CAAI Transactions on Intelligence Technology

WILEY

The Institution of Engineering and Technology

ORIGINAL RESEARCH OPEN ACCESS

Exploration and Exploitation: A Study on Sample Efficiency in Reinforcement Learning With Multifaceted Curiosity Rewards and Adaptive Experience Replay Utilisation in Sparse Reward Environments

Jingyi Huang[1] | Guipeng Xi[1] | Meixiu Lin[1] | Haohui Zhang[2] | Bo Li[1] | Evgeny Neretin[3]

[1]School of Electronics and Information, Northwestern Polytechnical University, Xi'an, China | [2]Beijing Institute of Spacecraft System Engineering, Beijing, China | [3]Moscow Aviation Institute, Moscow, Russia

Correspondence: Bo Li (libo803@nwpu.edu.cn)

Received: 11 December 2025 | **Revised:** 27 April 2026 | **Accepted:** 21 May 2026

Keywords: adaptive experience replay | curiosity reward | exploration-exploitation trade-off | mixed exploration strategy | reinforcement learning sample efficiency

ABSTRACT

In recent years, the widespread application of deep reinforcement learning (DRL) in autonomous systems has highlighted the importance of achieving high sample efficiency under sparse reward conditions. To improve sample efficiency in sparse reward environments, this paper proposes a reinforcement learning framework built upon the Soft Actor Critic architecture, which integrates multifaceted curiosity rewards (MCR) and adaptive experience replay utilisation (AERU) (MCR-AERU SAC). MCR combines multi-level intrinsic motivational signals, such as state prediction error and model uncertainty, to provide rich exploration incentives, encouraging the controlled entity to deviate from existing trajectories and discover new high-reward behaviours. AERU dynamically adjusts the experience replay priorities based on posterior temporal difference error (TD error), focusing on utilising 'partially successful' transitions that are easily overlooked. The synergy between MCR and AERU enables the proposed framework to achieve an optimal balance between exploration and exploitation, significantly accelerating policy convergence and improving sample utilisation. Extensive experiments in complex dynamic training environments demonstrate that the proposed MCR-AERU SAC algorithm achieves up to 1.41 times the early-stage gain rate and a 55.56% improvement in task success rate compared to the HER-SAC baseline, demonstrating superior sample efficiency and excellent robustness in large-scale sparse reward environments.

1 | Introduction

With technological advancements, artificial intelligence (AI) has rapidly grown [1], becoming a powerful tool for solving complex problems that traditional methods struggle with [2–5]. Deep reinforcement learning (DRL) [6] combines reinforcement learning's optimal decision-making [7] with deep learning's ability to represent high-dimensional data, enabling algorithms to learn optimal value functions or policies through interactive, sample-based training. It integrates perceptual and cognitive intelligence, achieving human-comparable performance in areas like UAV applications [8], robotics [9], natural language processing [10] and finance [11]. However, in sparse reward environments with random noise and other challenges, DRL struggles to sample valuable state-action pairs, resulting in

This is an open access article under the terms of the Creative Commons Attribution-NonCommercial License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited and is not used for commercial purposes.

© 2026 The Author(s). *CAAI Transactions on Intelligence Technology* published by John Wiley & Sons Ltd on behalf of The Institution of Engineering and Technology and Chongqing University of Technology.

inefficient training and poor policy learning [12]. In real-world applications, sparse or nonexistent reward signals make it hard for the learning policy to collect meaningful feedback, especially when starting with limited environmental knowledge [13, 14].

To address sample inefficiency and exploration challenges in sparse and delayed reward environments, various goal-based exploration methods and imitation learning approaches have been proposed [15]. For example, Wu [16] introduced Goal Exploration Augmentation via Pre-trained Skills (GEAPS), which combines pretrained skills with raw actions to improve exploration in long-horizon tasks by maximising entropy. Zhang [17] developed the GFA-RFE algorithm, which uses intrinsic rewards and a weighted learning mechanism for exploration in the absence of external rewards. Albaba [18] proposed a framework that combines imitation learning with inverse RL (RILE), allowing the algorithm to converge to near-expert levels with fewer demonstrations. However, these methods often have limited impact on task performance and increase data acquisition costs.

Recently, intrinsic motivation from psychology has been applied to DRL reward design to address exploration issues [19]. In sparse reward environments, intrinsic motivation provides the RL policy with feedback through intrinsic rewards like prediction errors and information gain [20]. When incorporated into value iteration or policy gradient algorithms, intrinsic rewards can reduce sample complexity for convergence [21]. These rewards, based on state representation features rather than task-specific signals, adapt well to environmental noise and changes. When the learnt state representation is stable and hierarchical, intrinsic rewards can provide reliable incentives for unobserved states, reducing the need for re-exploration in new scenarios [22]. Raileanu [23] proposed 'empowerment' as an intrinsic reward to encourage actions that alter state representations, outperforming random algorithms in tasks like MiniGrid. Li [24] combined intrinsic rewards with external goals to reduce reliance on external feedback. However, a single intrinsic reward may not meet task needs in sparse reward settings, leading to instability and excessive exploration.

To address sample inefficiency in sparse reward environments, the hindsight experience replay (HER) algorithm [25] has been proposed. HER replaces the goal with the state actually achieved during the episode, turning previously invalid experiences into valuable learning opportunities, thus increasing positive samples and improving sample utilisation. This also stabilises the value function gradient by ensuring that more transitions contribute meaningful estimates, avoiding degradation from zero temporal difference error (TD error) values. Unlike exploration strategies requiring complex environment modelling or reward engineering (e.g., imitation learning [26], inverse RL [27], and model-predictive exploration [28]), HER relies on offline relabelling, making it more efficient and widely applicable. HER-based methods, such as RIG (RL with imagined goals) [29] and CHER (curriculum hindsight experience replay) [30], help the policy identify actions leading to goal achievement by setting the final state as the goal, improving causal chain tracing. However, if the gap between HER-generated goals and actual task goals is too large, learning from irrelevant experiences leads to low sample efficiency.

To overcome the aforementioned limitations—namely, the instability of single-metric intrinsic rewards and the inefficiency of poorly targeted experience replay—this paper proposes a novel synergistic framework that firmly bridges exploration and exploitation. While numerous intrinsic reward functions have been discussed in existing literature [19–24], they predominantly rely on singular metrics (e.g., pure prediction error or pure empowerment), rendering them vulnerable to environmental stochasticity (such as the 'Noisy-TV' problem). To fundamentally differentiate our work from these prior arts, we introduce a coupled architecture: multifaceted curiosity rewards (MCR) for robust, directed exploration and adaptive experience replay utilization (AERU) for targeted exploitation of the discovered rare transitions. By coupling multifaceted exploration with adaptive exploitation, our approach forms a complete, highly sample-efficient closed loop, significantly distinguishing itself from conventional methods that address only one side of the sparse reward challenge.

Specifically, we integrate these coupled innovations into the Soft Actor Critic (SAC) [31] backbone, proposing the MCR-AERU SAC algorithm. The main contributions of this paper are summarised as follows:

1. We propose an MCR exploration strategy: MCR provides diverse exploration incentives by structurally integrating state prediction error, spatial visitation counts and policy uncertainty. Furthermore, we employ a policy curriculum learning mechanism to dynamically adjust the ratio between intrinsic and extrinsic rewards. This formulation establishes a comprehensive exploration strategy, empowering the policy to conduct multidimensional exploration and effectively circumventing the limitations of single-metric intrinsic rewards.
2. We introduce the AERU mechanism: AERU dynamically adjusts the priority of experience replay based on post hoc TD error, focusing on utilising those 'partially successful' transitions that are typically overlooked. It assigns new learning value to previously ineffective experiences, resolving the issue of information disparity among samples. This ensures that the policy focuses on valuable learning signals, thereby significantly improving the algorithm's data utilisation efficiency [32].
3. We develop a unified training framework combining MCR-AERU and SAC: The synergistic effect of MCR (targeted exploration) and AERU (adaptive exploitation) allows the algorithm to achieve a superior exploration–exploitation balance [33]. This integrated framework not only accelerates policy convergence but also fundamentally strengthens the capability of RL methods to solve complex problems in severe sparse reward environments.

2 | Problem Description

2.1 | Complex Dynamic Training Environment

To verify the effectiveness of the algorithm under controlled conditions, we use the maze environment [34] as the task environment, focusing on two scenarios: a 3×3 grid (hereafter referred to as Scenario 1) and a 5×5 grid (Scenario 2), as shown in Figure 1.

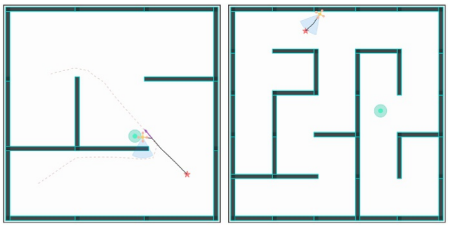


FIGURE 1 | The 3 × 3 maze (Scenario 1) and the 5 × 5 maze (Scenario 2). Visualisation legend: orange controlled entity (current position), red star (starting point), green double circle (target), short black line (trajectory), orange dashed line (trajectory of the previous episode) and thick black-green line (walls/obstacles).

In both environments, the goal is to find the optimal movement strategy that enables the controlled entity to navigate from the starting point to the target along a collision-free path, with rewards given only when the controlled entity reaches the target. The objective of this work is to accelerate the learning process in these two environments by leveraging intrinsic compound curiosity and adaptive experience replay. Complex task spaces often involve multiple controllable or interfering factors, which exhibit structured characteristics. Both scenarios in this study are created using a depth-first search algorithm, with random starting and target positions generated within the continuous workspace. In this environment, the algorithm faces challenges as outlined in Table 1.

2.2 | Markov Decision Process

In this work, we define the problem as an MDP, using the same formalism as Strudel [34]. The state space S encompasses all possible configurations of the environment C, and O_{Env} denotes the observable information obtained from the environment, which includes both environmental information and goal g information. In this work, the maze simulation yields an observation vector O_{Env} of dimensionality 101 for Scenario 1 and 245 for Scenario 2. O_{Env} can be used to indirectly infer the underlying state S. The action space A consists of the effective velocity actions, with the algorithm outputting continuous velocities along the x and y axes within the range of -1 to 1 during the simulation process.

Given an environment configuration $q \in C_{\text{free}}$ (representing the current state) and a velocity vector v, we define $q(v)$ as the expected state achieved by applying v to the controlled entity over a fixed time period, assuming no obstacles. Because the system may collide with obstacles, we introduce $q_{\text{free}}(v)$, a function that returns the last collision-free state along the path from q to $q(v)$. The actual resulting state p is defined as $p(q, v) = q_{\text{free}}(v)$.

The reward function is defined as follows. Given a physical target position (denoted mathematically as the goal $g \in \mathcal{C}$), we define $r_{\text{velocity}}(q, v) = -\|v\|_2$ and $r_{\text{task}}(q, v, g)$. Furthermore, let $\varepsilon_{\text{goal}}$ denote the distance threshold: The controlled entity is considered to have reached the target when its current position is within $\varepsilon_{\text{goal}}$ of the target.

$$r_{\text{task}}(q, v, g) = \begin{cases} r_{\text{goal}} & \text{if } \left\| q_{\text{free}}(v) - g \right\| \le \varepsilon_{\text{goal}}, \\ r_{\text{free}} & \text{else if } [q, q(v)] \subset C_{\text{free}}, \\ r_{\text{collision}} & \text{else.} \end{cases} \quad . \tag{1}$$

When $r_{\text{goal}} > 0$, $r_{\text{free}} < 0$ and $r_{\text{collision}} < 0$ are considered, the reward function is defined as $r(q, v, g) = r_{\text{velocity}}(q, v) + r_{\text{task}}(q, v, g)$, which serves as the primary external reward r_{ext} for the learnt policy optimisation in later sections. The r_{task} reward corresponds to the action of reaching the goal g and penalises actions that cause collisions. Given two collision-free paths leading to the goal, the shortest path will accumulate the most reward $r_{\text{task}}(q, v, g)$. Maximising this reward encourages the policy to avoid collisions and achieve the shortest possible path. The reward r_{task} defined above is sparse with respect to the goal, as positive feedback is only received when the controlled entity reaches the target during a training episode, which may occur with low probability in challenging environments.

3 | Method

3.1 | Adaptive Experience Replay Utilisation Mechanism

3.1.1 | Hindsight Experience Replay-Based Soft Actor Critic Algorithm

In this work, we base our approach on the precise SAC + HER algorithm used by Strudel [34] as the foundational algorithm for sparse reward environments. Specifically, Soft Actor Critic (SAC) is an off-policy algorithm built upon the maximum entropy reinforcement learning framework. Unlike standard RL, which solely maximises the expected sum of rewards, SAC optimises an augmented objective function that jointly maximises both the expected return and the policy's entropy:

$$J(\pi) = \mathrm{E}_{\tau \sim \pi}\left[\sum_{t=0}^{\infty} \gamma^t (r(s_t, a_t) + \alpha H(\pi(\cdot|s_t))) \right], \tag{2}$$

where γ is the discount factor, $H(\pi(\cdot|s)) = -\mathrm{E}_{a \sim \pi}[\log \pi(a|s)]$ represents the stochastic policy's entropy and the temperature parameter α determines the relative importance of the entropy term, thereby naturally encouraging exploration to prevent premature convergence to local optima. Under this Actor Critic framework, the policy π and the soft Q-function are parameterised as independent networks and trained via alternating optimisation.

However, while the inherent entropy maximisation of SAC provides fundamental exploration, it remains insufficient for environments with severe reward sparsity. To overcome this limitation, HER [25] is integrated into the baseline. This technique inherently empowers off-policy RL algorithms to extract valuable learning signals from failed trajectories in goal-conditioned tasks, thereby vastly improving sample efficiency. After an episode consisting of a transition sequence $(s_0, a_0, s_1, a_1, ..., s_T)$, the controlled entity might have reached the goal g. For the i-th transition (s_i, a_i, s_{i+1}) in the sequence, a future time step $k \in \{i + 1, ..., T\}$ is randomly sampled, and the goal is redefined as $g^* = \phi_{\text{goal}}(s_k)$, where $\phi_{\text{goal}}(\cdot)$ represents the 'actually achieved goal' extracted from the original observation, such as the learnt position in a path planning task. In this process, the unreached physical target (desired goal g) is replaced by the actually achieved goal $g*$, and the transition's reward and goal conditions are recalculated accordingly.

TABLE 1 | Challenges and key algorithm requirements in maze environment.

Challenge types	Specific manifestations	Algorithm key requirements
Repeated path exploration	In complex mazes, the policy must explore more areas and frequently backtrack along the same corridors. Only after finding the optimal path, it can successfully reach the target.	Efficient exploration capability and state access counting mechanism
Long-term delays	Successfully navigating this environment requires the algorithm to maintain a representation of key junction locations.	Hierarchical decision-making capability
Sparse rewards	The environment has sparse rewards, with positive feedback only provided when the controlled entity reaches the exit. This sparsity of rewards requires the actor to explore a large portion of the space before receiving feedback, making the training process slower and more complex [35, 36].	Effectively rewarding signal processing and exploration incentives
Complex action and observation space	The continuous action space and high-dimensional state representation present greater challenges for policy during learning, particularly when exploration is insufficient [37].	Effective path planning and state space search
Scene dynamics	After each epoch, the scene is dynamically generated with random starting and target positions. Obstacle placement is randomly determined based on simulation arena size, number of obstacles and the distance between the starting point and target points.	Algorithm generalisation capability
Uncertainty in sample validation	The environment generates valid starting and target positions that satisfy specific distance constraints. Should the start target configuration fail to meet the conditions, new positions are resampled. This makes it difficult for the actor to find the optimal path, particularly under sparse reward conditions where the policy may become trapped in suboptimal states, thereby delaying the learning process [38, 39].	Adaptive exploration–exploitation balance to avoid getting stuck in suboptimal strategies.

3.1.2 | *Adaptive Experience Replay Utilisation*

To enhance sample utilisation efficiency and mitigate the limitations of HER, we propose an adaptive experience replay mechanism (AERU), which introduces a prioritised experience replay (PER) [40] technique into the HER-SAC experience replay buffer. PER sets the sampling probability $p_i \propto \left|\delta_i^{\text{her}}\right|^{\alpha_{\text{per}}}$ for each transition based on the TD error δ of the transition.

In HER-SAC experience replay buffer, the TD error δ for the i-th transition $\left(s_i^{\text{her}}, a_i, r_i^{\text{her}}, s_{i+1}^{\text{her}}\right)$ is computed as follows.

$$\delta_i^{\text{her}} = r_i^{\text{her}} + \gamma\left[\min_j Q_{\bar{\theta}_j}\left(s_{i+1}^{\text{her}}, \hat{a}\right) - \alpha \log \pi_\phi\left(\hat{a}|s_{i+1}^{\text{her}}\right)\right] - Q_{\theta_k}\left(s_i^{\text{her}}, a_i\right) \tag{3}$$

The action $\hat{a} \sim \pi_\phi\left(\cdot|s_{i+1}^{\text{her}}\right)$ is sampled from the current policy at the next state s_{i+1}^{her}, $Q_{\bar{\theta}_j}$ is the target network and the minimum value from Q_{θ_k}, $k \in \{1, 2\}$, is chosen to reduce overestimation bias. A Q network is selected to compute the current Q value. The priority p_i for the i-th transition is then assigned and computed as follows:

$$p_i = \left(\left|\delta_i^{\text{her}}\right| + \varepsilon_{\text{per}}\right)^{\alpha_{\text{per}}}, \tag{4}$$

where $\varepsilon_{\text{per}} > 0$ is a small constant to avoid a priority of 0 when $\delta_i^{\text{her}} = 0$, and $\alpha_{\text{per}} \in [0, 1]$ controls the 'sharpness' of the priority. When $\alpha_{\text{per}} = 0$, it reverts to uniform sampling. All transition priorities are normalised to derive the sampling probability $P(i)$, where transitions with higher priority are more likely to be sampled.

$$P(i) = \frac{p_i}{\sum_j p_j}. \tag{5}$$

By prioritising transitions that are most likely to correct the current value function error, the algorithm replays these samples more frequently, which accelerates convergence in the early learning phase, reduces optimality errors more rapidly and improves gradient convergence by lowering variance. Compared to uniform sampling, this mechanism speeds up initial convergence by $O(1/t)$ levels [41].

Furthermore, by multiplying each sampled sample by the importance sampling weight, we correct the bias introduced by

nonuniform sampling. The importance sampling weight w_i is computed as

$$w_i = \left(\frac{1}{N_{\text{num}}} \cdot \frac{1}{P(i)}\right)^{\beta_{\text{per}}} = (N_{\text{num}} \cdot P(i))^{-\beta_{\text{per}}}, \tag{6}$$

where N_{num} is the number of samples in the buffer, and β_{per} increases linearly from 0 to 1 to eliminate bias. PER restores an unbiased estimate of the original distribution and ensures convergence to the same limit solution as uniform sampling when β_{per} reaches 1. Each sampling selects a minibatch of transitions according to $P(i)$, and the loss function is weighted by w_i:

$$Loss_{\text{SAC}} = \frac{1}{|B|}\sum_{i\in B} w_i\left(\delta_i^{\text{her}}\right)^2, \tag{7}$$

where B represents the minibatch of transitions sampled, and $|B|$ denotes the size of this batch.

PER enhances HER by prioritising high-value samples in reward transfer, with HER generating hindsight transitions by relabelling failed trajectory goals as the 'final achieved state', amplifying sparse positive rewards. However, only transitions close to successful paths contribute to Q-network updates. By combining PER with HER, transitions with large TD errors are prioritised, improving the efficiency of utilising 'partially successful' experiences, thus accelerating learning in sparse reward environments. The AERU mechanism combines HER's pseudo-samples with PER's learning focus in three ways: (1) prioritising transitions with high information content, (2) dynamically adjusting the sampling distribution as the policy converges and (3) maintaining sample diversity by balancing priority and uniform sampling.

3.2 | Multifaceted Curiosity Reward Exploration Strategy

3.2.1 | Reward Mathematical Models and Theoretical Guarantees

3.2.1.1 | Intrinsic Curiosity Reward. The Intrinsic Curiosity Module (ICM) [20] simulates the human brain's process of receiving and processing external information. ICM predicts the next state feature $\hat{\vartheta}(s_{t+1})$ based on s_t and a_t, utilising the prediction error to estimate the intrinsic reward.

The encoder consists of two neural networks, which output the feature vectors $\vartheta(s_t)$ and $\vartheta(s_{t+1})$, representing feature extraction and encoding for states s_t and s_{t+1}, respectively. The forward prediction model is a forward dynamics model, where the input is the raw feature vector $\vartheta(s_t)$ and the current action a_t, and the output is the predicted next state $\hat{\vartheta}(s_{t+1})$:

$$\hat{\vartheta}(s_{t+1}) = \Gamma[\vartheta(s_t), a_t|\theta_\Gamma], \tag{8}$$

where $\Gamma[\cdot]$ represents the forward prediction model, and θ_Γ are the parameters of this model's network structure. The corresponding loss function is

$$\text{Loss}_\Gamma\left(\hat{\vartheta}(s_{t+1}), \vartheta(s_t)\right) = \frac{1}{2}\left\|\hat{\vartheta}(s_{t+1}) - \vartheta(s_t)\right\|_2^2. \tag{9}$$

When the controlled entity encounters nearly identical but subtly different situations, such as a corner or dead end, the forward model's error Loss_Γ for a particular state transition tends to spike. This occurs because slight environmental differences lead to changes in the transition encoding $\vartheta(s_t)$ that are difficult to predict accurately with the current knowledge [33]. At this point, the robot considers the transition 'novel', treating the error as an intrinsic reward and significantly increasing the exploration priority for this transition to encourage further exploration of the area. Because the error primarily originates from uncertainties about local feature details, ICM tends to repeatedly attempt action-state transitions with significant environmental differences, thus driving fine-grained exploration in regions where local predictability is poor.

The intrinsic reward r_{ICM} is computed as the difference between the predicted state feature vector $\hat{\vartheta}(s_{t+1})$ and the actual state feature vector $\vartheta(s_{t+1})$ at time $t+1$:

$$r_{\text{ICM}} = \frac{1}{2}\left\|\hat{\vartheta}(s_{t+1}) - \vartheta(s_{t+1})\right\|_2^2. \tag{10}$$

This structure facilitates efficient learning of the intrinsic curiosity and exploration signals in sparse reward environments, improving the learnt ability to explore unknown states [42].

3.2.1.2 | Hash Count Reward. The hash count reward (Hash) uses a hash table to record the number of times a state is visited. States with higher visit counts are assigned lower rewards, encouraging exploration and avoiding misjudgements in similar areas [43]. The hash exploration reward $r_{\text{Hash}} : S \to R$ is added to the reward function, defined as

$$r_{\text{Hash}} = \frac{1}{\sqrt{N_{\text{Hash}}(s)+1}} \tag{11}$$

where $N_{\text{Hash}}(s)$ is the approximate number of state visits through the hash function. In continuous space, directly calculating $N_{\text{Hash}}(s)$ is infeasible, so the hash reward discretizes the state space using the hash function $M_{\text{Hash}} : S \to Z$, approximating it through the state density function $\hat{N}_{\text{Hash}}(s)$:

$$\hat{N}_{\text{Hash}}(s) = \sum_{i=1}^{t} \text{I}[\text{Hash}(s_i) = \text{Hash}(s)]. \tag{12}$$

The hash function employs a cryptographic hashing algorithm to encode the high-dimensional continuous state into a fixed-length discrete hash value. Specifically, we utilise the SHA-256 algorithm to map the quantised state representations, taking the modulus of the hash table size H_{size} to constrain the keys. The number of times each hash value is visited is recorded, with fewer visits yielding higher intrinsic rewards:

$$Hash(s) = \text{SHA}-256(s) \bmod H_{\text{size}}. \tag{13}$$

Initially, the visitation count $\hat{N}_{\text{Hash}}(\cdot)$ for all possible states within the state space is effectively set to 0. At any time step t, the count $\hat{N}_{\text{Hash}}(s_t)$ for the mapped state s_t increases by 1.

By mapping these 'approximately identical' states to the same bucket in the hash count, $\hat{N}_{\text{Hash}}(s)$ increases, $1\Big/\sqrt{\hat{N}_{\text{Hash}}(s)+1}$

will decay rapidly, reducing the exploration incentive for that area and guiding the policy to explore less-visited and unexplored regions within the scene [44].

3.2.1.3 | *Empowerment Reward.* In RL, empowerment (Emp) reward is typically calculated by maximising mutual information to measure the reduction in state uncertainty caused by the learnt actions [45]. Let the latent representation of the i-th sample in the state sequence S_{obs} be denoted as $z_t^i = f_\chi(s_t^i)$ and $z_{t+1}^j = f_\chi\left(s_{t+1}^j\right)$, with the goal of maximising L_{ij} [46], which is the mutual information between the current state and the next state.

$$L_{ij} = f_\psi\left(z_t^{(i)}, z_{t+1}^{(j)}\right), i, j = 1, ..., |S_{\text{obs}}|, \tag{14}$$

where f_ψ is the prediction scoring network. We calculate the cross-entropy loss utilising the Information Noise-Contrastive Estimation (InfoNCE) algorithm [47]. Specifically, we treat the diagonal scores L_{ii} as the outputs for positive samples and the off-diagonal scores $L_{ij}(j \neq i)$ as the outputs for negative samples. The loss $Loss_{\text{Emp}}$ is defined as

$$Loss_{\text{Emp}} = -\frac{1}{|S_{\text{obs}}|}\sum_{i=1}^{|S_{\text{obs}}|}\log\frac{\exp(L_{ii})}{\sum_{j=1}^{|S_{\text{obs}}|}\exp(L_{ij})}. \tag{15}$$

Maximising the likelihood of positive sample pairs while suppressing negative ones, information gain rewards enable the policy to be guided without the need for an environment model. By distinguishing between real transitions and random noise, the mutual information lower bound is maximised, driving the policy to perform effective exploration. The information gain reward is denoted as r_{Emp}.

$$r_{\text{Emp}} = \frac{1}{2}\tanh\left(\frac{L_{ii} - \mu}{\sqrt{\sigma^2 + \varepsilon_{\text{Emp}}}}\right) + 1, \tag{16}$$

where μ is the running mean, updated with exponential decay, where after each batch, $\mu \leftarrow \tau\mu + (1-\tau)\cdot\text{mean}(L_{ii})$, σ^2 is the running variance, normalised for different training stages, with $\sigma^2 \leftarrow \tau\sigma^2 + (1-\tau)\cdot\text{var}(L_{ii})$, τ is the momentum, and ε_{Emp} is a small constant to prevent division by 0. The information gain reward drives the policy to actively explore unknown states with potential information.

3.2.2 | *Multifaceted Exploration Strategy*

Hash focuses on the coverage of 'macro regions' quickly suppressing redundant exploration of the same region, thus avoiding misjudging similar areas like corridors in the task environment. ICM focuses on 'micro-detail' differences and responds rapidly to directional choices. Together, they ensure both breadth (global new areas) and depth (detailed corners) in exploration. Emp focuses on detecting path branching points, which complements the modelling of dynamic uncertainties, thus forming a multilayer, mixed exploration signal, as shown in Table 2.

The multi-faceted curiosity reward is defined as r_{int}, where the scaling hyperparameters $(\kappa_{\text{ICM}}, \kappa_{\text{Emp}}, \kappa_{\text{Hash}}) \in \mathrm{R}^3_{\geq 0}$ are designed to balance the respective contributions of each exploration metric, subject to $\kappa_{\text{ICM}} + \kappa_{\text{Emp}} + \kappa_{\text{Hash}} = 1$.

$$r_{\text{int}} = \kappa_{\text{ICM}}\cdot r_{\text{ICM}} + \kappa_{\text{Emp}}\cdot r_{\text{Emp}} + \kappa_{\text{Hash}}\cdot r_{\text{Hash}}. \tag{17}$$

3.2.3 | *Adaptive Internal and External Reward Adjustment*

In this section, we introduce a strategy curriculum learning (SCL) [48] implementation that adaptively adjusts the balance between intrinsic and extrinsic rewards during network updates. The specific steps are as follows:

1. When the external reward continuously increases, the weight of the intrinsic curiosity reward is decreased.
2. When the external reward enters a plateau phase, the intrinsic curiosity exploration incentives are automatically increased.
3. A minimum of 10% intrinsic curiosity weight is maintained to prevent stagnation in local optima.

The policy curriculum reward coefficients, represented by the time-dependent weight functions $W_{\text{ext}}(t)$ and $W_{\text{int}}(t)$ at training step t, are dynamically adjusted according to the aforementioned curriculum strategy. The total reward r_{total} is defined as follows:

$$r_{\text{total}} = W_{\text{ext}}(t)\cdot r_{\text{ext}} + W_{\text{int}}(t)\cdot r_{\text{int}}, \tag{18}$$

where r_{ext} represents the external reward provided by the environment based on the learnt actions during the interaction (i.e., corresponding to the task-specific reward function $r(q, v, g)$ defined in Section 2.2), and r_{int} denotes the intrinsic curiosity reward generated by the MCR module. The target optimisation function $J(\pi)$ is

$$J(\pi) = \mathrm{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t(r_{\text{total}}(s_t, a_t) + \alpha H(\pi(\cdot|s_t)))\right]. \tag{19}$$

3.3 | Implementation Details

3.3.1 | *Algorithm Flow*

The algorithm implementation pseudocode is shown in Table 3.

For the MCR-AERU SAC algorithm, the primary time and space complexity components are analysed as follows. The data collection step, E_{coll}, updated E_{coll}/U times (execute once every U steps), and the total time complexity TC are as follows:

$$\text{TC} = O\Big(E_{\text{coll}}\cdot M_{\text{net}} + \frac{E_{\text{coll}}}{U}\big(|B|^2\cdot l + |B|\cdot M_{\text{net}} + |B| \cdot \log N_{\text{her}}\big)\Big). \tag{20}$$

The total space complexity SC is

$$\text{SC} = O\big(N_{\text{SAC}}\cdot D_O + N_{\text{her}} + M_{\text{net}} + |B|^2\big), \tag{21}$$

where N_{SAC} is the number of samples in the replay buffer, N_{her} is the number of samples from HER, D_O is the dimension of O_{Env}, l is the dimension of the latent layer and M_{net} is the parameter size of the SAC network's actor/critic network (which primarily determines the forward/backward time). In the

TABLE 2 | Analysis of multifaceted exploration strategies.

Reward type	Bias	Proficient scenario	Computational overhead
ICM	Emphasise model predictability	Local fine-grained exploration	Medium
Hash	Emphasise state visit frequency	Avoiding repetitive visits	Low
Emp	Emphasise the enhancement of environmental knowledge	Discovering high-control regions	High

TABLE 3 | MCR-AERU SAC algorithm pseudocode.

Algorithm 1: MCR-AERU SAC Algorithm

Input: Environment *Env*; SAC networks (Policy π_ϕ, Q-networks θ_{Q1}, θ_{Q2}); MCR modules (ICM, Hash, Emp); Replay buffer $\boldsymbol{B}_{\text{replay}}$ and AERU buffer $\boldsymbol{H}_{\text{AERU}}$.

Initialize: Network parameters, target network weights, and curriculum weights $W_{\text{ext}}(t), W_{\text{int}}(t)$.

for each training iteration **do**

//Phase 1: Environment Interaction & MCR Computation

for each environment step **do**

Sample action $a_t \sim \pi_\phi(a|s_t)$ and execute it in *Env*.

Observe next state s_{t+1}, external reward r_{ext}, and *done*.

Compute raw intrinsic signals $(r_{\text{ICM}}, r_{\text{Hash}}, r_{\text{Emp}})$ using current state transitions via Equations (10), (11), (16).

Calculate the composite MCR r_{int} via Equation (17).

Store transition $(s_t, a_t, r_{\text{ext}}, r_{\text{int}}, s_{t+1}, done)$ into buffers $\boldsymbol{B}_{\text{replay}}$ and $\boldsymbol{H}_{\text{AERU}}$.

if *done* then reset environment.

end for

//Phase 2: Adaptive Experience Replay & Network Updates

for each gradient step **do**

Sample prioritized batch $B_{\text{batch_per}}$ from $\boldsymbol{H}_{\text{AERU}}$ based on priorities p_i via Equation (4).

Sample uniform batch $B_{\text{batch_SAC}}$ from $\boldsymbol{B}_{\text{replay}}$, and merge them to form mixed batch B.

Calculate total reward r_{total} for batch B using dynamic curriculum weights $W_{\text{ext}}(t), W_{\text{int}}(t)$.

Update SAC Q-networks, Policy network, and temperature by minimizing their respective losses using batch B.

Update target Q-networks with soft momentum update.

Update AERU transition priorities p_i with freshly computed TD-errors.

Update intrinsic module networks (ICM and Emp) using gradient descent.

end for

//Phase 3: Curriculum Adjustment & Evaluation

if curriculum milestone reached **then**

Dynamically adjust $W_{\text{ext}}(t), W_{\text{int}}(t)$ according to the curriculum schedule.

end if

Evaluate the deterministic SAC policy (without exploration noise) and record metrics.

end for

Output: Optimized policy parameters ϕ and Q-network parameters θ.

algorithm's simulation training, we control the batch size to manage the $|B|^2$ cost.

Figure 2 summarises the MCR-AERU SAC training loop. During data collection, the controlled entity samples actions from the current policy, interacts with the environment, computes intrinsic rewards (ICM, Hash and Emp) for each transition and stores transitions (with intrinsic rewards) in both the standard and AERU replay pools. During updates, experiences are sampled via AERU, mixed and weighted according to the current curriculum; the combined batch reward (extrinsic + intrinsic) is used to update the SAC Q-functions, policy and temperature. PER priorities are refreshed from the resulting TD errors, and the ICM and Emp networks are trained.

3.3.2 | *Implementation of Task Fusion Reward*

Using the observation vector O_{Env} returned by the environment, we design the intrinsic reward model that integrates the task. Specifically, the first n^2 dimensions of O_{Env} (the grid view, which is 8 and 16 for Scenario 1 and Scenario 2, respectively) are considered the grid part, and the remaining dimensions correspond to the extra state variables (extra part). The reward design varies as follows:

As shown in Figure 3a, by treating the grid view as a single-channel image and using a small CNN [49] to extract spatial features, the ICM model naturally captures spatial adjacency information. The remaining features, including the current position, goal vector and other state vectors, are processed using an MLP network. These two sets of features are concatenated and input into the forward and inverse model branches to compute rewards. As shown in Figure 3b, the learnt coordinates (x, y) are extracted from the O_{Env} grid view and used as the key to map the position to the grid cells. The hash reward is then computed based on the number of visits to each grid cell, without the need for random projection or full observation hashing. This approach avoids hash collision issues and reduces hash computation. The additional state is quantised and then encoded using SHA-256 hashing. Both keys are merged to ensure sensitivity to both the grid and other state information. As shown in Figure 3c, we encode states using CNN/MLP networks, construct a $|B| \times |B|$ contrastive matrix over the batch to compute a cross-entropy loss and derive the Emp reward from the matrix's diagonal logits.

3.3.3 | *Experience Pool Design*

Pan [41] pointed out that while PER accelerates convergence early on, it also faces issues such as 'outdated priorities' and 'insufficient coverage', which is similar to the risk of HER over-relying on a few 'successful' trajectories in later stages. In this work, we combine PER's priority update mechanism (periodic recalculation of TD error) with HER's goal sampling strategy (multi-goal subsampling) to ensure high-frequency replay of key transitions while ensuring sufficient exploration of all goal-state pairs. This approach theoretically balances sample efficiency and guarantees strategy convergence. The combination of HER's goal relabelling and PER's prioritised sampling achieves a balance between 'experience reuse' and 'exploration', as illustrated by the adaptive experience pool sampling process in Figure 4.

Robustness in dynamic environments: It is important to address how our historical data–dependent mechanisms (MCR and AERU) remain robust in environments with dynamic obstacles. A common concern in dynamic settings is that prior visitation counts (hash) or offline transitions (AERU) may become obsolete, leading to nonstationary learning. However, in our

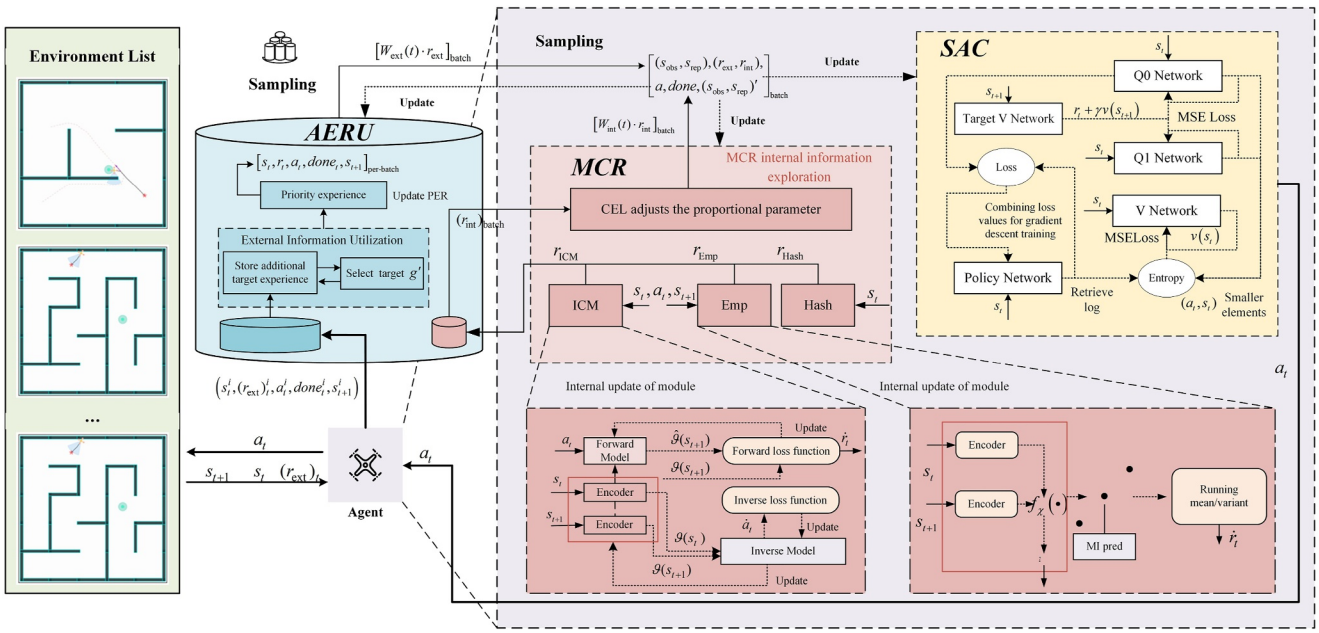


FIGURE 2 | MCR-AERU SAC algorithm training and update process.

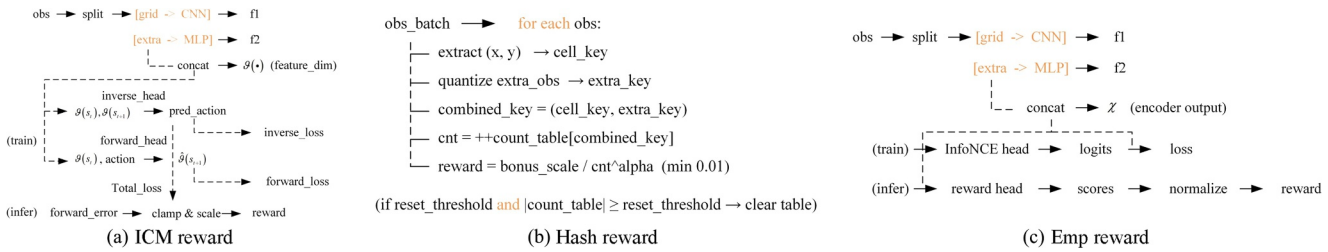


FIGURE 3 | The information flow diagram of the MCR reward. (a) ICM reward; (b) hash reward; (c) emp reward.

framework, this conflict is fundamentally resolved through the state representation. Because our state space s_t inherently incorporates real-time obstacle representations (encoded via the PointNet architecture), any movement of obstacles strictly generates a new, distinct state representation. Consequently, the hash function maps different obstacle topologies to uniquely different hash keys, naturally incentivising the policy to explore new pathways opened by moving obstacles rather than being suppressed by stale counts. Similarly, the transitions (s_t, a_t, r_t, s_{t+1}) stored in AERU accurately capture the exact environment configuration at that specific time step. Replaying these historical experiences does not introduce adverse effects; rather, it enriches the training distribution, enabling the policy to generalise effective collision-avoidance and path-planning behaviours across diverse dynamic obstacle configurations.

3.3.4 | *Algorithm Network and Parameter Design*

In this work, the neural network architecture is specifically tailored to align with the maximum entropy framework and continuous control requirements of the SAC algorithm. The architecture is primarily composed of a Q-network (Critic) and Policy Network (Actor):

1. Critic network (Q-Network): Designed to estimate the soft Q value $Q(s, a)$, the critic utilises a PointNet-based feature extractor followed by two multilayer perceptron (MLP) blocks (each comprising three linear layers). This deep structural design allows the network to effectively fuse the high-dimensional spatial state observations with the continuous action inputs, enabling precise evaluation of the expected return to guide policy improvement.

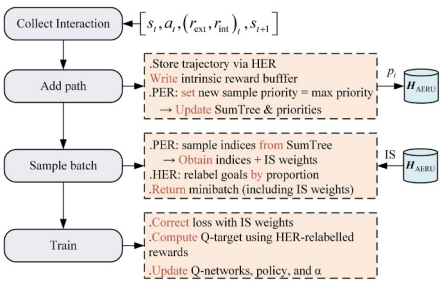


FIGURE 4 | Adaptive experience pool sampling process.

2. Actor Network (Policy Network): To fully support the stochastic exploration mechanism inherent in SAC, the Actor employs a Tanh Gaussian PointNet policy architecture. The state input is first processed through two stacked MLP blocks to extract high-level representations. To parameterise the continuous action space, the network then bifurcates into two parallel linear output heads: One computes the mean μ of the action distribution, and the other computes its log standard deviation $\log \sigma$. Based on these two parameters, the algorithm constructs a Gaussian policy distribution for action sampling. Finally, a Tanh activation function is applied to the sampled actions to enforce strict boundaries, ensuring that the outputs remain valid within the physical constraints of the simulation environment.

Table 4 details the key simulation hyperparameters, including learning rates, batch sizes and the specific scaling coefficients for the MCR and AERU modules. All settings were empirically optimised to ensure a stable balance between exploration efficiency and policy convergence across both Scenario 1 and Scenario 2.

4 | Experimental Design and Result Analysis

In our experiments, we selected the HER-SAC algorithm [34] as our primary baseline for comparison. We chose HER-SAC because it represents a standard and highly effective approach in sparse reward environments by integrating a hindsight experience replay mechanism and utilising a PointNet-based network architecture for learning obstacle representations. By comparing our proposed algorithm against this foundational baseline, we aim to clearly demonstrate the specific performance gains achieved by introducing our MCR and AERU mechanisms. In addition, we evaluated Cur HER-SAC (HER-SAC enhanced with curriculum learning), which tackles the task progressively by initialising training with a minimal start-to-target distance in an obstacle-free setting, subsequently increasing the distance and obstacle count until the target task complexity is reached.

4.1 | Algorithm Exploration and Sample Efficiency Experiments

4.1.1 | *Exploring Performance Analysis*

Through multiple experiments, the training and evaluation results of the MCR-AERU SAC algorithm after 1500 epochs of

TABLE 4 | Key parameter settings.

Parameter name	Scenario 1	Scenario 2	Parameter name	Scenario 1	Scenario 2
batch_size	256	256	ext_int_ratio_min	0.1	0.8
max_path_length	80	120	icm_base_weights	0.4	0.6
min_num_steps_before_training	1000	5000	hash_base_weights	0.3	0.6
num_expl_steps_per_train_loop	1000	3000	emp_base_weights	0.1	0.3
num_trains_per_train_loop	1000	1500	hash_kwargs_alpha	0.5	0.4
num_eval_steps_per_epoch	1000	2000	emp_kwargs_scale	1.0	5.0
num_epochs	1500	2000	replay_α_{per}	Decay from 0.6 to 0.25.	Decay from 0.8 to 0.25.
policy_lr/qf_lr	3e-4	1e-4	replay_β_{per}	increase from 0.4 to 1.0.	Increase from 0.2 to 1.0.
Discount (γ)	0.99	0.995	replay_eps	1e-6	1e-6
soft_target_tau	0.005	0.001	replay_frac_goal	0.8	0.6

training in Scenario 1 are shown in Figure 5, where 'Training batch rewards' refers to the mean intrinsic and extrinsic rewards of the transitions randomly sampled in the minibatches during network gradient updates (i.e., $^1/_{|B|}\sum_{i\in B} r_i$). This metric is specifically plotted to visualise the actual learning signals fed into the networks, thereby demonstrating the dynamic sampling effects of the AERU mechanism and the reward curriculum over time.

Under the MCR-AERU SAC training framework, using phased scheduling (strong intrinsic reward and high 'pseudo-sample' in the early stage and reducing intrinsic reward and more real goals in the later stage) allows for rapid improvement in sample efficiency during the mid-stage and achieving higher and more stable success rates in the later stage. At the same time, the average priority decreases with training, reflecting that TD error and sampling bias are effectively suppressed as learning progresses.

The algorithm performance comparison between MCR-AERU SAC and other algorithms is shown in the experiment results, as reflected in Table 5, Figures 6–8. The results reveal clear performance differences across algorithms under identical training conditions. MCR-AERU SAC achieves the best overall performance, effectively balancing exploration and exploitation to obtain high success rates and strong training efficiency. Its stable convergence and moderate exploration time indicate efficient policy learning.

The results in Table 6 present the task outcomes over 100 evaluation episodes for different algorithmic baselines under the same training random seed. The MCR-AERU SAC algorithm significantly outperforms the others, excelling across all evaluation metrics. These evaluation results emphasise the critical importance of integrating multifaceted intrinsic rewards with adaptive experience replay mechanisms, further validating the overall improvement in learning efficiency. This proposed framework significantly enhances the policy's performance, particularly in terms of task efficiency, path optimisation and obstacle avoidance.

The navigation trajectories obtained by testing the algorithm policy trained with Seed 1 in Scenario 1 are shown in Figures 8 and 9. Within MCR, ICM drives exploration toward dynamically unpredictable regions, producing behaviours such as wall-following and circular search; Hash rewards rare states, broadening coverage into edges and gaps; Emp encourages controllable high-impact states, yielding smoother and more structured trajectories. AERU complements this by providing virtual successes through relabelling and prioritising transitions with higher TD errors, enabling rapid early exploration followed by stable, direct behaviour. Overall, the MCR-AERU SAC algorithm achieves early access to rare and hard-to-predict states, high mid-stage sample efficiency and stable near-optimal behaviour later in training. The controlled entity reliably avoid obstacles and reach targets, initially exploring broadly (detours and wall-following)

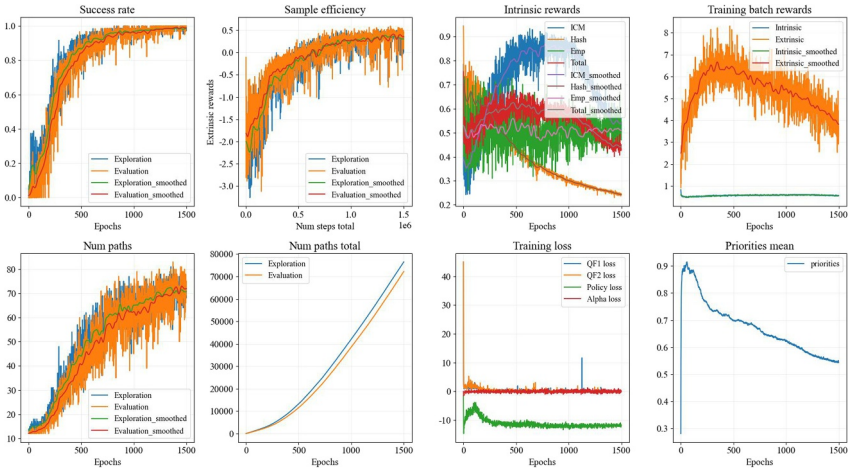


FIGURE 5 | Training result curve of the MCR-AERU SAC algorithm in Scenario 1. (In these figures, 'Evaluation' refers to the deterministic testing phase during training. In this phase, all exploration mechanisms and intrinsic rewards are disabled to measure the policy's true performance in the environment).

TABLE 5 | Comparison of key algorithmic metrics.

Algorithm	500 epochs SR	Final SR	ERM	FER	RM	FR	LPS	AS	PLM	ES (s)
MCR-AERU SAC	0.889	**1.0**	**−0.193**	**0.382**	**−16.371**	4.853	**1.322**	**0.60**	**25.80**	7.336
MCR HER-SAC	**0.890**	0.986	−0.541	0.174	−22.214	1.355	1.445	0.573	29.673	8.196
AERU SAC	0.821	0.986	−0.327	0.315	−23.923	**5.038**	1.343	0.590	28.812	6.896
Cur HER-SAC	/	0.476	−1.675	−2.042	−104.214	−97.491	1.163	0.457	57.90	4.419
HER-SAC	0.333	0.857	−1.143	−0.067	−60.345	−1.364	1.458	0.512	47.391	**3.408**

Note: Boldface indicates the optimal value among the indicator values in that column.
Abbreviations: AM, action mean; AS, actions Std; ERM, external reward mean; ES, exploration sampling; FER, final external reward; FR, final return; LPS, Log Pis Std; PLM, path length mean; RM, returns mean; SR, success rate.

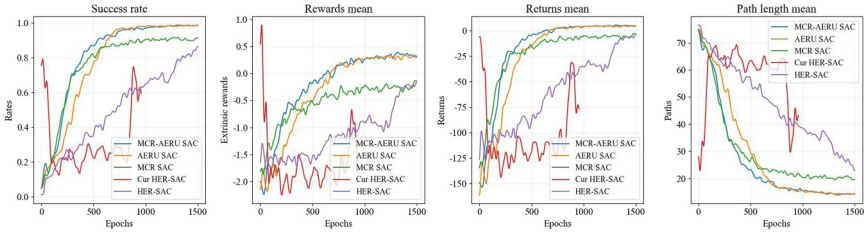


FIGURE 6 | Performance comparison of different algorithms in Scenario 1 (exploration phase, after smoothing).

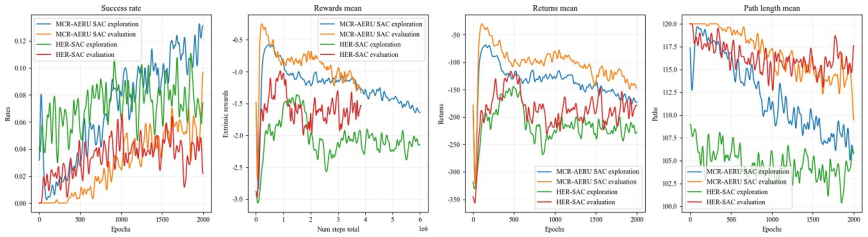
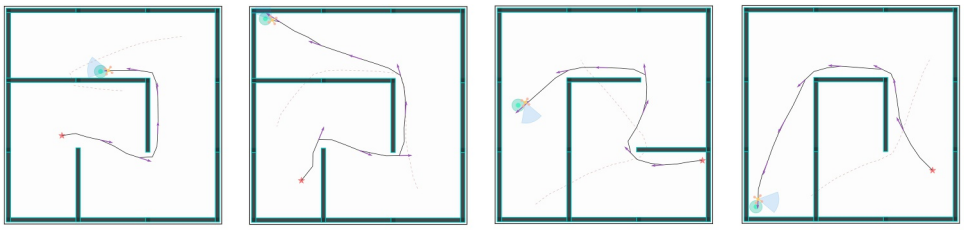


FIGURE 7 | Exploration and evaluation curves of MCR-AERU SAC and HER-SAC algorithm in Scenario 2 (after smoothing). In addition, for this highly complex task, the value of d in the Cur HER-SAC algorithm should be much greater than 0.8. However, the algorithm was forced to terminate with a low task success rate when $d = 0.4$ due to the high task complexity.



(a) Wall-following search (b) Frequent entry into the edges (c) The path is smoother (d) Long-distance exploration

FIGURE 8 | Navigation path trajectory driven by the MCR-AERU SAC algorithm (The purple arrows denote the velocity directions). (a) Wall-following search; (b) frequent entry into the edges; (c) the path is smoother; (d) long-distance exploration.

TABLE 6 | The task outcomes of evaluating algorithmic strategies for 100 episodes under the same training seed.

Algorithm	RM (↑)	MS (↓)	SR (↑)	Collisions (↓)	ML
MCR-AERU SAC	**6.474**	**12.8**	**1.0**	**11.8**	0.812
MCR HER-SAC	−11.082	19.39	0.912	18.48	0.837
AERU SAC	6.073	13.5	0.990	12.51	0.820
Cur HER-SAC	−191.014	65.67	0.215	65.46	0.407
HER-SAC	1.585	22.35	0.874	21.48	0.865

Note: Boldface indicates the optimal value among the indicator values in that column.
Abbreviations: ML, mean length; MS, mean steps; SR, success rate.

and later following smooth, direct paths as intrinsic rewards fade and AERU-guided exploitation dominates.

4.1.2 | *Sample Efficiency Analysis*

In DRL, sample efficiency refers to learning effective policies from limited interactions. It can be assessed either by comparing performance under a fixed number of interactions or by measuring the number of interactions needed to reach a given performance level. The sample efficiency experiments in this work follow this evaluation standard.

1. Steps to threshold (ST): We preset 5 sets of key success rate thresholds and recorded the number of environment interaction steps required for each algorithm to first reach the threshold.
2. Compare the area under the learning curve (AUC): For the first 500, 1000 and 1400 epochs, we calculate the AUC as a composite indicator of sample efficiency. For each curve, the value is numerically integrated $\int_0^{S_T} SR(y)\,dy$, where $SR(y)$ represents the success rate at step y. The final AUC value after multiple experiments is shown in Figure 10. In the same number of epochs, the larger the AUC, the higher the success rate maintained throughout the training process.
3. Early-stage gain (ESG): Focusing on early training, we calculate the task success rate's first improvement speed from the 10% threshold to the 40% threshold;

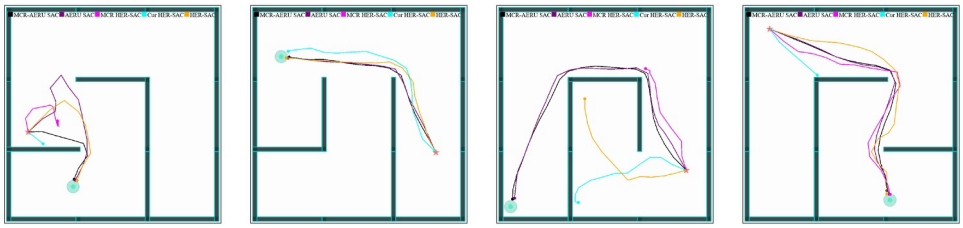


FIGURE 9 | Trajectories of the controlled entity driven by the algorithmic policies under the same random seed in Scenario 1.

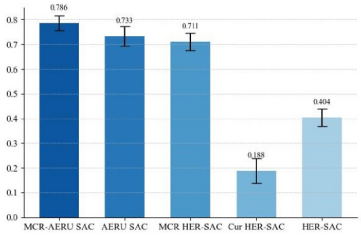


FIGURE 10 | The final AUC metric normalised results (evaluation phase).

ESG $= \frac{SR(S_{T_{40}}) - SR(S_{T_{10}})}{S_{T_{40}} - S_{T_{10}}}$. Here, $S_{T_{40}}$ represents the number of steps to reach a 40% success rate, and $S_{T_{10}}$ represents the number of steps to reach a 10% success rate. A larger ESG indicates higher success rates during the entire training process and a greater early-stage gain.

4. Convergence stability (CS): Three different seed experiments were run randomly, comparing the success rate variance of each algorithm under the same number of training steps to determine whether the improvement in sample efficiency comes at the cost of stability. A good algorithm should maintain low variance while being efficient. The success rates and variances of each algorithm under the same training steps are shown in Table 7 and Figure 11.
5. Average new state discovery: To quantify exploration in our continuous environments, the spatial workspace is evaluated by discretizing it into uniform grid cells. Entering a previously unvisited cell is recorded as discovering a 'new state' (duplicate visits filtered). State visitation heat maps tracking this progression across early, middle and late training stages are shown in Figures 12 and 13.

The state visitation heat maps visually capture the transition from broad exploration to focused exploitation. As illustrated in Figure 12 (Scenario 1), MCR-AERU SAC (top row) demonstrates a clear optimisation trajectory. During the first 50k steps, MCR drives broad exploration across the workspace. By the last 50k steps, AERU's adaptive replay effectively concentrates the policy's visits onto an optimal, obstacle-free path. In contrast, the baseline algorithms (middle and bottom rows) struggle to achieve this level of concentration, maintaining more scattered visitation patterns even in the late stages.

Figure 13 extends this analysis to the highly complex Scenario 2. It is important to note that plotting a 2D spatial heat map for a 245-dimensional continuous state space can only capture a limited projection of the policy's true behaviour. Due to this massive dimensionality and the intricate 5 × 5 dynamic topology, the visual changes are inherently more subtle. Collapsing to a single narrow path is practically impossible and would imply severe overfitting. Both MCR-AERU SAC (top row) and HER-SAC (bottom row) maintain broad spatial coverage. However, a closer inspection of MCR-AERU SAC reveals a gradual and meaningful density shift towards viable navigational regions as training progresses. While the 2D visual distinction is less pronounced than in Scenario 1, this subtle shift in exploration focus consistently corroborates the significant quantitative performance gap (e.g., success rate and sample efficiency) established in our learning curves.

4.2 | Exploration Versus Exploitation Trade-Off Experiment

4.2.1 | *Performance Analysis of the Trade-Off*

In RL, the exploration–exploitation trade-off refers to trying new actions during exploration, even if the rewards are uncertain, to discover better strategies, and selecting the currently optimal action during exploitation to gain certain high rewards. This trade-off is reflected in several algorithm performance metrics in policy entropy–based methods.

1. Policy entropy (Log pis): High entropy suggests a more exploratory policy (diverse actions), whereas low entropy indicates a more deterministic policy (exploiting known information). (2) Policy Gaussian distribution variance (Policy log std): Larger variance indicates stronger exploration (higher randomness), whereas smaller variance suggests more deterministic behaviour (strong exploitation). (3) Alpha: Alpha controls the balance between 'maximizing reward' and 'maximizing entropy'. A larger alpha favours exploration, whereas a smaller alpha favours exploitation.

As shown in Figure 14a–c, MCR-AERU SAC initially exhibits higher policy entropy, larger policy log-std and a larger α—signals of stronger early exploration that later stabilise as the policy converges towards more deterministic, reward-oriented behaviour. MCR HER-SAC displays smaller entropy fluctuations and larger log-std, indicating a stable yet persistent exploration–exploitation transition, while Cur HER-SAC maintains lower entropy, smaller log-std and smaller α, favouring early exploitation and faster convergence at the risk of premature commitment to suboptimal policies. In terms of exploratory intensity, AERU SAC ranks below MCR-AERU SAC but above the HER variants. Specifically, HER-SAC exhibits an intermediate variance—more exploratory than Cur HER-SAC, yet less than MCR-AERU SAC. This indicates that HER-SAC fails to optimally balance exploration and exploitation, potentially causing sample inefficiency during the later stages of training.

TABLE 7 | Sample efficiency analysis results (evaluation phase).

Algorithm	ST (different thresholds ≥) (↓)					AUC (↑)			ESG	CS	
	0.30	0.50	0.90	0.95	1.0	500k	1000k	1400k	(e^{-6}↑)	400k	900k
MCR-AERU SAC	129,534	**163,078**	**429,088**	**595,494**	**793,673**	**0.457**	**0.690**	**0.771**	1.741	0.593 ± 0.0295	**0.921** ± 0.0007
AERU SAC	94,834	243,268	580,919	664,208	755,562	0.333	0.612	0.715	1.374	0.555 ± 0.0409	0.732 ± 0.0086
MCR HER-SAC	**78,272**	187,975	540,238	984,794	/	0.430	0.628	0.699	**2.156**	**0.677** ± 0.0005	0.732 ± 0.0174
Cur HER-SAC	/	/	/	/	/	0.169	0.188	0.188	/	0.077 ± 0.0	0.421 ± 0.0
HER-SAC	131,297	416,411	1320,153	/	/	0.149	0.284	0.376	1.231	0.328 ± 0.0037	0.692 ± 0.0214

Note: Boldface indicates the optimal value among the indicator values in that column.

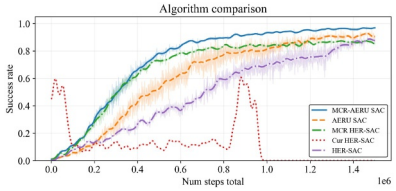


FIGURE 11 | Convergence stability curve (evaluation phase).

In summary, MCR-AERU SAC achieves a better balance between exploration and exploitation under the regulation of policy entropy, standard deviation and alpha values. Compared to other algorithms, it outperforms them in complex environments, particularly in tasks like path planning, where it can explore effectively in a wider state space and ultimately converge to a high-quality policy.

4.2.2 | *Module Weight Interaction*

MCR weight scan: To quantify the impact of various intrinsic reward modalities and external reward mixing weights on exploration–exploitation balance, we fixed the AERU module parameters in this experiment and selected six sets of reward weight parameters. The experimental results are shown in Table 8. AERU and MCR interaction: To verify the coupling effect between AERU module's sampling bias intensity and multifaceted intrinsic rewards, we fixed the MCR reward weight configuration in this experiment and selected five sets of AERU module parameters. The experimental results are shown in Table 8 and Figure 15.

As shown in Table 8, during training, the higher intrinsic reward and the goal replay ratio in the early stages enable the controlled entity to explore unknown areas of the environment more effectively, which is crucial for complex environments. A higher intrinsic reward ratio enhances early-stage exploration. As training progresses, the controlled entity gradually relies on external rewards to optimise decisions, reducing the risk of excessive exploration, and the higher external reward ratio helps improve success rates and optimise the quality of the paths. By adjusting the priorities and importance weights, the balance between exploration and exploitation is controlled.

MCR, by introducing multifaceted intrinsic rewards, effectively promotes the learnt early exploration and long-term learning stability, while AERU, through optimising the replay strategy of samples, significantly improves success rates, reward curve smoothness and training efficiency. The combination of these two mechanisms significantly enhances sample efficiency and convergence speed, especially in sparse reward environments.

4.2.3 | *Transfer and Generalisability*

To validate the consistency and robustness of the algorithmic strategy under multitask conditions, this section conducts the following generalisation experiments and analyzes the results.

1. The policy trained with Seed 1 in Scenario 1 is tested in a 100-seed environment.

Figure 16 illustrates the generalisation test trajectories of various algorithms in Scenario 1, evaluated under an unseen random configuration (Seed 100). As visually evident in the plots, baseline policies (such as HER-SAC and Cur HER-SAC) struggle to adapt to the novel obstacle topology. They frequently exhibit meandering behaviours, unnecessary detours or become completely trapped in local dead ends. In stark contrast, the policy driven by MCR-AERU SAC (represented by the black trajectory) successfully plans smooth, collision-free and highly efficient paths to the target. This striking behavioural difference explicitly demonstrates that our synergistic framework prevents the policy from merely memorising training layouts, equipping it with robust zero-shot generalisation capabilities in dynamic environments.

2. The policy trained with Seed 1 in Scenario 1 is evaluated for 100 episodes in an environment with 200 seeds.

Table 9 (cf. Table 6) shows that MCR-AERU SAC achieves superior transfer performance—average reward 6.204, success rate 1.00 and lower average steps and collisions (13.04 steps and 12.04 collisions)—indicating efficient, robust path planning under varying initialisations and layouts. Algorithms relying on a single mechanism (e.g., MCR HER-SAC and AERU SAC) suffer drops in reward and success and increased collisions after transfer, suggesting greater overfitting to training randomness or scene specifics, while Cur HER-SAC performs worst, failing to learn a scene-robust strategy. The HER-SAC yields moderate results but does not match MCR-AERU SAC. Mechanistically, MCR-AERU SAC's mixed intrinsic rewards combined with adaptive replay produce more diverse, informative experiences and improve sample utilisation, yielding greater robustness to environmental variation.

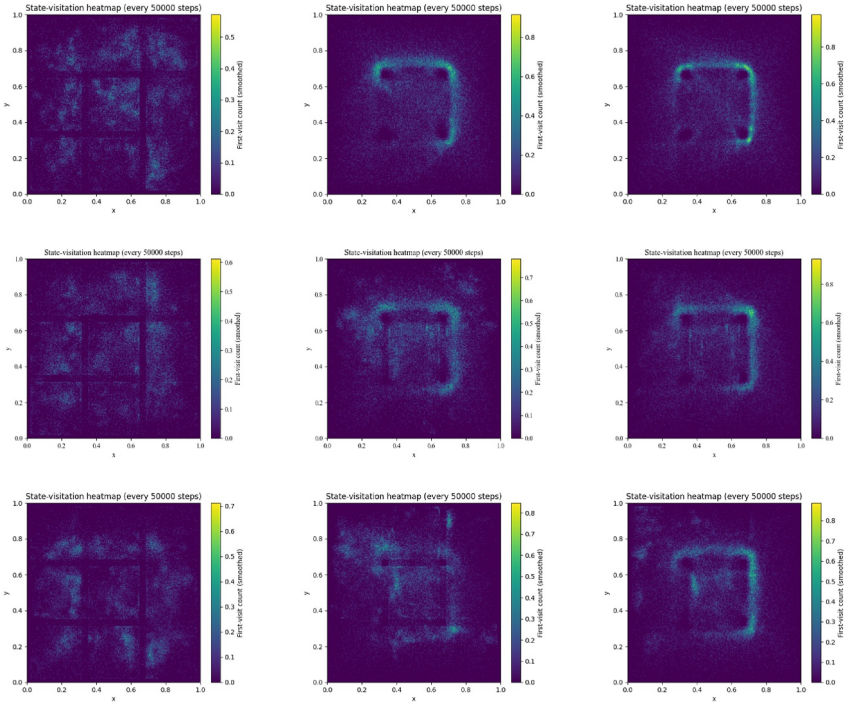


FIGURE 12 | State visitation heat maps tracking the exploration-to-exploitation transition in Scenario 1. From top to bottom: MCR-AERU SAC, MCR HER-SAC and HER-SAC. The top row clearly demonstrates a rapid and highly efficient concentration of visits onto an optimal path, contrasting sharply with the scattered baselines.

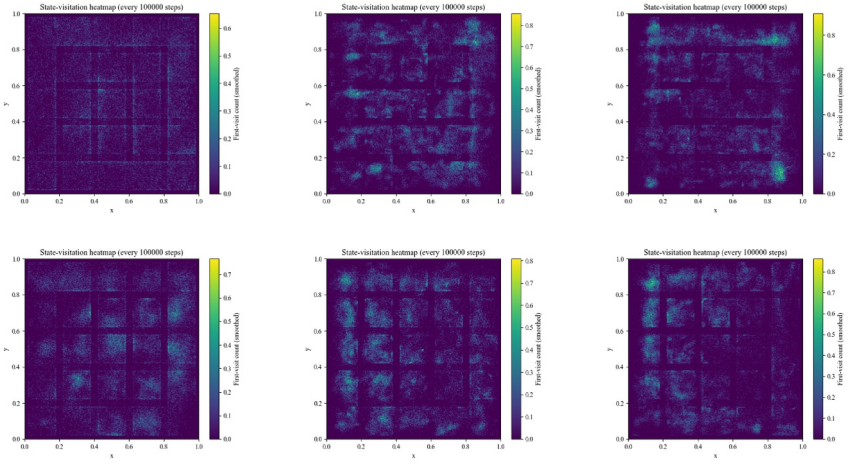
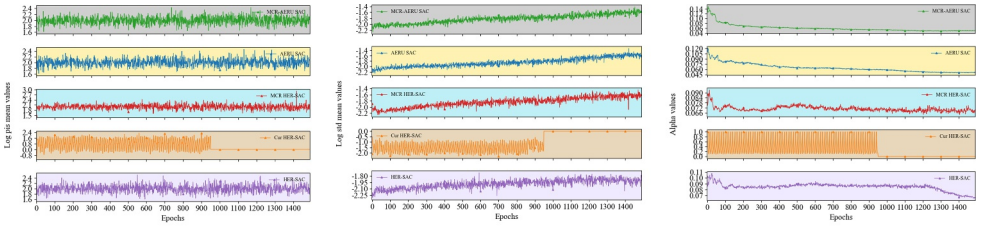


FIGURE 13 | State visitation heat maps in the highly complex Scenario 2 (top: MCR-AERU SAC; bottom: HER-SAC). Due to the 2D projection of a 245-dimensional state space, the visitation density shifts are visually more subtle than in Scenario 1. However, MCR-AERU SAC still exhibits a gradual concentration toward viable regions compared to the baseline.



(a) Policy entropy (b) Policy gaussian distribution variance (c) Alpha

FIGURE 14 | Exploration–exploitation trade-off performance curve (Cur HER-SAC performed 0-padding). (a) policy entropy; (b) policy gaussian distribution variance; (c) alpha.

TABLE 8 | Weight experiment results.

MCR parameter changes					Ratio of external rewards to internal rewards		AERU parameter changes				
ICM ratio	**1.0**	**0.0**	**0.0**	**1/3**							
Hash ratio	**0.0**	**1.0**	**0.0**	**1/3**							
Emp ratio	**0.0**	**0.0**	**1.0**	**1/3**	**0:1**	**1:1**	**α_{per}: 0.8 → 0.25**	**β_{per}: 0.2 → 1.0**	**eps:1e-5**	**frac_goal_replay:0.9**	**frac_goal_replay:0.6**
1000k steps SR	0.898	0.965	0.50	0.889	0.0	0.962	0.985	0.984	0.875	0.850	0.985
Final SR	0.902	0.984	0.421	0.920	0.143	0.964	0.985	0.986	0.945	0.943	0.987
ERM	−0.725	−0.534	−2.397	−0.675	−3.317	−0.311	−0.516	−0.259	−0.869	−0.993	−0.354
ERS	3.496	3.644	2.542	3.614	1.894	3.178	3.751	4.124	3.368	3.194	3.952
IRM	0.722	0.397	0.518	0.542	0.509	0.421	0.579	0.565	0.585	0.591	0.543
PLM	34.917	35.631	60.921	32.844	76.901	37.530	33.812	28.265	37.504	40.515	30.30
LPM	1.985	1.984	1.981	1.983	1.981	1.985	1.982	1.983	1.985	1.993	1.984
ES (s)	6.349	6.430	6.256	6.289	7.065	8.261	8.446	8.595	7.154	6.377	6.387
AM	0.0179	0.0085	0.0184	0.0162	0.1284	0.0107	0.0151	0.0142	0.0241	0.0187	0.0153
AS	0.551	0.555	0.582	0.561	0.628	0.442	0.566	0.587	0.544	0.550	0.567
RM	−35.90	−37.312	−146.194	−30.665	−255.071	−17.953	−34.884	−22.849	−45.073	−51.064	−24.962
NPT	21208.7	21589.6	12047.5	22873.7	9529.9	19298.5	22792.7	28659.3	19261.7	17442.7	24976.9
First 50k steps ECR	0.545	0.494	0.376	0.503	0.301	0.499	0.506	0.623	0.431	0.425	0.617

Note: Scenario 1, exploration phase, with the exception of parameter variations in the table; all other parameters remain consistent with those in Section 3.3.4.
Abbreviations: ECR, environmental coverage rate: ERS, external rewards Std; FER, final external reward; FR, final return; IRM, intrinsic rewards mean; LPM, Log Pis Mean; LPS, Log Pis Std; NPT, Num Paths Total.

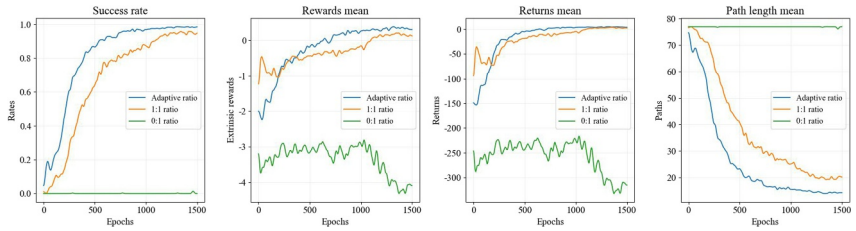


FIGURE 15 | Performance comparison diagram of different external and internal reward ratios.



FIGURE 16 | Trajectories of the controlled entity driven by the algorithmic policies under the same random seed in Scenario 1.

TABLE 9 | The task outcomes of evaluating algorithmic strategies for 100 episodes in Scenario 1 under the training seed of 200.

Algorithm	RM (↑)	MS (↓)	SR (↑)	Collisions (↓)	ML
MCR-AERU SAC	**6.204**	**13.04**	**1.0**	**12.04**	0.831
MCR HER-SAC	−8.521	23.97	0.843	23.13	0.771
AERU SAC	5.826	13.59	0.990	12.60	0.827
Cur HER-SAC	−202.037	63.7	0.247	63.46	0.432
HER-SAC	1.104	24.0	0.852	23.15	0.854

Note: Boldface indicates the optimal value among the indicator values in that column.

3. Preliminary physical validation via virtual-to-real synchronisation.

To evaluate generalisation and preliminary physical feasibility, we developed a virtual-to-real synchronisation system within a noise-isolated physical arena mirroring our simulation. Control commands generated by the learnt policy in simulation are transmitted in real time to synchronously drive a physical UAV along the planned trajectory (Figure 17). The dark blue background represents the scene boundaries, the red square denotes the starting position, and the yellow square represents the goal position. The red-and-white–dashed shape indicates the controlled entity UAV, which moves at a height of 0.2 m. The red dashed line depicts the learnt trajectory.

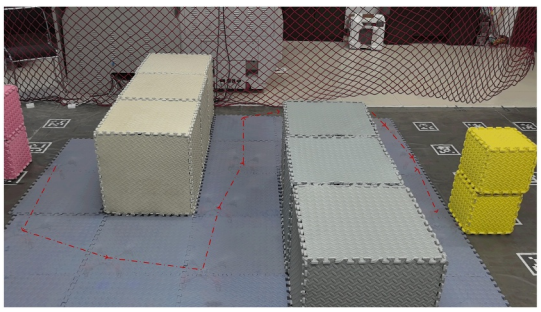


FIGURE 17 | The test results of the physical platform.

4. Strategies trained in Scenario 1 using the same training seed are tested in Scenario 2.

Figure 18 further details the generalisation test results of the MCR-AERU SAC algorithm in the much more complex Scenario 2. Observing the trajectories across four distinct unseen test layouts, while the policy may not perfectly reach the final destination in every zero-shot trial, it successfully navigates through intricate narrow corridors and intelligently bypasses multiple novel obstacle barriers, completing a substantial portion of the navigation task. This proves that dynamically changing training scenarios each epoch introduces crucial environmental diversity, effectively preventing overfitting. Furthermore, the synergy between MCR (encouraging novel state exploration) and AERU (efficiently reusing diverse experiences) helps the policy avoid local optima even in high-dimensional continuous spaces. Ultimately, this multi-environment training paradigm significantly enhances adaptability, making MCR-AERU SAC a highly robust solution for complex applications.

5. Robustness against environmental stochasticity

To validate the stability of MCR-AERU SAC, we conducted robustness evaluations under varying action (red arrows) and wind (blue arrows) noise intensities (Figure 19). While increasing noise causes noticeable trajectory deviations, the policy successfully executes dynamic corrections and reaches the target without being trapped in local optima, even under extreme perturbations (Figure 19d). This superior disturbance rejection stems from the MCR-AERU synergy: MCR effectively filters environmental stochasticity to maintain reliable exploration, while AERU prioritises replaying these noise-induced

'deviation-and-recovery' transitions. This enables the policy to master robust dynamic obstacle avoidance, demonstrating exceptional decision-making resilience.

5 | Discussion and Conclusions

5.1 | Discussion

When the MCR and AERU operate in parallel, we observed instances of training fluctuations and even short-term performance deviations at several experimental points. This does not indicate algorithmic failure. We have identified two primary causes for this phenomenon, providing corresponding explanations and mitigation strategies as shown in Table 10.

Additionally, although MCR-AERU SAC shows significant advantages over the HER-SAC algorithm in several metrics (Figure 7) within Scenario 2, the training and testing results did not reach ideal values. We attribute this to the following factors: Environmental complexity and difficulty in learning long-term temporal dependencies: The current environment features a complex layout with multiple paths and obstacles. In tasks with sparse rewards and long-delayed rewards, the learnt decisions often rely on long-term temporal dependencies. Because RL methods typically update policies based on TD error, long-delayed rewards may prevent the policy from effectively propagating reward signals to earlier time steps. This issue is exacerbated in high-dimensional or complex state spaces. The combination of multifaceted intrinsic rewards and adaptive experience replay in MCR-AERU SAC may further complicate the exploration phase when dealing with long-delayed rewards.

5.2 | Conclusions and Future Work

To address the issues of low-sample efficiency and susceptibility to local optima in sparse reward environments, this paper proposes the MCR-AERU SAC algorithm. In this approach, MCR

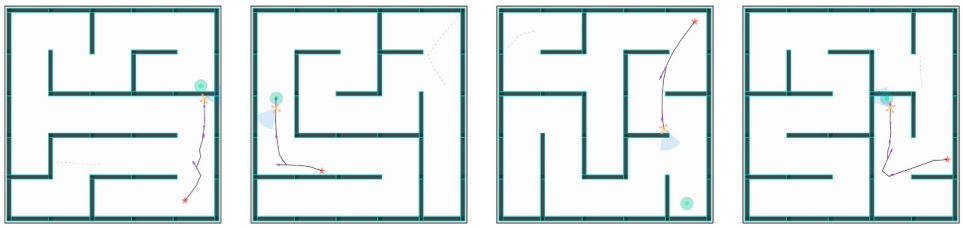


FIGURE 18 | Generalisation test results of the MCR-AERU SAC algorithm in Scenario 2.

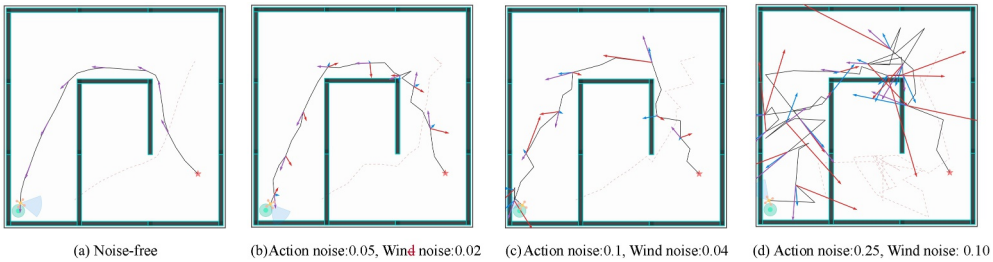


FIGURE 19 | Robustness evaluation results against environmental stochasticity. (a) noise-free; (b) action noise:0.05, wind noise:0.02; (c) action noise:0.1, wind noise:0.04; (d) action noise:0.25, wind noise: 0.10.

TABLE 10 | Analysis of causes for performance deviations.

Reason	Detailed description	Our solution
Over-optimisation of local objectives weakens global exploration capability.	Multifaceted intrinsic rewards encourage the discovery of novel states (driving broad exploration), while PER emphasises correcting high-error regions (driving local exploitation). When both mechanisms coexist, the network may be repeatedly torn between "extremely novel" and "extremely high-error" samples, potentially neglecting samples with medium novelty or medium errors. This leads to a lack of both broad exploration and effective utilisation of medium-quality samples, resulting in an overall strategy that is neither robust nor efficiently refined.	We adopt a staged training approach (strong intrinsic rewards in the early stage and strong prioritisation of exploitation in the late stage) and design an adaptive weight scheduling mechanism (curriculum learning) to balance exploration and exploitation.

provides the actor with multi-level exploration signals during training, encouraging the actor to actively explore the environment and accumulate successful historical experiences, thereby enhancing the exploration capability of the algorithm. At the same time, the AERU prioritises transitions that generate significant deviations in the post hoc setting, further improving the efficiency of utilising 'partially successful' experiences and effectively handling sparse reward signals. These two mechanisms work together to optimise the policy, allowing the algorithm to learn feasible strategies more quickly in sparse reward environments, adaptively balancing exploration and exploitation and avoiding stagnation in suboptimal strategies. This enables the learning framework to effectively utilise historical information while actively exploring the environment, improving sample efficiency and accomplishing tasks in complex environments. This is particularly important for applications in fields such as UAVs and autonomous vessels. Experimental results demonstrate that compared to other advanced DRL algorithms, the MCR-AERU SAC algorithm is successful, and both sample efficiency and exploration–exploitation balance experiments further confirm the algorithm's effectiveness.

In the future, we shall further refine our algorithms, enhancing the efficiency of RL samples by addressing multiple problem dimensions. This will resolve the issue of suboptimal sample efficiency caused by long-delayed rewards, while simultaneously deepening our research into the exploration–exploitation balance problem.

Acknowledgements

The authors would like to acknowledge the Fundamental Research Funds for the Central Universities (Grant No. H20250607) , the National Key Laboratory of Air-based Information Perception and Fusion (Grant No. 202510), the Key Research and Development Program of Shaanxi Province (Grant No. 2023-GHZD-33) , the Open Project of the State Key Laboratory of Intelligent Game (Grant No. ZBKF-23-05) and the National Natural Science Foundation of China (Grant No. 62003267).

Conflicts of Interest

The authors declare no conflicts of interest

Data Availability Statement

The data that support the findings of this study are available from the corresponding author upon reasonable request.

References

1. O. K. Pal, M. S. H. Shovon, M. F. Mridha, and J. Shin, "In-Depth Review of AI-Enabled Unmanned Aerial Vehicles: Trends, Vision, and Challenges," *Discover Artificial Intelligence* 4, no. 1 (December 2024): 97, https://doi.org/10.1007/s44163-024-00209-1.

2. A. A. Laghari, A. K. Jumani, R. A. Laghari, H. Li, S. Karim, and A. A. Khan, "Unmanned Aerial Vehicles Advances in Object Detection and Communication Security Review," *Cognitive Robotics* 4 (2024): 128–141, https://doi.org/10.1016/j.cogr.2024.07.002.

3. J. Y. Huang, Y. J. Cui, G. P. Xi, et al., "GTrXL-SAC-Based Path Planning and Obstacle-Aware Control Decision-Making for UAV Autonomous Control," *Drones* 9, no. 4 (April 2025): 275, https://doi.org/10.3390/drones9040275.

4. Z. Hu, M. Z. Butt, N. Nasir, and M. B. Salleh, "Vision-Based Autonomous Navigation for Quadrotor UAVs in Unknown and Constrained Environments," *International Journal of Intelligent Robotics and Applications* 9, no. 4 (May 2025): 1743–1754, https://doi.org/10.1007/s41315-025-00462-2.

5. B. Li, J. M. Wang, C. Song, Z. Yang, K. Wan, and Q. Zhang, "Multi-UAV Roundup Strategy Method Based on Deep Reinforcement Learning CEL-MADDPG Algorithm," *Expert Systems with Applications* 245 (July 2024), https://doi.org/10.1016/j.eswa.2023.123018.

6. X. Wang, S. Wang, X. X. Liang, et al., "Deep Reinforcement Learning: A Survey," *IEEE Transactions on Neural Networks and Learning Systems* 35, no. 4 (April 2024): 5064–5078, https://doi.org/10.1109/TNNLS.2022.3207346.

7. R. S. Sutton and A. G. Barto, "Reinforcement Learning: An Introduction," *IEEE Transactions on Neural Networks* 9, no. 5 (September 1998): 1054, https://doi.org/10.1109/TNN.1998.712192.

8. B. Li, J. Huang, S. Bai, et al., "Autonomous Air Combat Decision-Making of UAV Based on Parallel Self-Play Reinforcement Learning," *CAAI Transactions on Intelligence Technology* 8 (2022): 1–18, https://doi.org/10.1049/cit2.12109.

9. X. W. Fu and Y. Sun, "A Combined Intrusion Strategy Based on Apollonius Circle for Multiple Mobile Robots in Attack-Defense Scenario," *IEEE Robotics and Automation Letters* 10, no. 1 (January 2025): 676–683, https://doi.org/10.1109/LRA.2024.3512361.

10. M. Cao, L. Shu, L. Yu, et al., "Beyond Sparse Rewards: Enhancing Reinforcement Learning With Language Model Critique in Text Generation," 19 (February 2024), https://doi.org/10.48550/arXiv.2401.07382.

11. X. Y. Liu, H. Yang, J. Gao, et al., "FinRL: Deep Reinforcement Learning Framework to Automate Trading in Quantitative Finance," in *Proceedings of the Second ACM International Conference on AI in Finance, Virtual Event* (ACM, 2021), 1–9, https://doi.org/10.1145/3490354.3494366.

12. P. Ladosz, L. Weng, M. Kim, and H. Oh, "Exploration in Deep Reinforcement Learning: A Survey," *Information Fusion* 85 (September 2022): 1–22, https://doi.org/10.1016/j.inffus.2022.03.003.

13. M. Yuan, R. Castanyer, B. Li, et al., "RLeXplore: Accelerating Research in Intrinsically-Motivated Reinforcement Learning," (April 2025), https://doi.org/10.48550/arXiv.2405.19548.

14. D. Pathak, P. Agrawal, A. A. Efros, et al., "Curiosity-Driven Exploration by Self-Supervised Prediction," (May 2017), https://doi.org/10.48550/arXiv.1705.05363.

15. X. Fu, J. Zhu, Z. Wei, H. Wang, and S. Li, "A UAV Pursuit-Evasion Strategy Based on DDPG and Imitation Learning," *International Journal of Aerospace Engineering* 2022 (May 2022): 1–14, https://doi.org/10.1155/2022/3139610.

16. L. Wu and K. Chen, "Goal Exploration Augmentation via Pre-Trained Skills for Sparse-Reward Long-Horizon Goal-Conditioned Reinforcement Learning," (December 2023), https://doi.org/10.48550/arXiv.2210.16058.

17. J. Zhang, W. Zhang, D. Zhou, et al., "Uncertainty-Aware Reward-Free Exploration With General Function Approximation," (June 2024), https://doi.org/10.48550/arXiv.2406.16255.

18. M. Albaba, S. Christen, T. Langarek, et al., "RILe: Reinforced Imitation Learning," (April 2025), https://doi.org/10.48550/arXiv.2406.08472.

19. B. Li, J. Y. Huang, H. H. Zhang, L. Huai, and E. Neretin, "Enhanced Exploration for Multi-UAV Cooperative Roundup: An I2C-MATD3 Reinforcement Learning Framework," *Defence Technology* 58 (October 2025): 374–389, https://doi.org/10.1016/j.dt.2025.10.032.

20. Y. Burda, H. Edwards, D. Pathak, et al., "Large-Scale Study of Curiosity-Driven Learning," (August 2018), https://doi.org/10.48550/arXiv.1808.04355.

21. A. L. Strehl and M. L. Littman, "An Analysis of Model-Based Interval Estimation for Markov Decision Processes," *Journal of Computer and System Sciences* 74, no. 8 (December 2008): 1309–1331, https://doi.org/10.1016/j.jcss.2007.08.009.

22. S. Mohamed and D. J. Rezende, "Variational Information Maximisation for Intrinsically Motivated Reinforcement Learning," (September 2015), https://doi.org/10.48550/arXiv.1509.08731.

23. R. Raileanu and T. Rocktäschel, "RIDE: Rewarding Impact-Driven Exploration for Procedurally-Generated Environments," (February 2020), https://doi.org/10.48550/arXiv.2002.12292.

24. J. Li and P. Gajane, "Curiosity-Driven Exploration in Sparse-Reward Multi-Agent Reinforcement Learning," (February 2023), https://doi.org/10.48550/arXiv.2302.10825.

25. M. Andrychowicz, F. Wolski, A. Ray, et al., "Hindsight Experience Replay," (February 2018), https://doi.org/10.48550/arXiv.1707.01495.

26. H. Oh, M. Murooka, T. Motoda, et al., "Self-Augmented Robot Trajectory: Efficient Imitation Learning via Safe Self-Augmentation With Demonstrator-Annotated Precision," 11 (September 2025), https://doi.org/10.48550/arXiv.2509.09893.

27. C. Finn, S. Levine, and P. Abbeel, "Guided Cost Learning: Deep Inverse Optimal Control via Policy Optimization," (March 2016), https://doi.org/10.48550/arXiv.1603.00448.

28. J. Lin, Y. Q. Du, O. Watkins, et al., "Learning to Model the World With Language," (May 2024), https://doi.org/10.48550/arXiv.2308.01399.

29. A. Nair, V. Pong, M. Dalal, et al., "Visual Reinforcement Learning With Imagined Goals," (December 2018), https://doi.org/10.48550/arXiv.1807.04742.

30. M. Fang, T. Zhou, Y. Du, et al., "Curriculum-Guided Hindsight Experience Replay," in *Proceedings of the 33rd International Conference on Neural Information Processing Systems* (Curran Associates Inc., 2019) 12623–12634, Article 1131.

31. T. Haarnoja, A. Zhou, K. Hartikainen, et al., "Soft Actor-Critic Algorithms and Applications," (January 2019), https://doi.org/10.48550/arXiv.1812.05905.

32. Z. J. Hu, X. G. Gao, K. F. Wan, Q. Wang, and Y. Zhai, "Asynchronous Curriculum Experience Replay: A Deep Reinforcement Learning Approach for UAV Autonomous Motion Control in Unknown Dynamic Environments," *IEEE Transactions on Vehicular Technology* (2023): 1–16, https://doi.org/10.1109/TVT.2023.3285595.

33. B. Norman and J. Clune, "First-Explore, Then Exploit: Meta-Learning to Solve Hard Exploration-Exploitation Trade-Offs," (November 2024), https://doi.org/10.48550/arXiv.2307.02276.

34. R. Strudel, R. Garcia, J. Carpentier, et al., "Learning Obstacle Representations for Neural Motion Planning," (November 2020), https://doi.org/10.48550/arXiv.2008.11174.

35. Z. J. Hu, X. G. Gao, K. F. Wan, Y. Zhai, and Q. Wang, "Relevant Experience Learning: A Deep Reinforcement Learning Method for UAV Autonomous Motion Planning in Complex Unknown Environments," *Chinese Journal of Aeronautics* 34, no. 12 (2021): 187–204, https://doi.org/10.1016/j.cja.2020.12.027.

36. R. Devidze, P. Kamalaruban, and A. Singla, "Exploration-Guided Reward Shaping for Reinforcement Learning Under Sparse Rewards," in *36th Conference on Neural Information Processing Systems (NeurIPS 2022)*, (Curran Associates, Inc., 2022).

37. H. Lin, C. Huang, and Z. Chen, "An Advantage-Based Optimization Method for Reinforcement Learning in Large Action Space," (December 2024), https://doi.org/10.48550/arXiv.2412.12605.

38. J. Yan, B. Luo, and X. Xu, "Hierarchical Reinforcement Learning for Handling Sparse Rewards in Multi-Goal Navigation," *Artificial Intelligence Review* 57, no. 6 (May 2024): 156, https://doi.org/10.1007/s10462-024-10794-3.

39. D. Yunis, J. Jung, F. Dai, et al., "Subwords as Skills: Tokenization for Sparse-Reward Reinforcement Learning," (October 2024), https://doi.org/10.48550/arXiv.2309.04459.

40. T. Schaul, J. Quan, I. Antonoglou, et al., "Prioritized Experience Replay," (February 2016), https://doi.org/10.48550/arXiv.1511.05952.

41. Y. C. Pan, J. C. Mei, A. M. Farahmand, et al., "Understanding and Mitigating the Limitations of Prioritized Experience Replay," (June 2022), https://doi.org/10.48550/arXiv.2007.09569.

42. A. Kayal, E. Pignatelli, and L. Toni, "The Impact of Intrinsic Rewards on Exploration in Reinforcement Learning," *Neural Computing & Applications* 37, no. 21 (July 2025): 16269–16303, https://doi.org/10.1007/s00521-025-11340-0.

43. H. R. Tang, R. Houthooft, D. Foote, et al., "#Exploration: A Study of Count-Based Explorationfor Deep Reinforcement Learning," (November 2016), https://doi.org/10.48550/arXiv.1611.04717.

44. M. G. Bellemare, S. Srinivasan, G. Ostrovski, et al., "Unifying Count-Based Exploration and Intrinsic Motivation," (November 2016), https://doi.org/10.48550/arXiv.1606.01868.

45. A. S. Klyubin, D. Polani, and C. L. Nehaniv, "Empowerment: A Universal Agent-Centric Measure of Control," in *2005 IEEE Congress on Evolutionary Computation* (IEEE, 2005), 128–135, https://doi.org/10.1109/CEC.2005.1554676.

46. K. Gregor, D. Rezende, and D. Wierstra, "Variational Intrinsic Control," (November 2016), https://doi.org/10.48550/arXiv.1611.07507.

47. A. V. D. Oord, Y. Z. Li, and O. Vinyals, "Representation Learning With Contrastive Predictive Coding," (January 2019), https://doi.org/10.48550/arXiv.1807.03748.

48. X. Wang, Y. Chen, and W. Zhu, "A survey on Curriculum Learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2021): 1, https://doi.org/10.1109/TPAMI.2021.3069908.

49. Y. Kim, "Convolutional Neural Networks for Sentence Classification," (September 2014), https://doi.org/10.48550/arXiv.1408.5882.