machines

MDPI

Article

Simulated Corrective Subgoal Supervision for Hierarchical Reinforcement Learning in Long-Horizon AntMaze Navigation

Lidong Sun [1,†], Ye Wang [2,†], Zheheng Fan [3] and Fuchun Sun [3,*]

[1] Naval University of Engineering, Wuhan 430033, China; d22181003@nue.edu.cn
[2] Department of Mechanical Engineering, Tsinghua University, Beijing 100084, China; wang-ye22@mails.tsinghua.edu.cn
[3] Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China; fzh18@tsinghua.org.cn
* Correspondence: fcsun@mail.tsinghua.edu.cn
† These authors contributed equally to this work.

Abstract

Long-horizon navigation requires a high-level policy to select locally reachable subgoals, yet a scalar task reward provides little information about how an unsuitable proposal should be changed. We introduce Simulated Corrective Subgoal Supervision for Hierarchical Reinforcement Learning (SCS-HRL), a two-level method in which a topology- and clearance-aware programmatic supervisor evaluates each proposed subgoal and returns both a scalar score and a continuous target in the same subgoal space. The score trains the high-level critic, and the target enters a masked regression term for the high-level actor. Primitive actions are always conditioned on the actor's subgoal; the supervisor is inactive during learned-policy evaluation. In AntMaze, using 6000 training episodes, five seeds, and 100 deterministic evaluation episodes per seed, SCS-HRL attained an $88.4 \pm 7.8\%$ final success rate (mean $\pm$ sample standard deviation; 95% Student-t confidence interval [78.7%, 98.1%]). The matched scalar-only condition and HIRO attained 0% rates. Applying the same route rule directly to the SCS-HRL low-level controllers yielded $82.2 \pm 9.9\%$ success; the paired difference favored the learned high-level policy by 6.2 percentage points (95% confidence interval [2.1, 10.3], $p = 0.013$). Across three matched seeds, nonzero corrective weights of 0.5, 1.0, and 2.0 remained stable, whereas 0.25 was seed-sensitive. Term-level ablations further show that the continuous target, rather than the exact scalar-shaping formula, was the principal additional signal. Separate fixed-policy tests obtained 0% success rates on two unseen maze layouts. These results indicate that continuous subgoal targets can encode task-specific route information in the source maze, while cross-layout transfer remains unresolved.

Keywords: hierarchical reinforcement learning; corrective supervision; subgoal learning; AntMaze; long-horizon navigation

Check for updates

Academic Editor: Adeel Mehmood

Received: 6 July 2026
Revised: 3 August 2026
Accepted: 10 August 2026
Published: 12 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

Reliable autonomous navigation requires an agent to convert a distant task objective into a sequence of locally achievable decisions while maintaining stable low-level control. This requirement becomes particularly demanding in long-horizon continuous-control tasks, where successful behavior depends on temporal abstraction, efficient exploration, and accurate coordination between planning and execution [1]. Deep reinforcement learning (DRL) has demonstrated strong capabilities in locomotion, navigation, and robotic

manipulation [2–4]. Nevertheless, its performance often deteriorates when task rewards are sparse, successful trajectories require many sequential decisions, and early exploratory behavior provides little information about eventual task completion. These limitations remain important even as modern navigation systems increasingly incorporate safety-aware planning, LiDAR-based scene understanding, and adaptation to dynamic crowds [5,6].

Two related difficulties are especially relevant to long-horizon robotic learning. First, the sparse reward problem makes successful experience rare and weakens temporal credit assignment, resulting in inefficient exploration and slow policy improvement [7]. Second, manually designing a dense reward that correctly represents every stage of a complex task is difficult. Reward functions intended to encourage obstacle avoidance, waypoint selection, and goal reaching can introduce unintended shortcuts or correlations that do not reflect task feasibility [8]. This problem is particularly evident in maze navigation; the Euclidean distance provides a simple measure of progress, yet a valid route around an obstacle may initially require the agent to move farther from the final goal. Consequently, a geometrically dense reward can still be topologically misleading.

Hierarchical reinforcement learning (HRL) provides a natural framework for addressing these challenges by decomposing a long-horizon task into temporally extended subproblems [9,10]. A high-level policy proposes intermediate subgoals, whereas a low-level policy learns the primitive controls required to reach them. This decomposition reduces the effective planning horizon and enables the low-level controller to reuse local behaviors across different parts of a task. However, the effectiveness of HRL depends critically on the quality of the generated subgoals. A high-level policy may repeatedly propose targets that are uninformative, unreachable, or incompatible with the topology of the environment. Recent approaches therefore emphasize reachability-aware replay, strict subgoal execution, primitive-informed regularization, and adaptive optimization of human-specified subgoal spaces [11–13]. These studies demonstrate that improving subgoal feasibility is not merely an auxiliary consideration but a central requirement for reliable hierarchical learning.

Human or synthetic feedback offers a complementary source of information when the task reward alone does not sufficiently characterize desirable behavior [14,15]. Preference-based and evaluative methods can indicate whether one behavior is preferable to another, but collecting feedback at every primitive step is expensive and places a substantial burden on the teacher. Planning-level feedback is potentially more scalable because it operates at the slower subgoal timescale; a supervisor can evaluate a proposed waypoint and, when necessary, indicate how it should be adjusted. Recent work has demonstrated the value of adaptive feedback acquisition in robot learning and the utility of synthetic labelers as reproducible development instruments [16,17]. Existing feedback-guided HRL methods, including HhP, MENTOR, PIPER, and DIPPER, primarily use preferences, scalar assessments, learned rewards, or preference optimization objectives [13,18–20]. Continuous targets that directly specify a locally improved subgoal, while remaining separate from the executed command, have received comparatively less attention.

To address this gap, we propose **Simulated Corrective Subgoal Supervision for Hierarchical Reinforcement Learning (SCS-HRL)**. At each high-level decision step, the learned actor proposes a relative subgoal offset. A structured supervisor returns a scalar assessment, a corrective target offset in the same continuous subgoal space, and a mask indicating whether the target should contribute to actor learning. The supervisory reward trains the high-level critic, whereas an active target provides a masked regression regularizer for the high-level actor. The low-level controller always receives the actor-generated subgoal; the corrective target affects only the high-level optimization objective. SCS-HRL thereby preserves policy-driven execution and distinguishes corrective learning from direct rule-based control.

https://doi.org/10.3390/machines14080928

Figure 1 shows the information flow among the high-level actor, supervisor, low-level controller, and environment. The supervisor participates only in high-level learning, while primitive actions remain determined by the hierarchical policy. Table 1 positions SCS-HRL relative to representative feedback-guided and feasibility-aware HRL methods. Unlike approaches based only on scalar evaluation or pairwise preference, SCS-HRL supplies an explicit continuous corrective target without requiring a learned reward model or converting the target into an execution-time command.

The main contributions of this work are summarized as follows:

- **We introduce a dual-channel supervisory interface for hierarchical reinforcement learning.** It provides two complementary planning-level signals: a scalar evaluation of subgoal quality and a continuous target indicating how an infeasible or inefficient proposal should be corrected.
- **We develop a masked corrective objective for feasible subgoal learning.** The objective regularizes the off-policy high-level actor when a continuous target is available. Because the target shapes optimization without replacing the actor-generated subgoal, corrective learning remains separate from direct external control.
- **We conduct a controlled, mechanism-level evaluation of scalar and continuous supervisory signals.** The experiments combine a five-seed fixed-budget comparison, a same-low-level direct-rule counterfactual, corrective-coefficient sensitivity, term-level reward ablations, observation and control perturbations, trajectory measurements, and zero-shot tests on two unseen layouts.

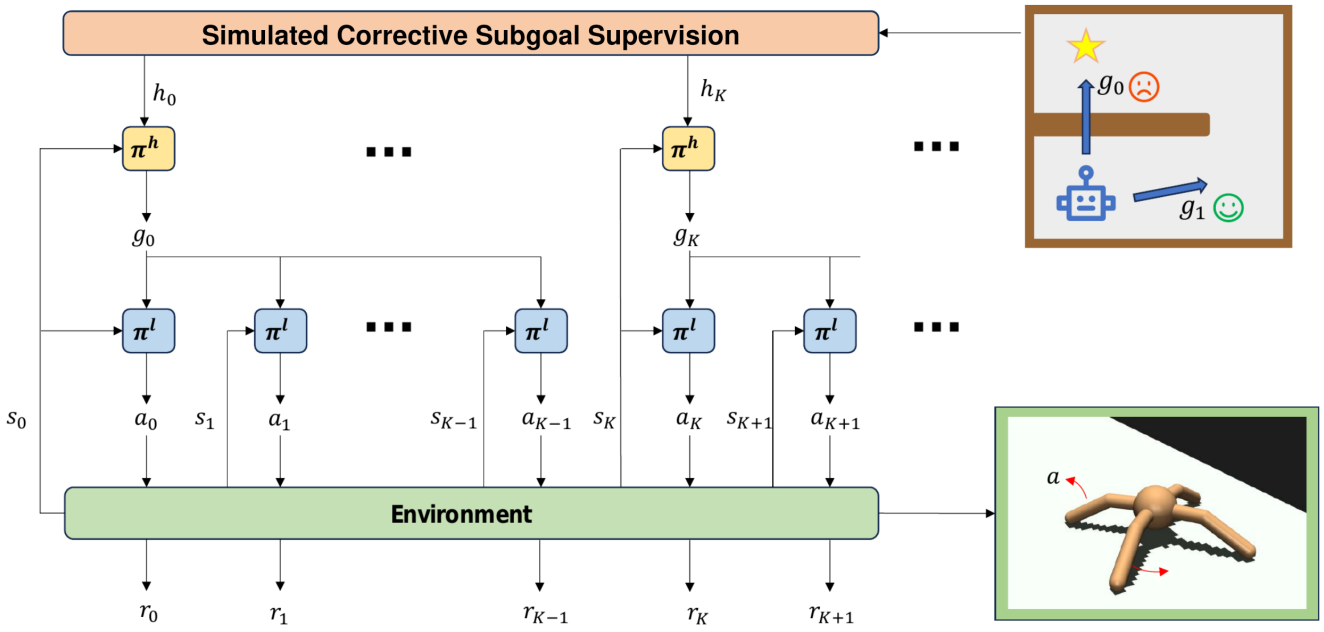


Figure 1. Overview of the SCS-HRL architecture. Every K primitive steps, the learned high-level actor proposes a relative subgoal. The supervisor supplies a scalar evaluation, a corrective target, and an activation mask; the target is used only in the high-level training objective when the mask is active. The low-level controller always executes the actor-generated subgoal rather than the supervisor-provided target. Black arrows indicate the flow of states, subgoals, actions, rewards, and supervisory information. The orange, yellow, blue, and green blocks denote the supervisor, high-level actor, low-level actor, and environment, respectively. In the upper-right inset, the yellow star denotes the task goal, the blue arrows show the proposed and corrected subgoal directions, and the sad and smiling symbols indicate an unsuitable proposal and its feasible correction.

Table 1. Comparison with representative feedback-guided and feasibility-aware HRL methods. Panel (**a**) summarizes the supervision signal and its learning role; panel (**b**) separates the corrective-target and feasibility mechanisms. An "explicit continuous corrective target" denotes a vector-valued target defined in the same continuous space as the high-level subgoal, rather than a scalar evaluation, pairwise preference, candidate ranking, or replay consistency correction. Boldface identifies the proposed SCS-HRL method and its corresponding characteristics.

(a) Supervision and Learning Role

Method	Supervision Signal	Learning Role	Supervised Level
HhP [18]	Human pairwise preferences	Reward learning	High-level behavior
HUGE [21]	Human state comparisons	State selection	Goal exploration
MENTOR [19]	Human preferences	High-level guidance	High-level subgoal
PIPER [13]	Primitive-informed preferences	Replay-reward relabeling	High-level replay
DIPPER [20]	Sequence preferences	Direct preference optimization	High-level policy
HUBS-GO [12]	Human-specified subgoals	Candidate refinement	Subgoal set
SSE [11]	Reachability and failure evidence	Planning refinement	Graph planner
SCS-HRL	**Scalar evaluation and continuous target**	**Critic and actor learning**	**High-level subgoal**

(b) Corrective Target and Feasibility Mechanism

Method	Explicit Continuous Corrective Target	Learned Feedback Model	Feasibility Mechanism
HhP [18]	No	Yes	Curiosity-driven low-level exploration
HUGE [21]	No	Yes	Preference-based breadcrumb selection
MENTOR [19]	No	Yes	Adaptive distance constraint
PIPER [13]	No	Yes	Hindsight relabeling and primitive regularization
DIPPER [20]	No	No	Bilevel optimization and low-level value regularization
HUBS-GO [12]	No	No	Bayesian candidate filtering
SSE [11]	No	No	Frontier replay and failure-aware refinement
SCS-HRL	**Yes**	**No**	**Topology and clearance rules; learned-policy execution**

2. Related Work

2.1. Hierarchical Reinforcement Learning and Subgoal Feasibility

Hierarchical reinforcement learning (HRL) addresses long-horizon decision making by decomposing a task across multiple temporal or behavioral levels. Early frameworks, including MAXQ [9] and the options framework [22], formalized value-function decomposition and temporally extended actions. Option-Critic framework subsequently enabled intra-option policies and termination conditions to be learned jointly [23], while FeUdal Networks introduced a manager–worker architecture in which higher-level goals support long-range credit assignment [10]. These approaches establish the central advantage of HRL: a high-level policy can reason over temporally abstract decisions, while a low-level policy handles primitive control.

Continuous-control HRL, however, introduces a coupled learning problem, because the meaning and attainability of a high-level subgoal change as the low-level policy evolves. HIRO addresses this non-stationarity through off-policy correction, which relabels previously collected high-level transitions according to the behavior induced by the current low-level policy [24]. Despite the shared term *correction*, this mechanism differs from the

corrective supervision studied here. HIRO's correction is a replay consistency mechanism, whereas SCS-HRL provides a task- and geometry-informed target describing how a proposed subgoal should be modified. The two mechanisms therefore address different sources of errors.

Recent HRL research has increasingly treated subgoal feasibility as a primary learning objective rather than an incidental outcome of temporal abstraction. HUBS-GO refines a set of human-specified subgoals by modeling statistical relationships among candidate subgoals [12]. Strict subgoal execution identifies reachability frontiers and exploits evidence of low-level failure to improve long-horizon planning [11]. PIPER and DIPPER likewise introduce primitive-aware regularization to reduce infeasible high-level predictions [13,20]. Collectively, these studies demonstrate that effective temporal decomposition alone does not guarantee useful subgoals; the high-level policy must also account for the capabilities of the low-level controller and the structural constraints of the environment. SCS-HRL follows this feasibility-aware perspective but learns from a continuous corrective target defined directly in the subgoal space.

2.2. Learning from Evaluative, Preference, and Corrective Feedback

Human-in-the-loop reinforcement learning supplements or replaces manually specified rewards with information supplied by a teacher. Existing feedback mechanisms can be broadly organized into evaluative, preference-based, and corrective signals. Evaluative approaches assign a scalar assessment to an agent's behavior. TAMER and Deep TAMER learn from immediate human evaluations of state–action decisions [25,26], whereas COACH and Deep COACH interpret feedback as policy-dependent advice for online policy improvement [27,28]. Such scalar feedback can indicate whether a decision is desirable, but it does not necessarily identify how an undesirable decision should be changed.

Preference-based approaches reduce the burden of absolute evaluation by asking a teacher to compare behaviors. A reward model can be learned from pairwise comparisons between trajectory segments and subsequently used to optimize the policy [14]. PEBBLE improves the feedback efficiency of this paradigm through unsupervised pretraining, active query selection, off-policy learning, and replay buffer relabeling [29]. T-REX and D-REX infer reward functions from ranked demonstrations, thereby avoiding the requirement for optimal expert trajectories [30,31]. Although evaluative and preference-based signals communicate relative behavior quality, they generally do not provide a vector-valued target specifying the direction and magnitude of a desired change in a continuous action or subgoal space.

Corrective feedback provides more structured information by indicating how an agent's proposal should be modified. Such information can be particularly valuable at the planning level, where a single correction can influence multiple primitive control steps. Nevertheless, human-provided correction is costly, subjective, and difficult to reproduce across experimental runs. Synthetic and programmatic supervisors therefore provide a useful intermediate instrument for controlled mechanism studies. Metcalf et al. examined the conditions under which synthetic preference labelers can serve as development proxies [17], while Anand et al. investigated adaptive querying through simulated studies, a user study, and robot experiments [16]. In this work, the supervisor is deliberately deterministic and programmatic; its purpose is to isolate the learning effect of continuous corrective information rather than model human behavior or establish the usability of a human feedback interface.

2.3. Feedback-Guided Hierarchical Reinforcement Learning

Several recent methods incorporate feedback into goal-conditioned or hierarchical learning, but they differ substantially in the representation and role of that feedback. HhP learns a high-level reward model from human preferences and combines it with curiosity-driven low-level exploration [18]. Human-guided exploration (HUGE) uses infrequent and potentially noisy human comparisons to select promising states as "breadcrumbs", separating feedback-guided exploration from self-supervised policy learning [21]. MENTOR introduces human preference guidance for high-level subgoal selection together with a dynamic distance constraint that adapts to the capability of the low-level controller [19]. These methods demonstrate that high-level feedback can reduce the burden associated with primitive-step supervision, although their feedback is primarily expressed through evaluations, comparisons, or candidate selection.

Primitive-informed preference methods provide a second relevant line of research. PIPER generates preferences from primitive-level information, learns a reward model, and uses that model to relabel the high-level replay buffer; primitive-informed regularization is additionally employed to discourage infeasible subgoals [13]. DIPPER avoids an explicit learned reward model by formulating high-level learning as a bilevel direct-preference optimization problem and regularizing the high-level policy using low-level value information [20]. From a theoretical perspective, Yan and Lipton studied how cardinal and ordinal supervisory information can be allocated across hierarchical levels [32]. These methods principally transform preferences or primitive-level evidence into rewards, replay labels, or preference optimization objectives. By contrast, SCS-HRL represents corrective information as a continuous target in the same coordinate space as the high-level subgoal and restricts its effect to a masked actor-training term.

Table 1 summarizes the positioning of SCS-HRL relative to representative feedback-guided and feasibility-aware HRL methods. The comparison focuses on the form of supervision, where it enters the hierarchy, and whether it supplies an explicit continuous corrective target.

SCS-HRL differs in both the representation and use of its supervisory signal. The scalar evaluation contributes to high-level value learning, whereas an active continuous target enters a masked auxiliary loss for the high-level actor. The target is not executed by the low-level controller; the learned high-level actor remains responsible for every subgoal used during interaction. This separation from direct waypoint intervention allows the learning effect of corrective information to be evaluated without attributing performance gains to externally controlled execution. The deterministic topology- and clearance-aware supervisor used in AntMaze also removes inter-annotator variability, providing a reproducible setting for controlled ablations of scalar evaluation, continuous correction, and direct guidance.

3. Methodology

3.1. Problem Formulation and Hierarchical Policy

We consider a goal-conditioned Markov decision process with a state space $\mathcal{S}$, primitive action space $\mathcal{A}$, goal space $\mathcal{G}$, transition kernel P, and task goal $g \in \mathcal{G}$. The policy is decomposed into a high-level actor π_h and a low-level actor π_l. Let $\phi_{xy} : \mathcal{S} \to \mathbb{R}^2$ extract the planar torso position from the state, and let the continuous subgoal dimension be $d_g = 2$. At a high-level decision time t, which occurs every K primitive steps, the deterministic high-level actor proposes a relative planar offset

$$o_t = \pi_h(s_t, g) \in \mathbb{R}^{d_g}, \qquad \tilde{g}_t = \phi_{xy}(s_t) + o_t, \tag{1}$$

where $\tilde{g}_t$ is the corresponding absolute waypoint. The actor output is bounded component-wise, and the low-level controller receives the relative displacement to $\tilde{g}_t$ rather than the supervisor's corrective target. This policy-driven execution path is shown in Figure 1.

3.2. Dual-Channel Supervisor Interface

At the same decision time, a planning-level supervisor evaluates the actor's proposal through the interface

$$F(s_t, o_t, g) = (y_t, c_t, m_t), \tag{2}$$

where $y_t \in \{-1, +1\}$ is a scalar evaluative label, $c_t \in \mathbb{R}^{d_g}$ is a corrective *target offset*, and $m_t \in \{0, 1\}$ indicates whether a continuous target is available for that replay transition. Thus, c_t is expressed in the same relative coordinate system as o_t; it is not an increment added to o_t. The implied correction vector is

$$\Delta_t = c_t - o_t. \tag{3}$$

The tuple is stored for learning, but the executed absolute waypoint remains $\tilde{g}_t = \phi_{xy}(s_t) + o_t$. The mathematical interface does not assume a particular feedback source; however, every experiment reported in this study uses the deterministic programmatic supervisor described in Section 4.2. In the fully simulated SCS-HRL condition, a target is supplied at every eligible high-level decision ($m_t = 1$); $m_t = 0$ covers unqueried decisions or feedback sources that provide only a scalar label. The mask therefore encodes target availability rather than the sign of y_t.

For the AntMaze instantiation, let κ_t^{sg} denote the Euclidean clearance of the proposed absolute waypoint $\tilde{g}_t$ from the nearest wall, and let κ_t^{path} denote the minimum clearance along the straight segment from $\phi_{xy}(s_t)$ to $\tilde{g}_t$. The normalized discrepancy between the actor proposal and the corrective target is

$$e_t = \min\left(\frac{\|o_t - c_t\|_2}{d_{\mathrm{cell}}}, e_{\max}\right), \tag{4}$$

$$\eta_t = \beta_{\mathrm{bad}}\mathbb{I}[y_t = -1] + \beta_{\mathrm{sg}}\mathbb{I}[\kappa_t^{\mathrm{sg}} < \delta_{\mathrm{sg}}] + \beta_{\mathrm{path}}\mathbb{I}[\kappa_t^{\mathrm{path}} < \delta_{\mathrm{path}}], \tag{5}$$

$$\hat{r}_t^h = \mathrm{clip}(b - w_e e_t - \eta_t, -r_{\max}, r_{\max}). \tag{6}$$

The score written to the high-level replay transition is

$$r_t^h = \alpha_h \hat{r}_t^h, \qquad \alpha_h = 0.1, \tag{7}$$

and thus the implemented critic reward is bounded to $[-1, 1]$ when $r_{\max} = 10$. Here, d_{cell} is the maze cell width, b is the base score, w_e weights the normalized target discrepancy, and $e_{\max}$ caps that discrepancy. The thresholds δ_{sg} and δ_{path} define insufficient waypoint and segment clearance, respectively; β_{bad}, β_{sg}, and β_{path} are the associated penalties. The indicator $\mathbb{I}[\cdot]$ equals one when its condition holds. Equation (6) produces the unscaled supervisor score, and Equation (7) gives the value stored for high-level critic learning. The feedback interface itself remains unchanged if this deterministic shaping rule is replaced by another source of evaluation and correction.

3.3. Two-Level Policy and Intrinsic Control Objective

At the start of a high-level interval, the low-level relative target is initialized as $o_t^l = o_t$. The low-level actor $\pi_l : \mathcal{S} \times \mathbb{R}^{d_g} \to \mathcal{A}$ maps the current state and relative target to a primitive control action. After the environment transitions from s_t to s_{t+1}, the target is updated as follows:

$$o_{t+1}^l = \phi_{xy}(s_t) + o_t^l - \phi_{xy}(s_{t+1}), \tag{8}$$

which preserves the same absolute waypoint throughout the interval. The low-level intrinsic reward is the negative residual distance after the primitive transition:

$$r_t^l = -\left\| \phi_{xy}(s_t) + o_t^l - \phi_{xy}(s_{t+1}) \right\|_2. \tag{9}$$

Accordingly, the low-level policy learns local displacement control, whereas the high-level policy learns which displacement should be requested to advance toward the task goal.

3.4. Off-Policy Actor–Critic Learning

Both levels use twin-delayed deterministic actor–critic updates with target policy smoothing. For a low-level replay transition $(s, o^l, a, r^l, s', o^{l\prime}, d)$, where d is the terminal indicator, the smoothed target action, Bellman target, and critic loss are

$$\tilde{a}' = \mathrm{clip}\Big(\bar{\pi}_l(s', o^{l\prime}) + \epsilon, -a_{\max}, a_{\max} \Big), \tag{10}$$

$$z_l = r^l + (1-d)\gamma_l \min_{i\in\{1,2\}} \bar{Q}_{l,i}(s', o^{l\prime}, \tilde{a}'), \tag{11}$$

$$\mathcal{L}_{Q_l} = \sum_{i=1}^{2} \mathbb{E}_{\mathcal{D}_l}\Big[\mathrm{Huber}\Big(Q_{l,i}(s, o^l, a) - z_l \Big) \Big], \tag{12}$$

where ϵ is clipped Gaussian noise, bars denote target networks, and Huber is the smooth-L_1 loss. The low-level actor minimizes

$$\mathcal{L}_{\pi_l} = -\mathbb{E}_{\mathcal{D}_l}\Big[Q_{l,1}(s, o^l, \pi_l(s, o^l)) \Big]. \tag{13}$$

Let k denote the start of a high-level interval, and let $q \le K$ be the number of primitive actions executed before the interval ends or the episode terminates. The high-level replay buffer stores $(s_k, g, o_k, r_k^h, s_{k+q}, d_{k+q}, c_k, m_k)$. The high-level discount γ_h is applied once per stored high-level transition, rather than being computed as γ_l^K. SCS-HRL retains the actor-generated subgoal o_k in replay and does not apply HIRO's likelihood-based high-level action relabeling. Its smoothed target subgoal, Bellman target, and critic loss are

$$\tilde{o}' = \mathrm{clip}\Big(\bar{\pi}_h(s_{k+q}, g) + \epsilon_h, -o_{\max}, o_{\max} \Big), \tag{14}$$

$$z_h = r_k^h + (1-d_{k+q})\gamma_h \min_{i\in\{1,2\}} \bar{Q}_{h,i}(s_{k+q}, g, \tilde{o}'), \tag{15}$$

$$\mathcal{L}_{Q_h} = \sum_{i=1}^{2} \mathbb{E}_{\mathcal{D}_h}[\mathrm{Huber}(Q_{h,i}(s_k, g, o_k) - z_h)]. \tag{16}$$

The corrective signal affects only the high-level actor objective. For a sampled high-level mini-batch $\mathcal{B}_h$, the actor loss is

$$\mathcal{L}_{\pi_h} = -\mathbb{E}_{\mathcal{D}_h}[Q_{h,1}(s_k, g, \pi_h(s_k, g))] + \lambda_{\mathrm{bc}} \frac{\sum_{j\in\mathcal{B}_h} m_j \| \pi_h(s_j, g_j) - c_j \|_2^2}{\max\Big(d_g \sum_{j\in\mathcal{B}_h} m_j, 1 \Big)}. \tag{17}$$

The mask m_j ensures that only transitions for which a continuous target was supplied contribute to the regression term. Division by $d_g \sum_j m_j$ averages the squared error over target-available samples and subgoal dimensions; the maximum with one prevents division by zero. The coefficient λ_{bc} controls the strength of corrective supervision. Setting $\lambda_{\mathrm{bc}} = 0$ removes continuous correction while retaining the scalar high-level reward, yielding the scalar-only ablation. When both the negative reward and nonnegative reward replay

partitions are nonempty, the implementation draws half of each mini-batch from each partition to reduce imbalance in high-level updates.

Actor and critic target parameters $\bar{\theta}$ are updated through Polyak averaging:

$$\bar{\theta} \leftarrow \tau\theta + (1-\tau)\bar{\theta}. \tag{18}$$

The corrective term changes the learned actor parameters through replay but never changes the subgoal already issued to the low-level controller.

Algorithm 1 summarizes the complete SCS-HRL training procedure and the interaction between the two replay buffers and the supervisor.

Algorithm 1 Training SCS-HRL with simulated corrective subgoal supervision

Require: high-level interval K; replay buffers $\mathcal{D}_l, \mathcal{D}_h$; supervisor F
Initialize π_l, π_h, twin critics, target networks, and replay buffers
for each training episode **do**
  Reset the environment; obtain s_0 and fixed task goal g
  for $t = 0, \ldots, T-1$ **do**
    if t mod $K = 0$ **then**
      Set interval start $k \leftarrow t$, $s_k \leftarrow s_t$, and counter $q \leftarrow 0$
      Propose and clip $o_k \leftarrow \pi_h(s_k, g)+$ exploration noise
      Obtain $(y_k, c_k, m_k) \leftarrow F(s_k, o_k, g)$; compute $\hat{r}_k^h$ and $r_k^h = \alpha_h \hat{r}_k^h$
      Initialize the low-level relative target $o_t^l \leftarrow o_k$
    end if
    Execute $a_t \leftarrow \pi_l(s_t, o_t^l)+$ exploration noise
    Observe s_{t+1} and compute r_t^l using Equation (9)
    Store the low-level tuple in $\mathcal{D}_l$ and update o_{t+1}^l
    Increment the interval counter $q \leftarrow q+1$
    Update low-level critics and the delayed actor after replay warm-up
    if $q = K$ or the episode terminates **then**
      Store $(s_k, g, o_k, r_k^h, s_{k+q}, d_{k+q}, c_k, m_k)$ in $\mathcal{D}_h$
      Update high-level critics and delayed actor from balanced replay
    end if
    On delayed actor updates, soft-update the corresponding target networks
    if the episode terminates **then**
      break
    end if
  end for
end for

4. Experiments

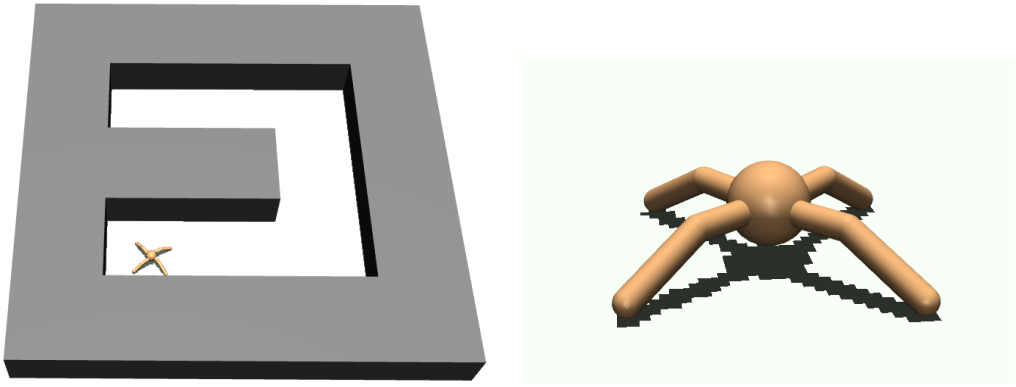
4.1. Experimental Environment

The experiments used the bundled AntMaze implementation derived from the public TensorFlow `efficient-hrl` environments associated with HIRO [24]. The physics engine was MuJoCo 2.1.0 [33]; the interface used OpenAI Gym 0.23.1 [34] and `mujoco-py` 2.1.2.14. The software stack comprised Python 3.10, NumPy 1.23.5, PyTorch 2.8.0, and CUDA 12.8. The AntMaze environment was a 5×5 grid with a cell width $d_{\text{cell}} = 4$ MuJoCo world coordinate units. The ant started near $(0, 0)$ and navigated to the fixed goal at $(0, 8)$ within 500 primitive steps. This was an online continuous-control task rather than an offline D4RL benchmark.

After executing a_t and observing s_{t+1}, the wrapper supplied the dense task reward $r_t^{\text{env}} = -\|\phi_{xy}(s_{t+1}) - g\|_2$. In the U-shaped layout, a valid route around the blocking wall could temporarily increase the Euclidean distance to the goal, making the reward topologically misleading. The benchmark therefore provided a focused test of the interaction between local locomotion and long-horizon subgoal selection. This complements

perception-rich navigation studies that address LiDAR histories, dense pedestrian traffic, and sim-to-real deployment [6].

Figure 2 illustrates the AntMaze layout and the simulated MuJoCo ant used in all experiments.



(**a**) AntMaze environment (**b**) Ant robot

Figure 2. AntMaze simulation used in the experiments. The left panel shows the U-shaped 5×5 maze with a cell width of 4 world coordinate units; the right panel shows the MuJoCo ant controlled through eight actuator commands. The start and fixed goal were approximately $(0, 0)$ and $(0, 8)$, respectively.

4.2. Subgoal Feedback Implementation

The supervisor contained a directed route table for every free maze cell. The world coordinates were converted to grid indices by the cell width and the maze origin. A proposed subgoal was accepted only if (1) it lay in a free cell, (2) its offset length was at most $1.25 d_{\text{cell}} = 5.0$, (3) it advanced along the directed route (a sufficiently aligned short step within the current cell, the next route cell, or at most two directed cells ahead), (4) its point clearance was at least $\delta_{\text{sg}} = 0.6$, and (5) the straight segment to it had clearance of at least $\delta_{\text{path}} = 0.35$.

Clearance is the Euclidean distance to the nearest axis-aligned wall rectangle. For the segment check, points were sampled uniformly from the current position through the proposed subgoal with a maximum spacing of 0.25 world coordinate units; the minimum sampled clearance was κ_t^{path}. This construction provided a consistent geometric feasibility test at the high-level decision timescale.

To construct c_t, the supervisor selected the center of the next cell in the directed route, subtracted the current position, and normalized the resulting two-dimensional offset to a magnitude d_{cell} (except in the terminal cell, where the unnormalized within-cell offset was used). The shaped reward in Equation (6) used $b = 10$, $w_e = 20$, $e_{\max} = 1$, $\beta_{\text{bad}} = \beta_{\text{sg}} = \beta_{\text{path}} = 2$, and $r_{\max} = 10$; the replay value was scaled by $\alpha_h = 0.1$ as specified in Equation (7). The scalar-only ablation received the same shaped reward but set $\lambda_{\text{bc}} = 0$. Two direct-rule controls were evaluated. The independently trained control executes the route-table offset during both low-level training and evaluation. The same-low-level control instead applies that route rule at evaluation time to the low-level controller from each corresponding SCS-HRL seed, isolating the effect of learned high-level subgoal selection from low-level-controller differences.

Because the route table, wall positions, and clearance rules were supplied explicitly, the supervisor is best understood as topology-aware reward shaping plus corrective behavioral supervision. Its decision count was logged separately from actual human queries. A terminal interface can collect human labels and optional two-dimensional corrections, but no output from that interface was used in the reported experiments.

4.3. State Space and Action Space Settings

The policy observation was 31-dimensional. It contained 29 MuJoCo generalized position/velocity values, the normalized inner-environment time feature, and the wrapper's raw episode-step counter. The two time channels were redundant but retained because they were part of the implementation used for all compared methods. The action was an eight-dimensional bounded MuJoCo actuator-control vector and not a vector of target joint angles. Table 2 gives the exact decomposition.

Table 2. AntMaze policy observation and action configuration

State Name	Dim	Description
torso_position	3	World-frame torso position (x, y, z)
torso_orientation	4	Torso quaternion
joint_position	8	Eight actuated-joint generalized positions
torso_linear_velocity	3	Torso translational generalized velocities
torso_angular_velocity	3	Torso rotational generalized velocities
joint_velocity	8	Eight actuated-joint generalized velocities
normalized_time	1	Inner-environment step multiplied by 10^{-3}
episode_step	1	Raw wrapper step in $[0, 500]$

Both actors and all critics received the 31-dimensional observation. The high-level actor additionally received the two-dimensional final task goal and output a two-dimensional relative planar target bounded componentwise to $[-5, 5]$. The low-level actor received this current two-dimensional relative displacement and output eight actuator controls.

4.4. Benchmark Algorithms

The primary comparison contained three learned-policy conditions and two direct-rule controls. HIRO used the same AntMaze implementation, high-level interval, TD3 backbone, and training budget [24]. Scalar-only retained the SCS-HRL architecture and shaped scalar reward but set $\lambda_{\mathrm{bc}} = 0$, and thus the high-level actor received no continuous target loss. SCS-HRL received route information only as a training target and executed its learned high-level actor during evaluation. The independently trained direct rule condition executed the route-table offset during both low-level training and evaluation. The direct rule + same low level condition replaced the learned high-level actor with the route rule only at evaluation time while reusing the corresponding SCS-HRL low-level controller. HIRO provides a standard learned HRL baseline, scalar-only isolates the continuous target, and the two direct-rule conditions separate online access to topology from low-level training compatibility.

4.5. Training and Evaluation Protocol

Actors and critics used two fully connected hidden layers of 300 ReLU units. The low-level actor and critic learning rates were 10^{-4} and 10^{-3}, respectively. For the high-level controller, HIRO retained the defaults of the evaluated implementation: actor and critic learning rates of 10^{-4} and 10^{-3}, respectively, and a Gaussian exploration noise standard deviation of 1.0. SCS-HRL instead used high-level actor and critic learning rates of 3×10^{-5} and 3×10^{-4}, respectively, and an exploration noise standard deviation of 0.4, with $\lambda_{\mathrm{bc}} = 1.0$. Thus, the high-level optimization settings were method-specific rather than identical. The scalar-only variant changed only λ_{bc} to zero relative to SCS-HRL. Target-policy smoothing used Gaussian noise with a standard deviation of 0.2 clipped to $[-0.5, 0.5]$; low-level actions were clipped to $[-1, 1]$, and two-dimensional high-level offsets were clipped component-wise to $[-5, 5]$. Both replay discounts were 0.99 per stored transition, the target update coefficient was $\tau = 0.005$, the replay capacity was

200,000 transitions, the batch size was 100, and the actors were updated once per two critic updates. High-level decisions and aligned replay transitions occurred every $K = 10$ primitive steps, and training started after 2500 primitive interactions.

Every trained main condition used 6000 episodes of at most 500 primitive steps, corresponding to at most 3,000,000 primitive interactions per seed. SCS-HRL, scalar-only, HIRO, and the independently trained direct rule condition used seeds 2–6. Direct rule + same low level was an evaluation-only counterfactual paired by seed with the five SCS-HRL low-level controllers. Each fixed episode-6000 checkpoint was evaluated for 100 episodes with deterministic policy actions. The primary metric was final success; the true torso position at the end of the episode (at step 500 unless the episode terminates earlier) had to be within the 5-world-unit success region centered on the fixed goal. We also report the final distance and success weighted by path length (SPL), defined over N evaluation episodes as $N^{-1} \sum_{i=1}^{N} S_i L_i^* / \max(L_i, L_i^*)$. Here, S_i is the final success indicator, L_i is the executed path length up to first entry into the success region, and L_i^* is the shortest collision-free reference-path length from the episode start to that success region. Failed rollouts therefore contributed zero to SPL, rather than being compared directly with successful path lengths. For the ablation metric "steps to first success", we included only episodes classified as successful at the end of the episode and recorded the first primitive step at which the trajectory entered the success region.

The statistical unit was the training seed and not the 100 evaluation episodes. Unless stated otherwise, entries are the mean and sample standard deviation of seed-level summaries; two-sided 95% confidence intervals used the Student-t distribution across training seeds. The main comparison used the fixed episode-6000 checkpoint without selecting the best checkpoint over training. Corrective-coefficient and reward-component studies used matched seeds 2–4, and they are presented with their individual seed observations, means, and sample standard deviations.

4.6. Five-Seed Fixed-Checkpoint Comparison

Table 3 and Figure 3 compare the five conditions at the fixed episode-6000 checkpoint. SCS-HRL reached an $88.4 \pm 7.8\%$ final success rate, whereas the scalar-only and HIRO frameworks remained at 0%. The independently trained direct rule condition reached only $4.8 \pm 10.7\%$, indicating that the deterministic route rule is not by itself sufficient when its low-level controller is trained separately. To isolate low-level-controller compatibility, direct rule + same low level applied the route rule to the five SCS-HRL low-level controllers and reached $82.2 \pm 9.9\%$.

Across the five paired seeds, SCS-HRL exceeded direct rule + same low level by 6.2 percentage points. The paired difference had a 95% confidence interval of [2.1, 10.3] percentage points, $t(4) = 4.24$, and $p = 0.013$. This comparison indicates that the learned high-level actor does more than reproduce the route-center target: jointly optimizing the critic objective and corrective loss yielded subgoals that were, on average, better matched to the learned low-level controller. This result does not imply that learned guidance is generally superior to planning rules; it is specific to the supplied route rule, the trained controller family, and the source AntMaze layout. The high clearance of the scalar-only model despite the 0% success rate further illustrates that clearance alone is not a measure of task completion.

Table 3. Fixed-checkpoint performance in the source AntMaze at episode 6000. Panel (**a**) reports task completion and path efficiency. Panel (**b**) reports geometric exposure. Every reported condition had five seed-level evaluation units (2–6), and every seed was evaluated for 100 deterministic episodes. Direct rule + same low level was paired by seed with SCS-HRL and reused its low-level controller; the other rows used independently trained controllers. Values are mean ± sample standard deviation across seeds. Clearance statistics use all recorded rollout states and should not be interpreted as success-conditioned path efficiency. Upward and downward arrows indicate that higher and lower values are preferable, respectively; boldface identifies the proposed method and its best task-completion results.

(a) Task Completion and Path Efficiency

Method	Final Success (%) ↑	SPL ↑	Final Distance (World Units) ↓
SCS-HRL	**88.4 ± 7.8**	**0.841 ± 0.106**	**1.48 ± 0.70**
Direct rule + same low level	82.2 ± 9.9	0.818 ± 0.117	1.96 ± 0.87
Direct rule + independent low level	4.8 ± 10.7	0.043 ± 0.096	8.92 ± 1.26
Scalar-only ($\lambda_{bc} = 0$)	0.0 ± 0.0	0.000 ± 0.000	8.88 ± 0.32
HIRO	0.0 ± 0.0	0.000 ± 0.000	7.16 ± 0.49

(b) Geometric Exposure over All Recorded Rollout States

Method	Mean Clearance (World Units) ↑	Near-Wall (%) ↓
SCS-HRL	1.55 ± 0.04	6.6 ± 2.8
Direct rule + same low level	1.55 ± 0.05	8.2 ± 2.8
Direct rule + independent low level	1.19 ± 0.11	37.9 ± 17.9
Scalar-only ($\lambda_{bc} = 0$)	1.52 ± 0.05	5.6 ± 3.0
HIRO	0.84 ± 0.33	65.8 ± 40.6

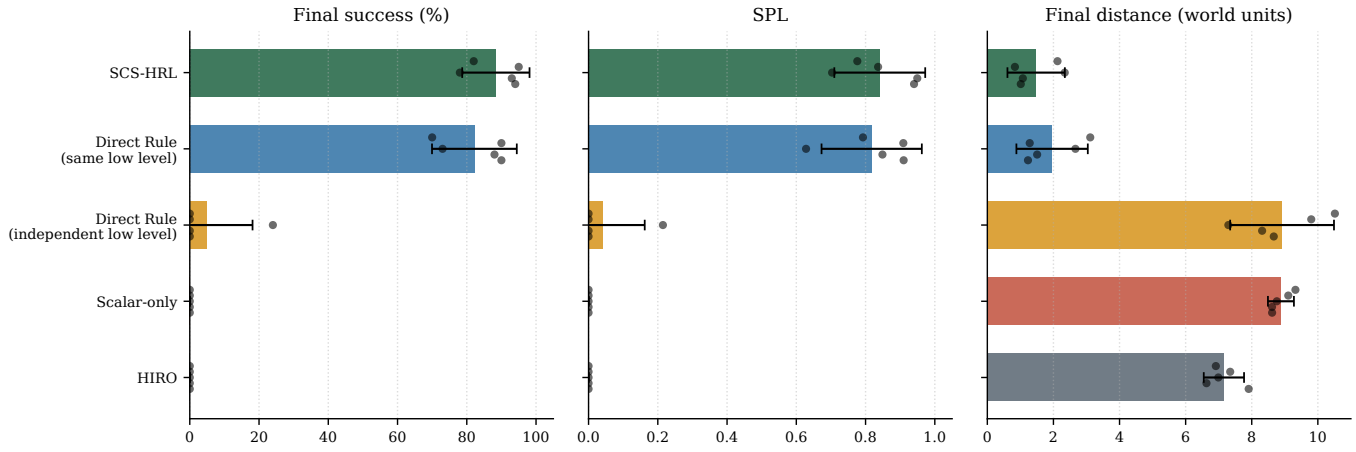


Figure 3. Fixed episode-6000 comparison. Bars show seed means, black points show the five seed-level values, and error bars show two-sided 95% Student-*t* confidence intervals. SCS-HRL and direct rule + same low level shared the same five low-level controllers, whereas the independently trained direct rule condition used separately trained low-level controllers.

4.7. Corrective-Coefficient and Reward-Shaping Ablations

Table 4 varies λ_{bc} while holding the 6000-episode budget and final evaluation protocol fixed. All five settings used the same three seeds (2–4), and thus differences were not confounded by unmatched random seeds.

Table 4. Sensitivity to the continuous-target coefficient at the episode-6000 checkpoint. All settings used matched seeds 2–4 and 100 deterministic evaluation episodes per seed. Values are seed-level mean ± sample standard deviation. Upward and downward arrows indicate that higher and lower values are preferable, respectively.

λ_{bc}	Final Success (%) ↑	SPL ↑	Final Distance (World Units) ↓	Near Wall (%) ↓
0.0	0.0 ± 0.0	0.000 ± 0.000	9.06 ± 0.28	6.1 ± 2.2
0.25	59.0 ± 51.1	0.589 ± 0.500	3.98 ± 3.83	11.0 ± 7.2
0.5	84.3 ± 6.0	0.847 ± 0.054	1.91 ± 0.76	9.2 ± 3.4
1.0	85.0 ± 8.9	0.772 ± 0.067	1.77 ± 0.81	7.9 ± 2.3
2.0	82.3 ± 8.6	0.842 ± 0.120	1.75 ± 0.55	7.3 ± 1.9

The matched sweep revealed a threshold-like pattern. Setting $\lambda_{bc} = 0$ eliminated success, and $\lambda_{bc} = 0.25$ was highly seed-sensitive; one seed failed, while the other two reached 88% and 89% success rates. Coefficients 0.5, 1.0, and 2.0 all exceeded 82% mean success with substantially smaller cross-seed variation. We retained $\lambda_{bc} = 1.0$ as the pre-specified central setting, rather than claiming that it is uniquely optimal.

The binary-score variant was numerically strongest, reaching 91.0 ± 4.0% success, but its paired three-seed difference from full shaping was not statistically established ($p = 0.428$). Removing individual scalar components likewise did not cause a uniform performance collapse. The largest stability loss occurred without the subgoal clearance penalty, where success varied from 41% to 94% across seeds. Removing the path clearance penalty approximately doubled the mean near-wall ratio from 7.9% to 15.6% and increased the variance in steps to success. Reward clipping was not empirically necessary in this sweep, and this is therefore interpreted as an implementation-level numerical bound rather than a source of performance. Together with the 0% scalar-only result, these outcomes identify the continuous corrective target—not the exact scalar-shaping formula—as the principal additional learning signal in the tested configuration.

Figure 4 visualizes the corrective-coefficient results reported in Table 4. Table 5 and Figure 5 summarize the term-level reward-shaping ablations.

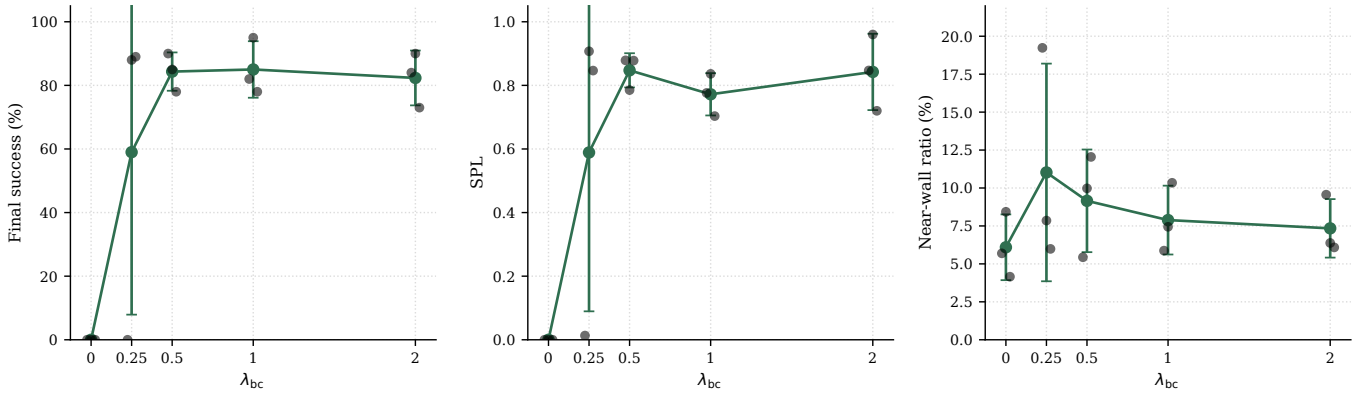


Figure 4. Corrective-coefficient sensitivity using matched seeds 2–4. Large markers and error bars show the mean and sample standard deviation; black points show the three seed-level observations. The wide variation at $\lambda_{bc} = 0.25$ was driven by one failed seed rather than by an inferential confidence interval.

Table 5. Term-level ablation of the shaped supervisor score in Equation (6). Panel (**a**) reports completion and path efficiency metrics. Panel (**b**) reports clearance metrics. All variants retained the continuous corrective target, used matched seeds 2–4, and were evaluated at episode 6000. "Steps to first success" was computed only for episodes classified as successful at the end of the episode and recorded the first primitive step at which the trajectory entered the 5-world-unit success region. Values are seed-level mean ± sample standard deviation. Upward and downward arrows indicate that higher and lower values are preferable, respectively.

(a) Completion and Path Efficiency

Variant	Final Success (%) ↑	Steps to First Success ↓	SPL ↑
Full shaped score	85.0 ± 8.9	150.1 ± 23.8	0.772 ± 0.067
Binary score	91.0 ± 4.0	132.4 ± 12.7	0.919 ± 0.042
Without distance term	86.3 ± 12.4	169.9 ± 24.6	0.817 ± 0.223
Without bad subgoal penalty	76.3 ± 15.6	161.2 ± 38.9	0.781 ± 0.152
Without subgoal clearance penalty	76.0 ± 30.3	142.8 ± 17.5	0.773 ± 0.246
Without path clearance penalty	83.7 ± 9.5	174.1 ± 49.1	0.824 ± 0.039
Without reward clipping	87.3 ± 5.0	139.5 ± 8.0	0.886 ± 0.047

(b) Geometric Exposure over All Recorded Rollout States

Variant	Mean Clearance (World Units) ↑	Near-Wall (%) ↓
Full shaped score	1.54 ± 0.04	7.9 ± 2.3
Binary score	1.54 ± 0.09	7.3 ± 3.6
Without distance term	1.52 ± 0.10	8.4 ± 6.8
Without bad subgoal penalty	1.47 ± 0.14	10.8 ± 6.4
Without subgoal clearance penalty	1.54 ± 0.01	6.0 ± 1.9
Without path clearance penalty	1.43 ± 0.06	15.6 ± 8.1
Without reward clipping	1.51 ± 0.08	5.9 ± 2.3

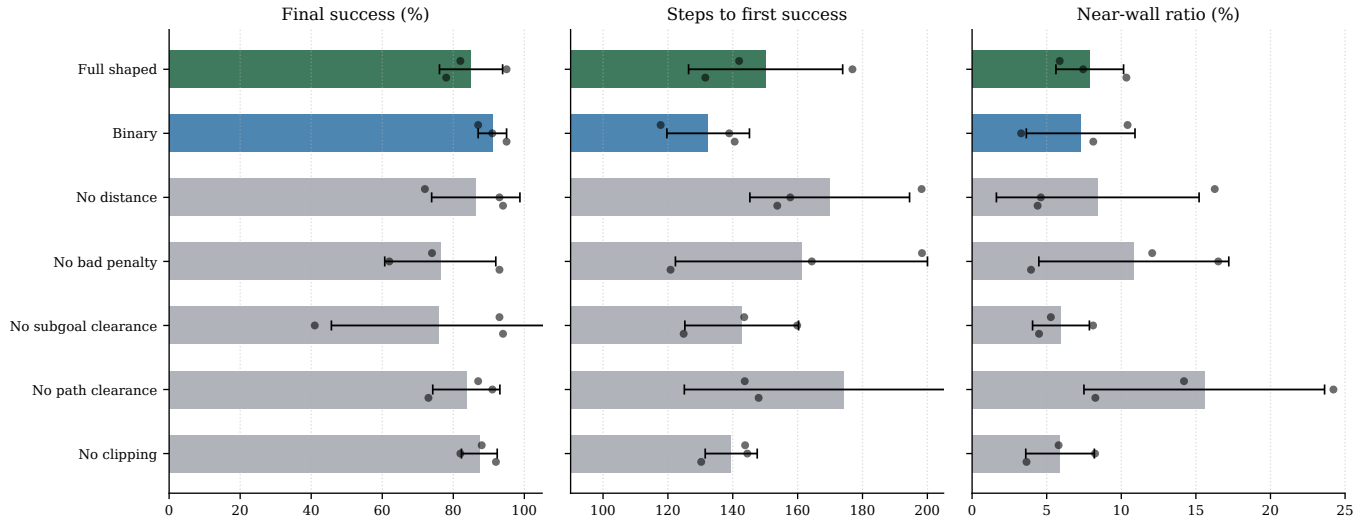


Figure 5. Term-level ablation of Equation (6) using matched seeds 2–4. Green bars denote the full shaped score, gray bars denote ablated variants, black points show individual seeds, and error bars show sample standard deviations. Steps to first success were calculated only for episodes classified as successful at the end of the episode and recorded the first entry into the 5-world-unit success region.

4.8. Observation and Control Noise Stress Test

As a secondary descriptive stress test, we evaluated two separately trained fixed SCS-HRL checkpoints under a clean condition, having planar localization noise (standard deviation of 0.25 world units), non-position sensor/state noise (0.05), actuator noise (0.10),

and their combination, without retraining. These two checkpoints were not pooled with the five-seed primary comparison. Each condition used 100 evaluation episodes per checkpoint, and the learned high-level actor was executed without supervisor intervention.

Table 6 reports the checkpoint-level results for the clean condition and all four perturbations.

Table 6. Noise stress test of fixed SCS-HRL checkpoints from training seeds 2 and 3. Values are the mean ± sample standard deviation across the two checkpoints; each checkpoint was evaluated for 100 episodes in the clean condition and under each perturbation.

Condition	Noise Standard Deviation	Final Success (%)	Final Distance
Clean	None	84.0 ± 8.5	1.62 ± 0.71
Action noise	Action 0.10	79.5 ± 7.8	2.21 ± 0.81
Localization noise	Position 0.25	82.0 ± 8.5	1.95 ± 0.58
Sensor/state noise	Non-position state 0.05	81.5 ± 6.4	1.95 ± 0.73
Combined	All three above	83.0 ± 12.7	1.88 ± 1.11

The clean two-checkpoint mean was 84.0%, and success ranged from 79.5% to 83.0% under the four perturbations. No tested perturbation produced a large numerical decrease relative to the clean two-checkpoint mean; owing to the small sample, this evidence is descriptive. The test did not cover dynamic obstacles, perception failures, or the sim-to-real shift.

Figure 6 displays the checkpoint-level observations, their mean, and their sample standard deviation. No inferential confidence interval is reported for this two-checkpoint comparison.

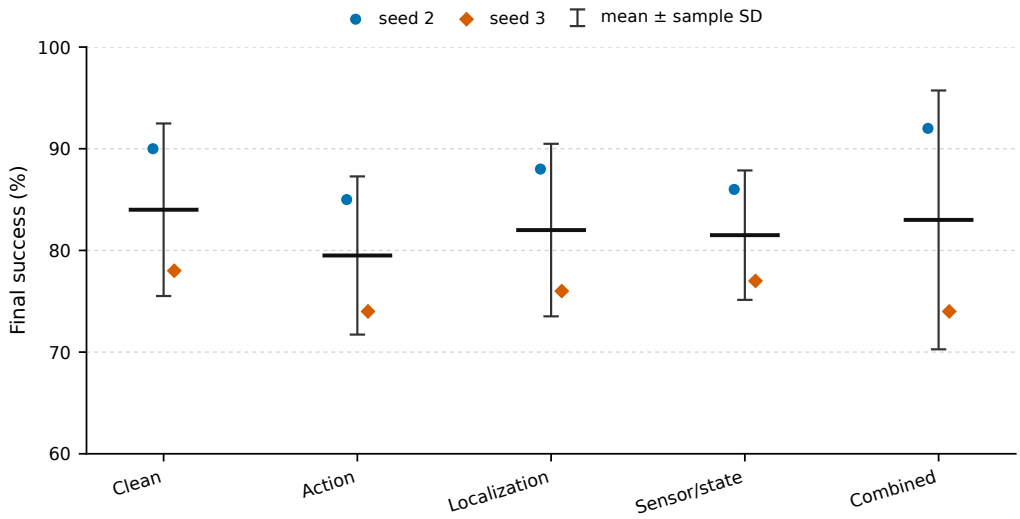


Figure 6. Observation and control noise stress test on two separately trained fixed SCS-HRL checkpoints. Bars show the checkpoint mean, black points show the two observations, and error bars show the sample standard deviation. No inferential confidence interval is reported because only two independently trained checkpoints were available.

4.9. Trajectory Metrics and Failure Handling

Table 3 reports SPL so that failed and successful rollouts were not compared through the raw path length. The success indicator was determined from the torso position at the end of the episode (step 500, unless termination occurred earlier). A failed rollout contributed zero through this indicator, whereas a final-success rollout was discounted when its executed path to first entry into the 5-world-unit success region exceeded the

shortest collision-free reference path from the same start to that region. The mean wall clearance was computed from the true torso position to the nearest axis-aligned wall rectangle over the recorded rollout states. The near-wall ratio is the percentage of those states with clearance below 1.0 world unit. These clearance quantities measure geometric exposure rather than task completion and are therefore reported alongside, but not substituted for, success and SPL.

SCS-HRL and direct rule + same low level had nearly identical mean clearance values (1.55 world units), while SCS-HRL had a lower near-wall ratio (6.6% versus 8.2%) and higher SPL (0.841 versus 0.818). The independently trained direct rule condition had both a low success rate and substantially greater near-wall exposure, demonstrating that the route rule and low-level controller must be evaluated as a coupled system. Scalar-only remained comparatively far from walls but never reached the goal, again showing why clearance from failed rollouts cannot be interpreted as navigation quality by itself.

Figure 7 restores a qualitative view of four fixed episode-6000 rollouts from the earlier trajectory archive. It shows one successful SCS-HRL rollout, one successful rollout from direct rule + independent low level, and representative failures from scalar-only and HIRO. The displayed seeds were selected for visualization in the earlier archive, and they are not redefined as the best seeds under the Plan-B statistics. Accordingly, these four traces were not used in Table 3, the paired test, or any quantitative estimate.

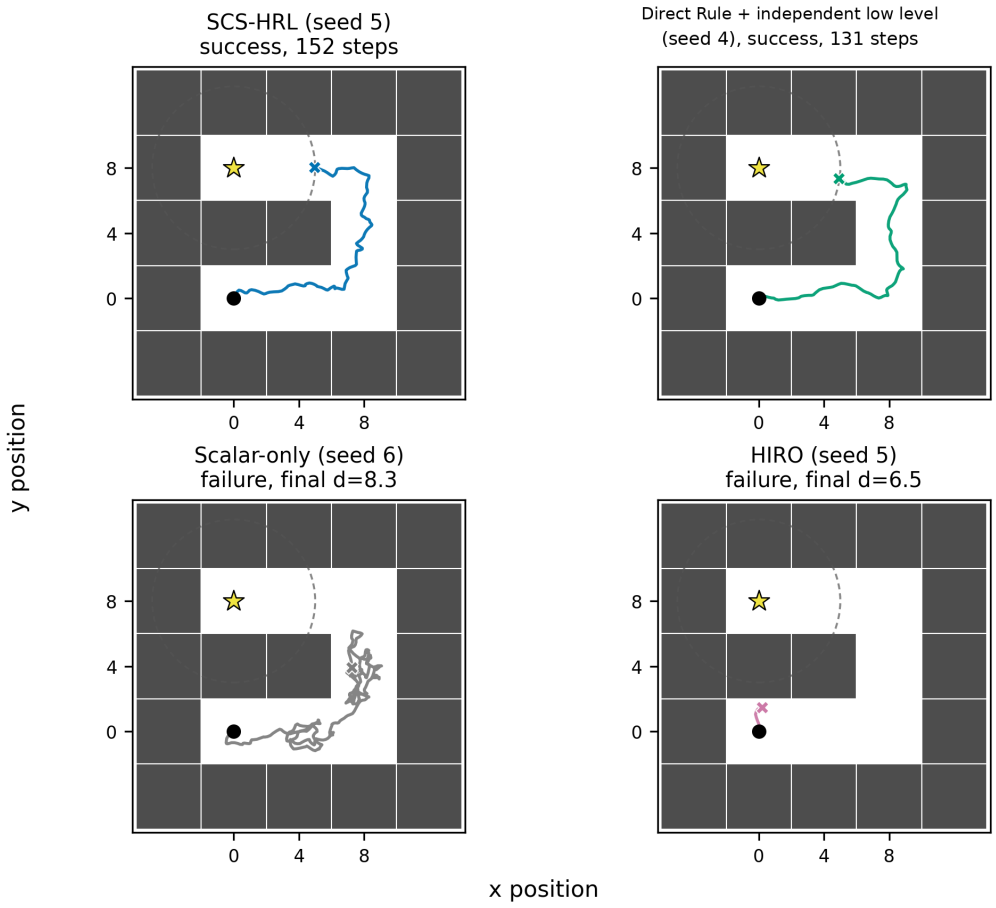


Figure 7. Qualitative deterministic AntMaze rollouts from fixed episode-6000 checkpoints in the earlier trajectory archive. The black circle and yellow star denote the start and goal, respectively, the dashed circle denotes the 5-world-unit success region, and the cross marks the displayed endpoint. Successful traces are truncated at first entry into the success region; failed traces contain all 500 primitive steps. The SCS-HRL, direct rule + independent low level, scalar-only, and HIRO panels used seeds 5, 4, 6, and 5, respectively. These examples illustrate behavior only and do not contribute to the Plan-B five-seed statistics or checkpoint selection.

4.10. Zero-Shot Cross-Layout Evaluation

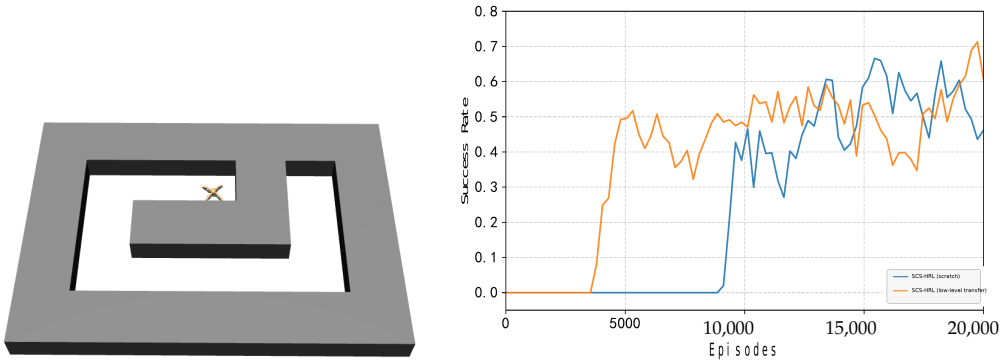
As a secondary transfer diagnostic, separately trained fixed source-AntMaze checkpoints were evaluated without gradient updates on MazeMid and MazeLarge. These checkpoints were not pooled with the five-seed primary comparison. SCS-HRL used two checkpoints, and HIRO used one. The supervisor was inactive at evaluation time, and thus the test measured the learned policy rather than access to target-layout route rules.

None of the fixed policies reached the goal in MazeMid or MazeLarge. SCS-HRL also finished farther from the goal than HIRO on both layouts, despite its strong performance in the source maze. The learned high-level mapping is therefore specific to the source topology. Training across procedurally varied layouts, conditioning the policy on a map representation, or learning a transferable supervisor are direct routes toward cross-layout transfer.

4.11. Adaptive Transfer in the AntMaze Extended Task

The zero-shot result above tested whether a complete source-layout policy transferred without updating. The earlier AntMaze Extended experiment addressed a different question: whether source-layout low-level locomotion parameters can accelerate *continued training* in a larger target maze. The archived target layout, shown in Figure 8, scales the maze to approximately 1.5 times the original size while retaining the observation and actuator interfaces. In both conditions, the high-level policy was trained in the target layout for up to 20,000 episodes using the target-layout programmatic supervisor. One condition trained the low-level controller from scratch, whereas the other initialized it from the source-AntMaze low-level controller. Thus, this experiment evaluated adaptive skill reuse rather than zero-shot high-level generalization.

The archived learning curves show that source low-level initialization reached nonzero target-layout success earlier than training both levels from scratch, although both curves remain variable. The archive contains the plotted diagnostic curves but not replicated seed-level summaries or raw per-seed evaluation records. We therefore report this result only as a legacy, single-run adaptation diagnostic; it is not pooled with the five-seed source-layout comparison, no confidence interval is inferred, and it does not overturn the 0% zero-shot result in Table 7. A replicated target-layout study is required before claiming transfer efficiency.



(**a**) Extended target layout. (**b**) Archived adaptation learning curves.

Figure 8. AntMaze Extended adaptive-transfer diagnostic. The scratch condition trains both hierarchical levels in the target layout; the source-low-level initialization condition transfers only the low-level locomotion parameters and continues target-layout training for both levels. The legacy curve labels were updated to the SCS-HRL terminology without changing the plotted data. Because replicated seed archives were unavailable, the curves are qualitative and are not used in the primary statistical comparison.

Table 7. Zero-shot evaluation of fixed source-layout checkpoints. Every checkpoint was evaluated for 100 episodes without adaptation. Single-seed HIRO entries are reported without artificial zero standard deviations.

Target Layout	Method	*n*	Final Success (%)	Final Distance
MazeMid	SCS-HRL	2	0.0 ± 0.0	11.71 ± 1.40
MazeMid	HIRO	1	0.0	6.89
MazeLarge	SCS-HRL	2	0.0 ± 0.0	33.80 ± 0.16
MazeLarge	HIRO	1	0.0	31.93

5. Discussion

The experiments provide a consistent mechanism-level pattern. Scalar-only received the same topology-aware reward as SCS-HRL but never succeeded, whereas adding the continuous target produced a 88.4% mean final success rate. A scalar score indicates whether a proposal is poor; the target additionally specifies a replacement direction in the two-dimensional subgoal space. This directional information is especially useful near the U-shaped obstacle, where moving directly toward the final goal is locally attractive but topologically incorrect.

The direct-rule controls clarify how much of the result comes from the evaluator and how much comes from learning. Direct rule + same low level had access to the route table at every evaluation decision yet reached 82.2% success, 6.2 percentage points below SCS-HRL in the paired five-seed comparison. The learned actor therefore does not merely copy a fixed cell-center route; the critic term and corrective regularizer jointly adapt subgoal selection to the controller's experience. Conversely, the independently trained direct rule condition reached only 4.8%, revealing strong coupling between high-level commands and low-level training. These findings are specific to the supplied route rule and do not establish a general advantage of learned policies over classical planning.

The coefficient and shaping studies further localize the source of improvement. Coefficients from 0.5 to 2.0 were stable across the three matched seeds, whereas 0.25 was insufficient for one seed. Binary scoring was numerically stronger than the full shaped score, and removing individual scalar terms did not produce a uniform collapse. The clearest component-level effects concern variability and wall exposure; removing subgoal clearance increased cross-seed instability, while removing path clearance approximately doubled the near-wall ratio. Together with the scalar-only result, this evidence identifies the vector-valued actor target—rather than a narrowly tuned scalar formula—as the principal additional signal. No tested perturbation produced a large numerical decrease relative to the clean two-checkpoint mean, but that stress test is descriptive because only two separately trained checkpoints were evaluated.

The principal limitation is zero-shot transfer. The supervisor uses the exact source-maze route table and wall geometry, and complete policies trained on that layout failed in both unseen mazes. The legacy AntMaze Extended diagnostic suggests that source low-level locomotion parameters may accelerate target-layout retraining, but the absence of replicated seed archives prevents a statistical transfer efficiency claim. The study also used simulated labels, a single MuJoCo task family, and no experiments with dynamic obstacles or physical robots. A fully queried 6000-episode run contained as many as 300,000 high-level supervisory decisions, which would be unsuitable for manual labeling. Practical extensions should therefore combine map-conditioned or learned supervisors with sparse querying, train across procedurally varied layouts, and test perception-based policies under dynamic and physical conditions. A faithful empirical comparison with recent preference-guided HRL methods in their validated software and task settings also remains for future work.

6. Conclusions

This paper presented SCS-HRL, which augments scalar high-level evaluation with a continuous corrective target while keeping execution under the learned hierarchical policy. Under a five-seed AntMaze protocol, SCS-HRL reached an $88.4 \pm 7.8\%$ final success rate, compared with 0% for the matched scalar-only condition and HIRO. Applying the route rule directly to the same five low-level controllers yielded $82.2 \pm 9.9\%$, while the independently trained direct-rule condition reached $4.8 \pm 10.7\%$.

The results provide evidence that continuous planning-level targets can improve high-level subgoal learning in the tested source maze. They also show the present boundary of the approach: a complete policy trained with a fixed route table supervisor does not transfer zero-shot to new layouts. An earlier single-run AntMaze Extended diagnostic indicates that reusing source-layout low-level parameters may accelerate target-layout retraining, but replicated evidence is still required. Extending corrective supervision to map-conditioned, learned, sparse, and human-provided targets is the next step toward broader navigation and robotic applications.

Author Contributions: All authors contributed to the study conception and design. Material preparation, data collection, and analysis were performed by L.S. and Y.W. The first draft of the manuscript was written by L.S. and Y.W. All authors reviewed and commented on previous versions of the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Nanjing Major Science and Technology Special Project, grant number 202405017.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The datasets generated and analyzed during the current study and the code supporting the findings are available from the corresponding author upon reasonable request.

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

References

1. Pateria, S.; Subagdja, B.; Tan, A.H.; Quek, C. Hierarchical reinforcement learning: A comprehensive survey. *ACM Comput. Surv.* **2021**, *54*, 1–35. [CrossRef]
2. Levine, S.; Finn, C.; Darrell, T.; Abbeel, P. End-to-end training of deep visuomotor policies. *J. Mach. Learn. Res.* **2016**, *17*, 1334–1373.
3. 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. [CrossRef] [PubMed]
4. Levine, S.; Pastor, P.; Krizhevsky, A.; Ibarz, J.; Quillen, D. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *Int. J. Robot. Res.* **2018**, *37*, 421–436. [CrossRef]
5. Gao, J.; Pang, X.; Liu, Q.; Li, Y. Hierarchical Reinforcement Learning for Safe Mapless Navigation with Congestion Estimation. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*; IEEE: New York, NY, USA , 2025. [CrossRef]
6. Meng, J.; Zou, J.; Wang, S.; Yang, R.; Kumar, A.; Kim, J. Deep Reinforcement Learning for Robust Robot Navigation in Complex and Crowded Environments. *J. King Saud. Univ. Comput. Inf. Sci.* **2025**, *37*, 333. [CrossRef]
7. Singh, A.; Yang, L.; Finn, C.; Levine, S. End-to-end robotic reinforcement learning without reward engineering. In Proceedings of the Robotics: Science and Systems XV, Freiburg im Breisgau, Germany, 22–26 June 2019.
8. Ibarz, B.; Leike, J.; Pohlen, T.; Irving, G.; Legg, S.; Amodei, D. Reward learning from human preferences and demonstrations in atari. *Proc. Adv. Neural Inf. Process. Syst.* **2018**, *31*, 8011–8023.
9. Dietterich, T.G. Hierarchical reinforcement learning with the maxq value function decomposition. *J. Artif. Intell. Res.* **2000**, *13*, 227–303. [CrossRef]
10. Vezhnevets, A.S.; Osindero, S.; Schaul, T.; Heess, N.; Jaderberg, M.; Silver, D.; Kavukcuoglu, K. Feudal networks for hierarchical reinforcement learning. In Proceedings of the International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; pp. 3540–3549.

11. Hwang, J.; Lee, S.; Kim, J.; Han, S. Strict Subgoal Execution: Reliable Long-Horizon Planning in Hierarchical Reinforcement Learning. In Proceedings of the 14th International Conference on Learning Representations (ICLR), Rio de Janeiro, Brazil, 23–27 April 2026.
12. Ma, H.; Vo, T.V.; Leong, T.Y. Mixed-Initiative Bayesian Sub-Goal Optimization in Hierarchical Reinforcement Learning. In Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems, Auckland, New Zealand, 6–10 May 2024; pp. 1328–1336.
13. Singh, U.; Suttle, W.A.; Sadler, B.M.; Namboodiri, V.P.; Bedi, A. PIPER: Primitive-Informed Preference-based Hierarchical Reinforcement Learning via Hindsight Relabeling. In Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria, 21–27 July 2024; Volume 235, pp. 45681–45695.
14. Christiano, P.F.; Leike, J.; Brown, T.; Martic, M.; Legg, S.; Amodei, D. Deep reinforcement learning from human preferences. In Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), Long Beach, CA, USA, 4–9 December 2017; Volume 30.
15. Kaufmann, T.; Weng, P.; Bengs, V.; Hüllermeier, E. A survey of reinforcement learning from human feedback. *arXiv* **2025**, arXiv:2312.14925. [CrossRef]
16. Anand, Y.; Nwagwu, N.; Sabbe, K.; Fitter, N.T.; Saisubramanian, S. Adaptive Querying for Reward Learning from Human Feedback. *Front. Robot. AI* **2026**, *12*, 1734564. [CrossRef] [PubMed]
17. Metcalf, K.; Sarabia, M.; Fedzechkina, M.; Theobald, B.J. Can You Rely on Synthetic Labellers in Preference-Based Reinforcement Learning? It Is Complicated. *Proc. AAAI Conf. Artif. Intell.* **2024**, *38*, 10128–10136. [CrossRef]
18. Bougie, N.; Ichise, R. Hierarchical learning from human preferences and curiosity. *Appl. Intell.* **2022**, *52*, 7459–7479. [CrossRef]
19. Zhou, X.; Yuan, Y.; Yang, S.; Hao, J. MENTOR: Guiding Hierarchical Reinforcement Learning with Human Feedback and Dynamic Distance Constraint. *IEEE Trans. Emerg. Top. Comput. Intell.* **2025**, *9*, 1292–1306. [CrossRef]
20. Singh, U.; Chakraborty, S.; Suttle, W.A.; Sadler, B.M.; Asher, D.E.; Sahu, A.K.; Shah, M.; Namboodiri, V.P.; Bedi, A.S. Direct Preference Optimization for Primitive-Enabled Hierarchical RL: A Bilevel Approach. *arXiv* **2026**, arXiv:2411.00361 . [CrossRef]
21. Torne Villasevil, M.; Balsells I Pamies, M.; Wang, Z.; Desai, S.; Chen, T.; Agrawal, P.; Gupta, A. Breadcrumbs to the goal: Goal-conditioned exploration from human-in-the-loop feedback. *Proc. Adv. Neural Inf. Process. Syst.* **2023**, *36*, 63222–63258. [CrossRef]
22. Sutton, R.S.; Precup, D.; Singh, S. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artif. Intell.* **1999**, *112*, 181–211. [CrossRef]
23. Bacon, P.L.; Harb, J.; Precup, D. The option-critic architecture. In Proceedings of the AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 4–9 February 2017; Volume 31.
24. Nachum, O.; Gu, S.; Lee, H.; Levine, S. Data-efficient hierarchical reinforcement learning. In Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), Montréal, QC, Canada, 2–8 December 2018; Volume 31.
25. Knox, W.B.; Stone, P. Interactively shaping agents via human reinforcement: The tamer framework. In Proceedings of the Fifth International Conference on Knowledge Capture, Redondo Beach, CA, USA, 1–4 September 2009; pp. 9–16.
26. Warnell, G.; Waytowich, N.; Lawhern, V.; Stone, P. Deep tamer: Interactive agent shaping in high-dimensional state spaces. In Proceedings of the AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; Volume 32.
27. MacGlashan, J.; Ho, M.K.; Loftin, R.; Peng, B.; Wang, G.; Roberts, D.L.; Taylor, M.E.; Littman, M.L. Interactive learning from policy-dependent human feedback. In Proceedings of the International Conference on Machine Learning (ICML), Sydney, Australia, 6–11 August 2017; pp. 2285–2294.
28. Arumugam, D.; Lee, J.K.; Saskin, S.; Littman, M.L. Deep reinforcement learning from policy-dependent human feedback. *arXiv* **2019**, arXiv:1902.04257.
29. Lee, K.; Smith, L.; Abbeel, P. PEBBLE: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training. In Proceedings of the 38th International Conference on Machine Learning, Virtual Event, 14–24 July 2021; Volume 139, pp. 6152–6163.
30. Brown, D.S.; Goo, W.; Nagarajan, P.; Niekum, S. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations. In Proceedings of the International Conference on Machine Learning (ICML), Long Beach, CA, USA, 9–15 June 2019; pp. 783–792.
31. Brown, D.S.; Goo, W.; Niekum, S. Better-than-demonstrator imitation learning via automatically-ranked demonstrations. In Proceedings of the Conference on Robot Learning (CoRL), Cambridge, MA, USA, 16–18 November 2020; pp. 330–359.
32. Yan, T.; Lipton, Z.C. A Theoretical Case-Study of Scalable Oversight in Hierarchical Reinforcement Learning. *Proc. Adv. Neural Inf. Process. Syst.* **2024**, *37*, 27295–27339. [CrossRef]

33. 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*; IEEE: New York, NY, USA, 2012; pp. 5026–5033.
34. Brockman, G.; Cheung, V.; Pettersson, L.; Schneider, J.; Schulman, J.; Tang, J.; Zaremba, W. OpenAI Gym. *arXiv* **2016**, arXiv:1606.01540.

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.