2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

MS-CRL: Multi-Scale Global Path Planning with Progressive Curriculum Reinforcement Learning

Nan Zhou[1], Xuqing Hu[1], Yixin Zhou[1], Rui Zhu[1], Fan Zhou[1], Ye Li[1], Guangqing Yin[1,2]

Abstract— **Global path planning provides high-level guidance for autonomous navigation, supplying reference paths for downstream navigation and control modules. Deep Reinforcement Learning (DRL) has shown strong potential in this domain, but existing methods struggle with multi-scale map inputs. This limitation arises from inconsistent representations across different map sizes and trajectory length variations, which hinder feature extraction, destabilize policy learning. To address these challenges, we propose the Progressive Multi-Scale Curriculum Reinforcement Learning (MS-CRL) framework. MS-CRL incorporates a progressive curriculum reinforcement learning algorithm (ProgCRL) that mitigates instability from trajectory length discrepancies, a unified multi-scale representation (UniMS) that normalizes spatial scales and resolves representation inconsistencies, and a Global-Local Fusion Network (GLFNet) that fully extracts both global and local features from the new representation for robust cross-scale policy learning. Extensive experiments on multi-scale map datasets demonstrate that MS-CRL enables effective global path planning, stabilizes policy learning, and achieves superior performance in path success rate, path quality, and planning efficiency, while significantly improving training efficiency and cross-scale adaptability compared with state-of-the-art baselines.**

I. INTRODUCTION

In recent years, embodied intelligence has experienced rapid development and has been widely applied in various domains such as robot navigation, autonomous driving, unmanned aerial vehicle (UAV) control, service robotics, and industrial automation [1], [2], [3]. Within embodied intelligence systems, path planning is a core technology for enabling autonomous movement and task execution. It directly affects the overall efficiency and safety of the system.

Path planning is generally divided into global and local planning. Global path planning is responsible for generating a path from the start to the goal based on a globally known map with complete environmental information [4]. It plays a guiding role for subsequent modules such as local planning and motion control, and significantly impacts the overall task performance[5], [6].

However, traditional path planning algorithms such as A*, RRT, and Dijkstra suffer from high computational costs. When the environment changes (e.g., map size, obstacle layout, or goal location), they must recompute the entire planning process from scratch, which limits their applicability in large-scale, high-dimensional environments with increasing spatial complexity and task constraints.

Deep reinforcement learning (DRL) has been widely applied across various domains, including social media information diffusion modeling and large-scale intelligent systems [7], [8]. In addition, DRL has shown promise in high-dimensional path planning tasks [9], [10], [11]. However, existing DRL-based path planning methods cannot effectively handle multi-scale map inputs. This limitation manifests in two aspects: (1) inconsistent representations across different map sizes hinder effective feature extraction by neural networks, and (2) significant trajectory length variations across scales lead to uneven reward discounting and inaccurate value estimation, thereby resulting in unstable policy updates.

To overcome these issues, we draw inspiration from the curriculum learning paradigm [12], which trains agents progressively from easier to harder tasks, and propose a **Progressive Multi-Scale Path Planning Curriculum RL (MS-CRL)** framework. Within MS-CRL, we design a progressive curriculum reinforcement learning algorithm (ProgCRL) that models task difficulty via map size and obstacle complexity to mitigate instability from trajectory length discrepancies; a unified multi-scale representation (UniMS) that normalizes spatial scales with agent-centered observations to resolve representation inconsistencies; and a Global-Local Fusion Network (GLFNet) built upon UniMS to integrate global and local features for robust cross-scale policy learning. In summary, the key contributions of our work are as follows:

- We are the first to emphasize the necessity of tackling representation inconsistency and trajectory length discrepancies in DRL-based global path planning on multi-scale maps. To this end, we propose the MS-CRL framework to address these challenges.
- We introduce ProgCRL and UniMS to address the key challenges in multi-scale path planning. ProgCRL stabilizes training by progressively adapting to trajectory length differences across scales, and UniMS provides a unified representation for heterogeneous inputs. Building on this new representation, GLFNet is designed to fully extract both global and local features for robust cross-scale policy learning.
- We conduct extensive experiments on a benchmark dataset with multi-scale map inputs, demonstrating that our framework significantly improves training efficiency, policy convergence, and cross-scale adaptability compared with state-of-the-art baselines.

*This work was supported by Shenzhen Science and Technology (KCXFZ202409030940-11015).

[1]The authors are with the University of Electronic Science and Technology of China, Chengdu, China.

[2]Guangqing Yin is also with the Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China, Shenzhen, China.

Corresponding author: Guangqing Yin yingq@uestc.edu.cn

II. RELATED WORK

Path planning is a fundamental problem in robot autonomous navigation. Traditional methods include graph-based algorithms such as Dijkstra [13] and A* [14], which guarantee optimal solutions but incur high computational costs on large maps, and sampling-based methods like RRT [15] and RRT* [16], which scale well in high-dimensional spaces but require frequent replanning in dynamic environments. Recent advances in deep reinforcement learning (DRL) have shown strong potential in path planning [17], e.g., UAV-based weed localization using DQN [18], urban IoT data collection with convolutional networks [19], hierarchical DRL for indoor navigation with LiDAR-based complexity metrics [20], attention-guided TERP for rugged terrains [21], and MCTS with convolutional networks for online replanning [22].

Although these studies have achieved notable progress, few explicitly consider multi-scale map observations. Most methods assume fixed-size or fixed-structure inputs, and even when the agent is placed at the map center to convert absolute positions to relative ones [18], [19], models trained on fixed-size maps often perform poorly on maps of different scales. In conventional deep learning tasks [23], input size inconsistencies are often addressed through preprocessing techniques to enforce a unified representation, which is generally unsuitable for path planning problems. In multi-scale scenarios, trajectory lengths can vary significantly across maps, amplifying the attenuation of cumulative rewards in long-horizon tasks [24]. This discrepancy makes long-term value estimation difficult and often destabilizes policy updates, even if the neural network can technically process inputs of varying sizes.

Several studies have addressed training instabilities in reinforcement learning. Florensa et al. [25] gradually expanded the learning range to alleviate sparse-reward stagnation, Kumaar [26] used a β-decay transfer learning framework to accelerate convergence, and Kerzel et al. [27] dynamically increased task difficulty while avoiding policy bias. Furthermore, curriculum reinforcement learning, as a method that gradually trains agents through sequences of tasks from easy to hard, is an effective approach to address low training efficiency and policy instability [12]. However, existing approaches do not directly address MDP stability issues caused by multi-scale map inputs, leaving a gap in systematically handling the interactions between multi-scale map inputs and policy learning.

III. PROBLEM FORMULATION

We consider global path planning for a ground mobile robot in a known map, aiming to find a feasible path from start $s_{start} = (x_s, y_s)$ to goal $s_{goal} = (x_g, y_g)$. The environment is a 2D grid $V \in \{0,1\}^{H\times W}$, where $V_{i,j} = 1$ denotes a traversable cell and 0 an obstacle. The robot moves in eight discrete directions, forming the action space $A = \{a_0, \ldots, a_7\}$.

A path τ is a sequence of states and actions avoiding obstacles, with cost

$$C(\tau) = \sum_{\tau} \sqrt{(s'_i - s_i)^2 + (s'_j - s_j)^2}, \tag{1}$$

and the optimal path τ^* minimizes $C_{\min}(\tau^*)$.

In the RL framework, the agent learns a policy $\pi(a|s)$ to maximize expected cumulative reward

$$J(\pi) = \mathbb{E}_\pi\Big[\sum_{t=0}^{\infty} \gamma^t r_t(s_t, a_t)\Big], \tag{2}$$

equivalently minimizing path cost. The optimal policy is

$$\pi^* = \arg\max_{\pi} J(\pi). \tag{3}$$

Global path planning is NP-hard, so in practice the goal is to efficiently find feasible paths with low cost, balancing optimality and computation.

IV. METHODOLOGY

A. Overview

The overall structure of MS-CRL is illustrated in Fig. 1. UniMS constructs unified agent-centered observations by normalizing spatial scales across maps of different sizes, ensuring scale-invariant state representations (Fig. 1(a)). These unified states are then processed by GLFNet (Fig. 1(b)), which consists of a global feature branch capturing large-scale structural information, a local feature branch focusing on fine-grained details around the agent, and a branch fusion module that integrates both streams into a joint representation. The fused features are subsequently passed to actor and critic networks to output the policy $\pi(a|s)$ and value function $V(s)$, enabling decision-making and value estimation to benefit from multi-scale global-local information.

ProgCRL (Fig. 1(c)) implements progressive curriculum learning. It includes difficulty modeling and task generation, which adjust map size and obstacle complexity, a curriculum sequence that arranges tasks from simple to complex, and PPO-based training leveraging this structured curriculum. This approach guides the agent to gradually accumulate experience, improve convergence, and achieve effective global path planning across multi-scale maps.

B. Progressive Curriculum Reinforcement Learning

To address poor adaptability and low training efficiency in deep reinforcement learning for multi-scale global path planning, we propose ProgCRL,with the structure shown in Fig. 2. This framework guides agents through a human-inspired "easy-to-hard" progressive learning process, progressively stabilizing policy updates.

1) Curriculum Task Generation: To implement the progressive curriculum learning process, we first define the curriculum task space $T = \{t_i\}_{i=1}^{N}$, where each subtask t_i corresponds to a Markov Decision Process (MDP). Based on the characteristics of path planning tasks, we construct two main task dimensions:

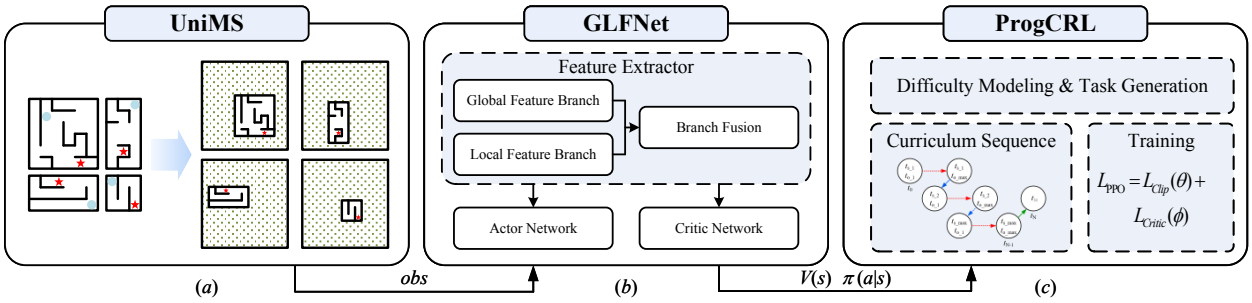

Fig. 1: Overview of the proposed MS-CRL framework. (a) UniMS generates unified agent-centered representations by normalizing spatial scales across different map sizes. (b) GLFNet extracts and fuses global and local features, providing shared representations for the actor and critic to produce the policy $\pi(a|s)$ and value function $V(s)$. (c) ProgCRL progressively models task difficulty by enlarging map sizes and increasing obstacle complexity, guiding the agent's learning from simple to complex scenarios.

Map Size Dimension: Define a task set $T_s = \{t_{s_1}, t_{s_2}, \ldots, t_{s_{\max}}\}$, representing the size of each map in the map set M.

Map Obstacles Dimension: Define a task set $T_c = \{t_{c_1}, t_{c_2}, \ldots, t_{c_{\max}}\}$, representing the obstacle density for maps of the same size.

2) Curriculum Sequence Generation: Based on the above task dimensions, three progressive curriculum units are designed:

Curriculum Unit 1: Obstacle Density Pre-training. In fixed-size maps, obstacle density gradually increases from empty to high-density, training the agent's basic obstacle avoidance and path optimization skills.

Curriculum Unit 2: Map Size Expansion Pre-training. Building on obstacle avoidance, map dimensions gradually increase from $\min(h_{\min}, w_{\min})$ to $\max(h_{\max}, w_{\max})$, with obstacle density adjusted, enabling smooth adaptation to larger, more complex maps.

Curriculum Unit 3: Multi-scale Fine-tuning. Pre-trained models are fine-tuned on maps of multiple sizes, consolidating the agent's ability to efficiently plan paths across varying scales.

3) Curriculum Training Module: The curriculum training module consists of three components:

Curriculum Content Module: Provides map sequences $\{M_{i,1}, \ldots, M_{i,j}\}$ for each subtask t_i in the curriculum sequence $T = \{t_i\}_{i=1}^{N}$, updating sequences as the subtask progresses.

Curriculum Monitoring Module: Evaluates episode statistics such as success rate and task completion to determine when to advance to the next subtask.

Vectorized Deep Reinforcement Learning: To improve training efficiency and sample collection speed, multiple environment instances are executed in parallel, each corresponding to different map samples and random seeds. Proximal Policy Optimization (PPO) [28] is used as the underlying reinforcement learning algorithm. PPO is based on the Actor-Critic architecture, providing good training stability and sample efficiency. The core loss function is:

$$L(\theta, \phi) = \mathbb{E}_t[L_{Clip}(\theta) - c_1 L_{Critic}(\phi)], \tag{4}$$

with the clipped loss defined as:

$$L_{Clip}(\theta) = \mathbb{E}_t[\min(r(\theta)\hat{A}_t, \text{clip}(r(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A}_t)], \tag{5}$$

where $r(\theta) = \pi_\theta(a|s)/\pi_{\theta_{\text{old}}}(a|s)$ is the importance sampling ratio and $\hat{A}_t$ is the advantage estimate. Parallel execution introduces randomness that increases data diversity, significantly accelerating policy learning while maintaining training stability.

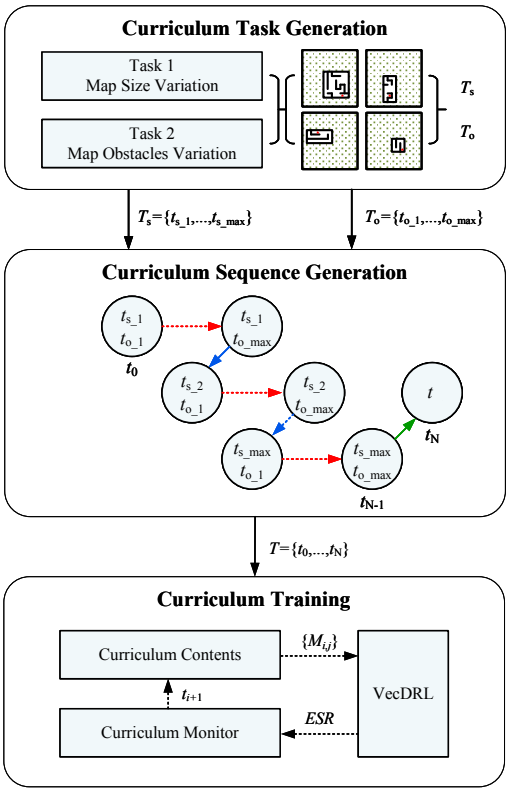

Fig. 2: Overview of ProgCRL.

C. Unified Multi-Scale Representation

Egocentric spatial representation is a core strategy in human navigation, encoding surrounding objects relative to the agent rather than a fixed coordinate system, and dynamically updating during movement while maintaining spatial consistency [29], [30]. Inspired by this, we design UniMS for deep reinforcement learning. By centering observations on the agent's current position, the method ensures locality and consistency, reducing dependence on map size and absolute coordinates. To standardize the input structure, maps are expanded into fixed-size matrices via scale extension techniques, preserving complete environmental information while enabling multi-scale adaptability.

Assume a set of maps $\mathbb{M} = \{M_i\}_{i=1}^{N}$, where each map M_i has dimensions $h_i \times w_i$. Let $h_{\max} = \max_i h_i$ and $w_{\max} = \max_i w_i$ denote the largest height and width in $\mathbb{M}$. The proposed UniMS constructs a unified state space, enabling maps of arbitrary sizes to be embedded into a standardized representation. This process can be summarized as follows:

$$S = \mathbb{R}^{H \times W}, \quad H \geq 2h_{\max} + 1, \; W \geq 2w_{\max} + 1. \tag{6}$$

Each map M_i is expanded to a matrix of size $H \times W$, with extended regions padded by obstacle values. The agent's position is centered in the expanded matrix, anchoring the state representation around the robot, as illustrated in Fig. 3. This ensures a consistent egocentric perspective during policy learning, allowing the policy to plan paths relative to the agent and goal. By standardizing the observation space, this method eliminates the influence of map-size variations on feature extraction and decision-making, improving the policy's stability and transferability across maps.

D. Global-Local Fusion Network

Building on the Global-Scale Anchored Observation, we propose GLFNet, which comprises a feature extractor, an actor network, and a critic network, as shown in Fig. 4.

1) Global Feature Branch: The global branch captures coarse-scale structural information using three convolutional blocks, each with convolution, Batch Normalization, and ReLU, and down-sampling to maintain a large receptive field.

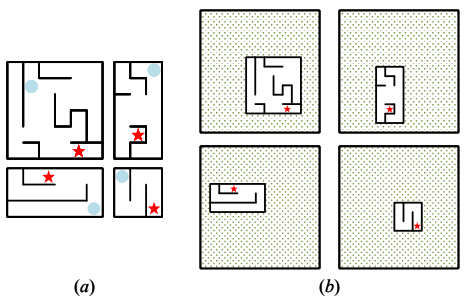

Fig. 3: Illustration of UniMS, showing the agent-centered multi-scale observation construction.

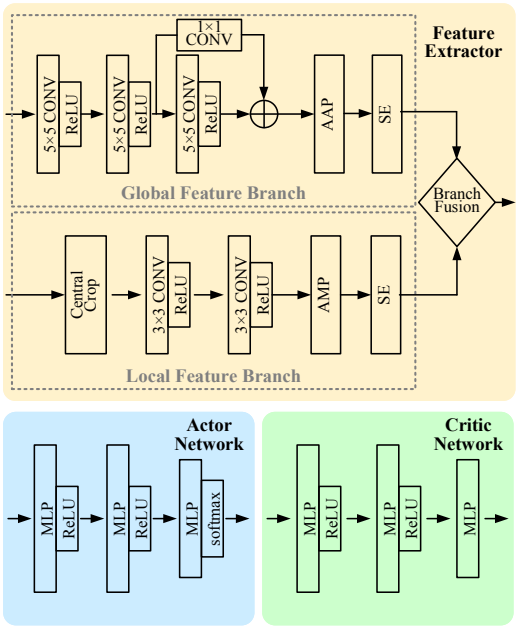

Fig. 4: Illustration of GLFNet, including the Feature Extractor, Actor network, and Critic network.

A cross-layer residual connection mitigates semantic loss by up-sampling the second block's output, aligning it with a 1×1 convolution, and adding it to the third block's output.

An adaptive average pooling (AAP) layer ensures fixed spatial dimensions, while a Squeeze-and-Excitation (SE) module [31] enhances channel-wise attention by weighting semantically rich channels and suppressing less informative ones.

2) Local Feature Branch: To capture fine-grained features around the robot, a local feature extraction branch is designed. A Central Crop module extracts a fixed-size window centered on the agent's position, simulating local perception (Fig. 5).

Owing to the limited spatial scope, the local branch employs a shallow two-layer convolution with 3×3 kernels, in contrast to the global branch's 5×5, capturing fine edges and spatial details while avoiding excessive abstraction.

An adaptive max pooling (AMP) layer produces feature maps with fixed spatial dimensions, preserving the most responsive key features within the local region. This is followed by a Squeeze-and-Excitation (SE) module to enhance channel-wise attention, further improving sensitivity to subtle environmental variations.

3) Feature Fusion: The Feature Extractor captures coarse global features and fine local features. A Feature Fusion mechanism integrates them into a high-dimensional semantic vector as input for the Actor and Critic networks. Specifically, the global and local feature maps are flattened and combined via learnable weights α:

$$f_{Fused} = \alpha \cdot f_{Global} + (1 - \alpha) \cdot f_{Local}. \tag{7}$$

During training, α is optimized via backpropagation to balance the contribution of global and local features.

4) Actor Network and Critic Network: The Actor and Critic networks use the feature vector from the Feature Extractor for policy learning and state-value estimation. Both share a fully connected architecture, with the Actor producing a probability distribution over actions via Softmax, and the Critic outputting a scalar state value.

V. EXPERIMENTAL

A. Experimental Setup

1) Datasets: The dataset is designed to support progressive curriculum learning and is divided into three units. Units 1 and 2 (Phases 1–6) each contain 256 maps per phase, where Unit 1 consists of 32×32 maps with 0, 3, and 5 obstacles (Phases 1–3) and Unit 2 consists of 64×64 maps with 0, 3, and 5 obstacles (Phases 4–6). Obstacles are randomly placed with randomized occupied areas. Unit 3 (Phase 7) includes 256 multi-scale maps of four sizes (32×32, 32×64, 64×32, and 64×64), each containing 5 randomly placed obstacles.

2) Metrics: The primary evaluation metrics used in the experiments include:

Episode Reward (ER): Average cumulative reward, reflecting policy learning and convergence.

Episode Success Rate (ESR): Proportion of successful episodes, indicating task completion effectiveness.

Path Cost Ratio (PCR): Normalized path cost; values near 1 indicate paths close to optimal.

Planning Time Ratio (PTR): RL planning time relative to Dijkstra's; lower values imply higher efficiency.

Success weighted by Path Cost (SPC): Balances success and efficiency by penalizing failures and long paths [32].

The Time-weighted SPC (TSPC) extends the standard SPC by incorporating planning time, using a logarithmic function to smooth extreme ratios. Its definition is given in Eq. (8):

$$TSPC = \frac{1}{N} \cdot \sum_{i=1}^{N} S_i \cdot \left(\frac{C_{\min}(\tau^*)}{C(\tau)} \cdot \log\left(1 + \frac{1}{PTR}\right) \right). \quad (8)$$

3) Eperimental Parameters: The parameters used in the experimental process are listed in Table I.

The reward function used in the experimental process is designed as follows:

$$r = r_{goal} + r_{distance} + r_{repeat} + r_{negative}. \quad (9)$$

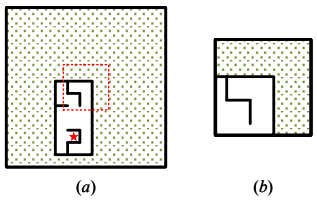

Fig. 5: Illustration of Central Crop module

TABLE I: Experimental Parameters

Parameters	Value	Parameters	Value
H, W	128	n_epochs	10
learning_rate	0.0003	gamma	0.99
num_envs	256	gae_lambda	0.95
n_steps	128	clip_range	0.2
batch_size	2048	vf_coef	0.5

TABLE II: Comparison of Path Planing Performance Across Methods

Methods	PCR	SPC	PTR	TSPC
Dijkstra	1.000	1.000	1.000	0.693
A*	1.000	1.000	0.159	1.764
RRT	1.145	0.872	0.142	1.832
Informed RRT	1.138	0.876	0.118	2.041
Ours(ESR=0.50)	1.196	0.850	0.0160	3.485
Ours(ESR=0.75)	1.194	0.852	0.0153	3.529
Ours(ESR=0.90)	**1.152**	**0.881**	**0.0154**	**3.637**

4) Implementation Details: To ensure experimental consistency, all experiments are conducted on the same software and hardware platform. The computing environment consists of Ubuntu 22.04.4 LTS, an Intel(R) Xeon(R) Gold 6226R CPU @ 2.90GHz, a single NVIDIA Tesla T4 GPU with 16GB memory, Python 3.12.0, and PyTorch 2.5.1 with CUDA 12.4 support.

B. Experimental Results

1) MS-CRL Performance: This experiment evaluates the performance of the proposed MS-CRL. Fig. 6 shows representative results on the multi-scale dataset, where blue trajectories denote our method and yellow ones the Dijkstra baseline. The proposed method consistently produces feasible paths across scales, aligning well with Dijkstra while ensuring efficiency in multi-scale scenarios.

From the training curves Fig. 7, the algorithm gradually acquires goal-reaching, obstacle avoidance, and path optimization skills. In Phases 1–3, higher obstacle density increases the ER minimum, indicating fewer collisions. ER drops at the 64×64 scale (Phase 4) but recovers quickly, showing effective transfer learning. The progressive curriculum stabilizes training, reduces collisions, and enhances sample efficiency, with phase transitions triggered at ESR 0.75 and final convergence at ESR 0.9.

Table II compares methods using ESR, PCR, SPC, PTR, and TSPC. Dijkstra and A* achieve optimal PCR and SPC but incur high planning costs. A* is faster (15.9% of Dijkstra's time) yet still yields lower TSPC. Our method, evaluated at ESR 0.50, 0.75, and 0.90, shows decreasing PCR (1.196→1.152) and improving SPC (0.850→0.881), indicating paths closer to optimal with training. PTR falls from 0.0160 to 0.0154, just 1.54% of Dijkstra's time, reflecting faster inference. Meanwhile, TSPC rises (3.485→3.637), confirming higher success, lower cost, and faster planning jointly.

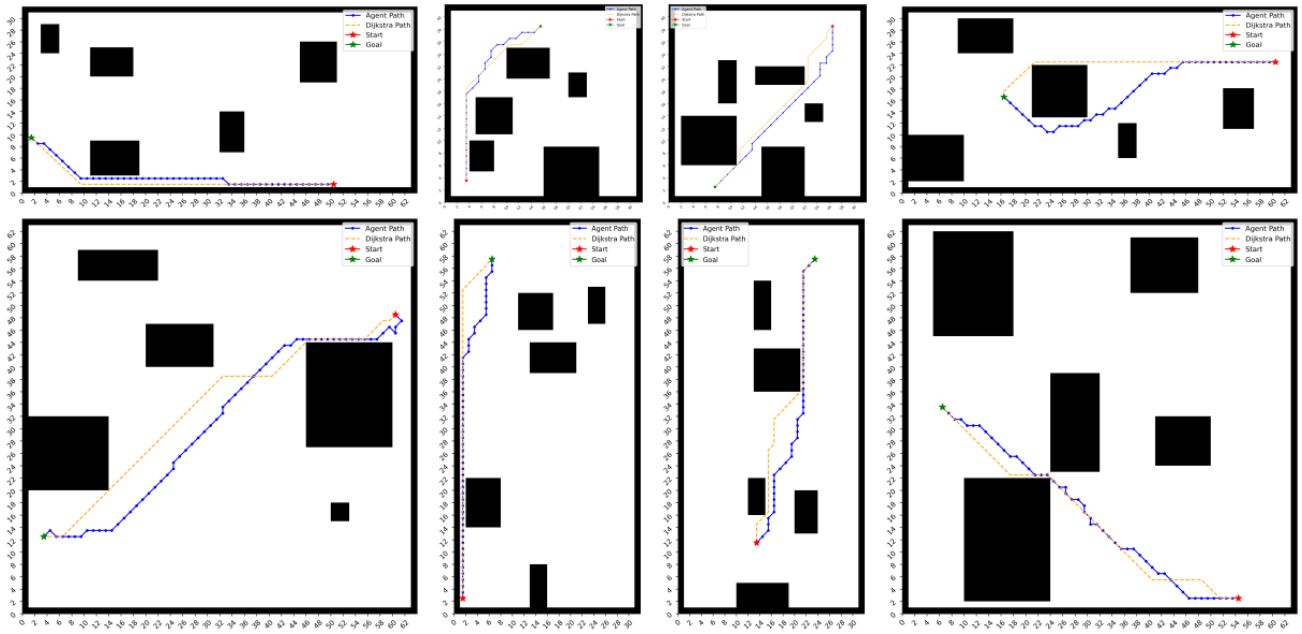

Fig. 6: Path planning results on multi-scale map datasets. Blue lines represent the paths planned by the proposed algorithm,while yellow lines correspond to paths generated by the Dijkstra algorithm. Only four map sizes (32×32, 32×64, 64×32, and 64×64) are shown, with two example maps per size due to space constraints.

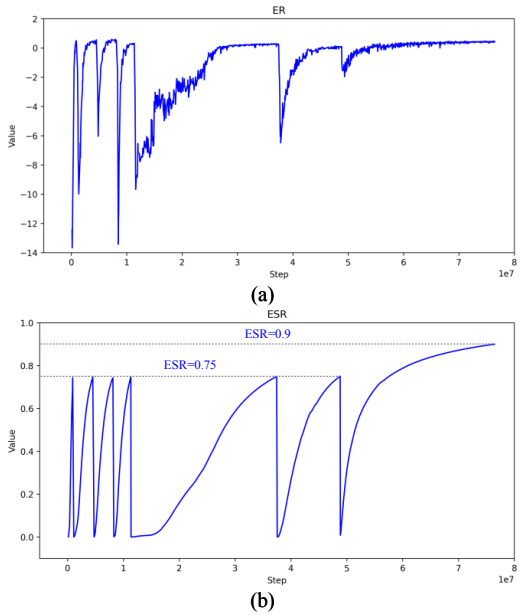

Fig. 7: ER and ESR curves during curriculum learning.

2) Impact of Curriculum Design: This experiment compares two training strategies: (1) progressive training from simpler maps to the target dataset (ProgCRL), and (2) direct training on the target dataset without curriculum. Fig. 8 compares the two, showing ER (a) and ESR (b). ProgCRL yields faster convergence and higher final performance, while direct training stalls at ER = -4.201 and ESR = 0.1003. As shown in Table III, our method achieves lower PCR and higher SPC, PTR, and TSPC, confirming that curriculum learning improves both training efficiency and planning quality.

TABLE III: Comparison of Path Planning Performance With Curriculum

Methods	PCR	SPC	PTR	TSPC
Without Curriculum	1.891	0.549	0.2680	1.984
Ours	**1.152**	**0.881**	**0.0154**	**3.637**

According to the transfer learning metrics in [12], ProgCRL achieves faster "time to threshold" and a clear jumpstart effect, with higher initial success rates and rewards on Unit 3 compared to training from scratch. This demonstrates that the proposed MS-CRL framework accelerates convergence, enhances policy quality, and improves robustness on more complex tasks.

3) Feature Extractor Comparison: This experiment evaluates the proposed Global-Local Multi-Scale Feature Fusion Network against standard MLP and CNN extractors, commonly used in RL-based path planning [33]. Simple architectures are often preferred to maintain online training efficiency and stable actor-critic optimization.

Table IV shows that the proposed network consistently achieves lower ER and higher ESR than CNN and MLP models, maintaining stable performance in later phases (5–7) where baselines stagnate or fail. With only 2.89M parameters—much fewer than MLP (563M) and CNN (5.06M)—it effectively leverages global and local multi-scale information

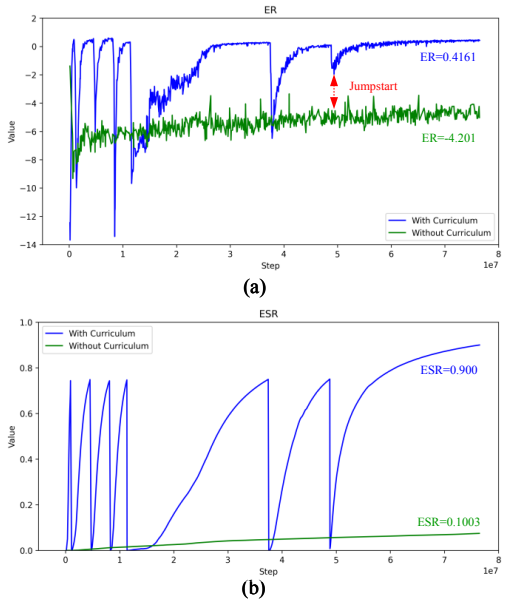

Fig. 8: ER and ESR curves for training with and without curriculum learning.

TABLE IV: Comparison of ER and ESR Across Curriculum Phases for Different Feature Extractors

Method	Metric	P1	P2	P3	P4	P5	P6	P7
CNN Based	ER	0.621	0.636	0.623	0.345	×	×	×
	ESR	0.701	0.730	0.747	0.723	×	×	×
MLP Based	ER	0.542	0.515	0.374	0.246	×	×	×
	ESR	0.725	0.748	0.750	0.734	×	×	×
Ours	ER	**0.492**	**0.537**	**0.527**	**0.303**	**0.252**	**0.271**	**0.416**
	ESR	**0.744**	**0.745**	**0.743**	**0.747**	**0.750**	**0.750**	**0.900**

for efficient and robust path planning.

4) Ablation Studies: This experiment conducts ablation studies by selectively removing or modifying the global branch, local branch, fusion modules, SE modules, and pooling layers, evaluating their impact on curriculum learning and path planning performance.

Table V summarizes ER and ESR across phases. Removing SE slows progression and reduces stability in phase 7 (ESR 0.744 vs. 0.900). Removing the residual connection in the Global Branch impairs convergence, and omitting the Local Branch limits training to phase 6, highlighting their necessity. Excluding Branch Fusion reduces efficiency and stability in later phases, showing its importance for integrating global and local features.

Pooling experiments indicate that AAP in the Global Branch combined with AMP in the Local Branch achieves the best trade-off between context and detail, improving training efficiency and stability. Other configurations (AAP+AAP, AMP+AMP, AMP+AAP) slow learning or destabilize training.

Regarding final performance (Table VI), removing SE or Branch Fusion reduces SPC (0.742, 0.738) and TSPC (3.040, 2.886). The AAP (Global) + AMP (Local) setting achieves the best balance (PCR 1.152, SPC 0.881, PTR 0.0154, TSPC 3.637), confirming that the Global-Local Dual Branch Feature Extractor, together with SE and Branch Fusion, enhances both training efficiency and path planning quality.

TABLE V: Comparison of ER and ESR Across Curriculum Phases for Different Ablation Settings

Methods	Metric	P1	P2	P3	P4	P5	P6	P7
SE	ER	0.467	0.563	0.522	0.320	0.214	0.260	-0.33
	ESR	0.731	0.741	0.747	0.745	0.750	0.750	0.744
Residual	ER	0.502	0.685	0.586	0.257	-0.011	-5.14	×
	ESR	0.729	0.749	0.738	0.737	0.750	0.015	×
Local Branch	ER	0.474	0.428	0.411	0.337	0.124	-2.45	×
	ESR	0.723	0.747	0.750	0.748	0.749	0.304	×
Branch Fusion	ER	0.235	0.539	0.623	0.352	0.213	0.128	-1.147
	ESR	0.706	0.740	0.749	0.749	0.749	0.754	0.440
AAP+AAP	ER	0.496	0.518	0.539	0.277	0.229	-0.098	0.353
	ESR	0.738	0.743	0.749	0.731	0.749	0.750	0.880
AMP+AMP	ER	0.412	0.554	0.550	0.345	0.253	-0.027	0.332
	ESR	0.729	0.743	0.743	0.750	0.748	0.750	0.742
AMP+AAP	ER	0.524	0.506	0.613	0.346	0.207	0.109	0.145
	ESR	0.741	0.748	0.750	0.747	0.748	0.750	0.791
Ours	ER	**0.492**	**0.537**	**0.527**	**0.303**	**0.252**	**0.271**	**0.416**
	ESR	**0.744**	**0.745**	**0.743**	**0.747**	**0.750**	**0.750**	**0.900**

TABLE VI: Comparison of Path Planning Performance for Different Ablation Settings

Methods	PCR	SPC	PTR	TSPC
SE	1.393	0.742	0.0164	3.040
Branch Fusion	1.378	0.738	0.0204	2.886
AAP+AAP	1.307	0.793	0.0157	3.271
AMP+AMP	1.358	0.758	0.0179	3.035
AMP+AAP	1.336	0.776	0.0176	3.125
Ours	**1.152**	**0.881**	**0.0154**	**3.637**

VI. CONCLUSION

This paper presents MS-CRL, a progressive Multi-Scale Curriculum Reinforcement Learning framework for global path planning across multi-scale maps. By integrating ProgCRL, UniMS, and GLFNet, the framework enables stable policy learning and efficient acquisition of multi-scale path planning strategies. Experiments demonstrate that MS-CRL improves path success rate, path quality, and planning efficiency, while effectively handling multi-scale map variability. Future work will focus on extending MS-CRL to multi-robot cooperative planning and richer environment representations.

REFERENCES

[1] Z. Li, W. Wu, Y. Guo, J. Sun, and Q.-L. Han, "Embodied multi-agent systems: A review," *IEEE/CAA Journal of Automatica Sinica*, vol. 12, no. 6, pp. 1095–1116, 2025.

[2] Y. Long, W. Wei, T. Huang, Y. Wang, and Q. Dou, "Human-in-the-loop embodied intelligence with interactive simulation environment for surgical robot learning," *IEEE Robotics and Automation Letters*, vol. 8, no. 8, pp. 4441–4448, 2023.

[3] L. Li, Y. Li, X. Zhang, Y. He, J. Yang, B. Tian, Y. Ai, L. Li, A. Nüchter, and Z. Xuanyuan, "Embodied intelligence in mining: Leveraging multi-modal large language models for autonomous driving in mines," *IEEE Transactions on Intelligent Vehicles*, vol. 9, no. 5, pp. 4831–4834, 2024.

[4] K. Karur, N. Sharma, C. Dharmatti, and J. E. Siegel, "A survey of path planning algorithms for mobile robots," *Vehicles*, vol. 3, no. 3, pp. 448–468, 2021.

[5] L. Liu, X. Wang, X. Yang, H. Liu, J. Li, and P. Wang, "Path planning techniques for mobile robots: Review and prospect," *Expert Systems with Applications*, vol. 227, p. 120254, 2023.

[6] Z. Jian, S. Zhang, S. Chen, Z. Nan, and N. Zheng, "A global-local coupling two-stage path planning method for mobile robots," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5349–5356, 2021.

[7] Z. Cheng, F. Zhou, X. Xu, K. Zhang, G. Trajcevski, T. Zhong, and P. S. Yu, "Information cascade popularity prediction via probabilistic diffusion," *IEEE Transactions on Knowledge and Data Engineering*, vol. 36, no. 12, pp. 8541–8555, 2024.

[8] Y. Cao, H. Zhao, Y. Cheng, T. Shu, Y. Chen, G. Liu, G. Liang, J. Zhao, J. Yan, and Y. Li, "Survey on large language model-enhanced reinforcement learning: Concept, taxonomy, and methods," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 36, no. 6, pp. 9737–9757, 2024.

[9] F. Zhao, Q. Yu, Z. Wu, and Z. Yang, "A hybrid a* algorithm with deep reinforcement learning for path planning and collision avoidance of mass in navigation," in *2025 5th International Conference on Artificial Intelligence and Industrial Technology Applications (AIITA)*, pp. 1682–1687, IEEE, 2025.

[10] R. Karthikeyan and B. S. Rani, "An innovative approach for obstacle avoidance and path planning of mobile robot using adaptive deep reinforcement learning for indoor environment," *Knowledge-Based Systems*, vol. 326, p. 114058, 2025.

[11] H. Xiao, L. Fu, C. Shang, Y. Lin, L. Yue, and Y. Fan, "Collaborative energy-saving path planning of unmanned surface vehicle cluster based on multi-head attention mechanism and multi-agent deep reinforcement learning," *Engineering Applications of Artificial Intelligence*, vol. 161, p. 112078, 2025.

[12] S. Narvekar, B. Peng, M. Leonetti, J. Sinapov, M. E. Taylor, and P. Stone, "Curriculum learning for reinforcement learning domains: A framework and survey," *Journal of Machine Learning Research*, vol. 21, no. 181, pp. 1–50, 2020.

[13] E. W. Dijkstra, "A note on two problems in connexion with graphs," in *Edsger Wybe Dijkstra: his life, work, and legacy*, pp. 287–290, 2022.

[14] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[15] S. LaValle, "Rapidly-exploring random trees: A new tool for path planning," *Research Report 9811*, 1998.

[16] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The international journal of robotics research*, vol. 30, no. 7, pp. 846–894, 2011.

[17] L. C. Garaffa, M. Basso, A. A. Konzen, and E. P. de Freitas, "Reinforcement learning for mobile robotics exploration: A survey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 8, pp. 3796–3810, 2021.

[18] R. van Essen, E. van Henten, and G. Kootstra, "Uav-based path planning for efficient localization of non-uniformly distributed weeds using prior knowledge: A reinforcement-learning approach," *Computers and Electronics in Agriculture*, vol. 237, p. 110651, 2025.

[19] H. Bayerlein, M. Theile, M. Caccamo, and D. Gesbert, "Uav path planning for wireless data harvesting: A deep reinforcement learning approach," in *GLOBECoM 2020-2020 IEEE global communications conference*, pp. 1–6, IEEE, 2020.

[20] P. Chen, Q. Liu, Y. Li, and S. Ma, "An environmental-complexity-based navigation method based on hierarchical deep reinforcement learning," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5119–5125, IEEE, 2024.

[21] K. Weerakoon, A. J. Sathyamoorthy, U. Patel, and D. Manocha, "Terp: Reliable planning in uneven outdoor environments using deep reinforcement learning," in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 9447–9453, IEEE, 2022.

[22] J. Rückin, L. Jin, and M. Popović, "Adaptive informative path planning using deep reinforcement learning for uav-based active sensing," in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 4473–4479, IEEE, 2022.

[23] X. Yang, S. Tu, B. Hu, S. Xu, Q. Yang, G. Yin, and Z. Wang, "Implicit authentication method based on image temporal features," *Pattern Recognition*, p. 112564, 2025.

[24] H. Jasso-Fuentes and T. Prieto-Rumeau, "Constrained markov decision processes with non-constant discount factor," *Journal of Optimization Theory and Applications*, vol. 202, no. 2, pp. 897–931, 2024.

[25] C. Florensa, D. Held, M. Wulfmeier, M. Zhang, and P. Abbeel, "Reverse curriculum generation for reinforcement learning," in *Conference on robot learning*, pp. 482–495, PMLR, 2017.

[26] A. N. Kumaar and S. Kochuvila, "Mobile service robot path planning using deep reinforcement learning," *IEEE Access*, vol. 11, pp. 100083–100096, 2023.

[27] M. Kerzel, H. B. Mohammadi, M. A. Zamani, and S. Wermter, "Accelerating deep continuous reinforcement learning through task simplification," in *2018 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–6, IEEE, 2018.

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

[29] R. L. Klatzky, "Allocentric and egocentric spatial representations: Definitions, distinctions, and interconnections," in *Spatial cognition: An interdisciplinary approach to representing and processing spatial knowledge*, pp. 1–17, Springer, 1998.

[30] R. F. Wang and E. S. Spelke, "Updating egocentric representations in human navigation," *Cognition*, vol. 77, no. 3, pp. 215–250, 2000.

[31] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7132–7141, 2018.

[32] P. Anderson, A. Chang, D. S. Chaplot, A. Dosovitskiy, S. Gupta, V. Koltun, J. Kosecka, J. Malik, R. Mottaghi, M. Savva, *et al.*, "On evaluation of embodied navigation agents," *arXiv preprint arXiv:1807.06757*, 2018.

[33] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, "Stable-baselines3: Reliable reinforcement learning implementations," *Journal of machine learning research*, vol. 22, no. 268, pp. 1–8, 2021.