Recurrent Actor-Critic Navigation of Unmanned Aerial Vehicles in Arbitrary-Shape Obstacle-Dense Environments via Temporally-Augmented Observation

Gabriele Gemignani, Adolfo Perrusquía, Lorenzo Pollini, Antonios Tsourdos

***Abstract*— Reactive navigation of Unmanned Aerial Vehicles (UAVs) in cluttered environments using only local sensor feedback is severely hampered by partial observability: irregularly-shaped obstacles can disappear from the sensor's instantaneous field of view, causing wrap-around collisions and local-minima traps. However, existing Reinforcement Learning approaches only partially address these challenges, as they commonly assume regularly shaped, often circular or convex, obstacles. This work proposes a lightweight recurrent extension of the Twin Delayed Deep Deterministic Policy Gradient (TD3) algorithm that tackles navigation in cluttered environments populated with arbitrary-shape obstacles. By augmenting the agent's observation with a sliding window of recent states, the resulting Temporally-Augmented Observation TD3 (TAO-TD3) feeds stacked observations through a recurrent feature backbone, enabling the policy to better infer obstacle structure from temporal information. Benchmark evaluations across five environments with increasing obstacle density and count demonstrate that TAO-TD3 consistently reduces collisions and improves goal-reaching rates relative to memory-less Reinforcement Learning.**

I. Introduction

Goal-oriented Unmanned Aerial Vehicles (UAVs) navigation in obstacle-dense environments increasingly relies on data-driven methods based on local perception. Classical path-planning algorithms require complete, pre-built maps of the workspace and scale poorly with grid resolution, limiting their applicability in unknown, dynamic, or frequently reconfigured scenarios [1], [2]. Model-free Reinforcement Learning (RL) addresses this limitation by mapping local sensor measurements directly to control commands, enabling policies to adapt online to previously unseen environments without relying on an explicit map [3]. Deep RL has been successfully applied to UAV navigation [4]–[8], as well as to multi-agent coordination [9]–[11] and dynamic obstacle avoidance [12].

Project co-funded by the European Union – Next Generation EU – under the National Recovery and Resilience Plan (NRRP), Mission 4 Component 1 Investment 4.1 – Decree No. 118 (2nd March 2023) of Italian Ministry of University and Research – Concession Decree No. 2333 (22nd December 2023) of the Italian Ministry of University and Research, Project code D93C23000450005, within the Italian National Program PhD Programme in Autonomous Systems (DAuSy).

Gabriele Gemignani, Adolfo Perrusquía, and Antonios Tsourdos are with the Faculty of Engineering and Applied Sciences, Cranfield University, MK43 0AL, Bedford, UK.

Gabriele Gemignani and Lorenzo Pollini are with the University of Pisa, Dept. of Information Engineering, 56126 Pisa, Italy. Gabriele Gemignani is a PhD student of the DAuSy National Programme for Polytechnic of Bari, Dept. of Electrical and Information Engineering, Via Re David 200, 70125 Bari, Italy, email: g.gemignani@phd.poliba.it

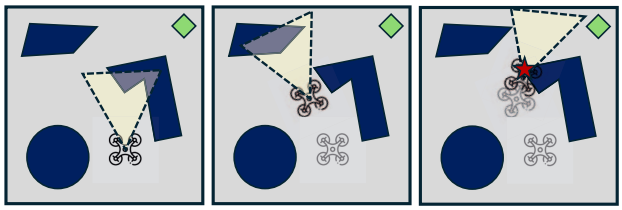


Fig. 1: Illustration of the *wrap-around effect*. As the UAV only detects some of the edges of an irregularly-shaped obstacle (left), it wraps around it, losing the other unseen edges from its sensor's field of view (yellow cone, center). When it subsequently turns towards the target, it collides with the unperceived obstacle's edge (right).

Most existing approaches equip the UAV with a local range sensor, such as a camera processed through convolutional feature extractors [7] or a multi-beam rangefinder [10], and train a feedforward neural network that selects actions based solely on the current observation. This reactive paradigm performs well when obstacles are circular or convex, as a single observation is often sufficient to describe the surroundings. However, in environments with irregular, non-convex, or densely packed obstacles, a single observation captures only part of the obstacle boundary within the sensor's limited field of view (FOV). As the UAV moves, previously observed surfaces leave the sensor cone while new, previously occluded facets appear, a phenomenon known as the *wrap-around effect* (Fig. 1). As a result, a memory-less policy may steer the vehicle into an unseen extension of the same obstacle it was trying to avoid. This issue becomes more pronounced when obstacle edges are sharp or when multiple non-convex objects are closely spaced, as consecutive observations can change abruptly and provide conflicting information. This failure mode is largely overlooked in works that approximate obstacles as circular or convex shapes [12]–[14].

Problem Statement. *Memory-less RL policies cannot retain past sensory information and are therefore vulnerable to wrap-around collisions, leading to degraded navigation performance in environments with irregular and non-convex obstacles.*

A. Partially Observable Navigation: Related Works

Recurrent neural architectures provide a mechanism to retain temporal information rather than relying solely on the current observation. Long Short-Term Memory (LSTM) networks

have been integrated into RL frameworks for UAV obstacle avoidance, showing reduced collision rates under limited sensor coverage [15], [16]. Singla et al. [17] explicitly model UAV navigation as a partially observable problem and propose a Deep Recurrent Q-Network with temporal attention, demonstrating that leveraging observation history improves collision-free navigation distance. Hodge et al. [18] report similar improvements using Proximal Policy Optimization with LSTM cells in indoor environments. Gated Recurrent Units (GRUs) have also been adopted to encode temporal information with lower computational cost [19]. More recently, attention- and transformer-based approaches have been explored to capture long-range spatial-temporal dependencies in navigation tasks [13], [20]. Overall, these works highlight the importance of temporal memory in mitigating the effects of partial observability.

B. Contributions and Paper Outline

This paper investigates goal-oriented UAV navigation in 2D environments populated by randomly generated, irregularly shaped obstacles, sensed through a multi-beam rangefinder. We formulate the task as a Partially Observable Markov Decision Process (POMDP) [21], which motivates the use of recurrent RL architectures. We employ TAO-TD3, a Temporally-Augmented-Observation extension of TD3 in which a fixed-length sliding window of past states is processed by a shared LSTM backbone for both actor and critic. Originally introduced in [22], TAO-TD3 is examined here for the first time in environments with arbitrary-geometry obstacles, specifically to mitigate the wrap-around ambiguity induced by irregular shapes, and is evaluated across maps with varying obstacle densities. Benchmark results demonstrate that TAO-TD3 achieves consistent improvements in collision avoidance and goal-reaching performance over feedforward RL baselines.

The remainder of this paper is organized as follows. Section II describes the problem formulation and environment modeling. Section III presents the proposed TAO-TD3 architecture and training scheme. Section IV reports the numerical results, and Section V concludes the paper and outlines future work.

II. Problem Setting

A. Scenario Definition and Problem Objective

Consider a square 2D map of side length L. Let $\mathbf{p}_k = (x_k, y_k)$ denote the UAV position and θ_k its yaw angle at discrete time k. It must be noted that the planar assumption is not restrictive for rotary-wing UAVs: these vehicles typically cruise at a fixed altitude and regulate height independently from the horizontal decision-making loop. The vehicle follows unicycle kinematics controlled via linear v_k and angular velocity ω_k, continuous and bounded respectively in $[0, \bar{v}]$ and $[-\bar{\omega}, \bar{\omega}]$:

$$\begin{aligned} \mathbf{p}_{k+1} &= \mathbf{p}_k + \Delta T\, v_k \begin{bmatrix} \cos\theta_k \\ \sin\theta_k \end{bmatrix}, \\ \theta_{k+1} &= \theta_k + \Delta T\, \omega_k, \end{aligned} \quad \text{s.t.} \begin{cases} 0 \le v_k \le \bar{v}, \\ |\omega_k| \le \bar{\omega}. \end{cases} \tag{1}$$

where ΔT is the sampling period. The problem's objective consists of finding a sequence of control commands that, from a random initial UAV's pose $\mathbf{p}_0$ and a randomly positioned target $\mathbf{p}^* = (x^*, y^*)$, produces a collision-free, minimum-time path to the goal.

Collisions occur either with the map boundaries or with obstacles. Obstacles are randomly generated as irregular polygons whose vertices are sampled stochastically around uniformly placed centers. The total number of obstacles Z and the fraction ϱ of the map area they occupy are design parameters that jointly control the scenario difficulty. Individual obstacle areas are Gaussian-distributed around $\varrho L^2/Z$, and placements are resampled whenever they overlap with the start or goal positions.

B. Partially Observable Markov Decision Process

Because the agent perceives the environment through a limited FOV rangefinder, the true environment state (map layout outside the sensor's FOV) is never fully accessible. The navigation task is therefore modeled as a POMDP operating at discrete time samples ΔT [21].

Action space. This represents the output of the navigation policy. The action at each time step is $\mathbf{u}_k = [v_k,\, \omega_k]$, i.e. the linear and angular velocities which drive the agent's kinematics via (1).

Observation space. The observation $\mathbf{o}_k = [\mathbf{o}_k^g \mid \mathbf{o}_k^\ell]$ is composed of a *global* and a *local* component. The global component provides absolute information around the relative target's location:

$$\mathbf{o}_k^g = \Big[d_k/L,\ (\cos\psi_k + 1)/2,\ (\sin\psi_k + 1)/2 \Big] \in [0,1]^3 \tag{2}$$

where $d_k = \|\mathbf{p}_k - \mathbf{p}^*\|_2$ is the Euclidean distance to the goal and $\psi_k = \mathrm{atan2}(y^* - y_k,\, x^* - x_k) - \theta_k$ is the heading error. The local component relies on a multi-beam rangefinder that measures the distance between the UAV and nearby obstacles [13]. Denoting by N the number of beams uniformly distributed around the UAV's yaw angle, over the symmetric angular sector $[\theta_k - \varphi,\, \theta_k + \varphi]$, each beam returns the quantity:

$$s^i = \begin{cases} \bar{s}, & \text{if beam } i \text{ intercepts no obstacles} \\ \text{distance} + \mathcal{N}(0,\ \sigma^s), & \text{otherwise} \end{cases} \tag{3}$$

with $\bar{s}$ the maximum sensor range and σ^s the sensor's noise. To reduce dimensionality without discarding safety-critical information, the N beams are partitioned into B equal angular groups and only the minimum reading per group is retained:

$$\mathbf{o}_k^\ell = \big[\, l_k^1/\bar{s},\ \ldots,\ l_k^B/\bar{s} \,\big] \in [0,1]^B, \quad l_k^b = \min_{i \in \mathcal{G}_b} s_k^i \tag{4}$$

where $\mathcal{G}_b$ is the index set of beams belonging to group $b \in \{1,\ \ldots,\ B\}$. Note that, from (2) and (4), all entries of $\mathbf{o}_k$ are $[0,1]$-normalized to be a compatible input for the neural network training.

Reward function. The scalar reward r_k combines sparse terminal signals with continuous shaping terms that encode the desired navigation behavior during training:

$$r_k = \begin{cases} +R & \text{if goal} \\ -R & \text{if collision} \\ \alpha\, v_k \;-\; \beta\,|\omega_k| \;-\; \lambda\,\Gamma\big(\min_b\, l_k^b\big) & \text{otherwise} \end{cases} \quad (5)$$

with the proximity penalty $\Gamma(c) = \max(1 - c,\ 0)$, which activates when the closest detected obstacle is within a threshold distance of one meter. The terminal reward $R = 100$ determines task success or failure and the restart of a new scenario. The three shaping terms serve complementary roles: the velocity coefficient $\alpha = 1$ encourages faster forward progress and thus, indirectly, shorter task duration; the angular-rate penalty $\beta = 0.5$ promotes smooth trajectories; the proximity weight $\lambda = 0.5$ rewards obstacle clearance and safe around-the-target (wrap-around) maneuvers.

III. Recurrent Actor-Critic Architecture and Training

This section presents the core contribution: an LSTM-featured actor-critic architecture that equips TD3 with memory through Temporally-Augmented Observation (TAO).

A. Temporally-Augmented Observation

At each time step k of a training episode, the agent maintains a First-In-First-Out (FIFO) queue $\mathcal{Q}$ of fixed capacity H. Upon episode reset, $\mathcal{Q}$ is filled with H copies of the initial observation $\mathbf{o}_0$. Then, each new observation $\mathbf{o}_k$ is pushed into $\mathcal{Q}$, evicting the oldest entry. We define the TAO as a matrix $\widetilde{\mathbf{O}} \in [0,1]^{H\times|\mathbf{o}|}$, with $|\mathbf{o}| = 3 + B$ the single-step observation dimension, obtained by vertically stacking the queue contents:

$$\widetilde{\mathbf{O}}_k = \begin{cases} \big[\,\mathbf{o}_0;\ \ldots;\ \mathbf{o}_0\,\big], & \text{if } k = 0 \\ \big[\,\mathbf{o}_{k-H+1};\ \ldots;\ \mathbf{o}_k\,\big], & \text{if } k \geq 1 \end{cases} \quad (6)$$

The resulting representation provides the policy network with a finite temporal context from which obstacle shape and motion trends can be implicitly inferred, without requiring explicit state estimation or map building.

B. Actor-Critic Architecture

The TAO-TD3 algorithm utilizes an actor-critic architecture, depicted in Fig. 2, with a recurrent feature-extraction backbone that operates on the TAO $\widetilde{\mathbf{O}}$.

Recurrent backbone. Each row of $\widetilde{\mathbf{O}}$ (i.e., each historical observation) is independently mapped to a feature vector by a temporal MLP (two layers, 128–64 units). The resulting sequence of H feature vectors is fed to a single-layer LSTM, whose final hidden state summarizes the temporal context encoded in the past trajectory [23].

Actor head. An MLP (two layers, 400–300 units) maps the last LSTMS's hidden state to the continuous policy output $\pi_\zeta(\widetilde{\mathbf{O}})$, parametrized by actor weights ζ. A $\tanh$ function is applied to produce a control bounded between -1 and 1,

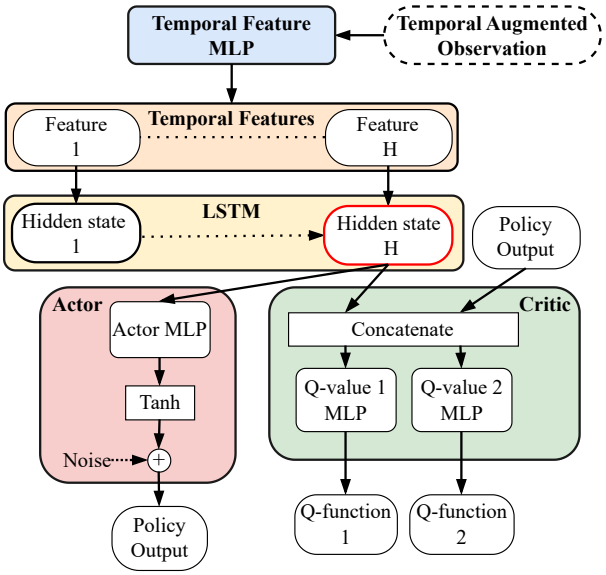


Fig. 2: TAO-TD3 actor–critic architecture. The augmented observation $\widetilde{\mathbf{O}}$ is first processed time step–wise by a temporal feature MLP and then by a single-layer LSTM, whose final hidden state summarizes the last H observations. The actor MLP maps LSTM's last hidden state to the continuous control action, whereas two critic MLPs concatenate it with a policy output from the replay buffer to yield twin Q-value estimates. LeakyReLU activations are used in all hidden layers.

then subsequently rescaled to the admissible control range in (1). During training, zero-mean clipped Gaussian noise is added to $\mathbf{u}$ for exploration.

Critic heads. Two independent MLPs (400–300 units each) receive the concatenated vector of the LSTM's final hidden state and the actor's policy, and output scalar Q-value estimates $Q_{\phi_1}(\widetilde{\mathbf{O}}, \mathbf{u})$ and $Q_{\phi_2}(\widetilde{\mathbf{O}}, \mathbf{u})$, with ϕ_1 and ϕ_2 the critic weights, implementing the twin-critic mechanism of TD3.

Implementation Detail. During training, a minibatch of TAO inputs has shape $(\text{batch size}, H, |\mathbf{o}|)$. To process all H temporal rows efficiently, the actor and critics reshape this tensor to $(\text{batch size} \times H, |\mathbf{o}|)$, apply the feature MLP independently to each row, and then reshape it back to $(\text{batch size}, H, \text{feature dimension})$ before feeding the sequence to the LSTM layer.

C. TAO-TD3 Training scheme

The training procedure follows the standard TD3 pipeline [24], with a single modification to the structure of the replay buffer. Because the actor and critics operate on TAO inputs, the replayed transitions must also contain these augmented states rather than single-step observations. The control $\mathbf{u}_k = \pi_\zeta(\widetilde{\mathbf{O}}_k)$, rescaled to match the control saturations (1), produces the environment transition:

$$\langle \widetilde{\mathbf{O}}_k,\ \mathbf{u}_k,\ r_k,\ \widetilde{\mathbf{O}}_{k+1} \rangle \quad (7)$$

which is subsequently stored in the replay buffer. This replaces the conventional tuple $(\mathbf{o}_k, \mathbf{u}_k, r_k, \mathbf{o}_{k+1})$ used in

vanilla TD3. Since training is performed off-policy, once the episode terminates, mini-batches of tuples (7) are sampled uniformly from the buffer and used to compute the actor and critics losses. The two critics Q_{ϕ_1} and Q_{ϕ_2} are optimized by minimizing the mean-squared Bellman residual:

$$\begin{aligned} \mathcal{L}_j^Q(\phi_j) &= \mathbb{E}\Big[(Q_{\phi_j}(\widetilde{\mathbf{O}}_k, \mathbf{u}_k) - y_k)^2\Big], \\ y_k &= r_k + \gamma \min_{j=1,2} Q_{\phi'_j}\Big(\widetilde{\mathbf{O}}_k,\, \pi_{\zeta'}(\widetilde{\mathbf{O}}_{k+1}) + \epsilon\Big) \end{aligned} \tag{8}$$

with discount factor γ, target networks (ϕ'_j, ζ'), and clipped white noise ϵ. The actor π_ζ is updated, at lower frequency than the critics for stability reasons, by maximizing the first critic's estimate:

$$\mathcal{L}^\pi(\zeta) = \mathbb{E}\Big[Q_{\phi_1}\big(\widetilde{\mathbf{O}}_k,\, \pi_\zeta(\widetilde{\mathbf{O}}_k)\big)\Big] \tag{9}$$

At the end of the single training iteration, target-network weights (ϕ'_j, ζ') are soft-updated with coefficient τ, i.e.:

$$\phi'_j \leftarrow \tau\phi_j + (1-\tau)\phi'_j, \qquad \zeta' \leftarrow \tau\zeta + (1-\tau)\zeta'.$$

A final consideration concerns the choice of the optimizer. Different map clutter levels induce different observation patterns, which increases the risk of overfitting to the specific topographies encountered during training. To promote generalization across environments with varying obstacle densities, both the actor and critics are optimized using AdamW, whose decoupled weight-decay mechanism provides effective regularization against overfitting [25].

IV. Numerical Results

A. Simulation and Training Setup

To systematically assess robustness under varying clutter conditions, we define five benchmark $10 \times 10\,\text{m}$ maps, each configured with a specific obstacles count Z and density ϱ (Table I). The *Training* environment was selected to be mildly cluttered, enabling us to detect potential overfitting, typically manifested as pronounced performance drops when transferring to increasingly complex environments. We additionally tested a *Dynamic* environment, in which a subset of obstacles moves over time with a Random walk model with velocities $\in [-0.2,\ 0.2]$, providing a challenging yet informative test case. All other fixed environment parameters used in simulation are reported in Table II.

During training, we set the TAO queue length to 20, corresponding to approximately 6 s of past trajectory given the sampling time $\Delta T = 0.3\,\text{s}$. This choice was fine-tuned but also informed by the average episode duration observed in the *Training* environment. We evaluated global performance metrics (success and collision rate) and compared our TAO-TD3 agent against standard RL feedforward baselines, namely TD3 [24] and Soft Actor–Critic (SAC) [26]. These baselines operate under a single-step observation input and do not feature the LSTM layer in the actor-critic architecture. The latter was included as a stochastic baseline policy as opposed to the TD3 paradigm. Common training hyperparameters were chosen equally among the different algorithms for the sake of a fair comparison (Table III).

TABLE I: Benchmark Environment: Obstacle Configurations

Scenario	Obstacles (Static/Dynamic)	Density ϱ
Static 1	7/0	0.20
Training	8/0	0.25
Static 2	10/0	0.30
Static 3	12/0	0.35
Dynamic	6/6	0.20

TABLE II: Environment and Sensor Simulation Parameters

Max. linear velocity $\bar{v}$	0.5 m/s
Max. angular velocity $\bar{\omega}$	1 rad/s
Sampling period ΔT	0.3 s
Rangefinder beams N	180
Max. range $\bar{s}$	7 m
Field-of-view half-angle φ	$\pi/2$
Measurement noise σ^s	0.08
Beam groups B	20

B. Training Convergence

Fig. 3 depicts the collision, goal rate and the episodic reward over $10\,000$ training episodes. The evaluation is performed on 30 random scenarios sampled at fixed intervals from the *Training* Environment. TAO-TD3 converges to a notably lower collision rate and higher goal rate and episodic reward than both feedforward baselines. While TD3 and SAC plateau after roughly half of the training process, reaching a goal rate of about 89%, TAO-TD3 continues to improve, ultimately achieving approximately 92% of goal rate with only 4% of collisions. No relevant difference is observed between the baseline policies, suggesting that the challenges caused by partial observability cannot be tackled simply by a different algorithmic choice.

C. Monte-Carlo Evaluation across Benchmark Environments

Each trained policy was evaluated over 3 000 randomized episodes per environment, using identical seeds across al-

TABLE III: Training Hyperparameters

Common Hyperparameters	
Actor / Critic learning rate	10^{-4}
Mini-batch size	64
Replay buffer capacity	5×10^4
Critic updates per episode	10
Actor updates per episode	5
Discount factor γ	0.99
Soft-update coefficient τ	0.005
Max episode duration	300
Algorithm-Specific Hyperparameters	
Temperature learning rate (SAC)	10^{-4}
Log-std clipping (SAC)	10^{-3}
Policy noise std. (TD3 / TAO-TD3)	0.15

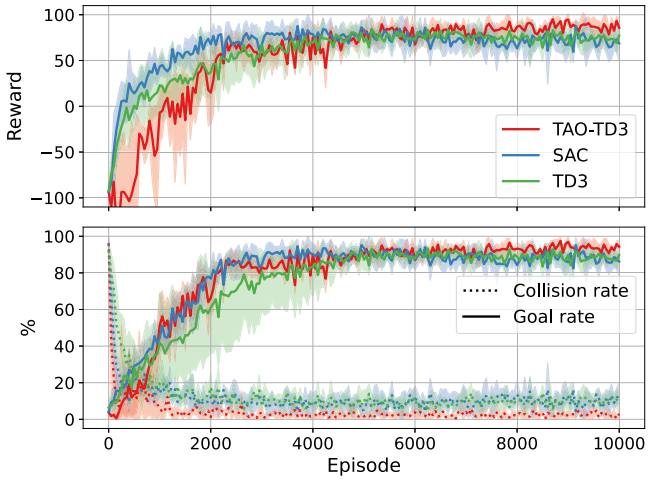


Fig. 3: Learning curve of the *Training* environment: the solid lines show the mean values across 10 training seeds, and the shaded areas indicate the 10th and 90th quantiles.

gorithms. Fig. 4 reports the key metrics averaged across all five benchmarks. TAO-TD3 achieves the highest goal rate and lowest collision rate in every environment. As regards the latter, the advantage widens as obstacle density increases, confirming that temporal memory is most beneficial in heavily cluttered maps. Notably, TAO-TD3 also scores the highest average forward velocity, over all successful episodes in every environment, demonstrating that its reduced collision rate is not obtained at the expense of overly conservative or slower maneuvers.

D. Qualitative Analysis

Fig. 5 presents representative trajectories that highlight the behavioral differences between the policies. We include several scenario types in which TAO-TD3 consistently outperforms the feedforward baselines. i) **Wrap-Around**: As

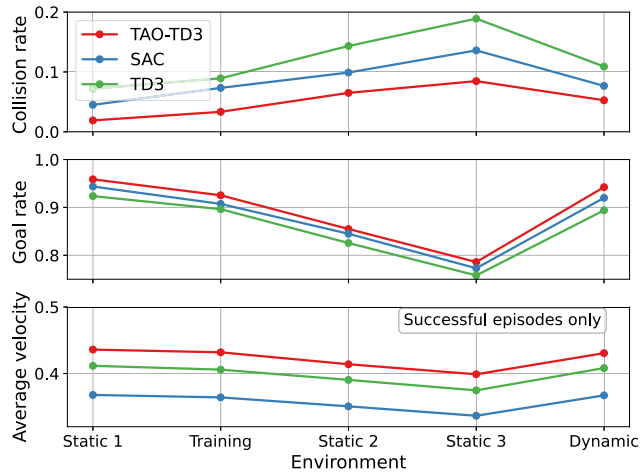


Fig. 4: Mean metrics of 3000 test samples across benchmark environments.

expected, feedforward policies often struggle with sharp, "edgy" obstacles, even in moderately cluttered environments, frequently colliding due to the wrap-around effect. TAO-TD3, by contrast, handles these situations more safely. ii) **Narrow Corridors**: TAO-TD3 also proves more effective when traversing tight passages formed by densely packed obstacles. In these cases, feedforward policies tend to slow down significantly or even become stuck and attempt alternative routes. iii) **Local Minima**: The recurrent architecture further helps the agent escape local-minimum configurations, such as dead-end, where memory-less policies commonly fall into an *infinite-loop* behavior.

E. Computational time comparison

A known drawback of recurrent architectures is their increased inference latency, traditionally one of the advantages of RL methods over non-learning controllers. Our TAO-TD3 variant inherits this limitation, as it processes a matrix of observations and incorporates an LSTM layer within the actor. For completeness, we report the average inference latency measured during the Monte-Carlo analysis. While feedforward TD3 and SAC require only $\approx$ 0.3 ms per evaluation, TAO-TD3 is approximately $100\times$ slower on an NVIDIA A100 GPU machine.

V. Conclusions

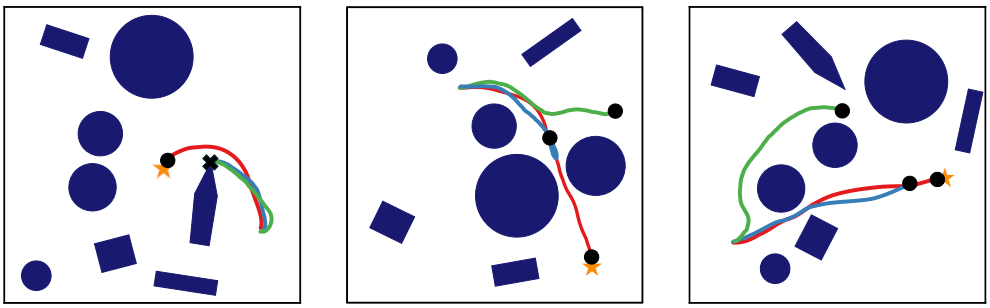
This work presented TAO-TD3, a recurrent actor-critic method for UAV navigation in a cluttered, partially observable environment with arbitrary-shape obstacles. By augmenting the TD3 observation with a sliding window of recent states and processing it through a lightweight LSTM backbone, the policy gains the ability to infer latent obstacle geometry from temporal cues. Evaluations on five benchmark environments with varying obstacle densities confirm that the approach consistently reduces collision rates and at the same time improves goal-reaching success. Additionally, experiments report a higher average velocity throughout navigation. The improvements are attributable to the recurrent architecture rather than the choice of base RL algorithm, as demonstrated by the comparable performance of feedforward TD3 and SAC. Future work will extend the current framework to three-dimensional navigation, dynamic obstacles, and multi-agent coordination. Moreover, the presented approach would benefit from ablation study on the architecture and hyperparameters and comparison with state-of-the-art recurrent networks.

References

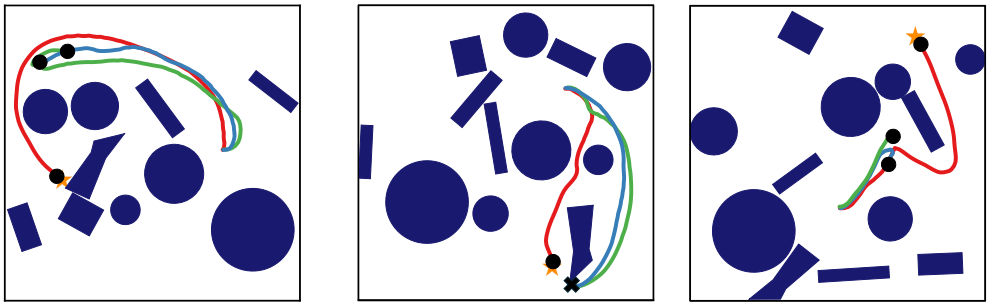
[1] A. Alexander, K. Venkatesan, J. Mounsef, and K. Ramanujam, "A comprehensive survey of path planning algorithms for autonomous systems and mobile robots: Traditional and modern approaches," *IEEE Access*, vol. 13, pp. 176 287–176 326, 2025.

[2] N. Hohmann, S. Brulin, J. Adamy, and M. Olhofer, "Three-dimensional urban path planning for aerial vehicles regarding many objectives," *IEEE Open Journal of Intelligent Transportation Systems*, vol. 4, 2023.

[3] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, 2015.



(a) Wrap-around ($t = 12\ s$). (b) Narrow corridor ($t = 17.7\ s$). (c) Narrow corridor ($t = 16\ s$).



(d) Cluttered map ($t = 30.3\ s$). (e) Wrap-around ($t = 15.9\ s$). (f) Local Minimum ($t = 22.8\ s$).

Fig. 5: Trajectories obtained across static benchmark environments with SAC (Blue), TD3 (Green) and TAO-TD3 (Red).

[4] G. Gemignani, M. Bongiorni, and L. Pollini, "An energy-aware decision-making scheme for mobile robots on a graph map based on deep reinforcement learning," in *2024 18th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, 2024.

[5] H. Han, J. Cheng, M. Lv, and Z. Xi, "Autonomous navigation of uavs in unknown 3d environments using deep reinforcement learning for path planning," *IEEE Transactions on Vehicular Technology*, 2025.

[6] K. Tamanakijprasart, A. Perrusquia, S. Mondal, and A. Tsourdos, "Autonomous path selection of unmanned aerial vehicle in dynamic environment using reinforcement learning," in *AIAA SCITECH 2025 Forum*, 2025.

[7] H. Wu, W. Wang, T. Wang, and S. Suzuki, "Model-free uav navigation in unknown complex environments using vision-based reinforcement learning," *Drones*, vol. 9, no. 8, 2025.

[8] X. Ren, N. Geng, Y. Zhang, L. Xiao, and D. Gong, "Pg-itd3: A potential field-guided deep reinforcement learning approach for uav path planning after disaster," *IEEE Transactions on Automation Science and Engineering*, vol. 22, 2025.

[9] J. Wang, P. Zhang, and Y. Wang, "Autonomous target tracking of multi-uav: A two-stage deep reinforcement learning approach with expert experience," *Applied Soft Computing*, vol. 145, p. 110604, 2023.

[10] Y. Xue and W. Chen, "Multi-agent deep reinforcement learning for uavs navigation in unknown complex environment," *IEEE Transactions on Intelligent Vehicles*, vol. 9, no. 1, 2024.

[11] G. Gemignani, S. Casini, V. Rosellini, G. Bucchioni, and L. Pollini, "Preliminary design of human-like decentralised task assignment for heterogeneous unmanned vehicles using reinforcement learning," in *AIAA SCITECH 2026 Forum*, 2026, p. 0326.

[12] S. Zhang, Y. Li, and Q. Dong, "Autonomous navigation of uav in multi-obstacle environments based on a deep reinforcement learning approach," *Applied Soft Computing*, vol. 115, p. 108194, 2022.

[13] W. Jiang, T. Cai, G. Xu, and Y. Wang, "Autonomous obstacle avoidance and target tracking of uav: Transformer for observation sequence in reinforcement learning," *Knowledge-Based Systems*, vol. 290, 2024.

[14] Z. Liu, Y. Cao, J. Chen, and J. Li, "A hierarchical reinforcement learning algorithm based on attention mechanism for uav autonomous navigation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 11, 2023.

[15] G. Tong, N. Jiang, L. Biyue, Z. Xi, W. Ya, and D. Wenbo, "Uav navigation in high dynamic environments: A deep reinforcement learning approach," *Chinese Journal of Aeronautics*, 2021.

[16] X. Yuntao and C. Weisheng, "A uav navigation approach based on deep reinforcement learning in large cluttered 3d environments," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 3, 2022.

[17] A. Singla, S. Padakandla, and S. Bhatnagar, "Memory-based deep reinforcement learning for obstacle avoidance in uav with limited environment knowledge," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 1, 2021.

[18] 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, 2021.

[19] S. Essaky, G. Raja, K. Dev, and D. Niyato, "Arresvg: Intelligent multi-aav navigation in partially observable spaces using adaptive deep reinforcement learning approach," *IEEE Transactions on Vehicular Technology*, vol. 74, no. 10, pp. 15 429–15 440, 2025.

[20] Z. Huang, Z. Yang, R. Krupani, B. Şenbaşlar, S. Batra, and G. S. Sukhatme, "Collision avoidance and navigation for a quadrotor swarm using end-to-end deep reinforcement learning," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

[21] M. J. Hausknecht and P. Stone, "Deep recurrent q-learning for partially observable mdps." in *AAAI fall symposia*, vol. 45, 2015, p. 141.

[22] G. Gemignani, A. Perrusquía, A. Tsourdos, and L. Pollini, "Temporal-augmented observation for navigation of unmanned aerial vehicles: a recurrent reinforcement learning architecture," in *2026 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2026.

[23] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition," *arXiv preprint arXiv:1402.1128*, 2014.

[24] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International conference on machine learning*. PMLR, 2018.

[25] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[26] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*. Pmlr, 2018.