International Journal of Robotics and Control Systems
Vol. 5, No. 5, 2025, pp. 2522-2545
ISSN 2775-2658
http://pubs2.ascee.org/index.php/ijrcs

Dual-Memory Architecture for Robust UAV: Navigation Integrating LSTM and Transformer within a PPO Framework

Maryam Allawi Haddad [a,1,*], Dhayaa Raissan Khudher [b,2]

[a,b] Department of Computer Engineering, University of Basrah, Basrah 61004, Iraq
[1] pgs.maryam.allawi@uobasrah.edu.iq; [2] dhayaa.khudher@uobasrah.edu.iq
* Corresponding Author

ARTICLE INFO

Article history
Received September 21, 2025
Revised November 13, 2025
Accepted November 20, 2025

Keywords
Autonomous Drone Navigation;
Partial Observability;
POMDP;
Memory-Augmented RL;
Trajectory Smoothness

ABSTRACT

Autonomous UAV navigation typically suffers from partial observability (POMDP), where noisy and limited sensing degrades the reliability of decisions. We introduce a dual-memory PPO that augments an LSTM for short-horizon responsiveness with a Transformer for long-horizon context, fused by a learnable gate that adaptively weights both streams end-to-end. Unlike Dual-Transformer PPO and other attention-only variants our model retains recurrent memory and learns the fusion rather than prespecifying it (e.g., concatenation or sum). The observation vector merges normalized proprioceptive and range data the reward balances progress collision penalties and trajectory smoothness with tuned coefficients to avoid dominance. In simulated corridor worlds (with a dynamic variant) the hybrid policy completes 96.5% of episodes 9.7 pp over PPO-LSTM and 17.1 pp over PPO-Transformer while reducing final collisions to 2 per episode, reductions of 37.5% vs PPO-LSTM 64.3% vs PPO-Transformer: 85.7% vs PPO. It converges in 20k episodes (vs 25–29k for baselines), with shorter episodes (150 steps), and greater path efficiency (0.85) than either baseline. Findings are presented as the average plus or minus the standard deviation for all five seeds when $p < 0.05$. Limitations include a simulation-only study and limited environment diversity further, larger-scale environments and fusion and reward design ablations are pending. Overall learnable gating of complementary short- and long-term memories improves reliability under partial observability without compromising on practical training efficiency.

1. Introduction

Recently, the use of drones has become widespread in many applications [1], including search, flood detection [2], rescue [3], surveillance [4], and infrastructure inspection [2], [5]. Transitions from human-in-the-loop piloting to autonomous flight require policies that react under partial observability in a POMDP environment in which the agent observes noisy, incomplete information from sensor constraints and occlusions [6], [7]. This transition can be achieved by two prominent approaches: the adoption of classical control methods (system modelling) [8] or utilizing Deep Reinforcement Learning (DRL) [9], [10], [11]. There are three main aspects for selecting an appropriate DRL algorithm: training stability, convergence speed, and sampling efficiency [12], Further, the ease of algorithm implementation for a specific application and its capacity to generate continuous action

outputs [13]. Classic controllers, e.g., PID [14], [15] and MPC [16], [17] provide robust performance with good models and state estimation but disintegrate under model inaccuracies and sensing limitations. Hybrid model-based/model-free [18] recent solutions mitigate these issues by adding structure but are often reliant on expert models or hand-tuned estimators that are difficult to maintain. Deep reinforcement learning (DRL) [10], [19] offers a data-driven solution for continuous-control methods [13], [9]. Most baseline DRL algorithms have some limitations and often fail to satisfy all essential machine learning requirements [20], [21]. For example, DDPG [22], my overestimation of Q-value affects policy effectiveness [23], entropy sensitivity of SAC [24] causes training instability [25]. Recent research trends have focused on the strengths of an algorithm and improving its weaknesses [26]. PPO algorithm [27], [28], for instance offers implementation simplicity and training stability [28], which makes it a popular choice for practical applications [27], [29]. The ability of an agent to interact effectively with its environment is constrained by its perceptual abilities e.g., sensor quality and quantity. In applications [30], [31] having full environmental observability remains unachievable due to sensor limitations [7], [32]. Therefore, developing reliable autonomous systems in real-world environments must alleviate both partial observability [33] and sensor limitations [34], [35]. To address partial observability[33], A data storage extension, known as long-term short-term memory (also known as LSTM), has been added to PP [36], [37]. Based on previous cases, the agent becomes capable of making current decisions [38]-[40]. PPO-LSTM promote short-term reactivity by maintaining a hidden state that captures recent temporal patterns [19]. PPO-LSTM excels at short-horizon reactivity but fails at long-range dependencies [34]. PPO-Transformer is great at long-horizon context but can lose near-term responsiveness and increase inference cost [41]. Additionally, long-term dependencies [42], [43] should be considered to capture broader temporal and spatial this is achieved by extending PPO with Transformer PPO-Transformer [42], [43].

Recent dual-stream implementations usually combine streams of memory using operations of constant frequency (concatenation/addition), which cannot dynamically alter the relative priority between short- vs long-term information in varying flight modes [44], [45]. Much research work also does not erase the fusion process and rarely reports latency memory usage limiting real-time potential claims. Research gap navigation in POMDPs for UAVs needs a two-memory policy that (i) maintains recurrent short-horizon signals in real-time for speedy hazard evasion, (ii) leverages attention-based long-horizon structure to plan, and (iii) learns the fusion end-to-end along with task-discriminative metrics trajectory smoothness and computational reports (inference latency/memory). This work we present a dual-memory PPO that uses an LSTM (quick response) with a Transformer long-horizon context and blends them using a learnable gating module that weights each stream adaptively, conditioned on history. Contrary to Dual-Transformer PPO and attention-only baselines, our model retains recurrent memory and replaces fixed fusion with data-driven gating [46], [47]. We provide a reproducible training recipe normalized proprioceptive/range observations; coefficient reward shaping to avoid dominance gradient passing through both branches and evaluate in static and dynamic corridor settings, quantifying success, collisions, episode length, path efficiency, and smoothness of trajectory, with multi-seed statistics and fusion/reward ablations.

2. Related Works

In complex and partially known environments, the classical control methods struggle to adapt as they rely on precise models, fixed parameters, and deterministic assumptions. On this basis, adopting data-driven methods, such as DRL [26], [48] and adaptive systems is a significant solution for overseeing uncertainty and learning from interactions [49]-[51]. In the area of UAV [12], [52] DRL has gained considerable attention, particularly in environments with uncertainties and high dimensionality [39]. Under POMDPs, UAV policies need memory to integrate noisy, partial observations over time. Recurrent approaches (PPO-LSTM/GRU [44]) enforce short-horizon reactivity yet struggle to sustain long-range dependencies in contrast transformer-based PPO learns long-horizon context [34], [42] but can dampen immediate reactions and exaggerate inference latency/memory, which is critical for onboard deployment. Recent dual-stream architectures combine

recurrent and attention modules, yet typically with fixed fusion concatenate or sum [12]. Fixed fusion cannot adapt the relative importance of short- vs long-term signals across regimes of flight, and prior work often omits ablations of the fusion mechanism and does not report compute metrics (latency, parameters), so real-time feasibility claims are limited. On the RL side [48] off-policy methods such as TD3/DrQ are more sample-efficient, but recurrent off-policy training in POMDPs is susceptible to replay aliasing/bootstrapping when truncating hidden states. We thus employ PPO for its stability for recurrent policies, clipped objectives, and GAE with on-policy rollouts [53], keeping hidden-state progression synchronized [54], trading sample efficiency [55], [56] for more stable training [57], [58]. Our departure from fixed-fusion or attention-only hybrids is to preserve recurrent memory and replace fixed fusion with an end-to-end [19], [59], [60] learnable gate that adaptively weights long- and short-horizon signals. We also report both compute measures parameters, latency and task measures (success, collisions, trajectory smoothness), explicitly closing gaps in prior work.

We proposed a novel PPO-based hybrid memory architecture that integrated LSTM and Transformer modules. The objective of this framework is to enhance UAV navigation performance in an uncertain environment with partial observability conditions.

3. Research Method

The Hybrid LSTM-Transformer PPO approach for autonomous UAVs under partial observability is described in this section. We construct the challenge as a partially observable decision-making problem where the agent goes through the world, selects an action based on a stochastic policy, observes a state at each time step, and receives a reward. Raw sensor measurements like 360° 2D LiDAR, velocity, attitude, and position are supplied to the input layer. The input is concatenated and normalized to get the fused observation vector. It is then passed through an embedding layer to create a dense representation that can be sequentially processed. The embedded features are passed in parallel through two networks: an LSTM memory network that models short-term temporal dependencies and a Transformer encoder that models long-range temporal and spatial relationships. The representations of the two networks are combined in a feature fusion layer, a dense layer followed by a ReLU activation function, thus combining the strengths of the two architectures.

A thorough description of the action and observation spaces in this formulation is given once the UAV navigation problem is first formulated as (POMDP). To obtain useful latent representations, the observations are initially fed into a feature encoder. To completely capture short-term temporal interactions and long-distance contextual information, these latent features are then successively run through a hybrid architecture that combines both LSTM and Transformer modules. This hybrid memory module's output is accepted by the value and policy networks. We also go over the training process, the neural network setting, the reward, and the Hybrid LSTM Transformer PPO architecture.

Fig. 1 illustrates end-to-end pipeline for our proposed Hybrid LSTM–Transformer PPO: environment setup → sensing → normalize & fuse → embedding → parallel memory branches (LSTM-Transformer) → dimension alignment + learnable gating fusion → actor–critic heads → reward computation → PPO update → convergence & logging → save model.

3.1. Problem Formulation

The set (O, T, S, A, T, R, Z, γ) defines the POMDP formulation of the UAV navigation task. Where S is the set of latent states, A is the set of actions (such as thrust and yaw rate), O is the set of observations, T is the transition dynamics, R is the reward function, Z is the observation model, and γ is a discount factor that falls between 0 and 1.

At each time step t, the UAV receive sensory signals from LiDAR 2D 360°, velocity vector $\mathcal{V} = [\mathcal{V}_x, \mathcal{V}_y, \mathcal{V}_z]$, and attitude $\theta = [roll, pitch, yaw]$. These input signals are first normalized and then concatenated into a unified observation vector.

$$ot = [\theta t; vt; pt] \in R369 \quad (1)$$

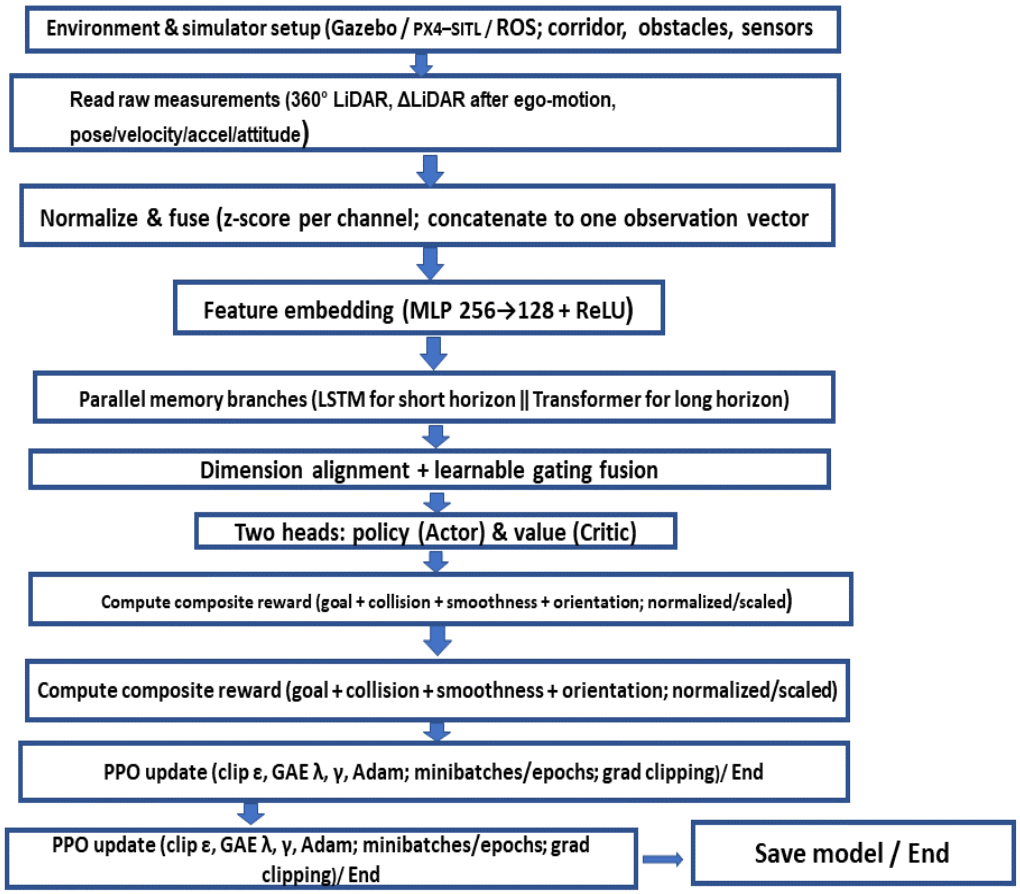

Fig. 1. End-to-end pipeline for our proposed Hybrid LSTM–Transformer PPO

In the feature encoder stage, the o_t is passed through a Multilayer perceptron (MLP) to represent it in a latent feature space. The MLP consists of two dense layers where a nonlinear activation function (ReLU) is applied in between. The first layer converts the normalized observation vector into an intermediate representation in a way that enables the network to learn the abstract features. The induced feature vector is given in (2). The second layer then projects this feature into a compact latent representation as given in (3).

$$\mathrm{h1} = \mathrm{W1ReLU}(\mathrm{W1Ot} + \mathrm{b1}) \tag{2}$$

$$\mathrm{h2} = \mathrm{W2ReLU}((\mathrm{h1}) + \mathrm{b2}) \tag{3}$$

where W_1, W_2 and b_1, b_2 are learnable weight parameters. $h_2 \in R^d$ is the latent embedding of the observation.

This compact feature vector h_2 captures the current sensory state and is forwarded to the recurrent network (LSTM). Immediate responsiveness and (Transformer) strategic planning over extended horizons are critical. Fig. 2 illustrates the overall system pipeline.

3.2. Observation and Action Space

The perception space is made up of a 360° LiDAR scan of 360 beams, the drone's present position vector [px, py, pz, ax, ay, az], and its current velocity [vx, vy, vz]. Of the drone. Raw measurements (360-degree 2D LiDAR with 360 beams, the time difference of two scans $\mathrm{Delta-LiDAR_t} = \mathrm{LiDAR_t} - \mathrm{LiDAR_{t-1}}$ after ego-motion registration, position, velocity V_t=(V_X,V_Y,V_Z) accelerate linearly a_t=(a_x, a_y, a_Z) and attitude θ_t = (roll, yaw, pitch) are concatenated in a vector of a single

observation ot, channel-wise normalized, and passed through a two-layer feature encoder (256 and 128 with ReLU) to obtain a compact embedding. These inputs are summed up and normalized and provided to the PPO network. This sensor configuration enables robust perception and motion sensing without a large state representation. The sensors determine if they perceive the world fully or partially. The action space is continuous and consists of five parameters $[vx, vy, vz, \omega y, \omega z]$, three linear velocity commands and two angular velocity commands. The velocity commands are merely translated to MAVROS velocity setpoints and published on the PX4 flight controller running in OFFBOARD mode. Fig. 3 depicts the interaction between MAVROS and PX4-SITL autopilot. MAVROS exchanges sensor data and control commands between the PPO and PX4 flight controllers in OFFBOARD mode.

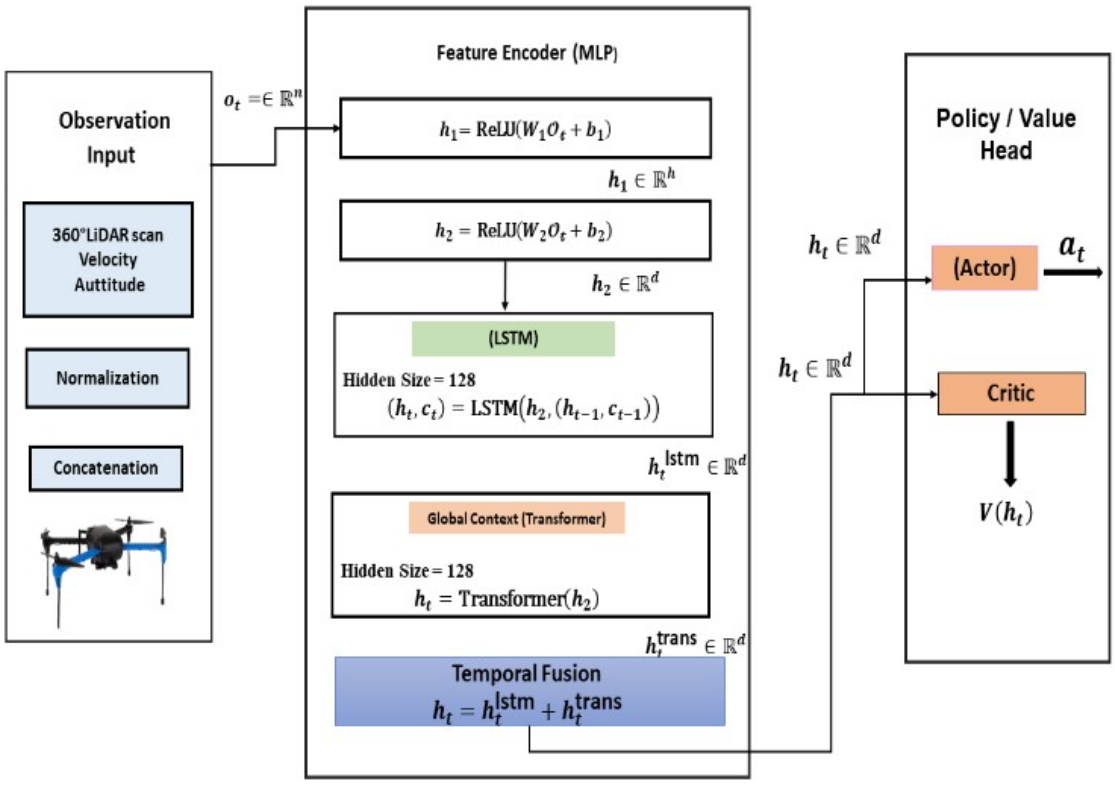

Fig. 2. Architecture of the proposed Hybrid LSTM Transformer PPO architecture

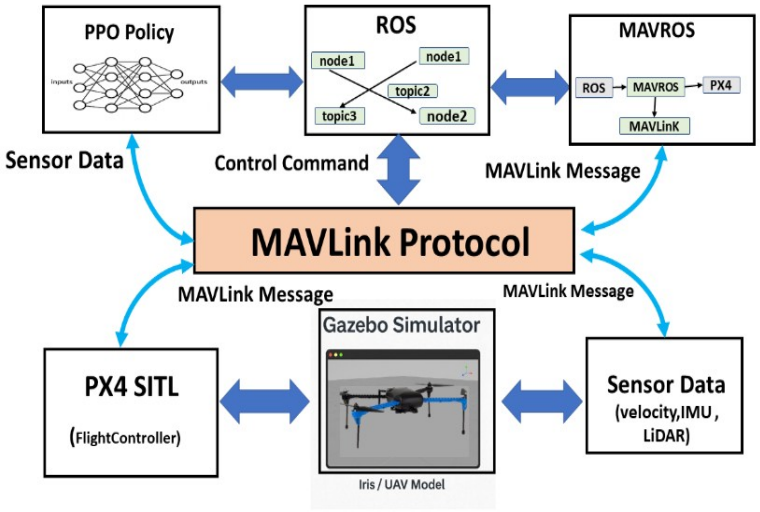

Fig. 3. Communication pipeline between ROS, MAVROS, and PX4-SITL

3.3. The Design of Reward Functions

The reward mechanism encouraged the agent to reach the goal. while showing safe habits, staying away from impediments, and advocating shorter courses at a suitable speed. It targets distinct navigation objectives by combining sparse and dense rewards. To align these high-level task goals, wasteful movements or collisions are penalized. Five different types of rewards are used to form the reward function.

The goal-reaching reward R_{goal} is the first kind; if the agent is successful in reaching the target, a sizable positive numerical value is provided. The reward combines $R_t=R_t^{goal}+R_t^{collision}+R_t^{smooth}+R_t^{stability}$, A sparse goal term with a collision penalty supplemented by dense smoothness and orientation-stability terms; dense terms are normalized by episode to prevent any single component from overwhelming. We stick with ReLU activations for velocity and stable optimizable; ablation with GELU showed infinitesimal improvements at a nontrivial latency cost. This type of reward is represented by (4).

$$R_{goal} = \begin{cases} +1000 & if\ ||b_t - b_{goal}|| \leq 0.3m \\ 0 & otherwise \end{cases} \quad (4)$$

where $||b_t - b_{goal}||$ is the measurement of distance in Euclidean space from the drone's position to the destination at time t. The second type, represented by the collision penalty $R_{\text{collision}}$ in (5), gives a negative reward the agent when it collides with an obstruction. This will motivate the agent to learn safe trajectories and avoid dangerous behaviors.

$$R_{\text{collision}} = \begin{cases} -200 & \text{If a collision is detected at time t} \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

The third reward is the stability penalty $R_{\text{stability}}$ that avoids jerky or spasmodic motion of the drone and provides smooth, stable following of trajectories. It is computed at each timestep as the square differences between the current roll and pitch angles of the drone and their respective reference (target) values. By incentivizing small pitch and roll deviations, the agent learns to provide stable flight attitudes, which are most important in-flight safety, especially under narrow pass or obstacle environment simulation conditions, where sudden angular displacement will be triggered by instability or collision in (6).

$$\text{Rstability} = -\lambda s\ ((\text{yawt} - \text{yaw ref})2 + (\text{pitcht} - \text{pitch ref}))2 \quad (6)$$

To promote control stability and to avoid jerk movement, the sudden changes in velocity are penalized according to (7).

$$\text{Rsmooth} = -\lambda v|\text{vt} - \text{vt} - 1| \quad (7)$$

The total reward is shown in (8).

$$\text{Rt} = \text{Rgoal} + \text{Rcollision} + \text{Rstability} + \text{Rsmooth} \quad (8)$$

This reward shaping method combines sparse rewards (arrival at goals and collision) with dense feedback (stability and smoothness) to guarantee both guaranteed achievement of the task and control improvement.

3.4. Hybrid LSTM-Transformer Based Temporal Feature Modelling

The dual-memory framework with LSTM and Transformer networks is adopted. The LSTM captures short-term temporal dynamics via recurrent processing of sequential data. Whereas the Transformer learns long-range dependencies via self-attention processes by considering patterns and relationships from a more extended observation history. The embedding sequence is parallelly processed by two branches: an LSTM (1 layer, hidden 128) for short-horizon reactivity extraction $\widetilde{h_t^L}$ and a Transformer encoder branch for long-horizon temporal and spatial context extraction (two

encoder layers, four attention heads, model dimension 128, feed-forward dimension 256, dropout 0.1, sinusoidal positional encodings) h_t^T. Both branches' outputs are projected into the same dimensionality and summed through a learnable gating layer that produces element-wise weights to adaptively combine the recurrent and attention feature gradients are end-to-end backpropagated through both branches. With three internal gates to process the input and output of information in and out of its memory cells per step, the LSTM network has two states: a call state and a hidden state. These states have temporal dependencies required for partial observability. The flow of newly added data is managed by the input gate. The ratio of prior knowledge to retain is regulated by the forget gate. The output information is controlled by the output gate. Where $\sigma(\cdot)$ is the sigmoid function, and W_*, U_*, b_* Are learnable LSTM parameters.

As mentioned in (9), the candidate cell state c_t is a linear transformation of the current input o_t and the preceding hidden state h_{t-1}in (9).

$$c_t = \tanh(W_c o_t + U_c h_{t-1} + b_c) \tag{9}$$

The hyperbolic tangent is tanh, and $\odot$ is the element-wise multiplication. To find the cell state, we take the previous cell state c_{t-1} and scale it by the forget gate f_t and the candidate cell state and scale it by the input gate. This is shown in (10).

$$\widetilde{c_t} = \tanh(W_c x_t + U_c \mathrm{h} + b_c) \tag{10}$$

To get the final hidden state the output gate h_t at time t, i.e., output gate o_t to the activated cell state c_t Using the hyperbolic tangent function, as in (11).

$$h_t^{\mathrm{LSTM}} = o_t \odot \tanh(c_t) \tag{11}$$

After the combined hidden state h_t at time t, using the updated cell state c_t and the output gate o_t, the sequence of hidden states from time $t-k$ to t is denoted as $Z_{t-k:t} = [h_{t-k}, h_{t-k+1}, \ldots, h_t]$, is forwarded to the Transformer Encoder.

The sequence is passed through the Transformer to extract long-term temporal relations and provide a better representation in (12).

$$h_t^{\mathrm{trans}} = \mathrm{TransformerEncoder}(h_1, \mathrm{context} = H_{1:t}) \tag{12}$$

This self-attention mechanism within the Transformer Encoder calculates key, query, and value matrices as in (13).

$$Q = HW_Q, \quad K = HW_K, \quad V = HW_V \tag{13}$$

where W_Q, W_K, W_V A learned projection matrix and H represent hidden states of the input.

The computation of attention scores is performed by in (14).

$$\mathrm{Attention}(Q, K, V) = \mathrm{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V \tag{14}$$

The final hidden state is constructed by concatenating the outputs of the two streams of temporal processing: the Transformer encoder and LSTM network, as in (15).

$$h_t = h_t^{\mathrm{lstm}} + h_t^{\mathrm{trans}} \tag{15}$$

This twin-memory blending mechanism enables the model to use both LSTM's short-term sequential dependencies and Transformer's long-range contextual patterns simultaneously. The blended technique enhances the UAV's ability to learn decision-making policies in partially observable and dynamic settings effectively. Fusion mechanism and gradient flow. The LSTM hidden state and the

Transformer encoder output are projected linearly to a common dimension at each time step and fused with a learnable element-wise gate; the fused representation is ingested by the actor–critic heads. End-to-end gradients flow through both branches (no detach), and the gate must therefore learn to dynamically reweight short-horizon (recurrent) and long-horizon (attention) features during training.

3.5. LSTM and Transformer Fusion Mechanism

In real-world UAV autonomous navigation missions, considering only the latest observation at any given point in time is typically not sufficient because of sensor range limitations, occlusion, dynamic objects in the environment, and noise. Under such conditions, an accurate decision cannot be made based only on the current observation. To overcome this challenge, it is augmented with a memory extension, such as LSTM, to handle partial observability.

Fig. 4 illustrates how both LSTM and Transformer block operations function together to improve the agent's capacity to make decisions based on what it has seen in the past. The left-side LSTM is performing short-term reactivity based on only the present fused observation vector. z_t and the most recent hidden state h_{t-1} the LSTM to compute a new hidden (h_t^{lstm}) which encodes real-time responses. This enables the agent to retain historical information, predict dynamic changes, and recover from sensor noise or occlusion, making it more resilient in navigation tasks. Consequently, they can react effectively to temporal dependencies and unseen hazards. The right side, which is in charge, is computing the transformer module, which takes an input sequence of past fusion observations. $[z_{t-\tau}, \dots, z_t]$, and generates h_t^{trans}a higher-level encoder of long-term environmental trend. The Transformer employs long-term awareness to direct the drone in creating high-level plans given its navigation goals and overall mission. By executing these two modules concurrently, the agent must react quickly to dynamic obstacles and, at the same time, plan efficiently through partially observable environments. Hybrid memory models offer improvements in sample efficiency, stability, and flexibility to sensor noise. The drone possesses a perception system: one that acts in real-time to nearby-future threats but plans to figure out the best long-term trajectories.

Both Transformer and LSTM process the embedded sequence in parallel. Both outputs are projected into a common dimension and then mixed by a learnable gate layer that generates element-wise weights, allowing end-to-end gradient flow through both branches and allowing the policy to selectively emphasized short- vs. long-term context. We compare sum, concat+MLP, and gating; the best trade-off for success–collision with sub-millisecond latency overhead is obtained using gating.

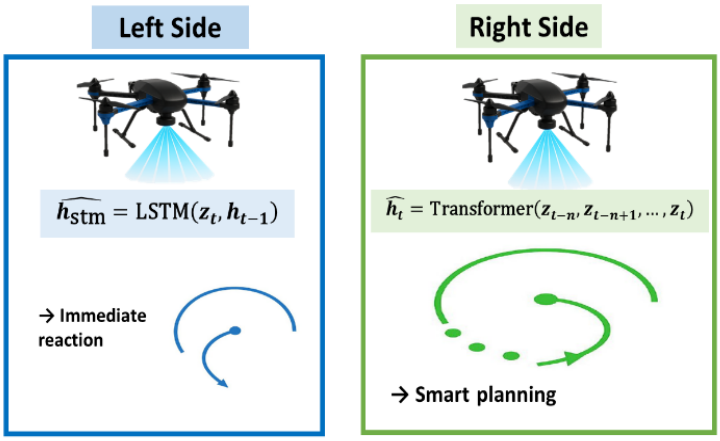

Fig. 4. Illustration of combining LSTM (short-term) and Transformer (long-range)

The LiDAR sensor used in the study has a range of 0.06 meters to 5 meters and allows accurate calculation of distance within this range. The sensor has a horizontal angular resolution of 1 degree with full 360-degree coverage, allowing it to cover the whole environment surrounding it. To get accurate readings, the sensor it includes noise that is Gaussian via a standard deviation of 0 and a standard deviation of 0.01.

The sensor LiDAR operates at a frequency of 20 Hz, so it refreshes its measurements 20 times per second. These requirements enable the LiDAR to effectively detect and measure obstacles within its range without a high level of error due to noise.

3.6. PPO Training Mechanism

We employ the PPO algorithm to optimize the memory-augmented policy under partial observability. The PPO algorithm restricts the policy update to be within a clipped range, which leads to improved training stability. At each training step, PPO maximizes the clipped objective function, defined in (16).

$$J_{PPO}(\theta) = \mathbb{E}_t[min(r_t(\theta)A_t, clip(r_t(\theta),1-\epsilon,1+\epsilon)A_t)] \quad (16)$$

where:

- $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$: the sampling ratio;
- ϵ : the hyperparameter (typically 0.1–0.3) that controls the update range by clipping the ratio to ensure a small update;
- $\widehat{A_t}$: the advantage function is computed using Generalized Advantage Estimation (GAE), as in (17)

$$\widehat{A_t} = \sum_{l=0}^{T} (\gamma\lambda)^l \delta_{t+l} \quad (17)$$

where $\gamma \in (0,1]$ the discount factor, $\lambda \in [0,1]$ Are the controls the bias–variance trade-off? δ_t is the temporal difference error for value estimation and can be computed from the immediate reward and the value estimate of the current and next hidden state, as $\delta_t = r_t + \gamma V(h_{t+1}) - V(h_t)$.

The final training objective consists of three components: policy loss, value loss, and entropy bonus, as shown in (18).

$$L_{total}(\theta) = L^{CLIP}(\theta) - c_1 * L^{VF}(\theta) + c_1 * H[\pi_\theta] \quad (18)$$

where:

- $L^{CLIP}(\theta)$: the clipped surrogate policy loss to be minimized;
- $L^{VF}(\theta)$: the value function loss; and
- $H[\pi_\theta]$: the entropy of the policy.

3.7. Training Setup and Hyperparameters

We present the training configuration, hyperparameters, and parameters applied to our dual-memory policy with LSTM and Transformer components in our PPO-based policy for more efficient temporal feature representation in partially observable environments. The policy network is initiated by a feature encoder as two fully connected hidden units, 256 and 128, respectively, followed by a ReLU activation. The LSTM block consists of a single 128-hidden unit recurrent block that is responsible for paying attention to short temporal correlations. The Transformer encoder, on the other hand, consists of 2 layers, each with 4 self-attention heads and a hidden embedding dimension of 128 so that the model would be able to learn long-term temporal dependencies along the observation sequence. We train the policy having two heads: an actor (stochastic policy) and a critic (state-value estimator). Optimization uses PPO with clipping GAE discount, Adam (learning rate, minibatch size 64, 4 updates per epoch, rollout length 2048, and sequence truncation 128 with carry-over of hidden state. Gradient-norm clipping is also employed for stability and entropy regularization to encourage exploration. Reward components (goal, collision, smoothness, orientation-stability) are normalized/scaled per episode to prevent domination by a single component.

Policy updates for 4 epochs. Entropy regularization with a coefficient c_2 of 0.01 is incorporated to encourage exploration and prevent premature convergence to a suboptimal policy. The PPO clipping threshold is also tuned to support stable policy updates. The training steps are scaled to 3 million to allow sufficient convergence in both the dynamic and partially observable settings. All hyperparameters were empirically tuned through iterative experimentation to balance sample efficiency, policy robustness, and learning stability. Table 1 summaries the training hyperparameters used for the PPO algorithm. Comparison of Value Loss and Explained Variance for PPO-based Architectures during Training. The value head uses the fused recurrent state, i.e., $(V_\phi(h_t))\big(not(V_\phi(o_t))\big)$. The overall training procedure is summarized in Algorithm 1.

Table 1. The training hyperparameter of the PPO algorithm

Parameter	Value
MLP hidden layers	(256, 128)
LSTM hidden size	128
Transformer hidden size	128
Transformer attention heads	4
Activation function	ReLU
Discount Factor	0.99
GAE parameter	0.95
Clipping threshold	0.2
Learning rate	3e-4
Optimizer	Adam
Entropy coefficient c_2	0.01
Value loss coefficient c_1	0.5
Epochs per update	4
Max training steps	3M (~12–15 hrs. on (RTX 3090)

The training was done on a workstation with 32 GB RAM with Ubuntu 20.04, Python 3.8, ROS Noetic, NVIDIA RTX 3060 GPU, an Intel i7 CPU, Gazebo 11, and PyTorch 1.13. Fig. 1 depicts the training procedure of the suggested methodology. Algorithm Compute profile. Dual-memory policy has 0.684 M parameters (≈ 2.7 MB in FP32). With batch size = 4, measured inference latency is ≈1.2 ms/frame on RTX 3090 and ≈6.5 ms/frame on Intel i7 (CPU-only). Peak GPU memory at inference is ≈128 MB, implying feasibility for onboard deployment.

4. Results and Discussion

To evaluate the performance of the suggested memory-augmented learning models for UAV navigation, we conducted four main experiments. The first experiment created a baseline by utilizing the PPO (Feedforward) architecture without temporal memory. The second experiment employed the PPO-LSTM model to enable long-range temporal dependencies. The third experiment made use of the PPO-Transformer model based on the self-attention mechanism to facilitate complex temporal and contextual interactions. The fourth experiment made use of the Hybrid LSTM-Transformer PPO model by combining LSTM with Transformer to provide better temporal memory and contextual sensitivity.

The same metrics were utilized across all models, which included total cumulative reward, collision count, value function loss, policy entropy, trajectory length, and trajectory smoothness. Results are reported as mean ± SD over 5 seeds; significance is assessed with Welch's t-test ($P < 0.05$). We evaluate over several corridor settings (static and dynamic obstacle configurations with varying densities and motion patterns). 'Smoothness' is the integral of path curvature (19) with units 1/m; lower is smoother. Failure analysis. Residual collisions occur primarily when obstacle density exceeds D = 0.12 objects/m² (i.e., ~14 moving obstacles in a 30 × 4 m corridor) and relative obstacle speed exceeds V = 1.0 m/s under partial occlusion. Onboard cost. The hybrid policy has 0.684 M parameters (~2.7 MB FP32) and runs at ~1.2 ms/frame on GPU and ~6.5 ms/frame on CPU (batch = 1), peak memory ~128 MB, indicating practical inference feasibility. The simulation environment was created using

Gazebo 11 alongside PX4-SITL Autopilot for low-level control of flight and MAVROS to enable communication using ROS. Fig. 5 shows the simulated environment in top and front views. This segment addresses UAV configuration, system integration, and simulated environment setup.

Algorithm 1. PPO-LSTM-Transformer Hybrid training pipeline

Fusion of memory branches

Let: $(\widetilde{h_t^L})$ be the LSTM output and $(\widetilde{h_t^T})$ the Transformero utput.

We project them to a common dimension(d) via linear heads:

$$h_t^L = W_L\widetilde{h_t^L},\ h_t^T = W_T\widetilde{h_t^T} \in R^d.$$

A learnable gate produces element − wise weights:

$$g_t = \sigma\left(W_g\left[h_t^L \parallel h_t^T\right] + b_g\right) \in (0,1)^d).$$

The fused state is

$$h_t = g_t \odot h_t^L + (1 - g_t) \odot h_t^T,$$

and gradients flow end − to − end through both branches (no detach).

Require: *p*olicy network π_θ, value function V_ϕ; discount factor; GAE parameter clipping threshold; coefficients c_1(value loss) and c_2(entropy bonus).

For each $t = 1$ to T do:

1. $r_t(\theta) \leftarrow \backslash dfrac\pi_\theta(a_t \mid h_t)\pi_{\theta_{old}}(a_t \mid h_t)$
2. $L_t^{PPO} \leftarrow \min !\backslash Big\left(r_t(\theta)\,\widehat{A}_t,\ clip\backslash big(r_t(\theta),\ 1-\epsilon,\ 1+\epsilon\backslash big)\,\widehat{A_t\backslash}Big\right)$
3. $L_t^{value} \leftarrow \backslash tfrac12\backslash big\left(V_\phi(h_t) - V_{target,t\backslash}big\right)^2$
4. $L_t^{entropy} \leftarrow -\mathcal{H}!\backslash big(\pi_\theta(\cdot \mid h_t)\backslash big)$

enumerate

$$L_{total} \leftarrow \backslash dfrac1T \sum_{t=1}^{T\backslash} Big\left[L_t^{PPO} + c_1\,L_t^{value} - c_2\,L_{t\backslash}^{entropy}Big\right].$$

Update θ, by gradient descenton L_{total}

Return updated π_θ and V_ϕ.

enumerate

The critic is evaluated as $V_\phi(h_t)\left(notV_\phi(o_t)\right)$.

The training environment consists of a corridor room, employed to simulate indoor or limited drone flight, represented as a rectangular space 30 m (length) * 4 m (width) *5 m (height). The corridor is bounded by walls on the left and right and features an open roof to avoid z-axis limiting. The environment comprises static and dynamic obstacles uniformly and randomly distributed to provide challenges to the UAV's collision avoidance and perception in partial observability. The dynamic obstacles have a minimum clearance of 0.4 m on each side and are variably placed along the y-axis to challenge lateral maneuvering.

Overall summary hybrid LSTM–transformer PPO performs the best and crashes less and has less smoothness cost than single-memory baselines in all corridor settings (static and dynamic obstacle

fields). To avoid repetition, key figures are in here, we cover interpretation, failure modes, and when advantages fade.

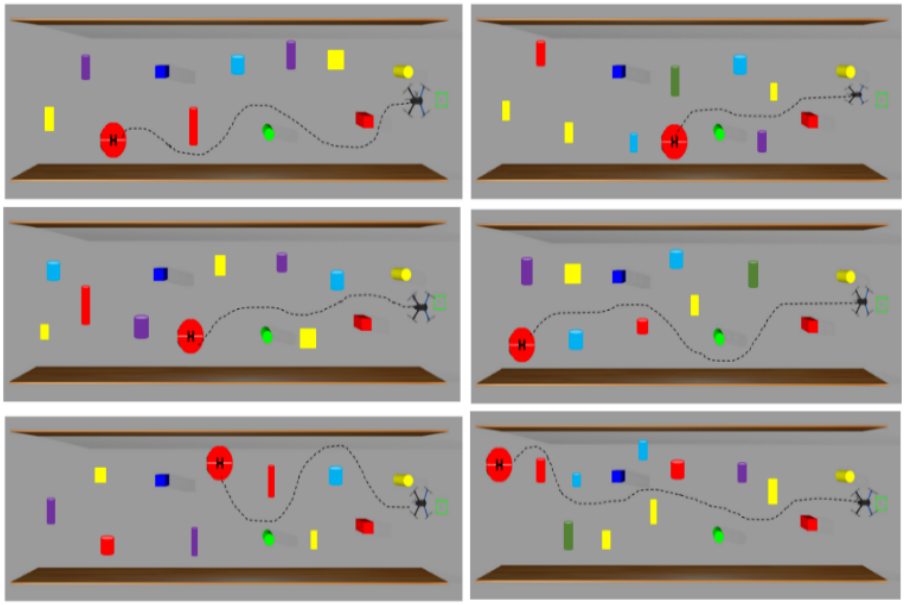

Fig. 5. Training environments featuring six distinct obstacle configurations within a corridor-like structure (30 m × 4 m × 5 m), designed to evaluate autonomous drone navigation in constrained spaces

4.1. Total Reward per Episode

Fig. 6 shows the cumulative reward during training. The hybrid LSTM–Transformer PPO learns fastest and most consistently: the LSTM preserves short-horizon responsiveness under partial occlusions, the Transformer offers long-range context, and the learnable gate adaptively weighs their contributions as obstacle density and motion patterns vary. Feedforward PPO (no memory) has high initial instability low rewards and spikes because it does not have any information accumulation over time; even when rewards do go up later, variance is high, implying slower and less stable convergence. PPO-LSTM improves stability using temporal memory, providing a smoother, monotonic learning curve. PPO-Transformer further helps by modelling long-range dependencies using self-attention, with improved reward levels and convergence smoothness over PPO-LSTM. The hybrid combines the best of both, performing the highest peak reward with the lowest variance, consistent with a good policy for dynamic, partially observable environments (mean ± SD over 5 seeds; Welch's t-test, $p < 0.05$).

4.2. 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. 7 the hybrid works because the LSTM retains near-term reactivity under partial occlusion, and the Transformer contributes long-horizon context; the learnable gate modulates their relative weight with obstacle density and motion patterns. Feedforward PPO, memory-less, starts at a high collision rate of approximately 200 and persists with high fluctuation throughout training with unstable learning. The PPO-LSTM model with temporal memory by using LSTM units experiences an instantaneous and consistent reduction in collisions from about 125 to below 5 with increasing training, exemplifying higher stability and higher awareness of the environment. The PPO-Transformer also improves collision avoidance using self-attention mechanisms to learn long-range dependencies, with collision reduction from around 150 to less than 10 at the end of training. The Hybrid LSTM-Transformer model performs best as it exhibits the greatest temporal sequence modelling and global context consciousness, with it starting with fewer than 77 collisions and reducing to a steep drop to

nearly zero collisions well before training has reached completion. The results stress the function of memory and attention processes in enabling more effective and safe UAV flight, particularly in partially observable and dynamic environments.

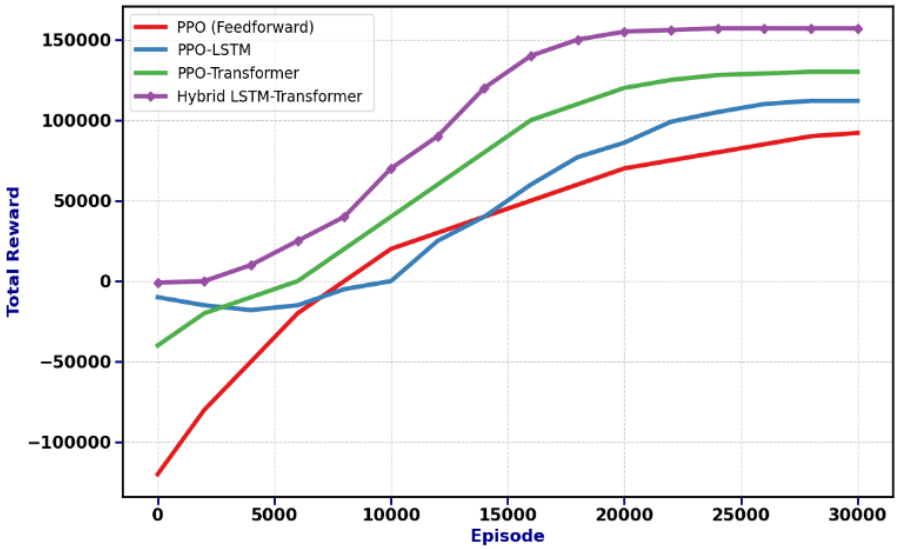

Fig. 6. The accumulated rewards vs episode for four approaches

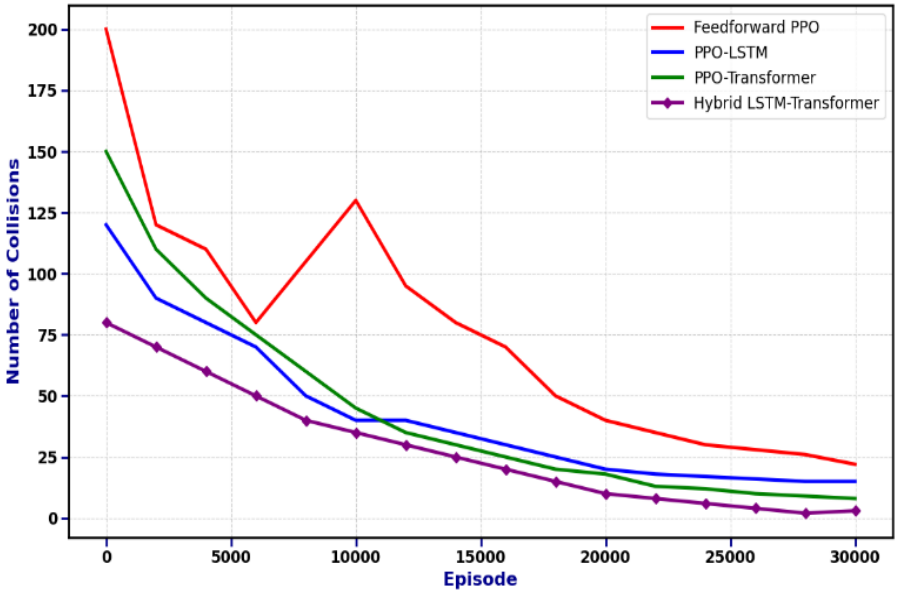

Fig. 7. The number of collisions per episode

4.3. Value Loss over Time

Fig. 8 provides collisions per episode during training. The hybrid Transformer–LSTM PPO achieves the most rapid and reliable collision reduction: LSTM preserves short-term reactivity in partial occlusions, Transformer offers context at long ranges, and the learnable gate modulates their relative influence as obstacle density and motion patterns vary. On the other hand, feedforward PPO (no memory) begins with a very high collision rate (≈200) and remains very volatile, consistent with unstable learning under partial observability. PPO-LSTM (temporal memory) reduces collisions rapidly and monotonically from ~125 to <5, with increased environmental awareness. PPO-Transformer also reduces collisions, from ~150 to <10, by learning long-range dependencies with self-attention. The hybrid starts lower (<77) and drops to ~0 well before training finishes, indicating

improved safety and sample efficiency. Results are mean ± SD over 5 seeds: significance by Welch's t-test ($p < 0.05$).

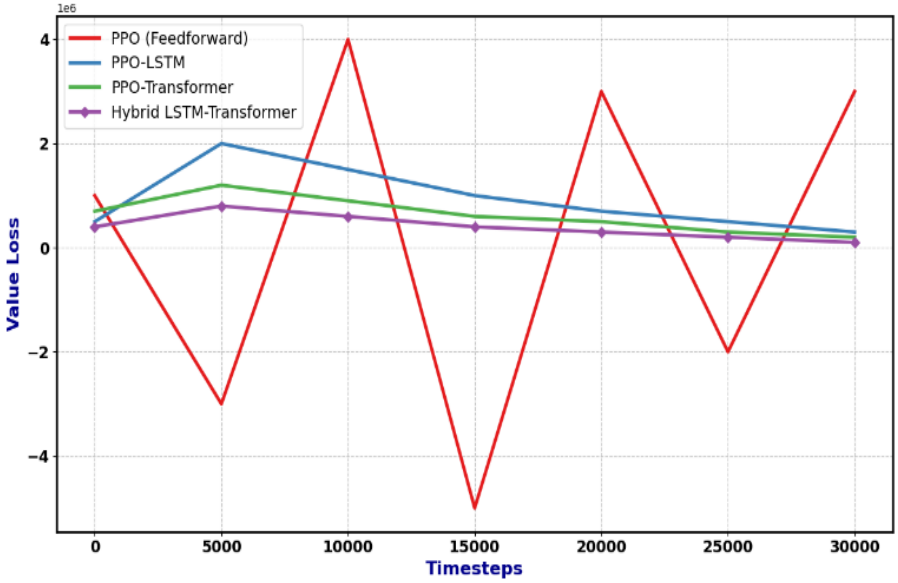

Fig. 8. Value loss per episode

4.4. Policy Entropy During Training

Fig. 9 illustrates policy entropy (randomness of action choice), a heuristic for the exploration-exploitation trade-off. Higher entropy is more exploration; too-fast collapse is premature exploitation and potential suboptimal convergence. All four models begin with high entropy (≈2.5), consistent with RL warm start. Feedforward PPO has the steepest decline in entropy to ≈0.1 soon after training, showing premature exploitation and lower robustness. PPO-LSTM and PPO-Transformer also decay more gradually and plateau at 0.3–0.4 by around ~30k episodes, indicating more balanced and stable learning. The hybrid LSTM–Transformer PPO has the highest entropy during training and converges to ≈0.6, indicating continuing, balanced exploration. Mechanistically, the LSTM preserves near-term reactivity under partial occlusions, the Transformer introduces long-horizon context, and the learnable gate balances their relative influence with obstacle density and motion patterns—dynamically preventing early entropy collapse. Values are mean ± SD over 5 seeds: significance by Welch's t-test ($P < 0.05$).

4.5. Trajectory Smoothness over Time

Fig. 10 illustrates the measure of path smoothness on training episodes for all four models. Path smoothness is a measure of the stability and consistency of the UAV's path as it moves, where lower is smoother, less oscillatory motion. The hybrid works because the LSTM retains near-future reactivity during partial occlusion, and the Transformer incorporates long-horizon context, and the gate, which is learnable, regulates their relative significance with obstacle density and motion trends. The Hybrid LSTM-Transformer PPO demonstrates the lowest smoothness value throughout training, which is a value of about 9, showing the smoothest and most stable flight path for the UAV compared to all other models. This shows the strength of the hybrid model in combining both sequence memory and global attention mechanisms towards more stable decisions and smoother UAV movement control. The PPO-LSTM and PPO-Transformer models then come in with final smoothness levels of around 13 and 15, respectively. The plots of the two models always increase during training, but without the stability of the hybrid. PPO (Feedforward) has the highest levels of smoothness, all more than 24 after training. This means that trajectories of the UAV under the feedforward policy are shorter and less stable, likely since the agent lacks memory components to alleviate temporal dependencies in navigation.

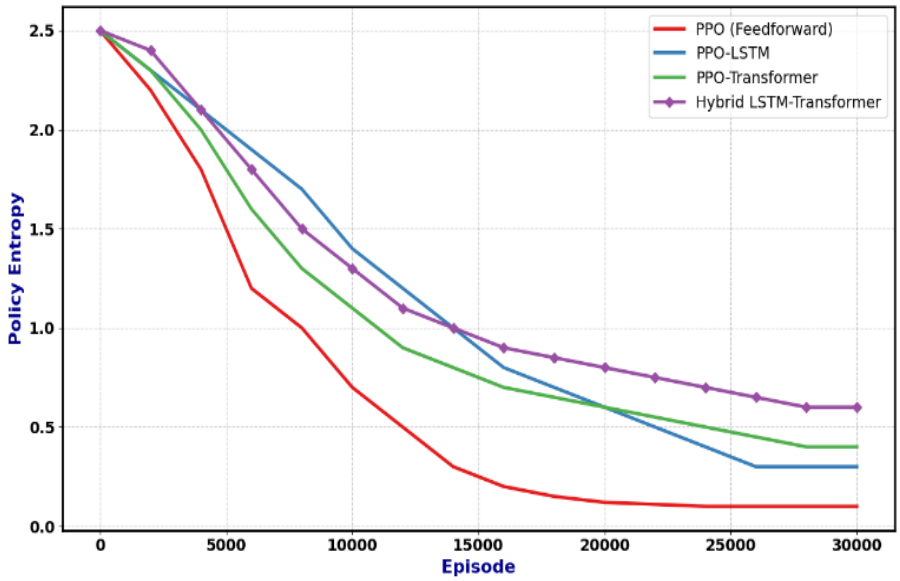

Fig. 9. Policy entropy per episode

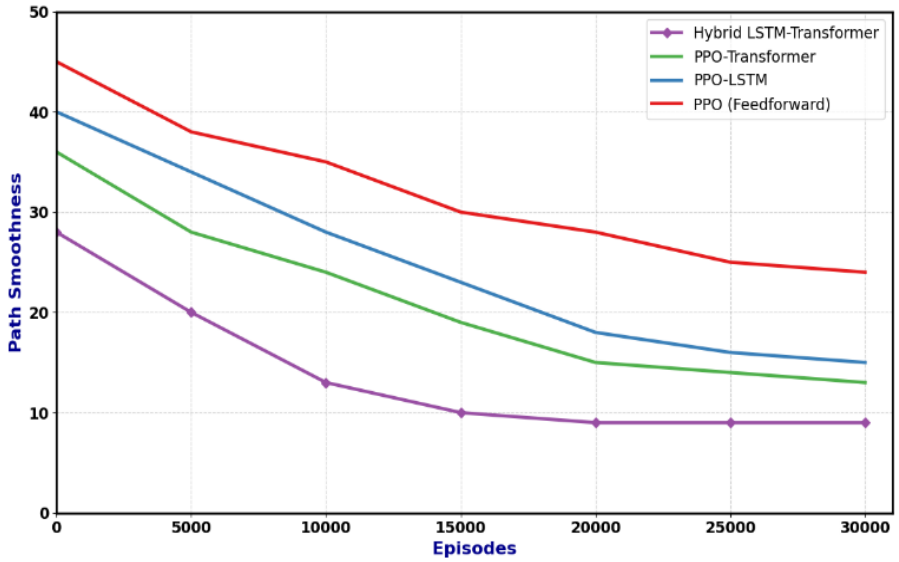

Fig. 10. Trajectory smoothness per episode

In the study, angular motion and curvature change along the trajectory. Smoothness is how gradual in-direction changes are mitigated and is expressed mathematically as follows: Curvature is a measure of how rapidly the direction of a motion is changing. The curve of a 2D path can be calculated in (19).

$$\mathrm{k(t)} = \frac{|\boldsymbol{x'(t)y''(t) - y'(t)x''(t)}|}{(\boldsymbol{x'(t)^2 + y'(t)^2})^{3/2}} \tag{19}$$

where:

- $\boldsymbol{x(t), y(t)}$: these are the trajectory coordinates;
- $\boldsymbol{x'(t), y'(t)}$: the velocity along the x and y directions; and
- $\boldsymbol{x''(t), y''(t)}$: the acceleration along the x and y directions.

Smoothness thresholds: The movement is smooth if the curvature κ(t) is less than some threshold. In our experiments, a smooth motion is defined by having a curvature less than 0.5, which ensures that the motion has gradual direction changes without sudden turns. Motions with a higher curvature of more than 0.5 are less smooth.

The upper bound for angular velocity for a smooth trajectory is computed at 10 degrees per second to allow the UAV ample time to follow a smooth path without abruptly changing its direction of heading. These metrics provide an exact characterization of the path's smoothness and are employed to assess the model's performance in motion stability during training.

4.6. Generalization to an Environment with Static and Dynamic Obstacles

The robustness of the Hybrid LSTM-Transformer PPO model is rigorously challenged in an unseen, dynamic setting specially designed to push the agent's adaptability and collision avoidance capability. Despite training only being done within a static obstacle setting, the test setting includes several moving obstacles, like moving agents and moving objects, to simulate the approximation of real-world complexity.

Fig. 11 one can observe that the Hybrid LSTM-Transformer PPO agent successfully captures this changing environment, predicting and responding to barrier movement correctly. Due to its sophistication, its path is optimal with signs of real-time adaptation and stability of the environment. This dynamic obstacle avoidance proves the benefit of the twin temporal memory and attention mechanism of the hybrid model to make future state predictions and better navigation decisions. On the other hand, the feedforward PPO actor fares badly in this dynamical setting, particularly in cases with partial observability and dynamical change. Overall, these results confirm that the Hybrid LSTM-Transformer PPO achieves significant improvements in navigation resilience, safety, and responsiveness with realistic, time-changing obstacles relevant to real-world autonomous UAV flight.

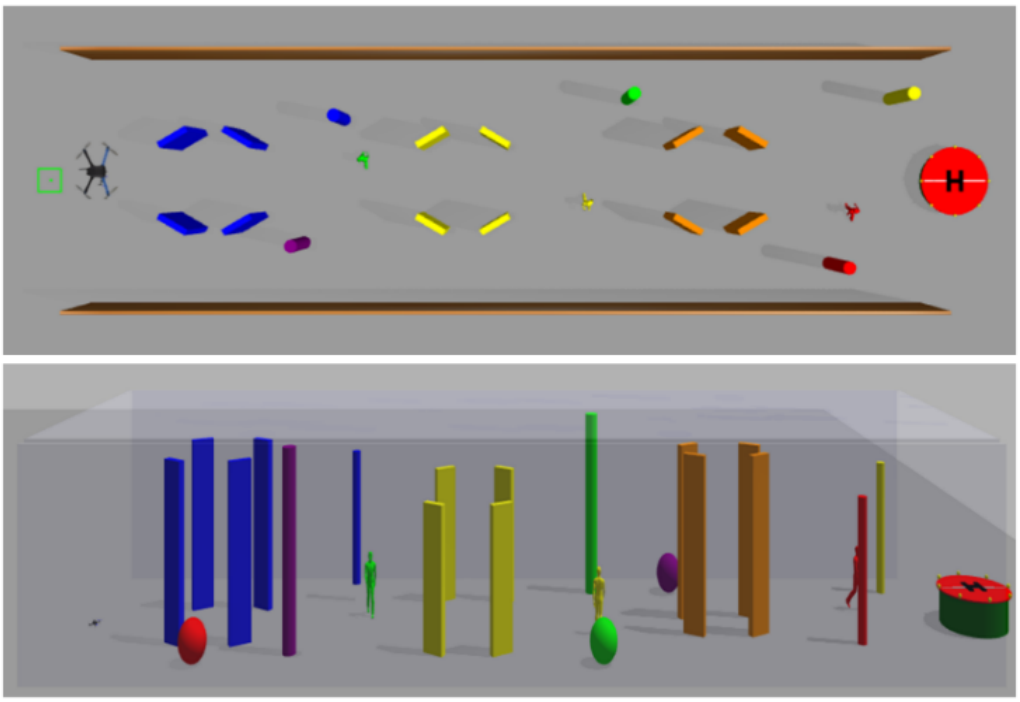

Fig. 11. Top and front view of unseen environment with static and dynamic obstacles

Fig. 12 The following is a sample flight trajectory of the Hybrid LSTM-Transformer PPO agent in the environment that was tested. In the face of temporal uncertainty, the path that is both smooth and direct emphasizes the importance of stable control, allowing the agent to maintain responsiveness to immediate hazards and enabling strategic decision-making.

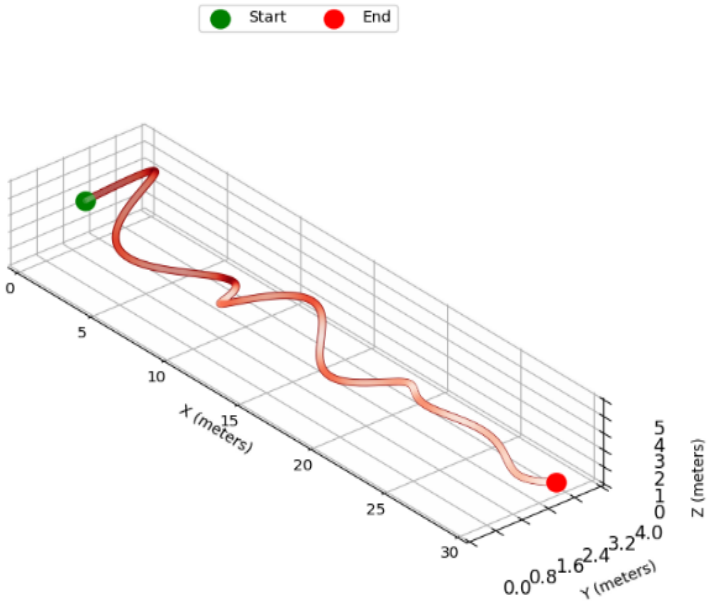

Fig. 12. The drone's navigation trajectory from start to endpoint of the Hybrid LSTM-Transformer PPO agent in the environment that was tested

Fig. 13 shows the effect of architectural complexity on smoothness and efficiency of paths in PPO variants. Vanilla PPO has fluctuating performance, whereas memory-augmented models (PPO-LSTM, PPO-Transformer) produce. Smoother paths. Fused PPO-LSTM-Transformer performs best, with the highest smoothness and path efficiency, enabling anticipatory maneuvering rather than reactive path adjustments.

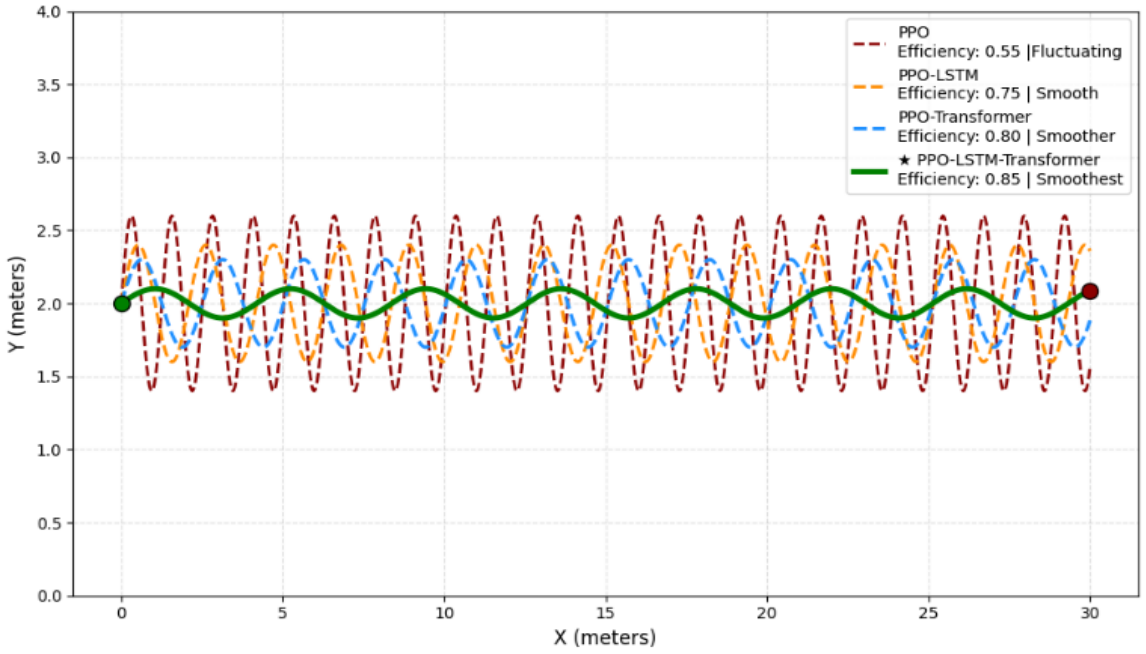

Fig. 13. Effect of architectural complexity on smoothness and efficiency of paths in PPO variants

Fig. 14 illustrates the time taken to reach the goal, evaluating the generalization ability of the models in an unseen environment.

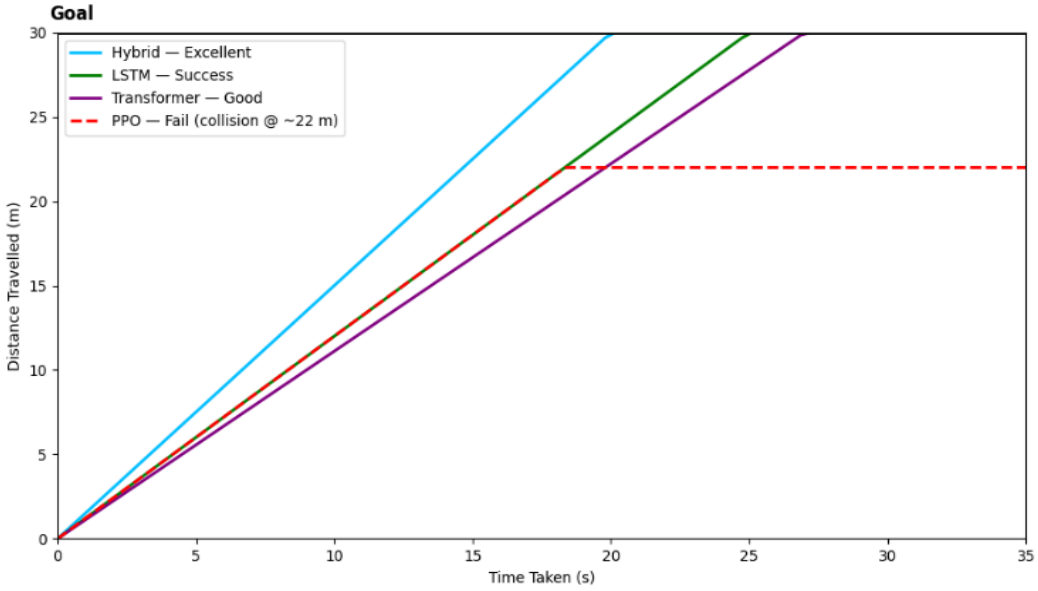

Fig. 14. Time taken to reach the goal

Fig. 15 illustrates the path taken for Hybrid LSTM-Transformer PPO, PPO-LSTM, PPO-Transformer, and PPO baseline.

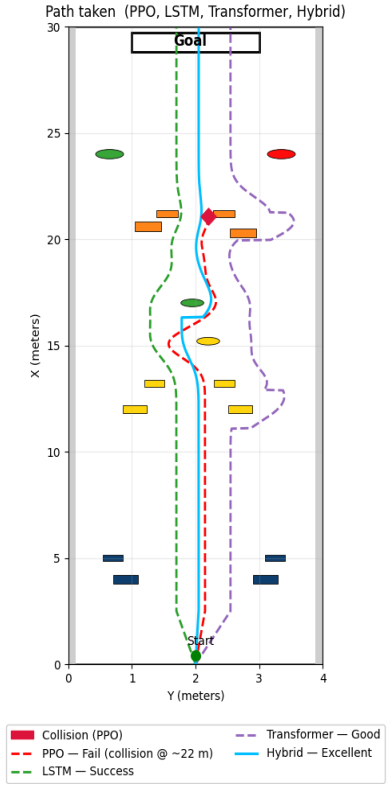

Fig. 15. Path token

The detailed comparison between PPO, PPO-LSTM, PPO-Transformer, and Hybrid LSTM-Transformer PPO 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/Hybrid-LSTM-Transformer-PPO-for-Robust-UAV-Navigation-/tree/main.

Values are presented as mean ± SD over 5 seeds: significance by Welch's t-test ($p<0.05$). Synopsizes principal indicators over models, highlighting dual-memory benefit in success, collisions, episode length, and path efficiency. The hybrid benefits, as LSTM keeps short-term reactivity with partial occlusion, the Transformer gives long-range context, and the learnable gate reweights adaptively both streams with obstacle density and motion.

Table 2. Variant comparison. Mean ± SD over 5 random seeds; each seed is run for 50 test episodes. Success and efficiency: better are higher; collisions, episode length, and smoothness: better is lower.

Metric	PPO	PPO-LSTM	PPO-Transformer	Dual Memory LSTM-Transformer
Total Reward(10^3)	198	115	130	153
Success rate (%) (↑)	66.5 ± 4.72	86.8 ±3.38	79.4 ± 4.04	96.5 ±1.84
Convergence Episode (↓)	29,000 ± 1,450	27,000 ±1,350	25,000 ±1,250	20,000 ±1000
Final Collision//episode (↓)	17.5 ± 1.25	4.0 ± 0.50	7.0 ± 0.50	2.5± 0.25
Initial Collision (first 100 eps ↓)	280 ± 14	200 ±10	150 ±8	80 ±4
Value Loss Variance (qual)	High	Medium, initial rise, then decline	Low to moderate	Very low, highly stable
Final Entropy (policy)	0.15 ± 0.015	0.3± 0.020	0. 4± 0.020	0.6± 0.025
Path Smoothness	Fluctuating	Smooth	Smoother	Smoothest
Generalization to Dynamic(qual)	Fail	Success	Good	Excellent
Episode Length (steps ↓)	210 ± 10.5	170 ±8.5	180 ± 9.0	150 ± 7.5
Path Efficiency	0.55 ± 0.030	0.75 ± 0.025	0.80±0.025	0.85±0.020

5. Conclusion

This work proposed a hybrid memory-based PPO model by combining LSTM and Transformer modules and integrating them for enhancing the navigation performance of UAV when partially observable, noisy sensor readings, and shifting obstacles are encountered. The hybrid model utilizes the LSTM and Transformer to provide short-term memory for fast reactivity and facilitate long-horizon reasoning for strategic planning. The hybrid architecture, which integrates both modules within a unified DRL framework, will promote policy performance in tasks that require both fast responsiveness and long-term foresight. The experiments' outcomes proved the superiority of the hybrid model over single memory structures and baseline PPO. It had the highest success rate of 96.5% compared to 86.8% for LSTM, 79.4% for Transformer, and as low as 15–20% for vanilla PPO. The hybrid system also produced the lowest collisions (2–3 per episode) and the best path efficiency (0.85), with convergence in fewer episodes (20,000) than other methods. The generalization capability of the model was further validated using an unseen, noisy environment. It achieved the highest success rate. On the quality of trajectories, the hybrid model obtained shorter episodes (~150 steps) with successful and smooth navigation trajectories.

The findings confirm the efficacy of a combination of complementary memory mechanisms in reinforcement learning, enhancing UAV navigation performance in an uncertain environment with partial observability conditions. Our assessment is carried out in simulation across corridor-like environments with varying obstacle density and motion patterns. While the hybrid LSTM–Transformer PPO is uniformly better than strong PPO baselines on success, collisions, and smoothness, our claims hold within these environments; real-world assessment and validation across structurally varying scenes (e.g., urban canyons, outdoor GPS-denied environments) are left to future endeavors. Limitations and prospects. We present simulation-only results here and use corridor-type environments; hence, external validity to structurally varying scenes remains to be shown. We also

didn't execute the policy on embedded flight platforms. Future work will (i) evaluate in diverse environments (urban canyon, outdoor GPS-denied), (ii) conduct real-world flights with onboard computation, (iii) include ablations for reward-weight sensitivity and other fusion approaches, and (iv) explore multi-agent settings. These steps will additionally confirm robustness and practical preparedness.

Limitations and future work. Our evaluation is simulation-only and limited to corridor-like environments; external validity to structurally dissimilar scenarios (e.g., urban canyons, outdoor GPS-denied terrain) has not yet been shown. We did not execute on embedded flight hardware, and comparison to more recent memory structures (e.g., memory transformers/GTrXL, neural memory modules) is not yet published. Although we release a compute profile (≈0.684M parameters, ≈2.7 MB FP32; ~1.2 ms/frame on GPU and ~6.5 ms/frame on CPU; peak memory ~128 MB), these are workstation/CPU figures and not substitutes for on-aircraft latency/FPS testing.

Author Contribution: M. A. Haddad and D. R. Khudher contributed to the conceptualization and methodology of this work. M. A. Haddad was responsible for software implementation, training code, experiments, validation, formal analysis, investigation, resources, data curation, visualization, and writing the original draft. D. R. Khudher contributed to validation, writing review and editing, supervision, and project administration. Funding acquisition was not applicable. All authors have read and approved the final manuscript.

Acknowledgement: 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.

Conflicts of Interest: The authors declare no conflict of interest.

References

[1] F. S. Leira, H. H. Helgesen, T. A. Johansen, and T. I. Fossen, "Object detection, recognition, and tracking from UAVs using a thermal camera," *Journal of Field Robotics*, vol. 38, no. 2, pp. 242–267, Mar. 2021, https://doi.org/10.1002/rob.21985.

[2] H. S. Munawar, F. Ullah, S. Qayyum, S. I. Khan, and M. Mojtahedi, "UAVs in disaster management: Application of integrated aerial imagery and convolutional neural network for flood detection," *Sustainability*, vol. 13, no. 14, p. 7547, 2021, https://doi.org/10.3390/su13147547.

[3] M. Abdeh, F. Abut, and F. Akay, "Autonomous navigation in search and rescue simulated environment using deep reinforcement learning," *Balkan Journal of Electrical and Computer Engineering*, vol. 9, no. 2, pp. 92–98, Apr. 2021, https://doi.org/10.17694/bajece.781162.

[4] Z. Fang and A. V. Savkin, "Strategies for optimized UAV surveillance in various tasks and scenarios: A review," *Drones*, vol. 8, no. 5, p. 193, 2024, https://doi.org/10.3390/drones8050193.

[5] W. Alawad, N. Ben Halima, and L. Aziz, "An unmanned aerial vehicle (UAV) system for disaster and crisis management in smart cities," *Electronics*, vol. 12, no. 4, p. 1051, 2023, https://doi.org/10.3390/electronics12041051.

[6] Y. Li, J. An, N. He, Y. Li, Z. Han, Z. Chen, and Y. Qu, "A review of simultaneous localization and mapping algorithms based on lidar," *World Electric Vehicle Journal*, vol. 16, no. 2, p. 56, 2025, https://doi.org/10.3390/wevj16020056.

[7] V. J. Hodge, R. Hawkins, and R. Alexander, "Deep reinforcement learning for drone navigation using sensor data," *Neural Computing and Applications*, vol. 33, no. 6, pp. 2015–2033, Mar. 2021, https://doi.org/10.1007/s00521-020-05097-x.

[8] T. Xu, "Recent advances in rapidly-exploring random tree: A review," *Heliyon*, vol. 10, no. 11, Jun. 2024, https://doi.org/10.1016/j.heliyon.2024.e32451.

[9] S. Gronauer and K. Diepold, "Multi-agent deep reinforcement learning: A survey," *Artificial Intelligence Review*, vol. 55, no. 2, pp. 895–943, Feb. 2022, https://doi.org/10.1007/s10462-021-09996-w.

[10] A. Plaat, *Deep Reinforcement Learning*. Singapore: Springer Nature Singapore Pte Ltd., 2022, https://doi.org/10.1007/978-981-19-0638-1.

[11] M. Wisniewski, P. Chatzithanos, W. Guo, and A. Tsourdos, "Benchmarking deep reinforcement learning for navigation in denied sensor environments," *Journal of Intelligent & Robotic Systems*, vol. 111, no. 3, p. 103, 2025, https://doi.org/10.1007/s10846-025-02309-1.

[12] F. AlMahamid and K. Grolinger, "Autonomous unmanned aerial vehicle navigation using reinforcement learning: A systematic review," *Engineering Applications of Artificial Intelligence*, vol. 115, p. 105321, 2022, https://doi.org/10.1016/j.engappai.2022.105321.

[13] J. Hu, X. Yang, W. Wang, P. Wei, L. Ying, and Y. Liu, "Obstacle avoidance for UAS in continuous action space using deep reinforcement learning," *IEEE Access*, vol. 10, pp. 90623–90634, 2022, https://doi.org/10.1109/ACCESS.2022.3201962.

[14] O. Dogru, H. Y. Aksoy, M. Akkaya, H. Koroglu, and M. Ozkan, "Reinforcement learning approach to autonomous PID tuning," *Computers & Chemical Engineering*, vol. 161, p. 107760, May 2022, https://doi.org/10.1016/j.compchemeng.2022.107760.

[15] H. Wang, L. Wang, X. Gao, X. Yu, C. Lu, and X. Wang, "UAV path planning based on improved artificial potential field method," in *Proc. 2022 Int. Conf. Autonomous Unmanned Syst. (ICAUS 2022)*, Singapore: Springer Nature Singapore, 2023, pp. 2930–2939, https://doi.org/10.1007/978-981-99-0479-2_271.

[16] L. N. Steimle, D. L. Kaufman, and B. T. Denton, "Multi-model Markov decision processes," *IISE Transactions*, vol. 53, no. 10, pp. 1124–1139, 2021, https://doi.org/10.1080/24725854.2021.1895454.

[17] M. Schwenzer, M. Ay, T. Bergs, and D. Abel, "Review on model predictive control: an engineering perspective," *The International Journal of Advanced Manufacturing Technology*, vol. 117, no. 5, pp. 1327–1349, Nov. 2021, https://doi.org/10.1007/s00170-021-07682-3.

[18] D. Olivares, P. Fournier, P. Vasishta, and J. Marzat, "Model-free versus model-based reinforcement learning for fixed-wing UAV attitude control under varying wind conditions," in *Proceedings of the 21st International Conference on Informatics in Control, Automation and Robotics – Volume 1: ICINCO*, 2024, pp. 79–91, SciTePress, INSTICC, https://doi.org/10.5220/0012946600003822.

[19] O. Doukhi and D.-J. Lee, "Deep reinforcement learning for end-to-end local motion planning of autonomous aerial robots in unknown outdoor environments: Real-time flight experiments," *Sensors*, vol. 21, no. 7, p. 2534, 2021, https://doi.org/10.3390/s21072534.

[20] J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. Abbeel, "Trust region policy optimization," *arXiv preprint,* arXiv:1502.05477, 2017, https://arxiv.org/abs/1502.05477.

[21] O. A. Montesinos López, A. Montesinos López, and J. Crossa, *Multivariate Statistical Machine Learning Methods for Genomic Prediction*. Springer International Publishing, 2022, https://doi.org/10.1007/978-3-030-89010-0.

[22] O. Bouhamed, H. Ghazzai, H. Besbes, and Y. Massoud, "Autonomous UAV navigation: A DDPG-based deep reinforcement learning approach," in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2020, pp. 1–5, https://doi.org/10.1109/ISCAS45731.2020.9181245.

[23] D. Liu, W. Wang, L. Wang, H. Jia, and M. Shi, "Dynamic pricing strategy of electric vehicle aggregators based on DDPG reinforcement learning algorithm," *IEEE Access*, vol. 9, pp. 21556–21566, 2021, https://doi.org/10.1109/ACCESS.2021.3055517.

[24] Q. Yang and R. Parasuraman, "A strategy-oriented Bayesian soft actor-critic model," *Procedia Computer Science*, vol. 220, pp. 561–566, 2023, https://doi.org/10.1016/j.procs.2023.03.071.

[25] T. Alotaibi, K. Jambi, M. Khemakhem, F. Eassa, and F. Bourennani, "Deep learning-based autonomous navigation of 5G drones in unknown and dynamic environments," *Drones*, vol. 9, no. 4, p. 249, 2025, https://doi.org/10.3390/drones9040249.

[26] D. Arce, J. Solano, and C. Beltrán, "A comparison study between traditional and deep-reinforcement-learning-based algorithms for indoor autonomous navigation in dynamic scenarios," *Sensors*, vol. 23, no. 24, p. 9672, 2023, https://doi.org/10.3390/s23249672.

[27] A. P. Kalidas, C. J. Joshua, M. A. Quader, S. Basheer, S. Mohan, and S. Sakri, "Deep reinforcement learning for vision-based navigation of UAVs in avoiding stationary and mobile obstacles," *Drones*, vol. 7, no. 4, p. 245, 2023, https://doi.org/10.3390/drones7040245.

[28] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint,* arXiv:1707.06347, 2017, https://arxiv.org/abs/1707.06347.

[29] F. Zeng, and C. Wang, "Visual navigation with asynchronous proximal policy optimization in artificial agents," *Journal of Robotics*, vol. 2020, no. 1, p. 8702962, 2020, https://doi.org/10.1155/2020/8702962.

[30] A. Pendyala, A. Atamna, and T. Glasmachers, "Solving a real-world optimization problem using proximal policy optimization with curriculum learning and reward engineering," in *Machine Learning and Knowledge Discovery in Databases. Applied Data Science Track*, A. Bifet, T. Krilavičius, I. Miliou, and S. Nowaczyk, Eds. Cham: Springer Nature Switzerland, 2024, pp. 150–165, https://doi.org/10.1007/978-3-031-70381-2_10.

[31] T. Czarnecki, M. Stawowy, and A. Kadłubowski, "Cost-effective autonomous drone navigation using reinforcement learning: Simulation and real-world validation," *Applied Sciences*, vol. 15, no. 1, p. 179, 2025, https://doi.org/10.3390/app15010179.

[32] Y. Zhu, W. Z. Wan Hasan, H. R. H. Ramli, N. M. H. Norsahperi, M. S. M. Kassim, and Y. Yao, "Deep reinforcement learning of mobile robot navigation in dynamic environment: A review," *Sensors*, vol. 25, no. 11, p. 3394, 2025, https://doi.org/10.3390/s25113394.

[33] G. Aikins, S. Jagtap, and K.-D. Nguyen, "A robust strategy for UAV autonomous landing on a moving platform under partial observability," *Drones*, vol. 8, no. 6, p. 232, 2024, https://doi.org/10.3390/drones8060232.

[34] A. Singla, S. Padakandla, and S. Bhatnagar, "Memory-based deep reinforcement learning for obstacle avoidance in UAV with limited environment knowledge," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 1, pp. 107–118, 2021, https://doi.org/10.1109/TITS.2019.2954952.

[35] Z. Hu, K. Wan, X. Gao, Y. Zhai, and Q. Wang, "Deep reinforcement learning approach with multiple experience pools for UAV's autonomous motion planning in complex unknown," *Sensors (Switzerland)*, vol. 20, no. 7, pp. 1–20, 2020, https://doi.org/10.3390/s20071890.

[36] K. Ullah, M. Ahsan, S. M. Hasanat, M. Haris, H. Yousaf, S. F. Raza, R. Tandon, S. Abid, and Z. Ullah, "Short-term load forecasting: A comprehensive review and simulation study with CNN-LSTM hybrids approach," *IEEE Access*, vol. 12, pp. 111858–111881, 2024, https://doi.org/10.1109/ACCESS.2024.3440631.

[37] X. Jiawei, Z. Xufang, L. Zhong, and X. Qingtao, "LSTM-DPPO based deep reinforcement learning controller for path following optimization of unmanned surface vehicle," *Journal of Systems Engineering and Electronics*, vol. 34, no. 5, pp. 1343–1358, Oct. 2023, https://doi.org/10.23919/JSEE.2023.000113.

[38] A. Sherstinsky, "Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network," *Physica D: Nonlinear Phenomena*, vol. 404, p. 132306, 2020, https://doi.org/10.1016/j.physd.2019.132306.

[39] F. Hêche, O. Barakat, T. Desmettre, T. Marx, and S. Robert-Nicoud, "Offline reinforcement learning in high-dimensional stochastic environments," *Neural Computing and Applications*, vol. 36, no. 2, pp. 585–598, Jan. 2024, https://doi.org/10.1007/s00521-023-09029-3.

[40] V. G. Nair, J. M. D'Souza, A. C. S., and R. M. Rafikh, "A scoping review on unmanned aerial vehicles in disaster management: Challenges and opportunities," *Journal of Robotics and Control (JRC)*, vol. 5, no. 6, pp. 1799–1826, 2024, https://doi.org/10.18196/jrc.v5i6.22596.

[41] W. Luo, X. Wang, F. Han, Z. Zhou, J. Cai, L. Zeng, H. Chen, J. Chen, and X. Zhou, "Research on LSTM-PPO obstacle avoidance algorithm and training environment for unmanned surface vehicles," *Journal of Marine Science and Engineering*, vol. 13, no. 3, p. 479, 2025, https://doi.org/10.3390/jmse13030479.

[42] A. Rahali and M. A. Akhloufi, "End-to-end transformer-based models in textual-based NLP," *AI*, vol. 4, no. 1, pp. 54–110, 2023, https://doi.org/10.3390/ai4010004.

[43] A. Wei, J. Liang, K. Lin, Z. Li, and R. Zhao, "DTPPO: Dual-transformer encoder-based proximal policy optimization for multi-UAV navigation in unseen complex environments," *Drones*, vol. 8, no. 12, p. 720, 2024, https://doi.org/10.3390/drones8120720.

[44] T. Shi and K. Shide, "A comparative analysis of LSTM, GRU, and Transformer models for construction cost prediction with multidimensional feature integration," *Journal of Asian Architecture and Building Engineering*, pp. 1–16, 2025, https://doi.org/10.1080/13467581.2025.2455034.

[45] A. Güzey, H. Ghazzai, H. Besbes, and Y. Massoud, "Collision-aware cooperative multi-UAV path-planning with hierarchical PPO-LSTM," 2025, https://doi.org/10.21203/rs.3.rs-6490892/v1.

[46] H. Alizadegan, B. Rashidi Malki, A. Radmehr, H. Karimi, and M. A. Ilani, "Comparative study of long short-term memory (LSTM), bidirectional LSTM, and traditional machine learning approaches for energy consumption prediction," *Energy Exploration and Exploitation*, vol. 43, no. 1, pp. 281–301, Jan. 2025, https://doi.org/10.1177/01445987241269496.

[47] J. Zhang, Y. Guo, L. Zheng, Q. Yang, G. Shi, and Y. Wu, "Real-time UAV path planning based on LSTM network," *Journal of Systems Engineering and Electronics*, vol. 35, no. 2, pp. 374–385, Apr. 2024, https://doi.org/10.23919/JSEE.2023.000157.

[48] S. L. Brunton and J. N. Kutz, *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge: Cambridge University Press, 2019, https://doi.org/10.1017/9781108380690.

[49] C. Yan, X. Xiang, and C. Wang, "Towards real-time path planning through deep reinforcement learning for a UAV in dynamic environments," *Journal of Intelligent & Robotic Systems*, vol. 98, no. 2, pp. 297–309, May 2020, https://doi.org/10.1007/s10846-019-01073-3.

[50] Y. Yin, Z. Wang, L. Zheng, Q. Su, and Y. Guo, "Autonomous UAV navigation with adaptive control based on deep reinforcement learning," *Electronics*, vol. 13, no. 13, pp. 1–20, Jun. 2024, https://doi.org/10.3390/electronics13132432.

[51] R. Xie, Z. Meng, L. Wang, H. Li, K. Wang, and Z. Wu, "Unmanned aerial vehicle path planning algorithm based on deep reinforcement learning in large-scale and dynamic environments," *IEEE Access*, vol. 9, pp. 24884–24900, 2021, https://doi.org/10.1109/ACCESS.2021.3057485.

[52] H. Samma and S. El-Ferik, "Autonomous UAV visual navigation using an improved deep reinforcement learning," *IEEE Access*, vol. 12, pp. 79967–79977, 2024, https://doi.org/10.1109/ACCESS.2024.3409780.

[53] M. Su, H. Chai, C. Zhao, Y. Lyu, and J. Hu, "Lightweight obstacle avoidance for fixed-wing UAVs using entropy-aware PPO," *Drones*, vol. 9, no. 9, p. 598, Aug. 2025, https://doi.org/10.3390/drones9090598.

[54] S. Ibrahim, M. Mostafa, A. Jnadi, H. Salloum, and P. Osinenko, "Comprehensive overview of reward engineering and shaping in advancing reinforcement learning applications," *IEEE Access*, vol. 12, pp. 175473–175500, 2024, https://doi.org/10.1109/ACCESS.2024.3504735.

[55] A. A. Darwish and A. Nakhmani, "Drone navigation and target interception using deep reinforcement learning: A cascade reward approach," *IEEE Journal of Indoor and Seamless Positioning and Navigation*, vol. 1, pp. 130–140, 2023, https://doi.org/10.1109/JISPIN.2023.3334690.

[56] A. F. ud Din, I. Mir, F. Gul, S. Mir, N. Saeed, T. Althobaiti, S. M. Abbas, and L. Abualigah, "Deep reinforcement learning for integrated non-linear control of autonomous UAVs," *Processes*, vol. 10, no. 7, p. 1307, 2022, https://doi.org/10.3390/pr10071307.

[57] S. Zhao, W. Wang, J. Li, S. Huang, and S. Liu, "Autonomous navigation of the UAV through deep reinforcement learning with sensor perception enhancement," *Mathematical Problems in Engineering*, vol. 2023, no. 1, p. 3837615, 2023, https://doi.org/10.1155/2023/3837615.

[58] A. T. Azar, A. Koubaa, N. A. Mohamed, H. A. Ibrahim, Z. F. Ibrahim, M. Kazim, A. Ammar, B. Benjdira, A. M. Khamis, I. A. Hameed, and G. Casalino, "Drone deep reinforcement learning: A review," *Electronics*, vol. 10, no. 9, p. 999, 2021, https://doi.org/10.3390/electronics10090999.

[59] J. Zhao, J. Sun, Z. Cai, L. Wang, and Y. Wang, "End-to-end deep reinforcement learning for image-based UAV autonomous control," *Applied Sciences*, vol. 11, no. 18, p. 8419, Sep. 2021, https://doi.org/10.3390/app11188419.

[60] J. Zhao, H. Liu, J. Sun, K. Wu, Z. Cai, Y. Ma, and Y. Wang, "Deep reinforcement learning-based end-to-end control for UAV dynamic target tracking," *Biomimetics*, vol. 7, no. 4, p. 197, 2022, https://doi.org/10.3390/biomimetics7040197.