

Article

Multimodal History-Window Gated-Attention Soft Actor–Critic for Urban Low-Altitude UAV Navigation

Xi You and Wenjun Yi *

National Key Laboratory of Transient Physics, Nanjing University of Science and Technology,
Nanjing 210094, China; youxi@njust.edu.cn

* Correspondence: wjy@mail.njust.edu.cn

Highlights

What are the main findings?

- HW-GA-SAC achieved 96% success in the five-seed seven-method MuJoCo comparison and 98% success on ten additional generated city layouts.
- Actor-side safety inputs and front-view visual cues improved task completion, while feedforward SAC produced shorter successful paths.

What are the implications of the main findings?

- Fixed-window gated attention and current-frame safety access are useful design choices for multimodal urban low-altitude UAV navigation in simulation.
- Future validation should cover integrated onboard perception, latency, energy use, and flight-safety layers before physical deployment.

Abstract

Urban low-altitude unmanned aerial vehicle (UAV) navigation combines partial observability, building occlusion, wind disturbance, and continuous control. This study develops and evaluates HW-GA-SAC, a multimodal history-window Soft Actor–Critic (SAC) policy for procedurally generated three-dimensional MuJoCo cities. A Gated Transformer-XL (GTrXL)-inspired gated-attention encoder processes a fixed eight-step navigation history, while a current-frame safety branch supplies vertical clearance, sparse Light Detection and Ranging (LiDAR)-like range sectors, and handcrafted safety cues directly to the actor and critic. The policy uses obstacle-related observations and reward shaping to support collision avoidance; it does not include constrained policy optimization or a separate runtime safety filter. In a seven-method comparison using five training seeds and five evaluation layouts, HW-GA-SAC achieved a $96\% \pm 3\%$ success rate, 207 ± 16 average return, and $3\% \pm 4\%$ timeout rate. Feedforward SAC achieved $92\% \pm 11\%$ success and a $7\% \pm 10\%$ timeout rate, but its successful paths were more direct. Five-seed learning curves, city-split evaluation, wind sensitivity, sensing perturbations, inference profiling, and ablation studies further characterize the method. Within this simulation protocol, HW-GA-SAC provides the strongest completion-oriented performance, with a measurable trade-off between task completion and path directness.

Keywords: UAV autonomous navigation; Soft Actor-Critic; GTrXL-inspired gated-attention; fixed-window gated attention; multimodal perception; obstacle avoidance; partial observability; urban low-altitude environments



Academic Editor: Oleg Yakimenko

Received: 30 June 2026

Revised: 3 August 2026

Accepted: 4 August 2026

Published: 5 August 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article

distributed under the terms and

conditions of the [Creative Commons](#)

[Attribution \(CC BY\)](#) license.

1. Introduction

Urban low-altitude UAV navigation combines partial observability, building occlusion, local obstacle avoidance, wind disturbance, and continuous control. Recent UAV control studies show a shift from low-level stabilization toward autonomous perception, planning, and obstacle-aware decision making in complex environments [1]. Path-planning and trajectory-optimization surveys further indicate that environmental modeling, task constraints, and trajectory quality directly affect the practical value of UAV systems [2,3]. Multisensor navigation and localization remain central to reliable autonomous unmanned systems [4]. Compared with open airspace, urban low-altitude scenes contain high-rise buildings, narrow street canyons, rooftop structures, skybridges, and frequent local occlusions. Buildings also reshape the local wind field, producing street-canyon acceleration, wake turbulence, and crosswind shear. Navigation in such scenes is a coupled problem of multimodal perception, continuous control, disturbance adaptation, and sequential decision making, not merely shortest-path planning.

Traditional planning methods often assume a sufficiently accurate map, an explicit geometric model, or a tractable local cost function. These assumptions can be fragile in unknown or heavily occluded urban blocks. Deep reinforcement learning offers an alternative route for learning continuous control policies through interaction. Soft Actor–Critic (SAC), in particular, optimizes a maximum-entropy objective and has shown strong stability in many continuous-control tasks [5]. Its automatic temperature adjustment and practical implementation details make it suitable for robotic control [6]. Standard SAC policies, however, are usually feedforward: the action is predicted from the current observation alone. This assumption is limiting for urban UAV navigation, where a single frame may not reveal the extension of a street corridor, whether the vehicle is closing on the target, how the safety margin is changing, or how wind is biasing the motion. Without short-term context, the policy can become myopic near obstacles or under disturbances.

Temporal modeling can mitigate partial observability by letting the policy use recent observations and actions to infer closing trends, safety-margin changes, and control bias introduced by wind or inertia. Self-attention models relationships across time steps directly, and the gated residual design of GTrXL offers a useful stabilization mechanism for reinforcement learning [7,8]. Here, this mechanism is adapted as a compact GTrXL-inspired gated-attention module over a fixed history window; Section 3.3 defines its exact scope.

We study a multimodal history-window SAC policy with current-frame obstacle-related inputs for urban low-altitude UAV target navigation. Its observation includes a front-view grayscale image, target-relative information, UAV motion state, sparse LiDAR-like range sectors, handcrafted safety cues, and the previous action. A GTrXL-inspired gated-attention encoder processes navigation features over a fixed history window, while a current-frame safety branch preserves immediate access to local clearance information. The complete policy is referred to as HW-GA-SAC. Throughout the paper, safety-related terms refer to obstacle observations and reward terms; collision and clearance remain empirical metrics. The policy does not enforce constraints, use a separate safety filter, or provide formal guarantees. Our evaluation covered task outcomes, successful-path stretch, average return, learning dynamics across independent training seeds, sensor perturbation, inference cost, and ablation studies. PPO, TD3, SAC, SAC-Large, GRU-SAC, and LSTM-SAC were tested under the same final main-comparison protocol. The evidence and claims were limited to MuJoCo simulation and procedurally generated city layouts; real-flight reliability was not evaluated in this study.

Relative to prior GTrXL-SAC UAV control work, this study focuses on the design and controlled evaluation of a multimodal urban-navigation policy. The SAC objective and scaled dot-product attention remain standard; the contribution lies in the task-specific

representation, fixed-window temporal interface, current-frame safety branch, and their evaluation under a unified protocol. The main contributions are as follows:

1. A multimodal reinforcement-learning task is formulated for three-dimensional urban low-altitude target navigation. The observation integrates front-view grayscale images, target-relative state, UAV motion state, sparse LiDAR-like range sectors, handcrafted safety cues, and action history in a partially observable continuous-control setting.
2. A current-frame safety branch is designed so that both the actor and critic receive vertical clearance, sparse LiDAR-like range sectors, and handcrafted safety cues. This representation makes local obstacle information available during action selection.
3. A GTrXL-inspired gated-attention encoder is integrated into SAC-family policies to evaluate fixed-window temporal context without claiming the persistent-memory mechanism of Transformer-XL or the original GRU-style GTrXL gate.
4. A unified simulation protocol is used to analyze algorithms, network capacity, recurrent and attention-based history modeling, safety and visual inputs, learning dynamics, city-layout split, additional generated cities, wind sensitivity, sensing perturbations, and inference cost.

2. Related Work

2.1. UAV Autonomous Navigation and Deep Reinforcement Learning

UAV autonomous navigation usually requires path planning, obstacle avoidance, and continuous motion control. Classical methods include graph search, heuristic search, sampling-based planning, and potential-field obstacle avoidance. Dijkstra, A*, probabilistic roadmaps, rapidly exploring random trees, and artificial potential fields are representative examples [9–13]. These methods are interpretable and controllable, but they often rely on strong environmental assumptions when the scene is unknown, occluded, or highly structured.

Deep reinforcement learning provides a complementary route for decision making in complex environments because the control policy is learned through interaction. DDPG extends deterministic policy gradients to high-dimensional continuous action spaces [14]; TD3 reduces function-approximation error in actor–critic learning [15]; PPO improves training stability by constraining policy updates [16]; and SAC balances return maximization and exploration through a maximum-entropy objective. Studies on mobile robots, hybrid aerial-underwater vehicles, and multi-UAV path planning show that deep reinforcement learning can support mapless navigation, continuous control, and obstacle avoidance, although algorithmic behavior varies across two-dimensional, three-dimensional, and multi-agent tasks [17–19]. SAC-based and memory-enhanced UAV navigation studies also suggest that temporal context can improve performance under partial observability [20,21].

2.2. Multimodal Perception for Urban Navigation

In the considered urban low-altitude setting, relying on a single modality can be fragile. Prior mapless-navigation studies have used sparse range measurements and target-relative states to support continuous obstacle avoidance, while recent learning-based navigation systems also combine range and visual cues when the task requires both local clearance and scene understanding [22,23]. Visual input captures building facades, street direction, skybridges, and traversable space. Range observations encode near-obstacle distance and collision risk. Target-relative state and UAV motion state provide goal-directed control signals, and previous actions expose recent control responses when wind disturbance or inertia creates delayed effects. The design problem is not simply how to concatenate these

signals, but which information should affect immediate action selection and which should be processed through temporal context.

The proposed representation keeps this separation explicit. Navigation-related variables and front-view images are fused into a 128-dimensional navigation feature. Vertical clearance, sparse LiDAR-like range sectors, and handcrafted safety cues are encoded by the current-frame safety branch, keeping near-obstacle information available to the actor while the navigation branch benefits from history-window modeling.

2.3. Temporal Modeling Under Partial Observability

Partially observable navigation requires the policy to infer hidden context from recent measurements. Recurrent architectures such as GRU and LSTM can summarize observation histories, but their hidden-state dynamics may be difficult to optimize in off-policy reinforcement learning. Transformer-based architectures offer an alternative by directly attending to multiple past steps [24]. Transformer-XL extends the attention context across segments [7], while GTrXL introduces gated residual connections to stabilize transformer learning in reinforcement-learning settings [8]. Existing work has applied GTrXL-SAC to UAV path planning and obstacle-aware control, indicating that attention-based temporal structures can be combined with continuous control [25].

The present work follows this line with a GTrXL-inspired gated-attention encoder, and we evaluated the resulting fixed-window design in a multimodal, safety-conditioned urban UAV task.

2.4. Safety-Conditioned Reinforcement Learning for Collision Avoidance

Formal safe reinforcement learning treats constraint satisfaction through methods such as constrained policy optimization, analytic safety layers, and separate recovery policies [26–28]. HW-GA-SAC uses none of these mechanisms. Its actor and critic receive sparse LiDAR-like range sectors and handcrafted safety cues through a current-frame branch, and the shaped reward penalizes obstacle proximity. Here, “safety-conditioned” describes this use of observations and rewards; it does not imply certified constraints or guaranteed collision avoidance.

Table 1 positions this work relative to the prior GTrXL-SAC UAV study in [25]. The comparison emphasizes task formulation and system-level validation.

Table 1. Positioning of this work relative to the prior GTrXL-SAC UAV study in [25].

Aspect	Prior GTrXL-SAC UAV Study [25]	This Work
Task	Path planning/obstacle-aware control	Urban low-altitude target navigation
Observation	Relatively simpler state representation	Image, state, sparse LiDAR-like range sectors, handcrafted safety cues, and action history
Temporal model	GTrXL/SAC	GTrXL-inspired gated-attention encoder over a fixed window
Current-frame safety branch	Not explicitly isolated in ablation	Included for actor and critic and evaluated by ablation
Urban wind	Not a primary evaluation factor	Parameterized urban wind disturbance
Evaluation focus	Navigation and obstacle-aware control	Robustness, failure modes, and path behavior

3. Problem Formulation and Technical Preliminaries

3.1. Reinforcement-Learning Formulation

The three-dimensional urban UAV target-navigation task is modeled as a partially observable Markov decision process (POMDP) [29,30]. A POMDP can be written as

$$\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{O}, P, \Omega, r, \gamma), \quad (1)$$

where $\mathcal{S}$ is the hidden state space, $\mathcal{A}$ is the continuous action space, $\mathcal{O}$ is the observation space, $P(s_{t+1}|s_t, a_t)$ is the transition kernel, $\Omega(o_t|s_t)$ is the observation kernel, r is the reward function, and γ is the discount factor. At each time step, the agent receives an observation, outputs a continuous action, and the environment updates the state under both control input and external disturbance. Because local vision, local range, and target-relative information do not fully describe the environment state, the policy conditions on a finite history,

$$h_t = \{o_{t-K+1}, a_{t-K+1}, \dots, o_{t-1}, a_{t-1}, o_t\}, \quad (2)$$

instead of the unavailable full state s_t .

The policy objective is

$$J(\pi) = \mathbb{E}_{\tau \sim \pi} \left[\sum_{t=0}^{T-1} \gamma^t r(s_t, a_t) \right], \quad (3)$$

where τ is a trajectory induced by policy π , and T is the maximum episode length. In urban navigation, the reward must account for goal reaching, progress, safety margin, and avoidance of inefficient wandering.

3.2. Soft Actor–Critic

SAC is an off-policy actor–critic algorithm based on maximum-entropy reinforcement learning [5,6]. Equations (4)–(7) restate the standard SAC optimization used in the implementation; the objective, Bellman target, and entropy mechanism are not modified. They are included to define how the proposed observation representation z_t interfaces with the unchanged SAC backbone. The simulator reward is defined by the hidden physical state and event flags, whereas the policy receives the finite-history navigation representation and current-frame safety feature detailed in Section 4.4. The maximum-entropy objective is

$$J_{\text{SAC}}(\pi) = \mathbb{E}_{\tau \sim \pi} \left[\sum_{t=0}^{T-1} \gamma^t (r(s_t, a_t) + \alpha \mathcal{H}(\pi(\cdot|z_t))) \right], \quad (4)$$

where α is the entropy temperature and $\mathcal{H}(\pi(\cdot|z_t))$ is policy entropy. A larger temperature encourages exploration, while a smaller temperature makes the policy more deterministic. Automatic temperature adjustment is used in all SAC-family experiments.

Two Q networks reduce overestimation, and target networks with soft updates improve training stability. For a replay transition $(z_t, a_t, r_t, z_{t+1}, \delta_t)$, where δ_t is 1 for episode termination and 0 otherwise, the soft target is

$$y_t = r_t + \gamma(1 - \delta_t) \left[\min_{i \in \{1,2\}} Q_{\bar{\theta}_i}(z_{t+1}, a'_{t+1}) - \alpha \log \pi_{\phi}(a'_{t+1}|z_{t+1}) \right], \quad (5)$$

where $a'_{t+1} \sim \pi_{\phi}(\cdot|z_{t+1})$, and $\bar{\theta}_i$ denotes target-network parameters. The critic minimizes the squared Bellman residual

$$J_Q(\theta_i) = \mathbb{E}_{\mathcal{D}} \left[(Q_{\theta_i}(z_t, a_t) - y_t)^2 \right], \quad (6)$$

while the actor is updated by

$$J_{\pi}(\phi) = \mathbb{E}_{z_t \sim \mathcal{D}, a_t \sim \pi_{\phi}} \left[\alpha \log \pi_{\phi}(a_t|z_t) - \min_{i \in \{1,2\}} Q_{\theta_i}(z_t, a_t) \right]. \quad (7)$$

In the present task, the actor outputs normalized continuous control actions, and the critic estimates long-term value from multimodal observation features and actions.

3.3. Fixed-Window Gated-Attention Temporal Encoding

Self-attention lets the current decision attend directly to relevant historical time steps [24]. GTrXL introduces gated residual connections that adaptively combine input features and transformed temporal features [8]. The present encoder uses this gating principle but not the full GTrXL architecture.

This study uses a GTrXL-inspired gated-attention encoder over the most recent $K = 8$ navigation features. Unlike Transformer-XL/GTrXL, it passes no persistent memory state across replay samples or test segments. Its residual gate is also simpler than the original GRU-style GTrXL gate: Equation (11) uses a sigmoid interpolation between the block input and transformed feature. These choices define a fixed-window encoder rather than a new SAC objective or a complete GTrXL implementation; hence the name HW-GA-SAC.

Given a feature sequence,

$$F_t = \{f_{t-K+1}, f_{t-K+2}, \dots, f_t\}, \quad (8)$$

standard scaled dot-product self-attention computes temporal context from query, key, and value projections:

$$Q = F_t W_Q, \quad K_F = F_t W_K, \quad V = F_t W_V, \quad (9)$$

$$\text{Attn}(F_t) = \text{softmax}\left(\frac{QK_F^\top}{\sqrt{d_k}} + M\right)V, \quad (10)$$

where M is a causal mask and d_k is the key dimension. The implemented residual interpolation is

$$G(x, y) = (1 - u) \odot x + u \odot y, \quad u = \sigma(W_u[x, y] + b_u), \quad (11)$$

where x is the block input and y is the transformed feature. The encoder has three layers, four attention heads, $d_{\text{model}} = 128$, a 256-dimensional feedforward sublayer, causal masking, and no dropout. The final token represents the current decision. Replay sequences contain six burn-in steps and two learning steps, giving the complete eight-step history window. The task-specific modification is therefore the interface among multimodal navigation encoding, fixed-window gated attention, and the current-frame safety branch, rather than a change to the underlying attention equation.

3.4. Urban Wind Disturbance Model

The experiments used MuJoCo rigid-body dynamics and a parameterized urban wind field for controlled policy comparison [31]. At the beginning of each episode, the horizontal base wind was sampled as

$$v_b = V_b [\cos \psi_b, \sin \psi_b, 0]^\top, \quad V_b \sim \mathcal{U}(2, 4), \quad \psi_b \sim \mathcal{U}(0, 2\pi). \quad (12)$$

The temporally correlated horizontal gust followed a zero-mean Ornstein–Uhlenbeck process [32]:

$$g_{t+1} = g_t - \theta g_t \Delta t + \sigma \sqrt{\Delta t} \varepsilon_t, \quad (13)$$

where $\theta = 0.42$, $\sigma = 0.95$, $\Delta t = 0.02$ s was the MuJoCo substep, and $\varepsilon_t \sim \mathcal{N}(0, I)$; the two horizontal gust components were clipped to ± 3.2 m/s and the vertical component was zero.

The local wind field depended on position p_t , nearby building bounding boxes, relative building height, corridor width, and inflow direction. Its implemented form can be summarized as

$$v_w(p_t, t) = [1 - \lambda_w(p_t)] \{G_c(p_t)[v_b + g_t] + v_{sh}(p_t, t)\} \eta(p_t, t), \quad (14)$$

where $G_c \geq 1$ was the street-canyon acceleration gain, $\lambda_w \in [0, 1)$ attenuated mean flow in downwind wakes, v_{sh} was a cross-canyon shear term, and η was clipped zero-mean turbulence whose variance increased in canyon and wake regions. The candidate buildings were searched within 95 m. Canyon acceleration was activated when sufficiently tall buildings bounded both sides of the flow-aligned corridor; wake strength decayed quadratically with normalized downwind distance and expanded laterally with building height. Four wind regimes—nominal, canyon-accelerated, wake-turbulent, and crosswind-shear—were resampled every 4–8 s and scaled the corresponding mean-flow, gust, shear, and turbulence terms. This construction introduced reproducible temporal and spatial disturbance variation consistent with the role of structured wind fields in UAV simulation [33].

At each physics substep, the relative airflow was converted to translational acceleration:

$$a_{\text{wind}} = k_l v_{\text{rel}} + k_q \left(\frac{\|v_{\text{rel}}\|}{v_{\text{ref}}} \right) v_{\text{rel}}, \quad v_{\text{rel}} = v_w - v_{\text{uav}}, \quad (15)$$

with $k_l = 0.32$, $k_q = 0.06$, and $v_{\text{ref}} = 8$ m/s. The external term ma_{wind} was added to the first three generalized-force entries together with the commanded translational acceleration. No wind-induced torque was added. Applying the disturbance at the center of mass deliberately isolated translational disturbance rejection and avoided introducing rotor-level moments without calibrated aerodynamic coefficients. Accordingly, the wind experiment evaluated policy behavior under structured force disturbances, rather than claiming rotor-resolved or full six-degree-of-freedom aerodynamic fidelity.

Table 2 summarizes the principal parameters of the default urban wind model.

Table 2. Principal parameters of the default urban wind model.

Component	Value	Role
Base wind speed and heading	2–4 m/s; 0–360°	Episode-level horizontal inflow
OU gust	$\theta = 0.42$, $\sigma = 0.95$, clip 3.2 m/s	Temporally correlated horizontal variation
Wind-regime duration	4–8 s	Switching among nominal, canyon, wake, and shear modes
Canyon gain	Maximum 1.65	Flow-aligned acceleration in bounded street corridors
Cross-canyon shear	Maximum 1.8 m/s	Lateral disturbance near asymmetric canyon geometry
Wake attenuation	Maximum 0.72	Mean-flow reduction downstream of buildings
Wake turbulence scale	1.75; total clip 4.5 m/s	Increased stochastic variation in wake regions
Aerodynamic mapping	$k_l = 0.32$, $k_q = 0.06$, $v_{\text{ref}} = 8$ m/s	Relative-airflow-to-acceleration conversion

4. Method

4.1. Overall Framework

HW-GA-SAC combines multimodal observation encoding, history-window temporal modeling, and SAC-based continuous control. At each time step, the environment outputs a 42-dimensional low-dimensional state vector and a front-view image. The low-dimensional state includes UAV position and velocity, attitude, vertical clearance, remaining time, yaw rate, target-relative position, target-navigation indicators, handcrafted safety cues, sparse LiDAR-like range sectors, and the previous action. The image provides forward-looking information about buildings, streets, and traversable space.

The policy encodes the low-dimensional vector and image through separate branches followed by gated fusion. Depending on the method variant, the current navigation feature is either used directly by a feedforward SAC policy or passed through a GRU, LSTM, or GTrXL-inspired gated-attention encoder. The history-enhanced navigation feature is concatenated with the current-frame safety feature to form a joint policy representation. The actor outputs a continuous action, and the critic estimates the action value. Figure 1 summarizes this closed-loop data flow:

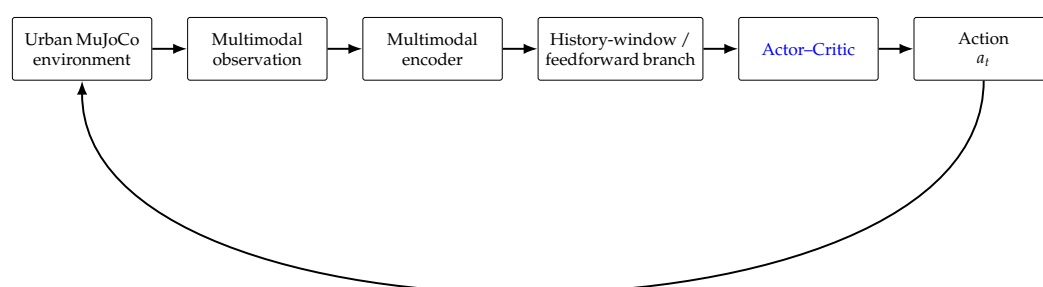


Figure 1. Overall data flow of the multimodal UAV navigation framework. The policy encodes image, navigation-state, current-frame safety, and action-history information, applies feedforward or fixed-window history processing depending on the method variant, and outputs a continuous UAV control action through the actor–critic.

4.2. Multimodal Observation Input

The observation is written as

$$o_t = \{s_t, I_t\}, \quad (16)$$

where s_t is a 42-dimensional low-dimensional state vector and I_t is a front-view grayscale image. Table 3 lists the state-vector composition used in the final experiments.

Terminology is used as follows throughout the manuscript. The current-frame safety branch denotes the complete 16-dimensional branch input and its encoder; it contains two vertical-clearance values, eight sparse LiDAR-like range sectors, and six handcrafted safety cues. The latter are minimum obstacle distance, obstacle closing speed, and front, rear, left, and right safety distances. The phrase actor-side safety inputs refers collectively to these current-frame quantities when discussing ablations. In these component names, “safety” refers only to obstacle-proximity information, not to formal constraint enforcement or certification. These inputs are computed from local range observations and geometric proximity queries, not from a privileged global map. The front-view input is a 96×128 single-channel grayscale image. Combining the low-dimensional navigation state with local visual information reduces the burden on pure vision and improves scene representation compared with state-only input [34,35].

Table 3. Composition of the low-dimensional state vector.

Category	Dim.	Meaning
Position and velocity	6	UAV position and linear velocity, describing self-motion state
Attitude	4	Attitude quaternion
Vertical clearance	2	Downward and upward local clearance, normalized by maximum detection range
Remaining time and yaw rate	2	Remaining episode-time ratio and current yaw rate
Target-relative position	3	Three-dimensional target position relative to the UAV
Target-navigation indicators	7	Target distance, horizontal distance, altitude error, sine/cosine of bearing error, closing speed, and lateral speed
Handcrafted safety cues	6	Minimum obstacle distance, obstacle closing speed, and front/back/left/right safety distances
Sparse LiDAR-like range sectors	8	Local obstacle distances in eight horizontal sectors, normalized by maximum range
Previous action	4	Previous normalized control action, representing control continuity

4.3. Action Space and Control Interface

The action space is continuous. The policy outputs a four-dimensional normalized vector corresponding to body-frame forward velocity command, body-frame lateral velocity command, vertical velocity command, and yaw residual or yaw-rate command. Each action component lies in $[-1, 1]$. The body-frame control interface supports local maneuvers such as forward motion, lateral avoidance, climbing or descending, and heading adjustment. The simulator maps normalized actions to physical velocity or yaw commands and applies them through the MuJoCo dynamics.

4.4. Multimodal Encoder with Current-Frame Safety Inputs

The low-dimensional state is divided into navigation and current-frame safety components. In the body-frame observation setting used here, absolute position is not fed directly into the navigation encoding branch. The navigation branch receives the remaining non-safety navigation features, denoted as $x_t^{\text{nav}} \in \mathbb{R}^{23}$. The current-frame safety branch receives vertical clearance, handcrafted safety cues, and sparse LiDAR-like range sectors, denoted as $x_t^{\text{safe}} \in \mathbb{R}^{16}$.

The navigation vector is encoded by an MLP:

$$f_t^x = \Phi_x(x_t^{\text{nav}}), \quad f_t^x \in \mathbb{R}^{128}. \quad (17)$$

The image branch uses three convolutional layers. The first layer is a 3×3 convolution with stride 1, padding 1, and 32 output channels. The second layer is a 3×3 convolution with stride 2, padding 1, and 48 output channels. The third layer is a 3×3 convolution with stride 1, padding 1, and 48 output channels. A 2×2 adaptive average pooling layer is followed by a linear projection, LayerNorm, and ReLU to obtain a 128-dimensional image feature:

$$f_t^I = \Phi_I(I_t), \quad f_t^I \in \mathbb{R}^{128}. \quad (18)$$

Gated fusion injects visual information into the navigation feature:

$$f_t^{\text{nav}} = \Phi_f([f_t^x, g_t \cdot f_t^l]), \quad g_t = \sigma(W_g[f_t^x, f_t^l]). \quad (19)$$

The gate controls the strength of visual feature injection. The current-frame safety vector is encoded by a separate MLP:

$$f_t^{\text{safe}} = \Phi_s(x_t^{\text{safe}}), \quad f_t^{\text{safe}} \in \mathbb{R}^{48}. \quad (20)$$

When GRU, LSTM, or HW-GA-SAC is enabled, the temporal module operates on the 128-dimensional navigation-feature sequence and outputs the history-enhanced navigation feature h_t^{nav} . In feedforward SAC, $h_t^{\text{nav}} = f_t^{\text{nav}}$. The final policy representation is

$$z_t = [h_t^{\text{nav}}, f_t^{\text{safe}}], \quad z_t \in \mathbb{R}^{176}. \quad (21)$$

Both the actor and critic use this representation:

$$a_t \sim \pi_\phi(\cdot | z_t), \quad (22)$$

$$Q_{\theta_i} = Q_{\theta_i}(z_t, a_t), \quad i \in \{1, 2\}. \quad (23)$$

The current-frame safety branch is concatenated after temporal encoding so that near-obstacle information remains available during action selection. If these quantities were passed only through the temporal encoder, their effect could depend on temporal attention allocation. The resulting representation lets both the actor and critic use sparse LiDAR-like range sectors and handcrafted safety cues from the current observation. Figure 2 details the navigation branch, temporal module, and current-frame safety branch. Similar structured representations are also useful in multi-robot path-planning studies [36].

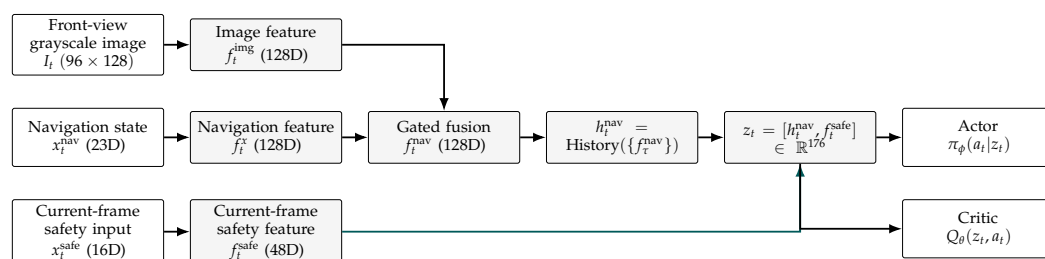


Figure 2. Multimodal encoding, fixed-window history modeling, and the current-frame safety branch. The current-frame safety feature is concatenated with the history-enhanced navigation representation before the actor and critic.

4.5. Policy Variants

Seven methods are compared: PPO, TD3, SAC, SAC-Large, GRU-SAC, LSTM-SAC, and HW-GA-SAC. The environment, observation input, action space, reward function, training budget, and evaluation protocol are kept consistent wherever the algorithms allow. PPO and TD3 serve as common continuous-control baselines. SAC is the feedforward maximum-entropy baseline. SAC-Large tests whether a larger feedforward network is sufficient. GRU-SAC and LSTM-SAC represent recurrent memory, and HW-GA-SAC is the GTrXL-inspired gated-attention variant over a fixed history window.

5. Reward Design

The task reward supplies dense learning signals for long-range navigation. Classical potential-based shaping preserves the optimal policy when the additional reward has the form

$$F(s_t, s_{t+1}) = \gamma\Phi(s_{t+1}) - \Phi(s_t), \quad (24)$$

where Φ is a potential function [37]. The reward used here also encodes safety margin, terminal-event priority, and effective progress, and therefore it is not claimed to preserve the optimum of an unshaped objective [37,38]. It is an engineering task specification rather than a control-theoretic safety constraint; safety is assessed empirically through collision, clearance, and ablation metrics. Curriculum learning can also improve difficult long-horizon control tasks [39], but the reward parameters are kept fixed across all methods.

The reward consists of goal-progress reward, closing-speed reward, safety penalty, living cost, and terminal reward:

$$r_t = R_{progress} + R_{closing} + R_{safe} + R_{living} + R_{terminal}. \quad (25)$$

The progress term clips the one-step target-distance reduction before scaling:

$$\Delta d_t = \text{clip}(d_{t-1} - d_t, -v_{cap}\Delta t, v_{cap}\Delta t), \quad (26)$$

$$R_{progress} = w_p \frac{\Delta d_t}{d_{norm}}, \quad (27)$$

where d_t is the target distance and d_{norm} is a distance-normalization scale computed from the initial target distance, current target distance, goal threshold, and a 1 m lower bound. The closing-speed reward is gated by local clearance so that direct target approach is not encouraged near obstacles:

$$R_{closing} = w_c \frac{\text{clip}(v_{goal}, 0, v_{cap})}{v_{cap}} g(d_{clear}), \quad (28)$$

$$g(d_{clear}) = \left[\text{clip}\left(\frac{d_{clear}/d_{safe} - r_{danger}}{r_{full} - r_{danger}}, 0, 1\right) \right]^{\kappa_g}. \quad (29)$$

Here, v_{goal} is the target-closing speed, d_{clear} is the current minimum obstacle clearance, and $g(d_{clear})$ releases the closing-speed reward only when the UAV is sufficiently clear of nearby obstacles. The safety penalty uses a speed-dependent dynamic safety margin:

$$d_{safe} = \text{clip}(d_0 + t_r v + v^2 / (2a_{safe}), d_{min}, d_{max}), \quad (30)$$

where v is UAV speed, t_r is reaction time, and a_{safe} is safe deceleration. To avoid numerical explosion at very small clearances, an effective distance is used:

$$d_{eff} = \max(d_{clear}, \max(\min(0.5, 0.1d_{safe}), 10^{-3})). \quad (31)$$

The safety penalty is then

$$R_{safe} = -w_s \max\left(\frac{1}{d_{eff}} - \frac{1}{d_{safe}}, 0\right) d_{safe} m_v, \quad (32)$$

$$m_v = 1 + \beta_s \text{clip}\left(\frac{\max(v_{app}, 0)}{v_{cap}}, 0, 1\right), \quad (33)$$

where v_{app} is the obstacle-approach speed. This term penalizes high-speed obstacle approach more strongly while leaving receding motion unamplified. A small living cost discourages inefficient hovering:

$$R_{living} = -c_l, \quad c_l = 0.0025. \quad (34)$$

Terminal rewards include success, collision, and timeout. If multiple terminal conditions occur at the same simulation step, collision has priority over success and timeout. Reaching the goal and colliding in the same step is counted as collision, following a safety-first interpretation. Table 4 lists the main numerical settings used by the reward function in all the reported experiments.

Table 4. Main reward-function parameters.

Parameter	Value	Meaning
w_p	100.0	Goal-progress reward weight
w_c	0.012	Closing-speed reward weight
w_s	0.6	Safety penalty weight
c_l	0.0025	Per-step living cost
$R_{success}$	150.0	Terminal reward for reaching the goal
$R_{collision}$	−150.0	Terminal penalty for collision
$R_{timeout}$	−90.0	Terminal penalty for timeout
v_{cap}	8.0 m/s	Closing-speed clipping upper bound
d_0	4.0 m	Base safety distance
t_r	0.3 s	Reaction time
a_{safe}	5.0 m/s ²	Safe deceleration
d_{min}, d_{max}	5.0 m, 12.0 m	Lower and upper bounds of dynamic safety margin
r_{danger}, r_{full}	0.4, 1.0	Danger and full-release ratios in the safety gate
κ_g	1.5	Safety-gate curve exponent
β_s	0.75	Obstacle-closing-speed amplification factor

All the reward parameters were shared by all the methods, and they were fixed before the final comparisons using physical-scale constraints and short pilot runs on the training city. Equal-magnitude success and collision terminal values balanced the goal-reaching reward against the collision penalty, while the smaller timeout penalty discouraged prolonged wandering without equating it to collision. Independently, the terminal-event priority rule classified simultaneous goal reaching and collision as collision. The progress weight made target-distance reduction the dominant dense term on the 850–1000 m task. The closing-speed coefficient was small and was released only when the clearance gate was open. The dynamic safety-margin parameters followed a reaction-distance heuristic with $t_r = 0.3$ s, $a_{safe} = 5.0$ m/s², and a 5–12 m clipped margin. No method-specific reward re-tuning was used.

6. Experimental Setup

6.1. MuJoCo Three-Dimensional Urban Environment

The environment covered a 2000 m × 2000 m world with an altitude range from 2 m to 180 m. The city contained 12 × 12 blocks, including major roads, minor streets, high-rise buildings, super-high-rise buildings, rooftop structures, plazas, and skybridges. The UAV started at 50 m altitude, and the target altitude was also 50 m. The target distance range was 850 m to 1000 m, with a minimum start-goal distance of 700 m. Each episode had a maximum length of 2200 steps, and the goal-reaching threshold was 10 m. These values were taken from the training configuration.

The environment enabled eight sparse LiDAR-like range sectors with a maximum range of 120 m. It also used a front-view grayscale camera with a resolution of 96 × 128. The UAV radius was 1.0 m, the mass was 1.5 kg, the maximum horizontal speed was 18.0 m/s, and the maximum vertical speed was 10.0 m/s. The action space

was a four-dimensional normalized continuous vector in $[-1, 1]$. The control interval was $\Delta t = 0.10$ s; each control step contained five MuJoCo substeps, so the physics timestep was 0.02 s and the policy was queried at 10 Hz during evaluation. Figure 3 shows representative views of the three-dimensional urban environment.

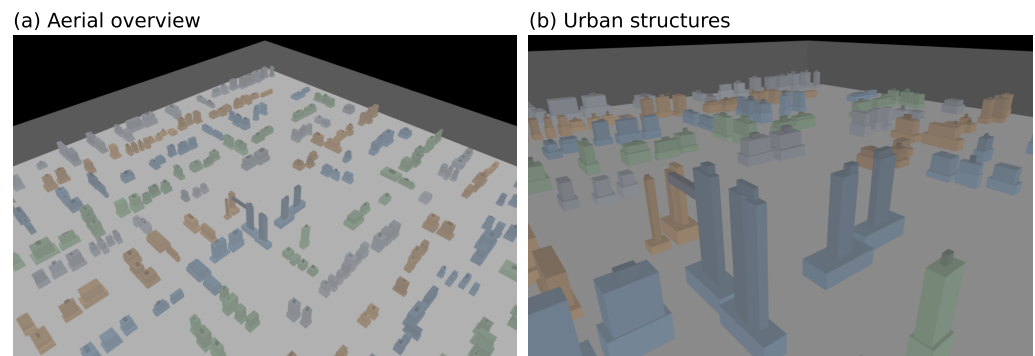


Figure 3. Actual three-dimensional MuJoCo environment used in the experiments: (a) aerial overview of the procedurally generated city and (b) a closer view showing varied building heights, rooftop structures, street corridors, and skybridges.

6.2. Urban Wind Environment

Wind disturbance was enabled in the default setting using the formulation and parameters in Section 3.4 and Table 2. The no-wind condition removed the complete disturbance term. The strong-wind condition retained the same spatial urban-wind construction and multiplied gust intensity by 1.75, allowing the sensitivity test to vary disturbance strength without changing obstacle geometry or the learned policy.

6.3. Compared Methods and Evaluation Protocol

The seven methods were trained and evaluated under the same urban environment, observation input, action space, reward function, training budget, and evaluation procedure. Each method used five independent training seeds. Evaluation was performed on five urban layouts: city 1, city 2, city 3, city 4, and city 42. The layout city 42 was the training city and was used as a training-city replay test, while cities 1–4 were unseen layouts for cross-layout evaluation. The main five-city averages mixed four unseen layouts with one training-city replay layout. Because training used only one city layout, the cross-city results provide preliminary robustness evidence within this procedural setting rather than large-scale distributional generalization.

For each training seed and city layout, 200 evaluation episodes were run. The full evaluation contained $7 \text{ methods} \times 5 \text{ training seeds} \times 5 \text{ layouts} \times 200 \text{ episodes}$, totaling 35,000 evaluation episodes. Statistics were first averaged within each training seed over the five layouts, and then the mean and sample standard deviation were computed across the five training seeds. The independent statistical unit was the trained model seed, not a single episode. Because five trained seeds remain a modest sample for formal inference, the tables report mean $\pm$ sample standard deviation, and the comparisons are interpreted descriptively. The experimental results are rounded to metric-appropriate precision: percentage outcomes and large-scale quantities are normally reported to the nearest whole unit, while very small rates, path-stretch ratios, average speed, and inference latency retain the minimum precision needed for meaningful interpretation. Consequently, the independently rounded outcome rates may differ from a total of 100% by at most one percentage point. Rounding was applied after aggregation over seeds and layouts. For this reason, a pooled five-city value could not always be reconstructed exactly from the separately rounded city 42 and cities 1–4 values.

To avoid reporting an under-tuned PPO baseline, a limited PPO hyperparameter stabilization pass was conducted before the final main comparison. Compared with the preliminary PPO setting, the stabilized configuration used a larger rollout batch, a larger minibatch size, a slightly stronger entropy regularization coefficient, and fewer update epochs per rollout. As shown in Table 5, this improved the five-seed success rate from 8% to 27%. The stabilized PPO configuration was used in the final seven-method main comparison. A preliminary PPO actor-input diagnostic from the separate ablation protocol is reported in Appendix B.

Table 5. Limited PPO hyperparameter stabilization before the final seven-method comparison.

PPO Setting	Rollout Steps	Minibatch	Entropy Coef.	Update Epochs	Seeds	Success Rate/%
Preliminary PPO	8192	64	0.001	8	5	8
Stabilized PPO	16,384	256	0.005	6	5	27

Additional core-method evaluations were performed for TD3, SAC, and HW-GA-SAC on ten unseen city-generation seeds, 43–52, and under no-wind and strong-wind settings. These checks are reported separately from the seven-method comparison because they probed broader city and disturbance conditions for the selected core methods.

Two implementation-oriented checks were also added for the core methods. First, a sensor-perturbation screening check injected zero-mean Gaussian noise into the image and range observations with normalized standard deviations of 0.01 and 0.03, and it added structural observation latency of 10, 30, or 50 ms. Combined settings used mild noise with 30 ms latency and moderate noise with 50 ms latency. Second, inference profiling measured the actor–critic parameter count, checkpoint size, CUDA inference latency, and peak CUDA memory on the same NVIDIA L40S platform. These diagnostics covered mild sensing perturbations and isolated policy-network cost, while integrated onboard evaluation was left to staged physical testing.

The cross-algorithm comparison among PPO, TD3, and SAC was mainly an engineering baseline because these algorithms differ in objective, data usage, and update rules. Conclusions about network capacity and temporal modeling were drawn primarily from the SAC-family comparison among SAC, SAC-Large, GRU-SAC, LSTM-SAC, and HW-GA-SAC.

6.4. Metrics

Table 6 defines the evaluation metrics. Success, collision, and timeout rates describe task outcomes. Average return, speed, final target distance, successful-path stretch, and minimum clearance characterize behavioral differences.

Table 6. Evaluation metrics.

Metric	Definition
Success rate	Fraction of test episodes in which the UAV reached the goal region.
Collision rate	Fraction of test episodes ending with collision.
Timeout rate	Fraction of test episodes that ended without collision but failed to reach the goal within the maximum episode length.
Average return Average speed	Overall reward reflecting progress, safety, efficiency, and terminal outcome. Flight-efficiency indicator.
Final target distance	Distance between UAV and target at episode termination.
Successful-path stretch	Path-length ratio, computed as actual path length divided by straight-line start-goal distance for successful episodes; lower values indicate more direct successful trajectories.

Table 6. *Cont.*

Metric	Definition
Successful-path minimum clearance	Minimum obstacle clearance along successful trajectories.

6.5. Training Details and Hyperparameters

All the methods shared the same environment, reward, and evaluation configuration. Algorithm-specific parameters were taken from the corresponding configuration files. Training and evaluation were performed on a platform equipped with an NVIDIA L40S GPU. The SAC-family methods used CUDA and automatic mixed precision. In every final temporal-policy experiment, the fixed history length was $K = 8$; the replay sequences used six burn-in steps followed by two learning steps. Table 7 summarizes the shared training setup, evaluation cities, wind settings, and main algorithm hyperparameters.

Table 7. Training details and main hyperparameters.

Parameter	Value	Description
Hardware	NVIDIA L40S GPU	Platform used for all deep-reinforcement-learning training and evaluation.
Training city	city 42	City generation seed 42.
Main-test cities	cities 1–4, city 42	Four unseen layouts and one training-city replay layout.
Additional city seeds	city 43–city 52	Ten unseen layouts for core-method generalization tests.
Wind sensitivity	no wind, default, strong	Strong wind uses a gust multiplier of 1.75.
Total training steps	1.5 M	Same training budget for all seven methods.
Maximum episode length	2200	Environment <code>max_steps</code> .
Goal threshold	10 m	Environment <code>goal_threshold</code> .
Parallel environments	4	Four environments for parallel sampling.
Replay buffer	200,000	Off-policy replay capacity for SAC-family and TD3.
Batch size	64	Mini-batch size for parameter updates.
Warmup steps	10,000	Random sampling before off-policy updates.
Discount factor γ	0.998	Discount for cumulative return.
Soft update coefficient τ	0.002	Target-network soft update coefficient.
Actor learning rate	0.0001	Actor optimizer step size.
Critic learning rate	0.0002	Critic optimizer step size.
Temperature learning rate	0.0001	SAC entropy-temperature optimizer step size.
Automatic entropy temperature	Enabled; $\alpha_0 = 0.1$, $\alpha \in [0, 1]$	Log-temperature update with target entropy $-\dim(\mathcal{A})$.
PPO main parameters	$\lambda = 0.95$, clip 0.2	Stabilized setting: rollout steps 16,384, update epochs 6.
TD3 main parameters	Policy delay 2	Policy noise 0.2, noise clip 0.5.
Sequence setting	burn-in 6, learn 2	Shared sequence setting for temporal methods.
History length K	8	Used by LSTM-SAC, GRU-SAC, and HW-GA-SAC.
Gated-attention encoder	$d_{model} = 128$	GTrXL-inspired gated-attention; 4 heads, 3 layers, feed-forward dim 256, dropout 0.
GRU/LSTM encoders	$d_{model} = 128$	GRU has 3 layers, LSTM has 1 layer, both use $K = 8$.

The configuration files for SAC, SAC-Large, GRU-SAC, LSTM-SAC, HW-GA-SAC, PPO, and TD3 were fixed before training and archived with the implementation package. The reported baselines are reproducible under the same environment, reward, training

budget, and evaluation protocol; global hyperparameter optimization for each algorithm was outside the scope of this comparison.

6.6. Implementation Details

The implementation used Python 3.8.20, PyTorch 2.4.1 with CUDA 12.1, NumPy 1.24.3, and MuJoCo 3.2.3. All the policies used normalized continuous actions in $[-1, 1]$, and the evaluation used deterministic action selection rather than stochastic policy sampling. Random seeds were set for Python, NumPy, and PyTorch at the beginning of each training run; environment reset seeds were generated from the corresponding training seed. Deterministic PyTorch kernels were not forced, so the reported seed-level variability reflects both learning stochasticity and normal GPU execution variability.

The actor and critic instantiated separate observation encoders, so their encoder parameters were not shared during optimization. SAC-Large kept the same observation encoder as SAC but replaced the two-layer $[256, 256]$ MLP heads with wider $[640, 640, 512]$ heads. For the temporal methods, each replay sample contained a fixed-length sequence of eight steps; the first six steps provided burn-in context, and the last two steps contributed to the learning loss. Normalized actions were clipped to $[-1, 1]$, shaped by the body-frame forward, lateral, and reverse scaling factors, and they were mapped to the velocity limits reported in Section 6.1 and to the yaw-control limit specified in the archived configuration files. The archived configuration files also provided detailed network widths, action-scaling constants, and wind-model coefficients, including gust clipping and the linear and quadratic aerodynamic gains.

Training logs, model checkpoints, configuration files, and evaluation summaries were saved separately for each algorithm and seed. The aggregate tables were computed from archived evaluation CSV files, not from manually selected episodes. The processed data, final configurations, and plotting scripts used to reproduce the reported tables and figures are provided in the Supplementary Materials.

7. Results

7.1. Overall Independent Test Results

Table 8 reports the task-completion metrics for the final five-seed comparison. HW-GA-SAC reached the highest success rate, $96\% \pm 3\%$, and the highest average return, 207 ± 16 . Its timeout rate was $3\% \pm 4\%$, below the SAC baseline value of $7\% \pm 10\%$. SAC remained the strongest feedforward baseline, with $92\% \pm 11\%$ success.

Table 8. Task-completion metrics pooled across all five evaluation layouts. Values are mean $\pm$ sample standard deviation over five training-seed summaries and were rounded after aggregation.

Method	Success Rate/%	Collision Rate/%	Timeout Rate/%	Average Return
PPO	27 ± 33	44 ± 40	29 ± 38	-64 ± 138
TD3	75 ± 29	3 ± 6	22 ± 31	149 ± 64
SAC	92 ± 11	1 ± 1	7 ± 10	191 ± 40
SAC-Large	85 ± 27	1 ± 1	13 ± 27	173 ± 72
GRU-SAC	78 ± 44	0.3 ± 0.2	22 ± 44	161 ± 103
LSTM-SAC	74 ± 43	5 ± 9	21 ± 44	151 ± 100
HW-GA-SAC	96 ± 3	1 ± 1	3 ± 4	207 ± 16

Note: Standard deviations were computed over five independent training seeds. For methods with failed training seeds, the mean-plus-standard-deviation range may exceed 100%. This reflects cross-seed instability, not a success rate above 100%.

The same seed-level unit was used for the figures and the city-split analysis below.

Table 9 reports the trajectory-behavior metrics. HW-GA-SAC reached the highest average speed, 13.7 ± 1.0 m/s, and a final target distance of 21 ± 9 m. SAC had the

lowest successful-path stretch, 1.21 ± 0.08 , indicating more direct successful trajectories. The main advantage of HW-GA-SAC lay in task completion, average return, timeout control, and training stability, whereas SAC remained stronger on path directness.

Table 9. Trajectory-behavior metrics across training seeds and city layouts (mean $\pm$ standard deviation). Successful-path stretch is the path-length ratio for successful episodes; lower values indicate more direct successful trajectories.

Method	Average Speed/(m/s)	Final Distance/m	Successful-Path Stretch	Minimum Clearance/m
PPO	9.5 ± 1.7	432 ± 436	1.84 ± 0.43	9 ± 9
TD3	11.8 ± 2.3	25 ± 11	1.39 ± 0.18	7 ± 7
SAC	13.5 ± 1.3	23 ± 6	1.21 ± 0.08	19 ± 9
SAC-Large	13.3 ± 2.1	20 ± 8	1.33 ± 0.24	9 ± 7
GRU-SAC	12.5 ± 2.1	17 ± 9	1.24 ± 0.11	13 ± 7
LSTM-SAC	12.7 ± 1.8	31 ± 21	1.24 ± 0.28	12 ± 5
HW-GA-SAC	13.7 ± 1.0	21 ± 9	1.38 ± 0.19	13 ± 7

7.2. Training-City Replay and Unseen Layouts

Table 10 separates the training-city replay from the four unseen layouts under the same five-seed protocol. The split is useful because the main average combines one training layout with four unseen layouts.

Table 10. Five-seed city-split results for all seven methods. The main protocol contains four unseen layouts and one training-city replay layout.

Method	Split	Episodes	Success/%	Collision/%	Timeout/%
PPO	city 42 train replay	1000	27 ± 32	42 ± 40	31 ± 39
PPO	cities 1–4 unseen	4000	27 ± 33	44 ± 40	29 ± 37
TD3	city 42 train replay	1000	75 ± 27	3 ± 6	23 ± 29
TD3	cities 1–4 unseen	4000	75 ± 29	3 ± 7	22 ± 31
SAC	city 42 train replay	1000	92 ± 11	2 ± 2	6 ± 10
SAC	cities 1–4 unseen	4000	91 ± 11	1 ± 2	7 ± 10
SAC-Large	city 42 train replay	1000	85 ± 28	2 ± 2	14 ± 28
SAC-Large	cities 1–4 unseen	4000	85 ± 27	1 ± 1	13 ± 27
GRU-SAC	city 42 train replay	1000	77 ± 43	0.9 ± 0.8	22 ± 43
GRU-SAC	cities 1–4 unseen	4000	78 ± 44	0.2 ± 0.2	22 ± 44
LSTM-SAC	city 42 train replay	1000	73 ± 42	5 ± 9	22 ± 44
LSTM-SAC	cities 1–4 unseen	4000	74 ± 43	5 ± 9	21 ± 44
HW-GA-SAC	city 42 train replay	1000	96 ± 3	2 ± 1	2 ± 4
HW-GA-SAC	cities 1–4 unseen	4000	96 ± 3	1 ± 1	3 ± 4

After rounding, HW-GA-SAC reached $96\% \pm 3\%$ success on both city 42 and cities 1–4. Its collision rate was $2\% \pm 1\%$ on city 42 and $1\% \pm 1\%$ on cities 1–4; the pooled five-city rate in Table 8 was also $1\% \pm 1\%$. The tables use different aggregation scopes. Table 8 pools all five layouts before rounding, whereas the present table separates city 42 from the four unseen layouts. Because the unseen layouts contributed 4000 of the 5000 evaluation episodes, they carry more weight in the pooled estimate. The displayed collision rates are therefore consistent despite the difference introduced by aggregation and rounding. SAC shows a similar success split: $92\% \pm 11\%$ on city 42 and $91\% \pm 11\%$ on cities 1–4.

7.3. Task-Outcome Bar Charts

Figure 4 shows the success, collision, and timeout rates under the final evaluation protocol. The error bars denote the sample standard deviation over the five training seeds.

HW-GA-SAC achieved the highest mean success rate with a low timeout rate under the final protocol. SAC and TD3 also performed well when the actor received the current-frame safety inputs. The main difference was that HW-GA-SAC combined high task completion with lower seed-level dispersion in this comparison. GRU-SAC and LSTM-SAC showed large cross-seed variance, while PPO had both low success and high failure rates in this long-range urban task.

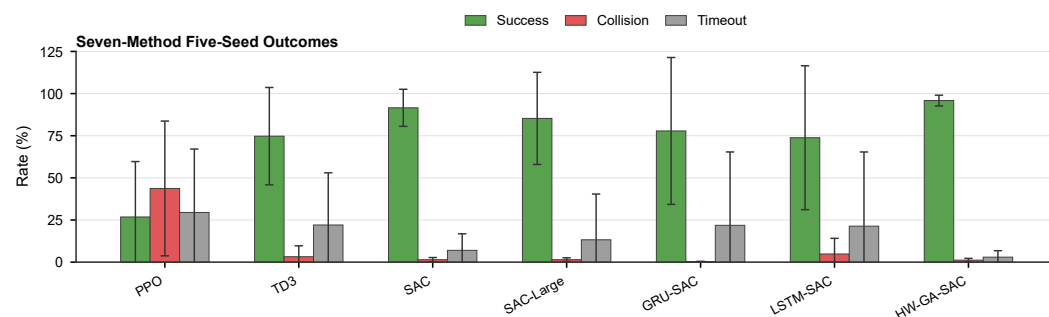


Figure 4. Success, collision, and timeout rates under the final evaluation protocol. The y-axis extends beyond 100% only to show untruncated mean-plus-standard-deviation error bars.

7.4. Stability Across Training Seeds

Figure 5 shows the seed-level outcome distributions for all seven methods. Each point represents one trained model evaluated over the five-city protocol.

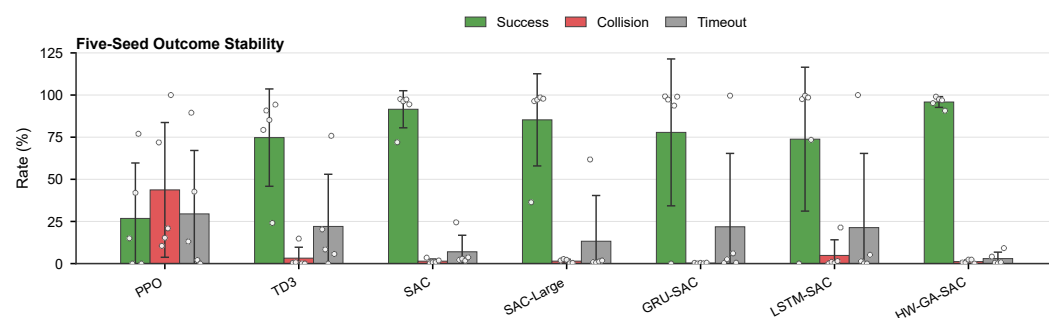


Figure 5. Five-seed success, collision, and timeout outcomes for all seven methods. Hollow points denote individual training seeds, and bars show mean $\pm$ sample standard deviation; the y-axis extends beyond 100% only to show untruncated error bars.

The HW-GA-SAC seeds remained in a high-success region with limited dispersion. SAC and TD3 also achieved strong average success, but SAC showed more cross-seed fluctuation. SAC-Large, GRU-SAC, and LSTM-SAC each contained low-success training seeds. In this task, increasing feedforward capacity or adding recurrent memory alone did not guarantee stable policy learning.

7.5. Five-Seed Training Dynamics

Figure 6 reports the HW-GA-SAC learning dynamics from all five archived TensorBoard runs. Within each seed, completed-episode return and success were aggregated in 50,000-environment-step bins; the solid line and shaded band show the cross-seed mean and sample standard deviation, and the faint lines show individual seeds. The mean return changed from -173 ± 34 in the first bin to 173 ± 39 at 0.475 M steps. The mean training success rose from 15% at 0.325 M steps to 76% at 0.475 M steps and remained predominantly above 80% after 0.75 M steps. The remaining oscillation and the wider band in some bins show why the final evaluation was reported across independently trained models rather than from a selected run.

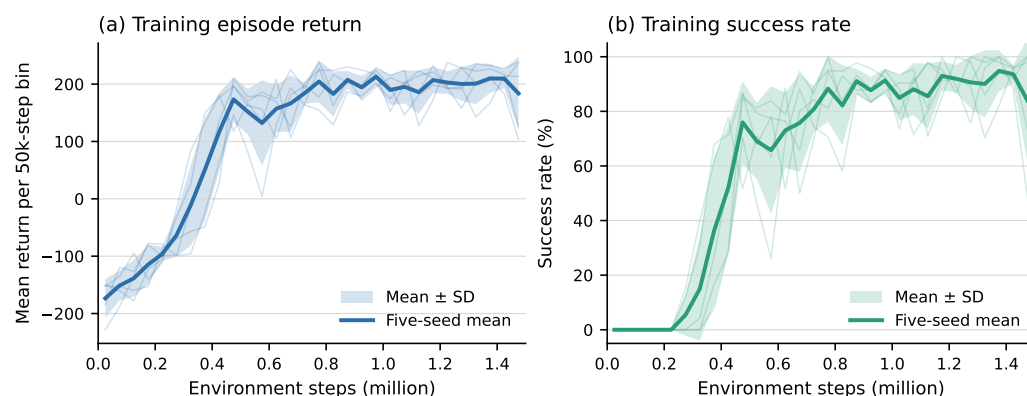


Figure 6. Five-seed HW-GA-SAC learning curves. Completed training episodes were aggregated into 50,000-environment-step bins for each seed. Thick lines and shaded regions denote mean $\pm$ sample standard deviation across five seeds; faint lines show the individual seed trajectories.

7.6. Behavioral Metric Analysis

The success–return relationship is provided in Appendix A. HW-GA-SAC occupies the high-success and high-return region in that plot, while PPO lies in the low-success and low-return region. The main text focuses on path efficiency because it is the clearest trajectory difference in the final comparison.

Figure 7 compares successful-path stretch. This path-length ratio was computed only on successful episodes, so it must be interpreted together with success rate.

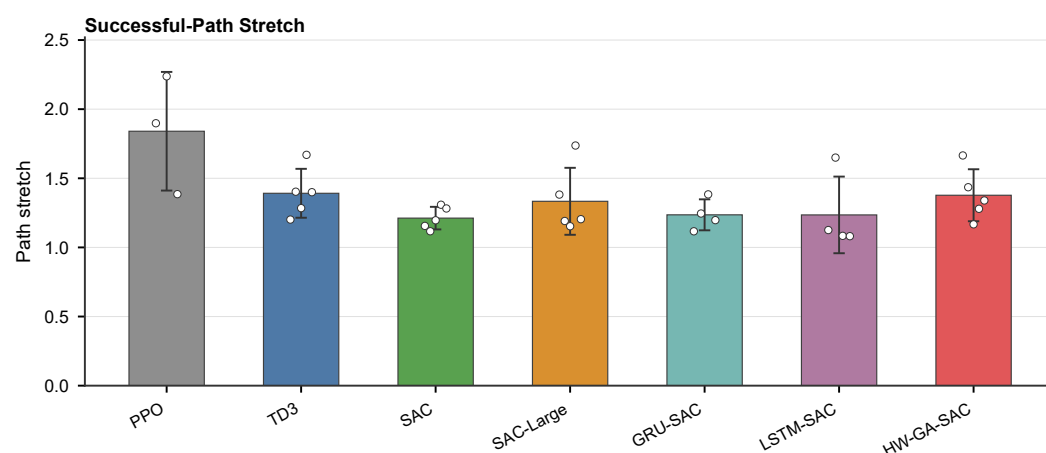


Figure 7. Successful-path stretch comparison. The metric is the path-length ratio for successful episodes; lower values indicate more direct successful trajectories. Hollow points denote individual training-seed values, and bars show mean $\pm$ sample standard deviation.

SAC had the lowest successful-path stretch, 1.21 ± 0.08 , while HW-GA-SAC reached 1.38 ± 0.19 . Table 11 examines this difference on the 5000 tasks shared by the two methods, aligned by training seed, city, and evaluation seed.

On the 4421 tasks completed by both policies, HW-GA-SAC remained less direct and used about 107 additional steps on average. Its mean minimum clearance was 13 m, compared with 21 m for SAC, so the added path length should not be attributed solely to wider geometric obstacle margins. A second component was task coverage: HW-GA-SAC completed 372 tasks on which SAC failed, and 85% of those SAC failures were timeouts; these HW-GA-SAC-only successes had a higher mean stretch of 1.57. The evidence therefore supports a completion–directness trade-off arising from both broader completion and different route selection, rather than a uniform claim of safer detours.

Table 11. Paired SAC–HW-GA-SAC path diagnostic on matched evaluation tasks. Metrics are reported only for the policy that succeeded in the corresponding category.

Paired Outcome	Tasks	SAC Stretch	HW-GA-SAC Stretch	SAC Steps	HW-GA-SAC Steps
Both succeeded	4421	1.21	1.36	807	914
Only HW-GA-SAC succeeded	372	–	1.57	–	1001
Only SAC succeeded	157	1.24	–	846	–
Neither policy succeeded	50	–	–	–	–

Figure 8 visualizes the success-rate split between the training-city replay layout and the unseen layouts.

**Figure 8.** Success-rate split between the training-city replay layout and the four unseen city layouts for the five-seed seven-method evaluation. The y-axis extends beyond 100% only to show untruncated mean-plus-standard-deviation error bars.

7.7. Behavioral Differences Among Policy Structures

With the current-frame safety branch, feedforward SAC already performed strongly, reaching $92\% \pm 11\%$ success. Maximum-entropy exploration and direct access to current-frame safety inputs appear well matched to continuous urban obstacle avoidance. SAC-Large did not reliably improve over SAC. Although a larger feedforward network increases representational capacity, it also increases cross-seed variance in this task. GRU-SAC and LSTM-SAC had lower mean success and much larger standard deviations. Their low collision rates and high timeout rates indicate that recurrent policies often fail by becoming conservative or failing to reach the target, rather than by crashing.

PPO remained weaker and more seed-sensitive than the off-policy baselines even after the stabilization pass in Table 5. Under the same 1.5M-step budget, two of the five stabilized PPO seeds still achieved 0% success, and the seed-level success standard deviation remained 33 percentage points, so the PPO result was treated as a fixed-budget baseline rather than a central comparison target.

Overall, HW-GA-SAC improves completion-oriented metrics and training stability, while SAC remains preferable when path directness is the primary criterion.

7.8. Additional City Generalization over Unseen Layouts

To further test additional procedural-layout generalization, the three core methods TD3, SAC, and HW-GA-SAC were evaluated on ten additional unseen city-generation seeds, 43–52. Each method used three trained model seeds in this protocol. For each trained model and city seed, 100 deterministic evaluation episodes were run under the default wind setting, giving 9000 additional episodes.

Table 12 reports the additional-city results for the three selected core methods. HW-GA-SAC kept the highest mean task-completion performance, with a success rate of $98\% \pm 2\%$, 8 percentage points higher than SAC and 13 percentage points higher than TD3. Its timeout rate was also lower, at $1\% \pm 2\%$, compared with $9\% \pm 12\%$ for SAC and $10\% \pm 9\%$ for TD3. Averaged over the training seeds, HW-GA-SAC had a narrow per-city success range from 97% to 98% across city seeds 43–52. SAC ranged from 86% to 93%, and TD3 ranged from 82% to 89%. This evaluation adds evidence for generalization within the same procedural-layout family; the seven-method comparison remains the broader algorithmic benchmark because PPO, SAC-Large, GRU-SAC, and LSTM-SAC are not re-run in this protocol.

Table 12. Additional city-generalization results on city seeds 43–52 under default wind (mean $\pm$ standard deviation over training seeds).

Method	Success/%	Collision/%	Timeout/%	Average Return	Final Distance/m
TD3	85 ± 3	6 ± 8	10 ± 9	160 ± 25	21 ± 11
SAC	90 ± 14	1 ± 2	9 ± 12	183 ± 50	20 ± 8
HW-GA-SAC	98 ± 2	1 ± 2	1 ± 2	213 ± 26	16 ± 6

7.9. Wind-Disturbance Sensitivity

The wind-sensitivity experiment evaluated TD3, SAC, and HW-GA-SAC on the same ten city seeds under no-wind and strong-wind conditions. The strong-wind condition kept the urban wind model active and multiplied gust intensity by 1.75. Each method, trained seed, wind level, and city seed was evaluated for 100 episodes, giving 18,000 additional episodes.

Table 13 indicates that strong wind mainly affected HW-GA-SAC through timeout increase rather than collision. Relative to no wind, its success rate decreased by 5 percentage points, its average return decreased by 11, and its timeout rate increased by 5 percentage points. The collision-rate change was small relative to its seed-level variability. SAC and TD3 did not show a consistent degradation in this protocol: their mean success rates increased slightly under strong wind, but the changes were small compared with their seed-level variability. HW-GA-SAC kept the highest mean success rate and average return among the three core methods under both wind conditions, while strong wind exposed a timeout-sensitive failure mode in the highest-success policy. The extended unseen-city and wind visualizations are provided in Appendix A.

Table 13. Wind-disturbance sensitivity on city seeds 43–52 (mean $\pm$ standard deviation over training seeds).

Method	Wind Level	Success/%	Collision/%	Timeout/%	Average Return
TD3	No wind	84 ± 4	5 ± 9	11 ± 10	158 ± 26
TD3	Strong	87 ± 6	6 ± 8	8 ± 8	165 ± 33
SAC	No wind	89 ± 15	2 ± 3	9 ± 12	180 ± 55
SAC	Strong	90 ± 14	2 ± 2	8 ± 12	183 ± 52
HW-GA-SAC	No wind	98 ± 2	1 ± 2	0.4 ± 0.5	215 ± 28
HW-GA-SAC	Strong	93 ± 7	1 ± 2	5 ± 8	203 ± 16

7.10. Limited Sensor-Perturbation Screening and Policy Inference Cost

Table 14 reports the sensor-perturbation results. Image and range noise were injected as zero-mean Gaussian perturbations with normalized standard deviations of 0.01 or 0.03. Latency re-used delayed observations at 10, 30, and 50 ms. The strongest combined setting used moderate noise with 50 ms latency. Each method-condition was summarized across five trained model seeds, with 20–50 episodes per seed-condition in the

archived perturbation grid. The protocol is intended as a compact screening check for mild synthetic perturbations.

The perturbation grid shows small non-monotonic changes in success rate. Under moderate noise plus 50 ms latency, HW-GA-SAC reached 96% success, 0.4% collision, and 4% timeout, while SAC reached 91% success and TD3 reached 62%. These increases are best read as sampling variation rather than evidence that noise or latency improves navigation. The tested perturbations were mild relative to the 100 ms policy-control interval, and the finite number of episodes per seed-condition could shift success rates by a few percentage points. Within this screening grid, none of the core policies showed a large mean drop under the tested settings. A compact perturbation visualization is shown later in Figure 9, and the full condition grid is provided in Appendix A.

Table 14. Sensor-noise and latency screening results for the five-seed core methods. Values are success rates in percent. The clean column uses the perturbation-grid sampling protocol without added noise or latency and is not the full 200-episode main-evaluation clean result.

Method	Clean	10 ms	30 ms	50 ms	Moderate Noise	Moderate Noise + 50 ms
TD3	63	68	66	65	63	62
SAC	91	92	92	93	91	91
HW-GA-SAC	94	96	95	96	95	96

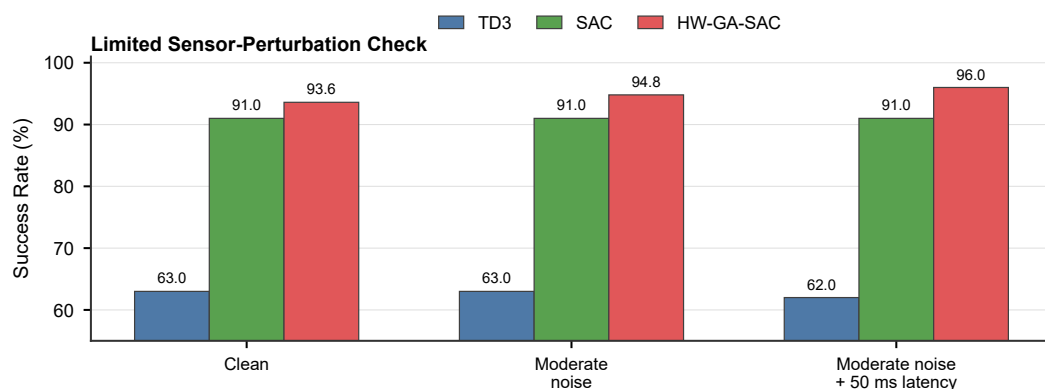


Figure 9. Success-rate comparison for the core methods under clean sensing, moderate Gaussian noise, and moderate Gaussian noise with 50 ms latency.

Table 15 summarizes the policy inference profile. HW-GA-SAC had about 4.1 times more actor–critic parameters than feedforward SAC and used more peak CUDA memory, but its measured mean and p95 inference latency remained close to the SAC baseline on the L40S GPU used in the experiments. These measurements characterize the isolated policy network on the tested GPU. They do not include image preprocessing, range estimation, state estimation, flight-controller communication, or onboard power use.

Table 15. Isolated policy-network inference profiling on the NVIDIA L40S platform. Latency was measured over 10,000 CUDA inference iterations after warm-up.

Method	Parameters	Checkpoint Size/MB	Mean Latency ms	p95 Latency ms	Peak CUDA Memory/MB
SAC	697,994	9.7	1.89	2.04	22
HW-GA-SAC	2,879,370	38.9	1.88	1.93	51

7.11. Actor-Side Safety-Input Ablation

To analyze the current-frame safety branch, a joint ablation removed sparse LiDAR-like range sectors, handcrafted safety cues, and vertical-clearance inputs from the actor

while keeping the critic input complete. The ablation protocol is reported separately from the final five-seed main comparison; its full-model rows were matched controls within the ablation setting. Training steps, environment, reward, city layouts, and evaluation procedure were fixed for each paired comparison. Each method-setting is summarized over three training seeds, five city layouts, and 200 deterministic evaluation episodes per seed–city group. Table 16 reports mean $\pm$ sample standard deviation over seed-level summaries.

Providing the current-frame safety inputs to the actor improved TD3, SAC, and HW-GA-SAC in this ablation. For the proposed method, success increased from $79\% \pm 4\%$ to $97\% \pm 2\%$, and collision rate decreased from $20\% \pm 5\%$ to $1\% \pm 1\%$.

The same pattern did not hold for every method. With the current-frame safety inputs, GRU-SAC and LSTM-SAC collided less often but timed out more often, suggesting more conservative policies. Preliminary PPO behaved differently: both the collision and timeout rates increased. This result points to sensitivity in that particular PPO configuration, not to a conservative shift or a general property of PPO. Appendix B reports the full numerical change and its limitations.

To separate the two safety modalities for the proposed architecture, Table 17 reports a follow-up ablation on HW-GA-SAC. The full model was compared with a variant without handcrafted safety cues and a variant without sparse LiDAR-like sectors.

Table 16. Joint actor-side safety-input ablation under the matched protocol. Values are mean $\pm$ sample standard deviation over three training seeds after seed-level aggregation across five city layouts; each seed–city group contained 200 evaluation episodes. Preliminary PPO diagnostic rows are reported in Appendix B.

Method	Current-Frame Safety Input	Success/%	Collision/%	Timeout/%	Average Return
TD3	No	31 $\pm$ 27	39 $\pm$ 25	30 $\pm$ 51	−71 $\pm$ 78
TD3	Yes	85 $\pm$ 6	5 $\pm$ 8	10 $\pm$ 10	161 $\pm$ 32
SAC	No	42 $\pm$ 42	55 $\pm$ 38	3 $\pm$ 5	−70 $\pm$ 164
SAC	Yes	89 $\pm$ 14	2 $\pm$ 2	10 $\pm$ 13	179 $\pm$ 51
SAC-Large	No	75 $\pm$ 21	24 $\pm$ 22	1 $\pm$ 1	99 $\pm$ 95
SAC-Large	Yes	77 $\pm$ 35	2.2 $\pm$ 0.5	21 $\pm$ 35	141 $\pm$ 81
GRU-SAC	No	71 $\pm$ 15	26 $\pm$ 12	3 $\pm$ 3	92 $\pm$ 53
GRU-SAC	Yes	66 $\pm$ 57	0.3 $\pm$ 0.2	34 $\pm$ 57	133 $\pm$ 135
LSTM-SAC	No	73 $\pm$ 6	26 $\pm$ 7	1 $\pm$ 1	18 $\pm$ 155
LSTM-SAC	Yes	66 $\pm$ 57	0.4 $\pm$ 0.5	34 $\pm$ 57	147 $\pm$ 134
HW-GA-SAC	No	79 $\pm$ 4	20 $\pm$ 5	0.7 $\pm$ 0.7	125 $\pm$ 22
HW-GA-SAC	Yes	97 $\pm$ 2	1 $\pm$ 1	2 $\pm$ 2	211 $\pm$ 19

Table 17. Separated safety-input ablation for HW-GA-SAC. Values are mean $\pm$ sample standard deviation over three training seeds after each seed was averaged over five city layouts; each seed–city group contained 200 episodes.

Variant	Seeds	Success/%	Collision/%	Timeout/%	Average Return	Final Distance/m
Full	3	98 $\pm$ 2	0.8 $\pm$ 1.0	2 $\pm$ 3	213 $\pm$ 19	20 $\pm$ 11
Without handcrafted safety	3	65 $\pm$ 28	20 $\pm$ 24	14 $\pm$ 22	80 $\pm$ 131	75 $\pm$ 71
Without LiDAR sectors	3	46 $\pm$ 28	54 $\pm$ 29	0.3 $\pm$ 0.3	22 $\pm$ 112	292 $\pm$ 126

The separated ablation suggests that both safety modalities matter, with sparse LiDAR-like range sectors appearing to carry the larger collision-avoidance burden in this implementation. Removing handcrafted safety cues reduced success by 32 percentage points and raised collision to 20%. Removing the sparse LiDAR-like range sectors reduced success by 51 percentage points and raised collision to 54%. The handcrafted cues appeared to refine

safety assessment, while the range sectors provided the primary local obstacle geometry. Figure 10 visualizes the success-rate comparison for this separated ablation.

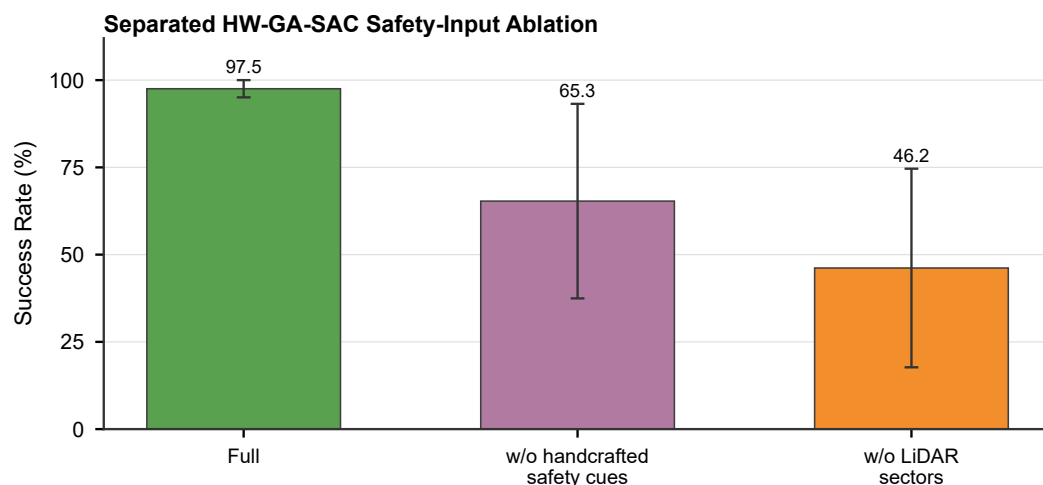


Figure 10. Success-rate comparison for the separated HW-GA-SAC actor-side safety-input ablation. The camera-removal ablation is reported separately in Table 18.

7.12. Visual Camera Input Ablation

A second ablation removed the front-view grayscale image branch from HW-GA-SAC while preserving the low-dimensional state, safety-related features, and action history. This protocol was separate from the final five-seed main comparison and used a matched full-model control for the camera-removal test. The training seeds, city layouts, evaluation episodes, and statistics were kept unchanged. Table 18 reports the effect of removing the camera branch from the matched HW-GA-SAC control.

Table 18. Visual camera input ablation for HW-GA-SAC under the matched ablation protocol. Values are mean $\pm$ sample standard deviation over three training seeds after seed-level aggregation across five city layouts; each seed–city group contained 200 evaluation episodes. Successful-path stretch was computed only over successful episodes.

Camera Input	Success/%	Collision/%	Timeout/%	Average Return	Final Distance/m	Successful-Path Stretch
Yes	97 $\pm$ 2	1 $\pm$ 1	2 $\pm$ 2	211 $\pm$ 19	22 $\pm$ 11	1.41 $\pm$ 0.25
No	85 $\pm$ 22	15 $\pm$ 22	0.1 $\pm$ 0.2	144 $\pm$ 93	98 $\pm$ 125	1.04 $\pm$ 0.19

Removing the camera input reduced success from 97% $\pm$ 2% to 85% $\pm$ 22%, increased collision from 1% $\pm$ 1% to 15% $\pm$ 22%, and lowered average return from 211 $\pm$ 19 to 144 $\pm$ 93. Final target distance also increased from 22 $\pm$ 11 m to 98 $\pm$ 125 m. Visual input contributed useful information about urban structure and traversable space. The lower successful-path stretch without the camera is not a better trajectory-quality result, because it was computed only on a smaller and less stable set of successful episodes.

8. Discussion and Limitations

HW-GA-SAC improved success rate, average return, timeout control, and cross-seed stability in the tested simulator. Similar performance on city 42 and cities 1–4 indicates that the main average was not driven by the training-city replay layout. The five-seed learning curves and the separated input ablations further connect the final evaluation to the training behavior and representation design. Policy quality nevertheless depends on mission priorities: SAC was more direct on successful episodes, whereas HW-GA-SAC

completed more tasks. The paired diagnostic shows that this difference reflects both broader completion coverage and different route selection; it is not explained by a uniformly larger minimum clearance.

The scope of the evidence was simulation-level. The training used one procedural city, the evaluation layouts used the same city generator, and the wind model applied structured translational forces rather than rotor-resolved aerodynamic moments. The sensing tests used synthetic perturbations, and the latency profile isolated policy-network execution on an NVIDIA L40S GPU. These choices provided controlled comparisons but did not establish reliability for real urban flight. The observed collision and clearance outcomes were empirical task metrics, not formal safety guarantees. A physical system would additionally require calibrated perception and state estimation, actuator and communication characterization, energy measurements, and an independent runtime safety mechanism such as geofencing, emergency braking, a control-barrier-function filter, or a recovery policy [26,28,40].

The staged next step will be controlled low-speed testing on a small UAV with on-board camera and range sensing, followed by broader outdoor urban-like trials. Policy compression [41], hybrid route planning [42,43], and higher-fidelity simulation [44] could be evaluated within that progression.

9. Conclusions

This study presented HW-GA-SAC, a multimodal history-window SAC framework for urban low-altitude UAV navigation. The framework fuses a front-view grayscale image, target-relative state, UAV motion state, sparse LiDAR-like range sectors, handcrafted safety cues, and previous actions. A GTrXL-inspired gated-attention encoder models an eight-step navigation history, while the current-frame safety branch allows near-obstacle information to influence both action selection and value estimation.

Under a unified MuJoCo evaluation protocol, seven methods were compared with five training seeds and five evaluation layouts, including cities 1–4 and the training city, city 42. HW-GA-SAC produced the strongest task-completion profile, and the city split showed similar performance on the training-city replay and unseen layouts. In additional simulation checks, the method achieved $98\% \pm 2\%$ success on city seeds 43–52 under default wind and retained the highest mean success among the three core methods under no-wind and strong-wind settings. Five-seed learning curves, sensor perturbations, and inference profiling provide complementary diagnostics.

The ablation studies showed that providing the current-frame safety inputs to the actor is important for TD3, SAC, and HW-GA-SAC, with sparse LiDAR-like range sectors carrying more of the collision-avoidance burden than the handcrafted safety cues in the separated ablation. Front-view visual input also contributes useful urban structural information. Taken together, the results support the conclusion that the current-frame safety branch, visual input, and fixed-window gated attention are associated with stronger task completion in this MuJoCo setting, while feedforward SAC remains more path-direct. These empirical outcomes do not establish formal safety guarantees.

Future work will extend this simulation study through staged physical UAV experiments, beginning with low-speed target navigation under independent safety supervision and progressing to onboard perception, communication, actuator, and energy evaluation.

Supplementary Materials: The following supporting information is available with this article: a reproducibility package containing the processed experimental data, final configuration files, and plotting scripts used to reproduce the reported tables and figures.

Author Contributions: Conceptualization, methodology, software, validation, formal analysis, investigation, writing—original draft preparation: X.Y.; supervision, writing—review and editing: W.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Processed data and plotting scripts supporting the reported tables and figures are available in the accompanying reproducibility package at https://github.com/Youx1/HW-GA-SAC_reproducibility_package (accessed on 5 August 2026). The package includes final configuration files, per-episode and aggregated CSV/JSON results, representative trajectory data, city-layout visualizations, sensor-perturbation summaries, inference-profiling summaries, processed five-seed learning-curve data, and the paired SAC–HW-GA-SAC path diagnostic. Raw training logs, replay buffers, model checkpoints, and the full MuJoCo training/evaluation pipeline are not included because of storage and maintenance constraints, but they can be made available from the corresponding author upon reasonable request.

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

Appendix A. Additional Figures

Figures A1, A2, A3, A4, and A5 provide the additional success–return, city-layout, sensor-perturbation, unseen-city, and wind-robustness visualizations, respectively.

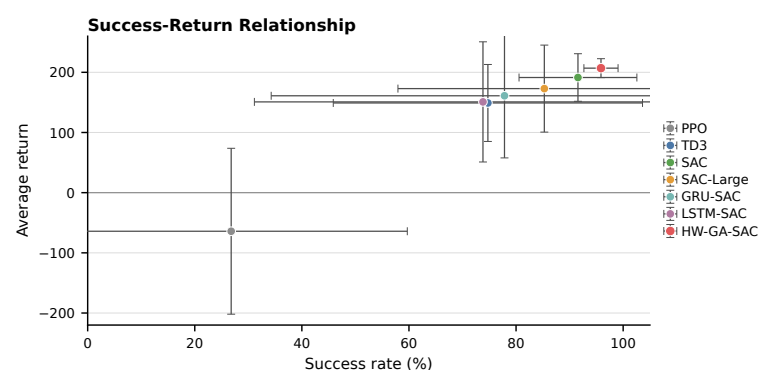


Figure A1. Seven-method relationship between success rate and average return. Methods near the upper-right region combine higher task completion and higher reward.

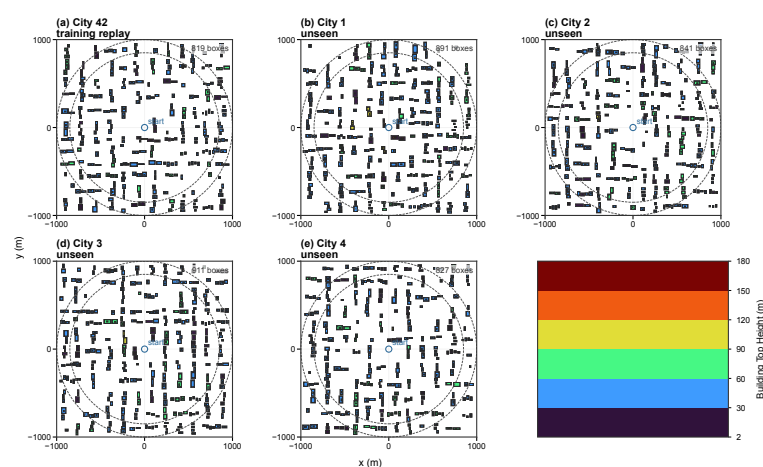


Figure A2. Top-down layout visualization for city 42 and cities 1–4. Discrete height bins are used to make building-height differences visible.

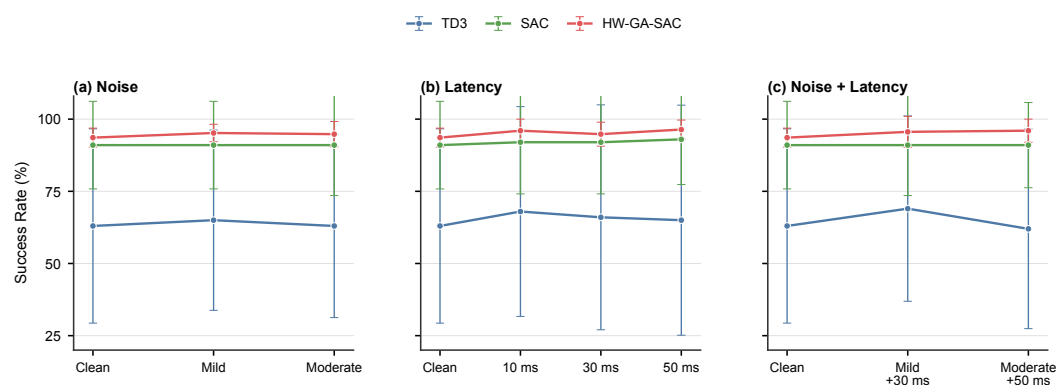
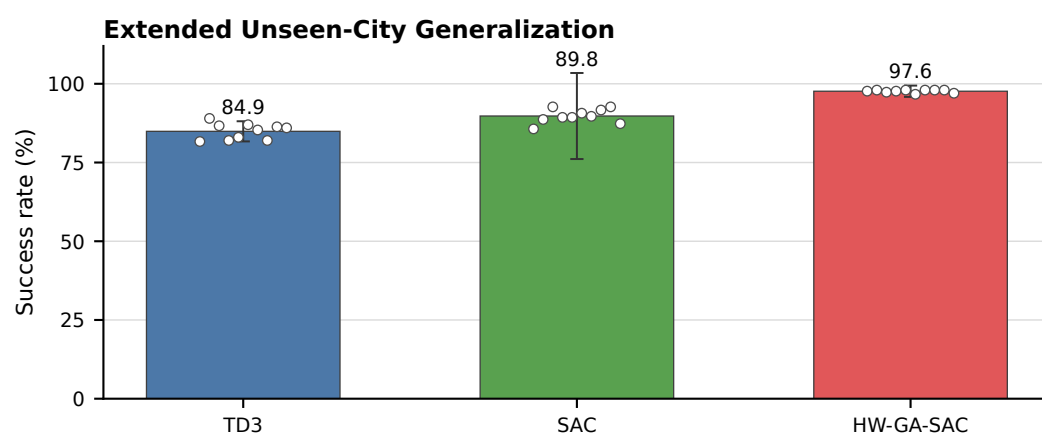


Figure A3. Full sensor-noise, latency, and combined perturbation grid for the core methods.



Hollow points: means for city seeds 43 to 52

Figure A4. Extended unseen-layout generalization over city seeds 43–52 under the default wind setting. Hollow points denote the mean success rates for individual city seeds 43–52, and bars show the mean $\pm$ sample standard deviation across training seeds.

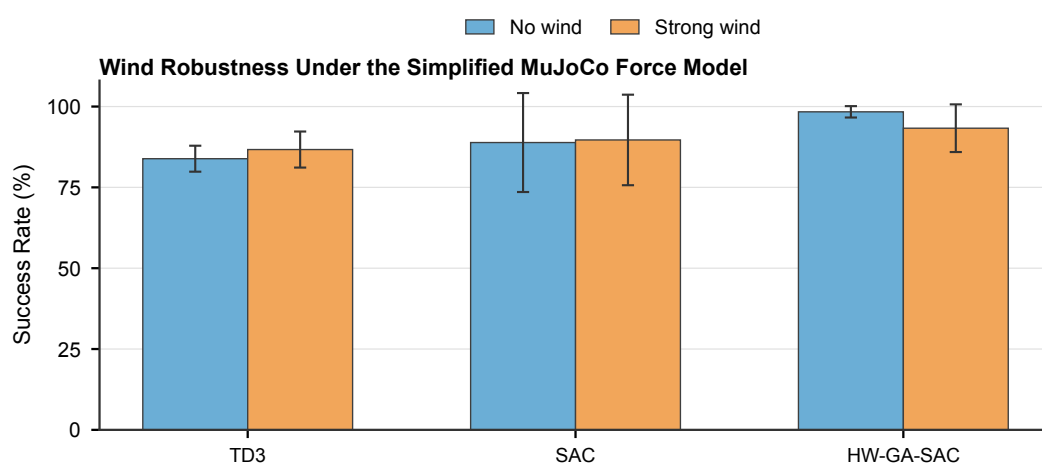


Figure A5. No-wind versus strong-wind success-rate comparison under the simplified force-based wind model.

Appendix B. Additional Diagnostic Tables

Table A1 reports the preliminary PPO actor-input diagnostic referenced in the main text.

Table A1. Preliminary PPO actor-input diagnostic under the matched three-seed ablation protocol. The critic retains the complete observation in both variants. Values are mean $\pm$ sample standard deviation after each training seed is aggregated across five city layouts. These rows use the preliminary PPO setting and were not part of the final stabilized five-seed PPO comparison.

Current-Frame Safety Input	Success/%	Collision/%	Timeout/%	Average Return
No	57 $\pm$ 33	22 $\pm$ 10	21 $\pm$ 31	59 $\pm$ 84
Yes	13 $\pm$ 19	34 $\pm$ 27	53 $\pm$ 11	−63 $\pm$ 109

With the current-frame safety inputs, the preliminary PPO configuration dropped from 57% $\pm$ 33% to 13% $\pm$ 19% mean success. Its timeout rate rose from 21% $\pm$ 31% to 53% $\pm$ 11%, its collision rate from 22% $\pm$ 10% to 34% $\pm$ 27%, and its average return fell from 59 $\pm$ 84 to −63 $\pm$ 109. Because collisions and timeouts both increased, a conservative shift alone cannot explain the result. Under the fixed 1.5 M-step budget, this PPO configuration did not learn to use the expanded actor observation reliably. Possible contributors include the changed input dimension and feature distribution, limited re-use of on-policy data, and the lack of input-specific PPO re-tuning; the diagnostic could not separate their effects. It also used only three training seeds, showed substantial cross-seed variability, and predated the stabilized PPO setting. We therefore treat the result as configuration-specific optimization sensitivity, not evidence that current-frame safety inputs generally impair PPO.

References

1. Nguyen, H.; Quyen, T.; Nguyen, C.; Le, A.; Tran, H.; Nguyen, M. Control algorithms for UAVs: A comprehensive survey. *EAI Endorsed Trans. Ind. Netw. Intell. Syst.* **2020**, *7*, e5. <https://doi.org/10.4108/eai.18-5-2020.164586>.
2. Ghambari, S.; Golabi, M.; Jourdan, L.; Lepagnot, J.; Idoumghar, L. UAV path planning techniques: A survey. *RAIRO Oper. Res.* **2024**, *58*, 2951–2989. <https://doi.org/10.1051/ro/2024073>.
3. Coutinho, W.P.; Battarra, M.; Fliege, J. The unmanned aerial vehicle routing and trajectory optimisation problem: A taxonomic review. *Comput. Ind. Eng.* **2018**, *120*, 116–128. <https://doi.org/10.1016/j.cie.2018.04.037>.
4. Konovalenko, I.; Kuznetsova, E.; Miller, A.; Miller, B.; Popov, A.; Shepelev, D.; Stepanyan, K. New approaches to the integration of navigation systems for autonomous unmanned vehicles. *Sensors* **2018**, *18*, 3010. <https://doi.org/10.3390/s18093010>.
5. Haarnoja, T.; Zhou, A.; Abbeel, P.; Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018*; PMLR: Stockholm, Sweden, 2018; Volume 80, pp. 1861–1870.
6. Haarnoja, T.; Zhou, A.; Hartikainen, K.; Tucker, G.; Ha, S.; Tan, J.; Kumar, V.; Zhu, H.; Gupta, A.; Abbeel, P.; et al. Soft actor-critic algorithms and applications. *arXiv* **2018**, arXiv:1812.05905. <https://doi.org/10.48550/arXiv.1812.05905>.
7. Dai, Z.; Yang, Z.; Yang, Y.; Carbonell, J.; Le, Q.; Salakhutdinov, R. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019*; Association for Computational Linguistics: Stroudsburg, PA, USA, 2019; pp. 2978–2988. <https://doi.org/10.18653/v1/P19-1285>.
8. Parisotto, E.; Song, H.F.; Rae, J.W.; Pascanu, R.; Gulcehre, C.; Jayakumar, S.M.; Jaderberg, M.; Kaufman, R.L.; Clark, A.; Noury, S.; et al. Stabilizing transformers for reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning, Online, 13–18 July 2020*; PMLR: Online, 2020; Volume 119, pp. 7487–7498.
9. Dijkstra, E.W. A note on two problems in connexion with graphs. *Numer. Math.* **1959**, *1*, 269–271. <https://doi.org/10.1007/BF01386390>.
10. Hart, P.E.; Nilsson, N.J.; Raphael, B. A formal basis for the heuristic determination of minimum cost paths. *IEEE Trans. Syst. Sci. Cybern.* **1968**, *4*, 100–107. <https://doi.org/10.1109/TSSC.1968.300136>.
11. Kavvaki, L.E.; Svestka, P.; Latombe, J.C.; Overmars, M.H. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. Robot. Autom.* **1996**, *12*, 566–580. <https://doi.org/10.1109/70.508439>.
12. LaValle, S.M. *Rapidly-Exploring Random Trees: A New Tool for Path Planning*; Technical Report TR 98-11; Iowa State University: Ames, IA, USA, 1998.
13. Khatib, O. Real-time obstacle avoidance for manipulators and mobile robots. *Int. J. Robot. Res.* **1986**, *5*, 90–98. <https://doi.org/10.1177/027836498600500106>.

14. Lillicrap, T.P.; Hunt, J.J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; Wierstra, D. Continuous control with deep reinforcement learning. In Proceedings of the 4th International Conference on Learning Representations, San Juan, Puerto Rico, 2–4 May 2016.
15. Fujimoto, S.; van Hoof, H.; Meger, D. Addressing function approximation error in actor-critic methods. In *Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018*; PMLR: Stockholm, Sweden, 2018; Volume 80, pp. 1587–1596.
16. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O. Proximal policy optimization algorithms. *arXiv* **2017**, arXiv:1707.06347. <https://doi.org/10.48550/arXiv.1707.06347>.
17. de Jesus, J.C.; Kich, V.A.; Kolling, A.H.; Grando, R.B.; Cuadros, M.A.S.L.; Gamarra, D.F.T. Soft Actor-Critic for navigation of mobile robots. *J. Intell. Robot. Syst.* **2021**, *102*, 31. <https://doi.org/10.1007/s10846-021-01367-5>.
18. Grando, R.B.; de Jesus, J.C.; Kich, V.A.; Kolling, A.H.; Bortoluzzi, N.P.; Pinheiro, P.M.; Neto, A.A.; Drews, P.L.J. Deep reinforcement learning for mapless navigation of a hybrid aerial underwater vehicle with medium transition. In *Proceedings of the 2021 IEEE International Conference on Robotics and Automation, Xi'an, China, 30 May–5 June 2021*; IEEE: Piscataway, NJ, USA, 2021; pp. 1088–1094. <https://doi.org/10.1109/ICRA48506.2021.9561188>.
19. Ali, M.A.; Maqsood, A.; Athar, U.; Khanzada, H.R. Comparative evaluation of reinforcement learning algorithms for multi-agent unmanned aerial vehicle path planning in 2D and 3D environments. *Drones* **2025**, *9*, 438. <https://doi.org/10.3390/drones9060438>.
20. Hwang, H.J.; Jang, J.; Choi, J.; Bae, J.H.; Kim, S.H.; Kim, C.O. Stepwise soft actor-critic for UAV autonomous flight control. *Drones* **2023**, *7*, 549. <https://doi.org/10.3390/drones7090549>.
21. Fu, C.; Xu, X.; Zhang, Y.; Lyu, Y.; Xia, Y.; Zhou, Z.; Wu, W. Memory-enhanced deep reinforcement learning for UAV navigation in 3D environment. *Neural Comput. Appl.* **2022**, *34*, 14599–14607. <https://doi.org/10.1007/s00521-022-07244-y>.
22. Tai, L.; Paolo, G.; Liu, M. Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation. In *Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems, Vancouver, BC, Canada, 24–28 September 2017*; IEEE: Piscataway, NJ, USA, 2017; pp. 31–36. <https://doi.org/10.1109/IROS.2017.8202134>.
23. Xue, H.; Hein, B.; Bakr, M.; Schildbach, G.; Abel, D.; Rueckert, E. Using deep reinforcement learning with automatic curriculum learning for mapless navigation in intralogistics. *arXiv* **2022**, arXiv:2202.11512. <https://doi.org/10.48550/arXiv.2202.11512>.
24. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2017; Volume 30, pp. 5998–6008.
25. Huang, J.; Cui, Y.; Xi, G.; Bai, S.; Li, B.; Wang, G.; Neretin, E. GTrXL-SAC-based path planning and obstacle-aware control decision-making for UAV autonomous control. *Drones* **2025**, *9*, 275. <https://doi.org/10.3390/drones9040275>.
26. Achiam, J.; Held, D.; Tamar, A.; Abbeel, P. Constrained policy optimization. In *Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017*; PMLR: Sydney, Australia, 2017; Volume 70, pp. 22–31.
27. Dalal, G.; Dvijotham, K.; Vecerik, M.; Hester, T.; Paduraru, C.; Tassa, Y. Safe exploration in continuous action spaces. *arXiv* **2018**, arXiv:1801.08757. <https://doi.org/10.48550/arXiv.1801.08757>.
28. Thananjeyan, B.; Balakrishna, A.; Nair, S.; Luo, M.; Srinivasan, K.; Hwang, M.; Gonzalez, J.E.; Ibarz, J.; Finn, C.; Goldberg, K. Recovery RL: Safe reinforcement learning with learned recovery zones. *arXiv* **2020**, arXiv:2010.15920. <https://doi.org/10.48550/arXiv.2010.15920>.
29. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*, 2nd ed.; MIT Press: Cambridge, MA, USA, 2018.
30. Kaelbling, L.P.; Littman, M.L.; Cassandra, A.R. Planning and acting in partially observable stochastic domains. *Artif. Intell.* **1998**, *101*, 99–134. [https://doi.org/10.1016/S0004-3702\(98\)00023-X](https://doi.org/10.1016/S0004-3702(98)00023-X).
31. Todorov, E.; Erez, T.; Tassa, Y. MuJoCo: A physics engine for model-based control. In *Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, Vilamoura, Portugal, 7–12 October 2012*; IEEE: Piscataway, NJ, USA, 2012; pp. 5026–5033. <https://doi.org/10.1109/IROS.2012.6386109>.
32. Gillespie, D.T. Exact numerical simulation of the Ornstein–Uhlenbeck process and its integral. *Phys. Rev. E* **1996**, *54*, 2084–2091. <https://doi.org/10.1103/PhysRevE.54.2084>.
33. Cole, K.; Wickenheiser, A.M. Spatio-temporal wind modeling for UAV simulations. *arXiv* **2019**, arXiv:1905.09954. <https://doi.org/10.48550/arXiv.1905.09954>.
34. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjeland, A.K.; Ostrovski, G.; et al. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529–533. <https://doi.org/10.1038/nature14236>.
35. Ha, D.; Schmidhuber, J. World models. *arXiv* **2018**, arXiv:1803.10122. <https://doi.org/10.48550/arXiv.1803.10122>.
36. Li, Q.; Gama, F.; Ribeiro, A.; Prorok, A. Graph neural networks for decentralized multi-robot path planning. In *Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems, Las Vegas, NV, USA, 25–29 October 2020*; IEEE: Piscataway, NJ, USA, 2020; pp. 11785–11792. <https://doi.org/10.1109/IROS45743.2020.9341668>.

37. Ng, A.Y.; Harada, D.; Russell, S. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the 16th International Conference on Machine Learning, Bled, Slovenia, 27–30 June 1999*; Morgan Kaufmann: San Francisco, CA, USA, 1999; pp. 278–287.
38. Devidze, R.; Kamalaruban, P.; Singla, A. Exploration-guided reward shaping for reinforcement learning under sparse rewards. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2022; Volume 35, pp. 5829–5842. <https://doi.org/10.52202/068431-0422>.
39. Bengio, Y.; Louradour, J.; Collobert, R.; Weston, J. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning, Montreal, QC, Canada, 14–18 June 2009*; ACM: New York, NY, USA, 2009; pp. 41–48. <https://doi.org/10.1145/1553374.1553380>.
40. Ames, A.D.; Xu, X.; Grizzle, J.W.; Tabuada, P. Control barrier function based quadratic programs for safety critical systems. *IEEE Trans. Autom. Control* **2017**, *62*, 3861–3876. <https://doi.org/10.1109/TAC.2016.2638961>.
41. Rusu, A.A.; Colmenarejo, S.G.; Gulcehre, C.; Desjardins, G.; Kirkpatrick, J.; Pascanu, R.; Mnih, V.; Kavukcuoglu, K.; Hadsell, R. Policy distillation. In *Proceedings of the 4th International Conference on Learning Representations, San Juan, PR, USA, 2–4 May 2016*.
42. Liu, J.; Xie, P.; Liu, J.; Tu, X. Task offloading and trajectory optimization in UAV networks: A deep reinforcement learning method based on SAC and A-Star. *Comput. Model. Eng. Sci.* **2024**, *141*, 1243–1273. <https://doi.org/10.32604/cmes.2024.054002>.
43. Zhou, Y.; Shu, J.; Zheng, X.; Hao, H.; Song, H. Real-time route planning of unmanned aerial vehicles based on improved soft actor-critic algorithm. *Front. Neurorobot.* **2022**, *16*, 1025817. <https://doi.org/10.3389/fnbot.2022.1025817>.
44. Shah, S.; Dey, D.; Lovett, C.; Kapoor, A. AirSim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and Service Robotics*; Hutter, M., Siegwart, R., Eds.; Springer: Cham, Switzerland, 2018; pp. 621–635. https://doi.org/10.1007/978-3-319-67361-5_40.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.