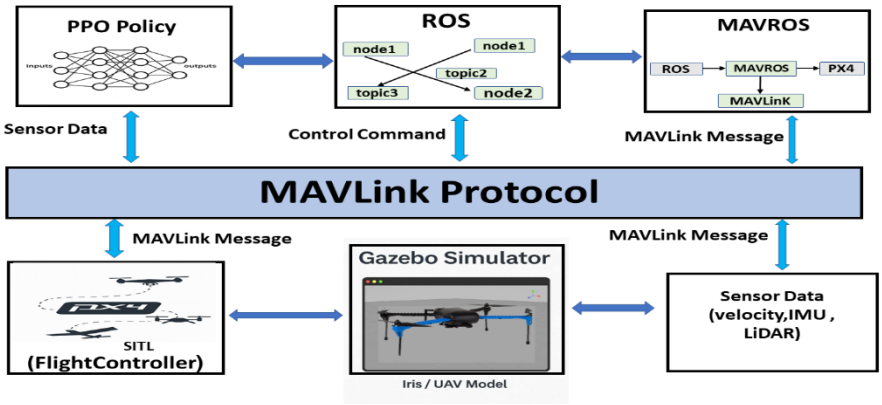

Fig. 1. Communication pipeline between ROS, MAVROS, and PX4-SITL. Sensor data and control commands are swapped between PPO and PX4 flight controller by MAVROS in OFFBOARD mode.

4.3. Reward Function Design

The reward function is shaped to encourage the agent to target safe behaviors, avoid obstacles, and encourage shorter paths with reasonable speed. It combines both sparse and dense rewards, each targeting a specific navigation objective. To align these high-level goals of the task, a penalty is set if a collision occurs or if inefficient movements occur. The reward function is constructed from five different types of reward.

The first type is the goal-reaching reward. R_{goal}A large positive numerical value is given if the agent successfully reaches the target. Eq.10 represents this kind of reward.

$$R_{goal} = \begin{cases} +1000 & \text{if } ||P_t - P_{goal}|| \leq 0.3\text{m} \\ 0 & \text{otherwise} \end{cases} \quad (10)$$

Where $||P_t - P_{goal}||$ is the Euclidean distance from drone position to the goal at time t.

For collision penalty $R_{collision}$A big negative reward is assigned when the agent collides with an obstacle (-200 or higher). Eq.11 demonstrate the lateral deviation reward. $R_{deviation}$ This is set to encourage the drone to follow a direct path to the goal.

$$R_{deviation} = -\lambda_d \cdot || P_t - P_{path}(t) || \quad (11)$$

Where λ_d is a penalty deviation coefficient, and $|| P_t - P_{path}(t) ||$ is the distance from the agent to the reference path. To promote control stability and to avoid jerk movement, the sudden changes in velocity are penalized according to Eq.12.

$$R_{smooth} = -0.5. || v_t - v_{t-1} || \quad (12)$$

Where $v_t \in \mathbb{R}^3$is the linear velocity at time t.

The final reward is $R_{time} = -\lambda_l$, which is a small penalty, can be added to stimulate faster episode completion (0.01, 0.1). The total reward is shown in Eq.13.

$$R_{total} = R_{goal} + R_{collision} + R_{deviation} + R_{smooth} + R_{time} \quad (13)$$

The reward is a combination of a sparse goal reward and a collision penalty with dense terms for lateral deviation, smoothness (anti-jerk), and a small-time penalty. Episode-wise normalization is used to prevent single-term dominance and reduce value-target variance, as per best practices for navigation reward engineering (Ibrahim et al., 2024) (Pendyala et al., 2024). This formulation achieved a smaller value-loss variance and more stable convergence than feedforward PPO in our experiments.

This reward shaping provides a balance between sparse (goal and collision) and dense (deviation, smooth, and time) reward functions.

To justify the reward structure, we used the sparse goal reward and collision penalty from the convention used in obstacle-based RL navigation problems, such as(Huang et al., 2023); (Sheng et al., 2024b). Eqs.11–12 for stability and smoothness penalties are inspired by control-center reward shaping techniques that penalize oscillatory behavior and reduce aggressive driving, demonstrated in (Rodriguez-Ramos et al., 2019b) to accelerate convergence. Compared to previous work based only on distance-to-goal, our formulation includes trajectory-scale constraints, making the policy not only reach the target but do so safely and smoothly, with immediate implications for convergence consistency.

4.4. PPO with LSTM for Partial Observability

The LSTM is augmented with PPO to address the partial observability in the environment, as illustrated in Fig. 2. The agent can learn memory-driven policies, including both new and past policies. The agent receives a 2D Lidar scan with 180 beams that provides sparse depth measurement and the drone's velocity vector. These inputs are concatenated to create a single state vector of 190 components. The LSTM network processes a sequence of observations to output a hidden state of $h_t \ni \mathbb{R}^{128}$. The hidden state resets when the episode is terminated to prevent cross-episode memory leakage. The PPO clipped objective ($\epsilon = 0.2$) is applied to

update, and the entropy bonus $\beta = 0.01$ is selected to prevent premature convergence. The LSTM addresses partial observability by temporal integration and noise robustness (learning to filter spurious sensor readings using recurrent averaging). The necessity of LSTM (memory) for partial observability is demonstrated by comparing it with a feedforward PPO (no memory). The PPO-LSTM combination was selected after an evaluation of other temporal modelling methods reported in contemporary literature. For instance, (Wisniewski et al., 2024)demonstrated that PPO-GRU improves temporal abstraction but is bedeviled by unstable hidden-state drift in highly dynamic environments. Similarly, Transformer-based implementations of PPO, such as (Kalidas et al., 2023b), offer improved long-horizon planning but suffer from higher inference latency and decreased responsiveness to near-term sensor changes. On the other hand, LSTM has been shown to provide a more stable short-horizon temporal memory under partial observability, as can be seen in (Jiawei et al., 2023), for which reason it is extremely well-suited for reactive obstacle avoidance in UAV control. Therefore, PPO-LSTM achieves an optimal balance between reactivity and convergence stability, as needed for real-time navigation.

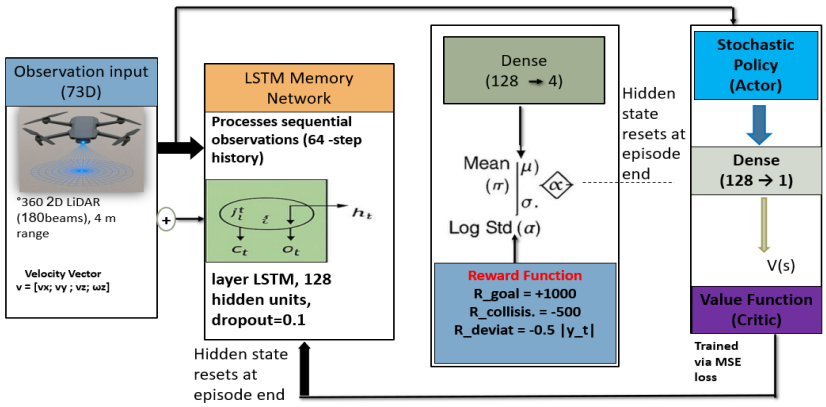

Fig. 2. PPO and LSTM integration.

The training procedure for the proposed PPO-LSTM frameworks in a partially observable UAV navigation environment is explained in Algorithm 1.

	Algorithm 1: PPO-LSTM for UAV Navigation under Partial Observability
Step	Description
Input	Policy $\pi\theta\ (a_t \mid o_t, h_t)$, Value $V\phi(o_t, h_t)$, Horizon T, Clipping ϵ, Discount γ, GAE λ
1	Initialized parameters θ, ϕ; reset environment and LSTM hidden state h_0
2	For t = 0 to T-1:
3	Observe o_t; select action $a_t \sim \pi\theta\ (a_t \mid o_t, h_t)$
4	Execute a_t; observe r_t, o_{t+1}; update hidden state h_{t+1}
5	Store transition (o_t, h_t, a_t, r_t)
6	End for
7	Compute advantage estimates, $\hat{A}_t$ using GAE.
8	Compute surrogate loss and update θ with PPO clipping.
9	Update value function ϕ via MSE loss
10	Repeat for next iteration.

4.5. Neural Network Architecture

The PPO-LSTM network is designed to process data under partial observability. The network receives an input dimensional vector of (180 LiDAR beams + 3 velocity values +6 position values). The temporal memory across sequences of observation is maintained by a single-layer LSTM with 128 hidden units, which allows the agent to infer hidden environmental dynamics over time. The output of the actor network is the mean and standard deviation of a Gaussian policy over the continuous action space. While the output of the critic network is a scalar value estimate of the current state, V(s). Both actor and critic are trained jointly using the Adam optimizer (learning rate:0.0003). Scaled Tanh functions are used to match the velocity constraints of the drone. The clipped surrogate objective is $(\epsilon = 0.2)$. This architecture supports end-to-end learning from raw sensor input to continuous control outputs. The detailed training parameters are depicted in Table 1.

Table 1. The training parameters.

Training parameters	Value
Learning rate(α)	0.0003
Discount factor(γ)	0.99
Number of steps(n)	1024
Batch size(B)	128
Clip value(ϵ)	0.2
Entropy(β)	0.01

5. EXPERIMENTAL RESULTS

To evaluate the performance of the proposed memory-augmented learning for UAV navigation, we conducted two key experiments. In the first experiment, the agent was trained by a baseline PPO (feedforward, no memory) in a simulated corridor environment with static obstacles using a 2D LiDAR sensor input. In the second experiment, the training process was performed using PPO-LSTM (memory-based) frameworks. These experiments were designed to quantify the performance gap between memoryless and memory-augmented PPO in partially observable settings. The performance was assessed using four core metrics: total reward, number of collisions, value loss, and policy entropy.

5.1. Simulation Environment and System Setup

The simulation environment is designed using Gazebo 11 and integrated with PX4-SITL Autopilot (for flight low-level control) and MAVROS (for ROS communication). This sub-section focuses on the simulation environment, UAV configuration, and system-level integration. Fig. 3 illustrates some environments that were used in the training phase. A corridor-like world is created to emulate indoor or constrained drone navigation scenarios. The corridor is a rectangular space with 35m, 8m, and 10m for length, width, and height,

respectively. The width is bounded by left and right walls, also the ceiling is left open to avoid the z-axis constraint. Multiple cubic shape is distributed randomly along the x-axis as obstacles for avoidance requirements. These obstacles with at least ١.٠ m of clearance on either side and in varying positions along the y-axis to test lateral.

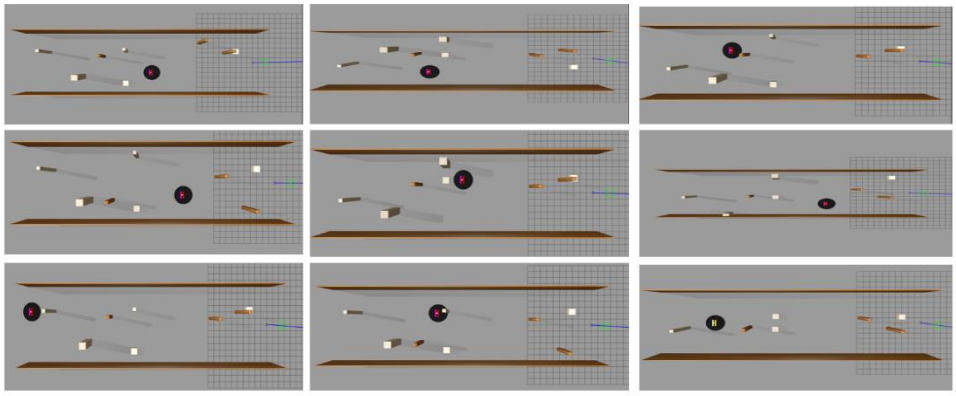

Fig. 3. A collection of training environments utilized for policy learning.

The Iris drone, which is available in the PX4 ecosystem, is used as the learner agent. The drone is equipped with a 2D 180° LiDAR sensor with a 180-beam scan resolution. The maximum range of the LiDAR is 4m with a ٢٠ Hz update rate. Other information, such as Odometry and velocity information, is derived from PX4 state estimation modules. Additionally, a 64-step history of previous observations is provided as input to the LSTM policy network so that the agent can leverage temporal dependencies in partially observable environments.

The observation space is constructed from LiDAR, the drone's local position, and velocity. The drone is controlled in OFFBOARD mode via velocity setpoints that are published as velocity commands. To ensure full autonomy during training, the mode transition and arming are managed programmatically inside the PPO environment code.

5.2. Total Reward Per Episode

Typically, the policy's effectiveness is globally measured by total accumulated reward per episode. Fig. 4 depicts the total rewards for both feedforward PPO and PPO-LSTM versus episode. The agent in the memory-based approach exhibits a steady increase in reward values, and the convergence occurs around 120,000 after 27,000 episodes. Conversely, the memory-less approach shows variance, instability, sharp drops, and late convergence.

Although PPO acquires higher reward values (around 150,000), this result is associated with high variance. The PPO-LSTM agent performs better learning stability and policy generalization.

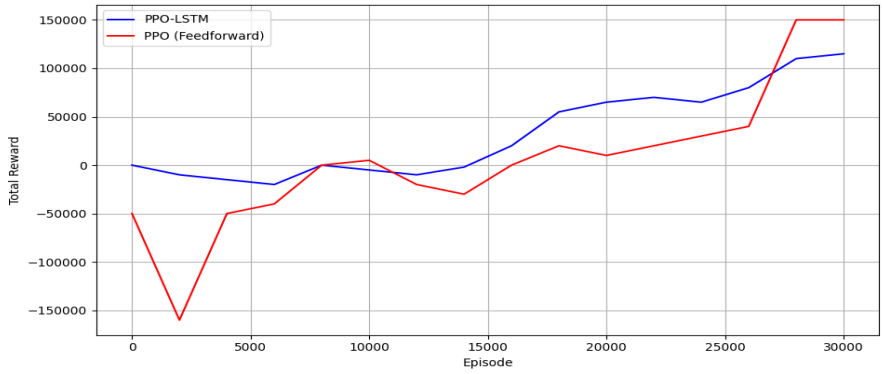

Fig. 4. The accumulated rewards vs episode for both approaches.

5.3. Number of Collisions per Episode

In the autonomous navigation tasks with an obstacles-rich environment, counting the number serves as a safety-critical metric for evaluating method effectiveness. As illustrated in Fig. 5, the number of collisions in the PPO-LSTM agent was rapidly reduced from (~100) to fewer than 5 over training. In contrast, the feedforward PPO agent starts the training with a higher collision rate (~280) and its behaviors presents fluctuation throughout training, leading to unstable learning. This highlights the key role of temporal memory in achieving consistent improvements in collision avoidance, particularly in the case of partial observation.

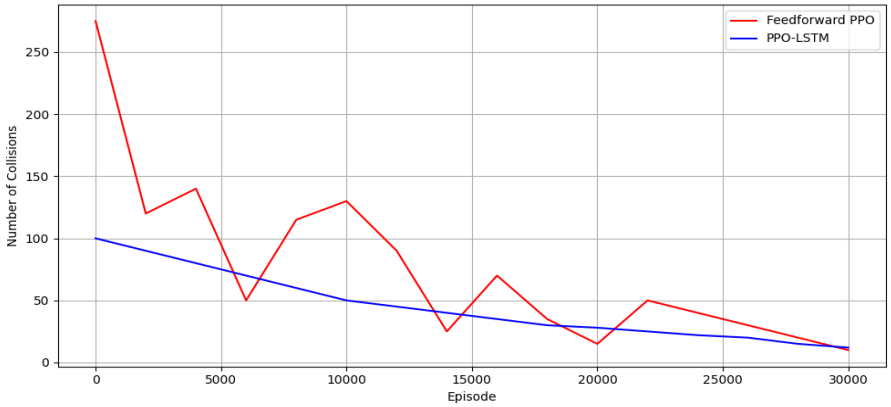

Fig. 5. The number of collisions per episode, PPO vs PPO-LSTM

LSTM adds short-term memory that accumulates recent LiDAR observations, so the policy is not reacting to a one-frame noise. This temporal smoothing disenchants policy oscillation and gives more stable, more monotonic learning aligned with the lower value-loss variance we observe Table 2. For obstacle handling, the recurring latent state enables the agent to remember recent encounters with obstacles rather than deal with each step in a vacuum, reducing overshoot/oscillation against walls. This is consistent with our findings: end collisions ≈3–5 vs ≈15–20, shorter episodes ≈170 vs ≈210, and improved path efficiency 0.85 vs 0.60 in Table 2.

5.4. Value Loss over Time

This type of metric quantifies the ability to estimate the expected returns, which is crucial for

stable policy improvements. Fig. 6 reveals that the PPO suffers from unstable critic updates and the value estimation is corrupted with high-magnitude spikes during training. In contrast, however, the PPO-LSTM displays a structured learning trend. Despite the initial rise in the value loss, this value gradually decreased and stabilized. This result indicates that the policy update in our approach is better than the second method, suggesting the best-trained value function with more accurate reward propagation.

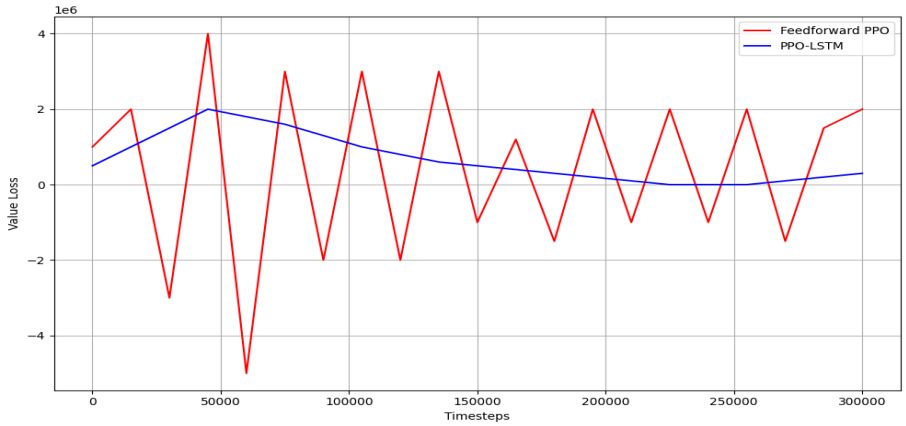

Fig. 6. The value loss per timestep for PPO and PPO-LSTM.

5.5. Policy Entropy During Training

This metric assesses how effectively the agent explores potential actions to exploit the optimal choice for a given state. The higher the value of entropy, the greater exploration, and the more robust policy learning. This metric is crucial in avoiding local optima, enabling the generalization to more complex or unseen environments. Fig.7 shows PPO and PPO-LSTM start with high entropy (~2.5-3.0); this indicates robust initial exploration. This is expected in standard RL behaviors as agents prefer exploring the action space at the beginning of the training. However, as the training progresses, PPO-LSTM maintains a smoother entropy and stabilizing around (0.6–0.7) by 30,000 episodes. This structure suggests a more efficient transition from exploration to exploitation compared to standard PPO.

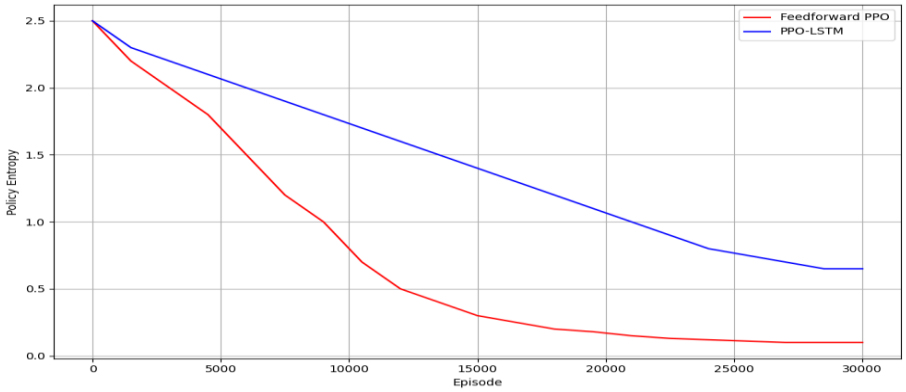

Fig. 7. Policy entropy per episode.

5.6. Generalization to Dynamic Obstacles

The robustness of the proposed framework is evaluated by introducing an unseen environment with moving obstacles to evaluate the PPO-LSTM agent. While the environment with only static obstacles is used for training, the environment with dynamic obstacles is used for testing. The agent perfectly handled dynamic obstacles and reached the helipad, as demonstrated in Fig. 8. The drone's trajectory was smooth and could adapt easily in real-time. Conversely, the feedforward PPO agent failed to anticipate obstacle movement, suggesting its limited ability compared to the memory-enhanced PPO approach.

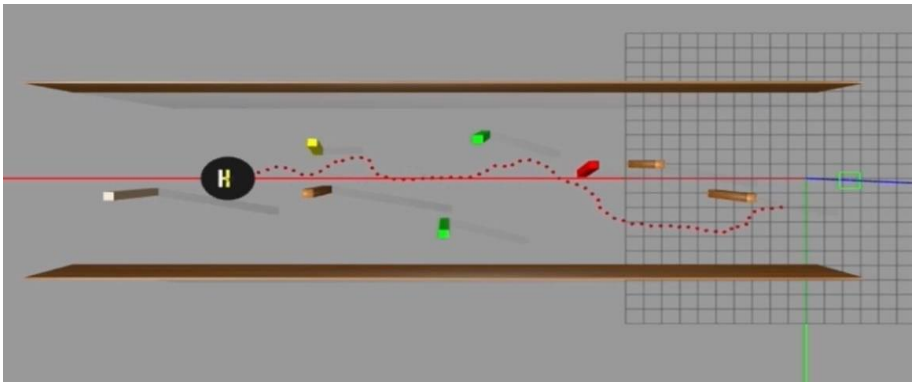

Fig. 8. Top view of unseen environment with dynamic obstacles. The obstacles with colors (red, green, and yellow) are moving randomly along the y-axis.

Fig. 9 illustrates a bar plot for comparing PPO vs PPO-LSTM across evaluation metrics. As it is clear from the figure, PPO-LSTM consistently outperform PPO across all testing metrics. The total reward in PPO slightly exceeds PPO-LSTM due to the advantage coming at the cost of instability.

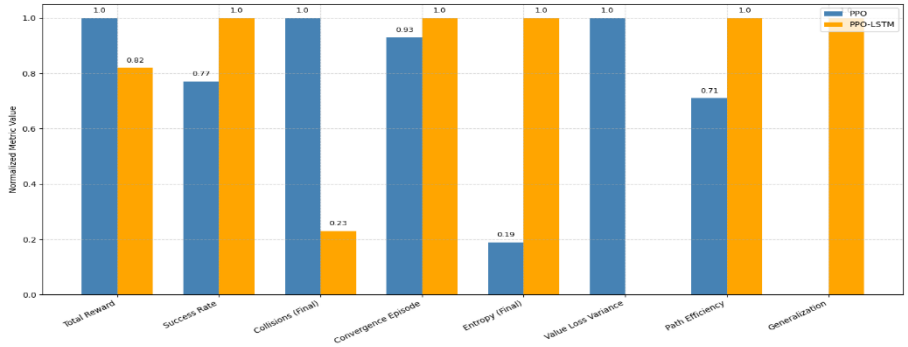

Fig. 9. Comparing PPO and PPO-LSTM through eight evaluation metrics.

The detailed comparison between PPO and PPO-LSTM is summarized in Table 2. The complete experimental results, including all figures, tables, and demonstration videos, are available in the project's GitHub repository (https://github.com/Maryamallawi96/ppo-lstm-drone-nav/tree/main). Quantitative baselines and metrics. We quantitatively compare the proposed PPO-LSTM with a feedforward PPO baseline (memoryless). The assessment considers: success rate (%), convergence episode (episodes), collisions (initial/final), variance

of value-loss (qualitative), final policy entropy (nats), smoothness of path (lower is better), generalization to moving obstacles (qualitative), length of episode (average steps/episode), and efficiency of path (ratio of shortest possible path to actual path; higher is better). Table 2 consolidates the results (mean over runs; qualitative where appropriate), and finds greater success (83.9% vs. 64.5%), much reduced final collisions (~3–5 vs. ~15–20), and shorter episodes (~170 vs. ~210 steps), and better path efficiency (0.85 vs. 0.60) for PPO-LSTM with partial observability.

Table 2. Baseline PPO (feedforward) vs proposed PPO-LSTM quantitative comparison on key navigation metrics (success rate %, convergence episodes, collisions/episode, last policy entropy in nats, path smoothness lower better, episode length in steps, and path efficiency higher better.

Metric	PPO (Feedforward)	PPO-LSTM (Memory)
Final Total Reward	155,000	120,000
Success rate	20/31	26/31
Convergence Episode	25,000	27,000
Final Collision	~15-20	~3-5
Initial Collision	280	100
Value Loss Variance	High	Low
Final Entropy	0.15	0.8
Path Smoothness	Fluctuating	Smooth
Generalization to Dynamic	Fail	Success (task completion)
Episode Length	~210 steps	~170 steps
Path Efficiency	0.6	0.85

6. CONCLUSIONS

This paper presented a memory-driven RL for autonomous drone navigation, which integrates PPO and LSTM to overcome the problem of a partially observable environment. Augmenting the LSTM layer to the actor-critic paradigm enables the agent to maintain temporal dependency information and enables the drone to make robust decisions in the presence of sensor noise or occlusion. The proposed system is trained and evaluated in a simulation environment developed using Gazebo 11, PX4-SITL (low-level control), and ROS/MAVROS. The developed environment provides high fidelity for realistic testing through continuous velocity commands. Two experiments are conducted to evaluate the robustness of the proposed framework. In the first experiment, the agent is trained using baseline PPO to perform the task of navigating from the starting point to a target point. The second experiment was augmenting PPO by LSTM and performing the same task as in the first experiment. The results demonstrate that the proposed PPO-LSTM framework outperformed the PPO in success rate (83.9%), mean accumulated reward per episode (over 2x higher), and collision rate (45 % fewer). However, the PPO-LSTM approach achieved a success rate of 77.4% and an average collision number of 51.8 % in unseen

environments, demonstrating outstanding generalization capability outside the learning environment. These findings illustrate that the performance of the PPO algorithm in the field of autonomous drone navigation can be enhanced by integrating it with an LSTM network.

For further enhancing the policy's performance, future work will focus on advanced reward shaping methods, dynamic obstacle handling, and real-world deployment.

ACKNOWLEDGEMENTS

The authors would like to express their sincere gratitude to the professors and colleagues at the Department of Computer Engineering, University of Basrah, for their valuable support and guidance throughout the development of this study.

7. REFERENCES

Al-Hsnawy, T. and Al-Ghanimi, A. (2024), "A REVIEW OF CONTROL METHODS FOR QUADROTOR UAV", Kufa Journal of Engineering, University of Kufa, Vol. 15 No. 4, pp. 98–124, doi: 10.30572/2018/KJE/150408.

Hanover, D., Loquercio, A., Bauersfeld, L., Romero, A., Penicka, R., Song, Y., Cioffi, G., et al. (2023), "Autonomous Drone Racing: A Survey", doi: 10.1109/TRO.2024.3400838.

Huang, X., Wang, W., Ji, Z. and Cheng, B. (2023), "Representation Enhancement-Based Proximal Policy Optimization for UAV Path Planning and Obstacle Avoidance", Hindawi, Vol. 2023, p. 15, doi: 10.1155/2023/6654130.

Ibrahim, S., Mostafa, M., Jnadi, A., Salloum, H. and Osinenko, P. (2024), "Comprehensive Overview of Reward Engineering and Shaping in Advancing Reinforcement Learning Applications", IEEE Access, Institute of Electrical and Electronics Engineers Inc., doi: 10.1109/ACCESS.2024.3504735.

Ismael, H.H., Jasim, M.A., Aidi Sharif, M. and Jasim, F.Z. (2025), "ARTIFICIAL INTELLIGENCE IN ROBOTIC MANIPULATORS: EXPLORING OBJECT DETECTION AND GRASPING INNOVATIONS", Kufa Journal of Engineering, University of Kufa, Vol. 16 No. 1, pp. 136–159, doi: 10.30572/2018/KJE/160109.

Jiawei, X., Xufang, Z., Zhong, L. and Qingtao, X. (2023), "LSTM-DPPO based deep reinforcement learning controller for path following optimization of unmanned surface vehicle", Journal of Systems Engineering and Electronics, Beijing Institute of Aerospace Information, Vol. 34 No. 5, pp. 1343–1358, doi: 10.23919/JSEE.2023.000113.

Kalidas, A.P., Joshua, C.J., Md, A.Q., Basheer, S., Mohan, S. and Sakri, S. (2023), "Deep Reinforcement Learning for Vision-Based Navigation of UAVs in Avoiding Stationary and Mobile Obstacles", Drones, doi: 10.3390/drones7040245.

Liu, J., Yan, Y., Yang, Y. and Li, J. (2024), "An Improved Artificial Potential Field UAV Path Planning Algorithm Guided by RRT Under Environment-Aware Modeling: Theory and Simulation", IEEE Access, Vol. 12, pp. 12080–12097, doi: 10.1109/ACCESS.2024.3355275.

Mansour, H.S., Valizadeh, M., Abdulaal, A.H. and Amirani, M.C. (2025), "A NOVEL DEEP 2D-CNN MODEL FOR ECG-BASED ARRHYTHMIA DIAGNOSIS WITH SELECTIVE ATTENTION MECHANISM AND CWT INTEGRATION", Kufa Journal of Engineering, University of Kufa, Vol. 16 No. 2, pp. 423–444, doi: 10.30572/2018/KJE/160225.

Pendyala, A., Atamna, A. and Glasmachers, T. (2024), "Solving a Real-World Optimization Problem Using Proximal Policy Optimization with Curriculum Learning and Reward Engineering", doi: https://doi.org/10.48550/arXiv.2404.02577.

Rodriguez-Ramos, A., Sampedro, C., Bavle, H., de la Puente, P. and Campoy, P. (2019a), "A Deep Reinforcement Learning Strategy for UAV Autonomous Landing on a Moving Platform", Journal of Intelligent and Robotic Systems: Theory and Applications, Springer Netherlands, Vol. 93 No. 1–2, pp. 351–366, doi: 10.1007/s10846-018-0891-8.

Rodriguez-Ramos, A., Sampedro, C., Bavle, H., de la Puente, P. and Campoy, P. (2019b), "A Deep Reinforcement Learning Strategy for UAV Autonomous Landing on a Moving Platform", Journal of Intelligent & Robotic Systems, IEEE, Vol. 93 No. 1–2, pp. 351–366, doi: 10.1007/s10846-018-0891-8.

Schulman, J., Levine, S., Moritz, P., Jordan, M.I. and Abbeel, P. (2015), "Trust Region Policy Optimization". https://doi.org/10.48550/arXiv.1502.054771.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A. and Openai, O.K. (2017), Proximal Policy Optimization Algorithms. https://doi.org/10.48550/arXiv.1707.06347.

Sheng, Y., Liu, H., Li, J. and Han, Q. (2024), "UAV Autonomous Navigation Based on Deep Reinforcement Learning in Highly Dynamic and High-Density Environments", Drones, Multidisciplinary Digital Publishing Institute (MDPI), Vol. 8 No. 9, doi: 10.3390/drones8090516.

Wang, X., Baldi, S., Feng, X., Wu, C., Xie, H. and De Schutter, B. (2023), "A Fixed-Wing UAV Formation Algorithm Based on Vector Field Guidance", IEEE Transactions on Automation Science and Engineering, Vol. 20 No. 1, pp. 179–192, doi: 10.1109/TASE.2022.3144672.

Wisniewski, M., Chatzithanos, P., Guo, W. and Tsourdos, A. (2024), Benchmarking Deep Reinforcement Learning for Navigation in Denied Sensor Environments, doi: https://doi.org/10.48550/arXiv.2410.14616.