IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

STAF-Navi: Vision-Based Spatio-Temporal Attention Fusion Navigation Framework

Haowen Zhang, Fanghong Liu, Chaoyu Zhang, and Qiuze Yu

***Abstract*—In cluttered, unknown, and partially observable environments, Uncrewed Aerial Vehicle (UAV) navigation encounters formidable challenges. To address these challenges, we propose an innovative spatio-temporal attention fusion navigation framework called STAF-Navi. The framework integrates spatio-temporal attention mechanisms to model sequential dependencies. It captures spatial and temporal correlations from historical observations and actions to improve navigation and obstacle avoidance. STAF-Navi employs deep collision encoding to compress high-dimensional depth images into informative low-dimensional latent states, and a single-site Transformer to model historical sensor inputs and states, enhancing the utility of current observations. By exploiting temporal dependencies, this integration enables early braking and stable hovering. Extensive simulation experiments show that the framework increases the navigation success rate by 10% and improves path efficiency by 7%. Finally, the successful deployment of the proposed strategy in real-world scenarios validates its effectiveness.**

***Index Terms*—UAV navigation, spatio-temporal attention fusion, cluttered environments, reinforcement learning.**

I. Introduction

UNCREWED aerial vehicles (UAVs) have significant potential for applications such as disaster rescue, urban exploration, and surveillance, enabling safe access to hazardous areas and providing real-time situational awareness [1]. Recently, deep reinforcement learning (DRL) methods have enabled UAVs to autonomously navigate dynamic environments directly from sensor data, offering adaptability without detailed maps or predefined rules [2]. A limitation of current DRL agents is their short-term memory. Over extended missions, they tend to forget target locations and previous cues, which impair long-range navigation performance [3].

Classical model-based UAV navigation typically decomposes tasks into perception, mapping, planning, and control [4]. These approaches frequently utilize simultaneous localization and mapping (SLAM) or depth sensing to construct 3D environmental representations, followed by path planners such as A$*$ or model predictive control for obstacle-free trajectories [5]. Despite providing precise and interpretable paths in static environments, these cascaded methods suffer from cumulative errors, computational overhead, and limited responsiveness under dynamic conditions [6]. To overcome these limitations, DRL-based UAV navigation frameworks have emerged, leveraging neural networks to map sensory inputs directly to control actions through environmental interaction [7], [8]. Approaches augment DRL agents with recurrent neural networks, enhancing memory and addressing partial observability in long-duration missions [3]. Purely reactive DRL systems without adequate memory struggle when action outcomes are correlated over time, indicating a need for improved spatiotemporal context modeling.

In this study, we propose a DRL-based UAV navigation framework incorporating explicit spatiotemporal information fusion. The approach integrates an internal memory and enhanced perceptual capabilities, enabling continuous situational awareness and goal retention. A Spatiotemporal Attention Fusion Module, utilizing Transformer-based self-attention mechanisms to model temporal dependencies in observation sequences [9], encodes historical observations and captures temporal dependencies between past states and current decisions. A deep collision encoder (DCE) compresses spatial sensor data into latent obstacle proximity representations, facilitating rapid collision-risk prediction and real-time path adjustments [10]. Domain randomization is employed during training to reduce the simulation-to-reality gap, supporting effective real-world deployment.

Contributions: Our main contributions are as follows:

- **Spatiotemporal Information Fusion Model**: A Transformer-based spatiotemporal fusion model that captures long-term temporal dependencies for globally informed navigation decisions.
- **DCE-Enhanced Temporal-Aware Navigation Framework**: Integration of DCE-compressed three-dimensional obstacle representations with temporal fusion for robust collision avoidance.
- **Sim2Real Deployment**: Successful sim-to-real deployment through domain randomization, validated in real-world UAV experiments.

II. Related Work

A. Planning-Based Navigation Framework

Traditional UAV navigation systems often rely on classical planning and control pipelines involving trajectory optimization, vision-based mapping, and heuristic-based motion

Received 25 June 2025; accepted 5 October 2025. Date of publication 24 October 2025; date of current version 3 November 2025. This article was recommended for publication by Associate Editor Z. Wu and Editor P. Vasseur upon evaluation of the reviewers' comments. This work was supported by the Major Program (JD) of Hubei Province, China under Grant 2023BAA025. *(Corresponding author: Qiuze Yu.)*

The authors are with the School of Electronic Information, Wuhan University, Wuhan 430072, China (e-mail: zhanghaowen@whu.edu.cn; 2018302080229@whu.edu.cn; chaoyv@whu.edu.cn; yuhenry007@whu.edu.cn).

This article has supplementary downloadable material available at https://doi.org/10.1109/LRA.2025.3625512, provided by the authors.

Digital Object Identifier 10.1109/LRA.2025.3625512

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

planning [11]. Techniques combining graph search and polynomial trajectory optimization have shown effectiveness in high-speed trajectory generation [12]. Real-time obstacle avoidance has been addressed using vision-assisted kinodynamic planning and gradient-based trajectory optimization methods [11]. Sampling-based planners have further improved robustness against dynamic obstacles by evaluating collision risks within uncertain environments [13]. However, these multi-stage systems typically suffer from cumulative errors and require extensive manual tuning, limiting their responsiveness and adaptability to dynamic or unknown scenarios [14].

B. Learning-Based Autonomous Navigation

Recent learning-based approaches leverage deep neural networks and reinforcement learning (RL) to learn UAV navigation policies directly from sensory data, reducing dependency on explicit environmental modeling [15]. CNN-based self-supervised strategies trained on collision data achieve reactive obstacle avoidance in cluttered indoor environments [16]. End-to-end RL methods directly map sensor observations to safe trajectories, mitigating latency issues inherent in classical pipelines [17]. Variational autoencoder-based approaches compress high-dimensional sensor data into informative latent representations, facilitating more effective and safer navigation [18]. Soft Actor-Critic (SAC)-based planners also learn reactive control policies for fast collision avoidance from depth sensors [19]. However, these controllers lack long-term planning, prompting research into hybrid methods that combine memory and prediction.

Recent progress in deep architectures integrating spatiotemporal modeling has significantly advanced fields like traffic management and financial forecasting [20]. However, explicit temporal modeling remains underexplored in UAV navigation. Motivated by these successes, we introduce a novel spatiotemporal fusion module into UAV navigation to enhance situational awareness, overcome short-term policy limitations, and ensure robust navigation performance in complex and dynamic scenarios.

III. Spatio-Temporal Attention Fusion Model

We propose an innovative spatiotemporal decision-making framework for UAV navigation tasks. The framework leverages the Single-Station-Transformer-Actor (SSTA) model. This model integrates positional encoding with Transformer-based attention mechanisms to fuse depth image inputs and low-dimensional state information over time. Depth images are compressed into a low-dimensional latent space [10]. In addition, both SSTA-based actor and GRU-based critic process historical data. This design enables the agent to incorporate contextual information from past observations and actions, ultimately allowing UAVs to undertake extended tasks that require multi-step planning and continuous feedback. The system and architecture diagram of STAF-Navi is introduced in Fig. 2.

A. Historical Observation Construction Across Time Steps

At each time step t, the agent records a comprehensive observation tuple capturing visual features, flight state, action taken, and goal-relative information shown in Fig. 1. The observation at time t:

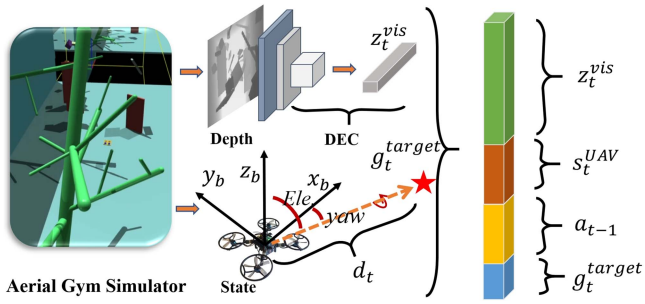

Fig. 1. Single-time observation interaction with the simulated environment. Including depth map embedding and relative distance of the target.

$$x_t = \left[z_t^{\text{vis}},\ s_t^{\text{UAV}},\ a_{t-1},\ g_t^{\text{target}} \right] \tag{1}$$

where the visual embedding $z_t^{\text{vis}} \in \mathbb{R}^{64}$ is a 64-dimensional feature vector extracted from the UAV's egocentric camera image at time t, generated by the DCE embedding network.

$$s_t^{\text{UAV}} = [\mathbf{v}_t,\ \boldsymbol{\omega}_t,\ \mathbf{q}_t,\ \bar{\rho}_t] \tag{2}$$

where $\mathbf{v}_t \in \mathbb{R}^3$ denotes the linear velocity, $\boldsymbol{\omega}_t \in \mathbb{R}^3$ the angular velocity, and $\mathbf{q}_t \in \mathbb{R}^4$ the UAV's orientation quaternion. The variable $\bar{\rho}_t \in \mathbb{R}$ represents the average absolute yaw change over the past N time steps, indicating the UAV's turning intensity or directional stability.

The a_{t-1} denotes the control action executed by the policy at the preceding time step, where a comprises roll ϕ_{t-1}, pitch θ_{t-1}, yaw ψ_{t-1} angles and total thrust T_{t-1}.

$$a_{t-1} = [\phi_{t-1}, \theta_{t-1}, \psi_{t-1}, T_{t-1}]^\top, \tag{3}$$

The target direction and distance vector g_t^{target}encodes relative goal position information at time t:

$$g_t^{\text{target}} = [\mathbf{u}_t,\ d_t], \tag{4}$$

where $\mathbf{u}_t \in \mathbb{R}^3$ is a unit vector pointing toward the target in the UAV's body frame, and $d_t \in \mathbb{R}$ is the UAV-target Euclidean distance.

At each time step, observation x_t integrates multimodal data. We aggregate the last H observations into a weighted historical sequence as input to the policy:

$$\mathcal{X}_{t-H+1:t} = \{(x_\tau, w_\tau) \mid \tau \in [t-H+1, t]\}, \tag{5}$$

where $w_\tau \in \mathbb{R}^+$ represents the temporal importance weight for observation x_τ, with normalization constraint $\sum_{\tau=t-H+1}^{t} w_\tau = 1$. This weighted sequence provides temporal context with adjustable importance for different time steps, enabling the model to prioritize relevant historical information.

B. Single-Station Transformer Actor Integration

The Single-Station-Transformer-Actor employs a Transformer-based temporal fusion mechanism to integrate information across H historical steps, capturing long-range temporal dependencies in the UAV's observations.

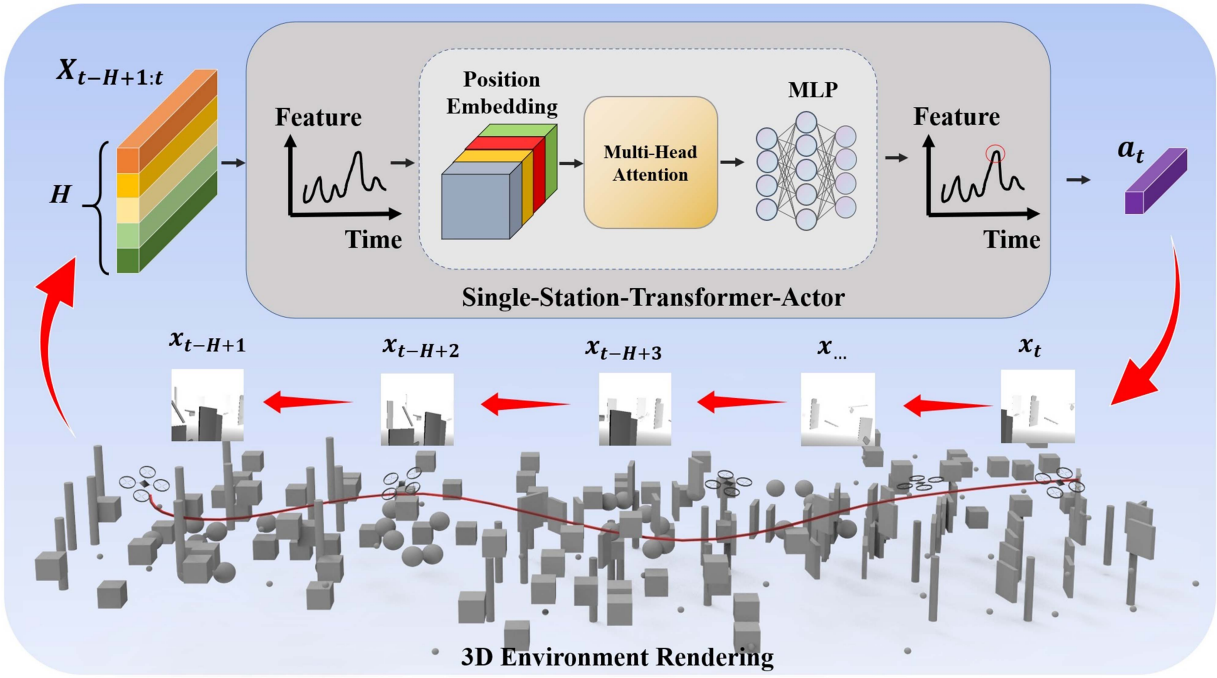

Fig. 2. **Method overview: learning vision-based navigation via Spatio-Temporal Attention Fusion Framework.** Demonstrate the decision-making process of integrating spatiotemporal information for navigation. Observation features from the past H timesteps are captured and their influence on the current timestep t is computed, which is then combined to generate a decision at time t. Each individual timestep comprises depth map rendering, state evaluation, action prediction, and quadrotor dynamics simulation.

Given a weighted historical sequence $\mathcal{X}_{t-H+1:t}$, each observation-weight pair (x_τ, w_τ) is processed by embedding the weighted observation into a d_h-dimensional token and combining with positional encoding:

$$h_\tau^{(0)} = W_e\,(w_\tau \cdot x_\tau) \;+\; p_{\tau-t+H-1}, \tag{6}$$

where $W_e \in \mathbb{R}^{d_h \times \dim(x)}$ is a learnable projection, and $p_k \in \mathbb{R}^{d_h}$ is the positional embedding at position k.

These embedded tokens $\{h_{t-H+1}^{(0)}, \ldots, h_t^{(0)}\}$ are processed by a single Transformer encoder layer as follows:

a) Linear Projections for Self-Attention: For each time step τ, compute:

$$Q_\tau = h_\tau^{(0)} W_Q, \quad K_\tau = h_\tau^{(0)} W_K, \quad V_\tau = h_\tau^{(0)} W_V. \tag{7}$$

where the weight matrices W_Q, W_K, $W_V \in \mathbb{R}^{d_h \times d_q}$ are learnable parameters.

b) Self-Attention Calculation: For each pair (τ, τ'), compute the attention weight:

$$\alpha_{\tau,\tau'} = \frac{\exp\left(\frac{Q_\tau \cdot K_{\tau'}}{\sqrt{d_q}}\right)}{\sum_{\kappa=t-H+1}^{t} \exp\left(\frac{Q_\tau \cdot K_\kappa}{\sqrt{d_q}}\right)}, \tag{8}$$

where d_q denotes the dimension of queries and keys. Then compute the attention output:

$$o_\tau = \sum_{\tau'=t-H+1}^{t} \alpha_{\tau,\tau'} V_{\tau'}. \tag{9}$$

c) Residual Connections and Feed-Forward Network: The attention outputs are combined with residual connections and layer normalization as:

$$\hat{h}_\tau = \text{LayerNorm}(o_\tau + h_\tau^{(0)}), \tag{10}$$

followed by a position-wise feed-forward network (FFN) with another residual connection and layer normalization:

$$h_\tau^{(1)} = \text{LayerNorm}\left(\text{FFN}(\hat{h}_\tau) + \hat{h}_\tau\right). \tag{11}$$

The final embedding at the latest time step $h_t^{(1)}$ summarizes the historical context. It is passed through a multilayer perceptron (MLP) to generate the policy's control action:

$$\pi_\theta(a_t \mid \mathcal{X}_{t-H+1:t}) = f_{\text{MLP}}\left(h_t^{(1)}\right), \tag{12}$$

where $f_{\text{MLP}}(\cdot)$ denotes the actor's output network. This temporal fusion mechanism allows the actor to effectively focus on relevant historical information, enhancing the UAV's decision-making capability.

C. GRU-Based Critic for Long-Horizon Value Estimation

The GRU-based critic estimates the long-horizon value of the UAV's state by processing the weighted historical sequence $\mathcal{X}_{t-H+1:t}$. Starting from an initial hidden state h_{t-H}^C, the GRU updates iteratively for $\tau = t-H+1, \ldots, t$:

$$h_\tau^C = \text{GRUCell}(w_\tau \cdot x_\tau,\, h_{\tau-1}^C), \tag{13}$$

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

with update equations:

$$\begin{aligned} r_\tau &= \sigma\left(W_r(w_\tau \cdot x_\tau) + U_r h_{\tau-1}^C\right), \\ z_\tau &= \sigma\left(W_z(w_\tau \cdot x_\tau) + U_z h_{\tau-1}^C\right), \\ \tilde{h}_\tau &= \tanh\left(W_h(w_\tau \cdot x_\tau) + U_h\left(r_\tau \odot h_{\tau-1}^C\right)\right), \\ h_\tau^C &= z_\tau \odot h_{\tau-1}^C + (1 - z_\tau) \odot \tilde{h}_\tau, \end{aligned} \tag{14}$$

where $\sigma(\cdot)$ denotes the sigmoid activation and $\odot$ denotes element-wise multiplication. The state value is computed from the final hidden state:

$$V(s_t) = W_v h_t^C + b_v, \tag{15}$$

representing the expected cumulative reward from time t onward.

D. Multi-Step Feedback and Task Dependency

Each action a_t influences future states through the transition function:

$$s_{t+1} = f(s_t, a_t). \tag{16}$$

Over multiple time steps, the state evolves as:

$$s_{t+n} = f^n(s_t, a_t, a_{t+1}, \ldots, a_{t+n-1}), \tag{17}$$

and observations are generated as:

$$o_{t+n} = h(s_{t+n}), \tag{18}$$

indicating that current actions impact future observations and rewards.

The Transformer-based actor attends to historical observations to approximate the optimal policy:

$$\pi_\theta(a_t \mid \mathcal{X}_{t-H+1:t}) \approx \pi^*(a_t \mid o_{1:t}). \tag{19}$$

The GRU-based critic approximates the value function through recurrent state aggregation:

$$V(s_t) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k} \mid s_t\right] \approx V_\phi(h_t^C), \tag{20}$$

where $h_t^C = F_\phi(h_{t-1}^C, w_t \cdot x_t)$ summarizes historical observations. Thus, the critic's recurrent structure captures state evolution under the policy, while the actor's attention explicitly leverages historical outcomes to enhance decision-making.

IV. Reinforcement Learning for Navigation Framework

We constructed a complete reinforcement learning loop, integrating our spatiotemporal fusion model into the overall navigation framework. This process comprises interactions with the simulator, state and reward collection, advantage estimation, and policy updates.

A. Problem Formulation

The UAV navigation task is formulated as a sequential decision-making problem that begins with safely navigating around obstacles, continues with guiding the vehicle to the designated goal, and concludes with maintaining a stable hover once the target is reached. Early-stage actions significantly impact subsequent stages, necessitating a policy that accounts for long-term action consequences rather than immediate rewards alone. The navigation task is formulated as a Partially Observable Markov Decision Process (POMDP) represented by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{O}, T, O, R, \gamma)$. Here, $\mathcal{S}$ denotes the hidden state space; $\mathcal{A}$ corresponds to the set of UAV control commands; $\mathcal{O}$ comprises sensor outputs. The system dynamics are characterized by state transition probabilities $T(s_{t+1} \mid s_t, a_t)$ and observation mappings $O(o_t \mid s_t)$, while the reward function $R(s_t, a_t)$ and discount factor $\gamma \in [0, 1]$ define the performance measure. In a reinforcement learning framework, this objective is written as:

$$\pi^* = \arg\max_\pi \mathbb{E}_\pi\left[\sum_{t=0}^{T} \gamma^t R(s_t, a_t)\right] \tag{21}$$

B. Reward Function Design

The reward function integrates various task objectives into a single scalar reward at each step t:

$$\begin{aligned} r_t = {} & r_{\text{distance}}(t) + r_{\text{time}}(t) + r_{\text{heading}}(t) \\ & + r_{\text{penalty}}(t) + r_{\text{hover}}(t) \end{aligned} \tag{22}$$

a) Distance-related reward: Encourages approaching the goal:

$$r_{\text{distance}}(t) = \lambda_1 e^{-\alpha d_t} + \lambda_2(d_{t-1} - d_t) \tag{23}$$

where d_t is the distance to the goal; $\lambda_1, \lambda_2, \alpha$ are scaling factors.

b) Time-related reward: Promotes efficient task completion:

$$r_{\text{time}}(t) = -\eta + \beta \frac{T_{\text{rem}}}{T_{\text{ep}}} \tag{24}$$

where η penalizes each step, β rewards quick completion, T_{rem} is the remaining time, and T_{ep} is the total episode length.

c) Heading reward: Incentivizes correct orientation:

$$r_{\text{heading}}(t) = \lambda_3 \cos(\theta_{\text{error}}) \tag{25}$$

where θ_{error} measures heading deviation from the target.

d) Penalty reward: Discourages unsafe or inefficient behaviors:

$$r_{\text{penalty}}(t) = -\lambda_4\, r_{\text{action}}(t) - \lambda_5\, \mathbf{1}_{\text{collision}}(t) - 4.0\, \lambda_6\, e^{-(d_{\text{pix}}(t))^2} \tag{26}$$

where $\mathbf{1}_{\text{collision}}(t)$ is an indicator function that equals 1 if a collision occurs at time step t, and 0 otherwise; $d_{\text{pix}}(t)$ is the minimum distance in pixel space to the nearest obstacle from the predicted trajectory points at time t.

e) Hovering reward: Encourages stable hovering near the target:

$$r_{\text{hover}}(t) = \begin{cases} +\lambda_7, & \text{if } d_t < \Delta_{\text{hover}},\ v_t < v_{\text{threshold}}, \\ 0, & \text{otherwise}, \end{cases} \tag{27}$$

where Δ_{hover} and $v_{\text{threshold}}$ are distance and velocity thresholds for stable hovering.

The necessity and contribution of each component are validated through ablation studies presented in Section V-A.

C. Training Details and Environment

The agent is trained using a Proximal Policy Optimization(PPO) [21] combined with Generalized Advantage Estimation (GAE) in an actor-critic framework. Training data are collected concurrently from 512 parallel simulation instances, each performing rollouts of 32 steps, resulting in approximately 64,000 timesteps per update. The PPO algorithm employs an Adam optimizer with a learning rate of 10^{-4}, discount factor $\gamma = 0.99$, and GAE factor $\lambda = 0.95$. Policy updates are constrained using a clipping parameter $\varepsilon = 0.2$ and an entropy bonus with coefficient c_E to balance exploration and policy determinism.Training proceeds iteratively, where both SSTA-based actor and GRU-based critic are updated using mini-batch gradient descent. The Deep Collision Encoder, responsible for extracting stable visual features, is pre-trained and frozen throughout reinforcement learning to maintain consistent visual embeddings.

The simulation environment is provided by the Aerial Gym Simulator [22], which offers a high-fidelity and physics-matched platform for multirotor UAV navigation, as illustrated in Fig. 1. The simulator generates randomized cluttered scenes with obstacles placed at varying positions, orientations, and densities. Start and goal positions are randomized for each episode to provide diverse navigation challenges. A curriculum learning approach progressively increases obstacle density based on agent performance, adjusting difficulty dynamically according to success and crash rates. Extensive domain randomization is applied to simulate real-world uncertainties, including random forces and torques on the UAV, sensor noise, pose variations, and visual corruptions. These randomizations mitigate overfitting to idealized simulator conditions and enhance the policy's robustness for real-world deployment.

V. Experiment

In this section, we evaluate the proposed spatiotemporal fusion model for UAV navigation through extensive simulations and real-world tests. Experiments validate the model's impact on navigation success, maneuverability, and hover stability, highlighting its effectiveness in cluttered environments.

A. Validation of Spatiotemporal Modeling in Navigation

To comprehensively validate the spatiotemporal fusion model, we conducted extensive experiments using the Aerial Gym Simulator examining temporal observation processing, architectural choices, and reward design. Navigation was considered successful if the UAV reached within 1.0 m of the target and maintained stable hover for at least 1 s.

1) Temporal Observation Window Optimization: The temporal observation length H determines the historical context available for decision-making. We evaluated $H \in \{1, 5, 10, 15, 20, 30, 50\}$ to identify the optimal balance between context richness and computational efficiency.

Fig. 3 and Table I demonstrate that performance peaks at $H = 20$ with a 71.3% success rate. Short windows lack sufficient context for trajectory planning, while longer windows introduce computational overhead without performance gains.

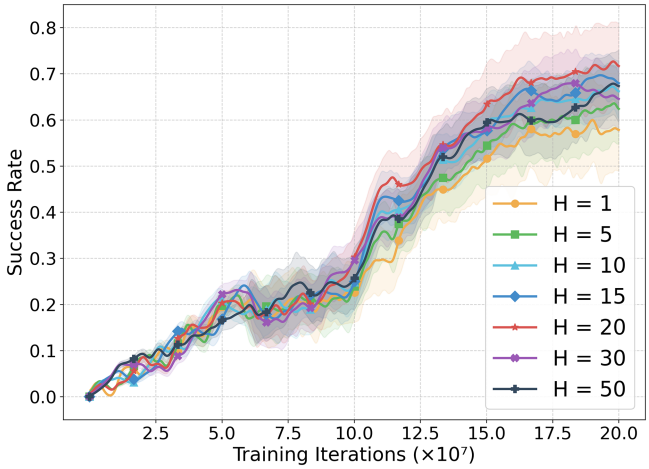

Fig. 3. Navigation success rate over training iterations for different observation window lengths H.

TABLE I
Effect of Window Size H on Navigation Performance

H	1	5	10	15	20	30	50
Success Rate (%)	57.6	61.2	65.3	67.4	**71.3**	66.1	62.8

TABLE II
Impact of Temporal Weighting Schemes

Weighting Scheme	Success Rate (%)	Improvement
Uniform (baseline)	71.3	–
Exponential (γ=0.95)	74.7	+3.4%
Exponential (γ=0.90)	70.2	-1.1%
Linear decay	72.1	+0.8%

TABLE III
Success Rate for Different H and Weighting Combinations

Weighting	H=10	H=15	H=20	H=25	H=30
Uniform	65.3	67.4	71.3	68.2	66.1
Exp(γ=0.95)	67.8	70.1	**74.7**	71.5	69.3
Improvement	+2.5%	+2.7%	+3.4%	+3.3%	+3.2%

2) Temporal Weighting Mechanisms: We investigated whether explicit temporal weighting could improve upon uniform observation treatment. We evaluated four weighting schemes with $H = 20$ in Table II.

Exponential weighting with $\gamma = 0.95$ achieves 3.4% improvement over uniform weighting. This gain arises from balancing immediate collision avoidance needs (requiring recent observations) with trajectory planning requirements (utilizing longer history). Stronger decay ($\gamma = 0.90$) degrades performance by excessively discounting valuable historical information. Linear decay shows marginal improvement, suggesting that gradual temporal discounting provides limited benefit.

3) Robustness Across Window Lengths: To investigate the relationship between window length and temporal weighting, we evaluated performance across multiple combinations in Table III.

Exponential weighting consistently improves performance across different window lengths, with gains ranging from 2.5%

TABLE IV
TEMPORAL ATTENTION FOCUS IN DIFFERENT ENVIRONMENTS

Environment Type	Last-5 Attention (%)
Sparse	32.4 ± 5.2
Moderate	48.7 ± 6.8
Dense	$\mathbf{71.3 \pm 8.1}$

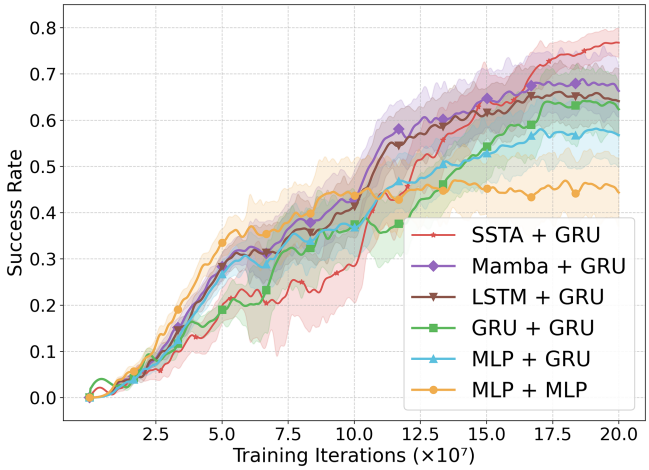

Fig. 4. Navigation success rate across training iterations for different architectures.

TABLE V
SUCCESS RATES OF DIFFERENT ACTOR-CRITIC ARCHITECTURES

Architecture	Success Rate (%)
SSTA + GRU	**74.7**
Mamba + GRU	67.8
LSTM + GRU	65.2
GRU + GRU	63.5
MLP + GRU	57.3
MLP + MLP	45.6

to 3.4%. This robustness demonstrates that temporal weighting enhances performance regardless of window size, providing resilience when the optimal H cannot be determined a priori.

4) Adaptive Attention Patterns: To assess implicit adaptability under a fixed window (H=20), we analyze attention during closed-loop inference on 1,000 test episodes. We extract attention weights from the self-attention layer and compute the percentage allocated to the most recent five tokens (steps 16-20in a H=20 window). Table IV shows mean and standard deviation percentages across environments classified by obstacle density: sparse (N_{obs}<100), moderate (100≤N_{obs}≤300), and dense (N_{obs}>300) per 100×50×8 m map.

Attention shifts toward recent frames as clutter increases (32.4% → 48.7% →71.3%), indicating context-dependent recency and practical flexibility within the fixed H=20 framework.

5) Architectural Ablation Study: Fig. 4 and Table V present the navigation success rates for each architecture. The SSTA architecture with exponential weighting achieves 74.7% success rate, outperforming the recent Mamba architecture by 6.9% and LSTM by 9.5%. While Mamba offers computational efficiency advantages, the Transformer's global attention mechanism provides superior temporal modeling for vision-based navigation. The attention mechanism enables the integration of spatial cues across the entire visual field, which proves critical in cluttered environments.

TABLE VI
HOVERING STABILITY METRICS

Architecture	Mean ± Std	
	Hover Duration (s)	Speed (m/s)
SSTA+GRU	$\mathbf{14.2 \pm 1.08}$	$\mathbf{0.05 \pm 0.02}$
Mamba+GRU	13.1 ± 1.65	0.06 ± 0.03
GRU+GRU	11.6 ± 2.23	0.06 ± 0.02
MLP+MLP	5.3 ± 3.91	0.15 ± 0.05

TABLE VII
IMPACT OF REWARD COMPONENTS ON NAVIGATION

Configuration	Success Rate (%)	Hover Duration (s)
Full Reward	74.7	14.2 ± 1.1
w/o r_{distance}	–	–
w/o r_{penalty}	12.4	13.8 ± 1.4
w/o r_{heading}	–	–
w/o r_{hover}	61.8	2.4 ± 0.8
w/o r_{time}	73.1	14.6 ± 1.3

6) Hover Stability Analysis: Hover stability was evaluated by measuring duration maintaining speed below 0.2 m/s within 15 s of target arrival:

As shown in Table VI, temporal modeling improves hover stability. The SSTA architecture with exponential weighting achieves the longest hover duration (14.2 s) with minimal speed fluctuation.

7) Reward Component Ablation: We validated the reward design through systematic ablation using the SSTA+GRU architecture with exponential weighting in Table VII. Distance and heading rewards are essential for training convergence; without either component, the agent fails to learn meaningful navigation behavior. Penalty reward ensures safety, with its removal causing frequent collisions. Hover reward critically affects terminal behavior, while time reward encourages efficient trajectories, with its absence leading to more conservative flight profiles.

In summary, our validation experiments examined temporal window length, weighting mechanisms, architectural choices, and reward components. Results show that temporal modeling with attention mechanisms improves navigation success and hover stability compared to baselines. Thus, incorporating temporal sequences with attention mechanisms effectively improves UAV navigation and target stabilization performance.

B. Benchmarking Against State-of-The-Art Navigation Methods

We benchmarked our proposed spatiotemporal navigation strategy against three state-of-the-art methods: DCE-RL [10], MAVRL [2], and EGO-Planner [23]. Experiments were performed on six distinct map configurations (100 m × 50 m × 8 m) with progressively increasing obstacle densities. UAV and target positions were randomly initialized at opposite ends, with a maximum UAV speed constrained at 2 m/s. Each scenario

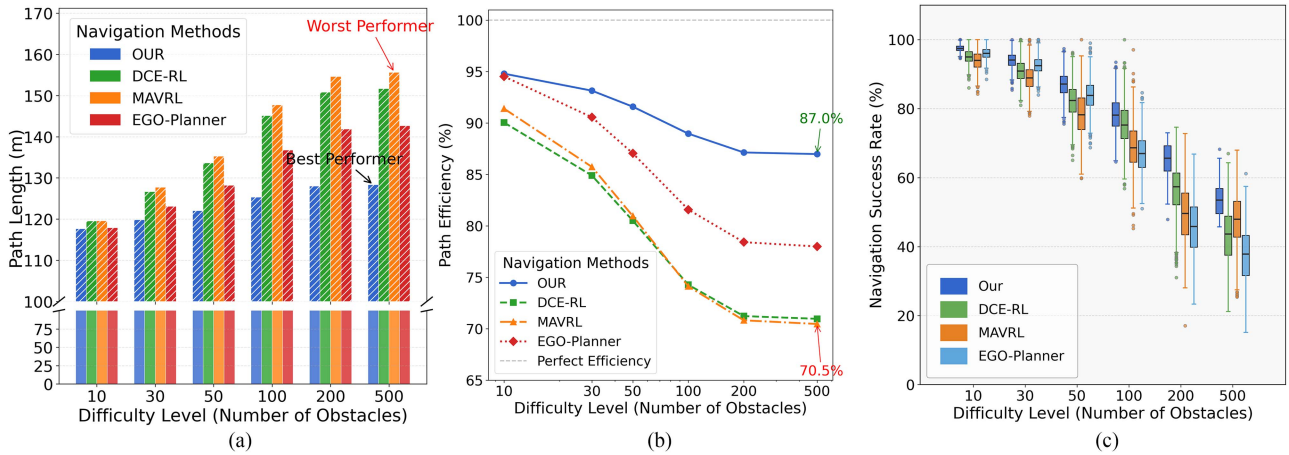

Fig. 5. (a) Comparison of path lengths for four navigation methods under different obstacle density environments. (b) Path efficiency of the four navigation methods. (c) Success rates for four navigation methods under different obstacle density environments.

involved 1000 trials across 200 rounds to ensure statistical robustness. Key evaluation metrics included navigation success rate, average path length, and path efficiency (ratio of direct target distance to actual traveled path length).

Fig. 5(c) summarizes navigation success rates. On the simplest map, all methods achieved nearly perfect performance with success rates exceeding 90%. In moderately cluttered maps, the proposed method achieves a success rate of approximately 75%, which is approximately 3% higher than DCE-RL, while MAVRL and EGO-Planner achieve success rates below 70%. In highly cluttered scenarios, our strategy consistently achieved success rates above 55%, outperforming all baselines. The narrower interquartile range in the proposed method's box plot indicates more stable and consistent performance, especially under challenging conditions.

Average path lengths and efficiencies for successful trials are presented in Fig. 5(a) and (b), respectively. As obstacle density grew, our approach yielded paths that were between 7% and 15.5% shorter than those generated by DCE-RL and MAVRL, implying fewer unnecessary detours. In complex maps, the proposed method achieved path efficiency values ranging from 80% to 90%, surpassing other approaches that recorded values between 70% and 80%. DCE-RL and MAVRL exhibited conservative maneuvering or frequent corrections, leading to less efficient routes. Although EGO-Planner achieved high efficiency in open spaces, it often generated significantly curved trajectories in cluttered scenes, sometimes resulting in path lengths similar to or longer than our method.

C. Real World Tests

The learned navigation policy was ported to a custom quadrotor equipped with an *NVIDIA Jetson Orin NX* and an Intel RealSense D435 depth camera streaming 640×480 depth frames at 30 Hz. Visual–inertial odometry (VIO) fused the D435 depth image stream with measurements from the onboard IMU at 60Hz, yielding centimeter-level positional accuracy and an attitude error of approximately 0.1° in laboratory calibration. The

Fig. 6. Demonstration of autonomous flight path navigation in diverse environments: (a) cluttered forest terrain, (b) dense vegetation, (c) building atrium, and (d) indoor corridor with obstacles.

end-to-end inference required 30.4 ± 1.7 ms, comprising 9.3 ms for perception (DCE) and 21.1 ms for policy inference. This 30.4 ms inference time enables 20 Hz control loops, providing decision updates approximately every 10 cm when flying at 2 m/s maximum speed, which proves sufficient for safe navigation in cluttered environments.

Flight experiments were conducted in visually and spatially cluttered environments representative of real-world, GPS-denied operation. During each mission, the quadrotor processed depth frames at 30 Hz on board, employed the DCE collision encoding to form a compact latent state, and applied the spatiotemporal attention module to generate velocity commands in real time. Across all test runs, the vehicle traversed obstacle-filled trajectories and reached its designated goal without human intervention, demonstrating the policy's capacity for fully autonomous navigation. Fig. 6 shows representative snapshots from the outdoor and indoor trials. A video demonstrating the UAV's navigation performance under these conditions is included in the attachment to this submission.

Transferring the learned policy from simulation to reality posed challenges including sensor noise, lighting differences, and dynamic discrepancies. The DCE module effectively mitigated these by extracting stable geometric representations from noisy depth images. Temporal fusion further enhanced robustness, enabling stable navigation despite transient sensor anomalies. However, the stereo-based depth sensor occasionally failed to detect fine, sun-lit obstacles such as wires or thin branches, which remain the dominant failure mode. As future work, we plan to integrate a lightweight solid-state LiDAR to provide texture-independent range measurements and fuse them with visual cues, thereby improving obstacle recall without sacrificing contextual richness.

VI. Conclusion

We presented a novel spatio-temporal attention fusion model combining a Single-Station Transformer-based policy with a GRU-based critic and DCE visual embeddings for UAV navigation in cluttered environments. By fusing spatiotemporal information, the UAV's decision-making in long-horizon tasks fully incorporates historical context, leading to remarkable improvements in both obstacle avoidance and path planning, as well as overall task efficiency and safety. Future work will further enhance robustness in highly dynamic scenarios, promoting practical deployment of autonomous UAV systems.

References

[1] M. Tranzatto et al., "CERBERUS in the DARPA subterranean challenge," *Sci. Robot.*, vol. 7, no. 66, 2022, Art. no. eabp9742.

[2] H. Yu, C. Wagter, and G. C. H. E. de Croon, "MAVRL: Learn to fly in cluttered environments with varying speed," *IEEE Robot. Automat. Lett.*, vol. 10, no. 2, pp. 1441–1448, Feb. 2025.

[3] A. Singla, S. Padakandla, and S. Bhatnagar, "Memory-based deep reinforcement learning for obstacle avoidance in UAV with limited environment knowledge," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 1, pp. 107–118, Jan. 2021.

[4] J. Li, C. Liao, W. Zhang, H. Fu, and S. Fu, "UAV path planning model based on R5DOS model improved A-star algorithm," *Appl. Sci.*, vol. 12, no. 22, 2022, Art. no. 11338.

[5] Z. Teed and J. Deng, "Droid-SLAM: Deep visual SLAM for monocular, stereo, and RGB-D cameras," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 16558–16569.

[6] Y. Chen, S. Lai, J. Cui, B. Wang, and B. M. Chen, "GPU-accelerated incremental euclidean distance transform for online motion planning of mobile robots," *IEEE Robot. Automat. Lett.*, vol. 7, no. 3, pp. 6894–6901, Jul. 2022.

[7] Stéphane Ross et al., "Learning monocular reactive UAV control in cluttered natural environments," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2013, pp. 1765–1772.

[8] Y. Ren, Y. Cai, F. Zhu, S. Liang, and F. Zhang, "ROG-map: An efficient robocentric occupancy grid map for large-scene and high-resolution LiDAR-based motion planning," in *Proc. 2024 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2024, pp. 8119–8125.

[9] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6000–6010.

[10] M. Kulkarni and K. Alexis, "Reinforcement learning for collision-free flight exploiting deep collision encoding," in *Proc. 2024 IEEE Int. Conf. Robot. Automat.*, 2024, pp. 15781–15788.

[11] Y. Wang, J. Ji, Q. Wang, C. Xu, and F. Gao, "Autonomous flights in dynamic environments with onboard vision," in *Proc. 2021 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 1966–1973.

[12] A. Bry, C. Richter, A. Bachrach, and N. Roy, "Aggressive flight of fixed-wing and quadrotor aircraft in dense indoor environments," *Int. J. Robot. Res.*, vol. 34, no. 7, pp. 969–1002, 2015.

[13] G. Chen, P. Peng, P. Zhang, and W. Dong, "Risk-aware trajectory sampling for quadrotor obstacle avoidance in dynamic environments," *IEEE Trans. Ind. Electron.*, vol. 70, no. 12, pp. 12606–12615, Dec. 2023.

[14] W. Sun, G. Tang, and K. Hauser, "Fast UAV trajectory optimization using bilevel optimization with analytical gradients," *IEEE Trans. Robot.*, vol. 37, no. 6, pp. 2010–2024, Dec. 2021.

[15] R. Penicka, Y. Song, E. Kaufmann, and D. Scaramuzza, "Learning minimum-time flight in cluttered environments," *IEEE Robot. Automat. Lett.*, vol. 7, no. 3, pp. 7209–7216, Jul. 2022.

[16] D. Gandhi, L. Pinto, and A. Gupta, "Learning to fly by crashing," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2017, pp. 3948–3955.

[17] A. Loquercio, E. Kaufmann, R. Ranftl, M. Müller, V. Koltun, and D. Scaramuzza, "Learning high-speed flight in the wild," *Sci. Robot.*, vol. 6, no. 59, 2021, Art. no. eabg5810.

[18] M. Kulkarni, H. Nguyen, and K. Alexis, "Semantically-enhanced deep collision prediction for autonomous navigation using aerial robots," in *Proc. 2023 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2023, pp. 3056–3063.

[19] K. Nakhleh et al., "Sacplanner: Real-world collision avoidance with a soft actor critic local planner and polar state representations," in *Proc. 2023 IEEE Int. Conf. Robot. Automat.*, 2023, pp. 9464–9470.

[20] K. Greff, R. K. Srivastava, J. Koutní k, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A search space odyssey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2222–2232, Oct. 2017.

[21] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.

[22] M. Kulkarni, W. Rehberg, and K. Alexis, "Aerial gym simulator: A framework for highly parallelized simulation of aerial robots," *IEEE Robot. Automat. Lett.*, vol. 10, no. 4, pp. 4093–4100, Apr. 2025.

[23] X. Zhou, Z. Wang, H. Ye, C. Xu, and F. Gao, "EGO-planner: An ESDF-free gradient-based local planner for quadrotors," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 478–485, Apr. 2021.