Guided Distillation and Risk Adaptive Evolution for Multi-Robot Navigation

Xuyang Li[1], Jianwu Fang[1], Lin Li[2], Boyuan Chen[1], Guangliang Li[3], Jianru Xue[1]*

[1]State Key Laboratory of Human-Machine Hybrid Augmented Intelligence, Institute of Artificial Intelligence and Robotics, Xi'an Jiaotong University

[2]School of Mechanical and Aerospace Engineering, Nanyang Technological University

[3]College of Electronic Engineering, Ocean University of China

lixuyang@stu.xjtu.edu.cn, fangjianwu@mail.xjtu.edu.cn, 2194111706@stu.xjtu.edu.cn

l.lin@ntu.edu.sg, guangliangli@ouc.edu.cn, jrxue@mail.xjtu.edu.cn

Abstract

Recent advancements in multi-robot navigation have explored methods that combine Large Language Models (LLMs) for tasks like scene understanding or high-level decision-making. However, these approaches face challenges with high inference latency and potential hallucinations. To address these challenges, we propose a knowledge-driven Reinforcement Learning (RL) framework, GUIDER, that utilizes an LLM in two different offline roles. First, we leverage the LLM as an *offline knowledge source*. Its expertise is distilled into a compact model, which is applied only when the RL agent is uncertain about its own value estimates and the model itself is confident in its prediction. Additionally, we utilize the LLM as an *offline semantic engine*. This process translates the LLM's high-level understanding of situational risk into a dynamic adjustment of the RL agent's behavioral style, evolving a function that optimally balances conservative and aggressive actions. We conduct extensive experiments in both Terrestrial and Maritime settings. Across all maritime scenarios (3–12 robots), GUIDER improves the task success rate and reduces the collision rate significantly compared to the state-of-the-art RL-based multi-robot navigation methods.

Code — https://github.com/lixuyang-m/GUIDER

Introduction

Safe and efficient multi-robot navigation is a challenging problem, especially in complex scenarios with dense obstacles and limited perception range (Garg et al. 2024; Rybczak, Popowniak, and Lazarowska 2024; Li et al. 2023). Current approaches ranging from classical planners like potential fields (Fan et al. 2020) and reciprocal velocity obstacles (Van den Berg et al. 2008) to modern Deep Reinforcement Learning (DRL) (Wang et al. 2024) are generally short of considering latent context or implicit rules of interaction. Relatively, the extensive world knowledge and reasoning capabilities of LLMs offer a promising avenue for incorporating such contextual understanding (Guo et al. 2024) into multi-robot navigation. Therefore, this raises a pivotal research question: **how can we effectively transfer the knowledge-driven capabilities of LLMs within multi-robot systems?**

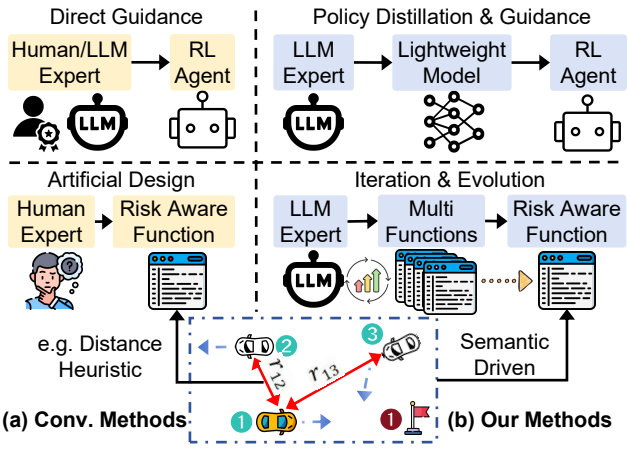

Figure 1: Conventional methods vs Ours. Different from conventional pipelines rely on slow external advice and hand-tuned risk rules (e.g., distance-based heuristics), our method first distills LLM knowledge for efficient and reliable guidance, and then repurposes the LLM as a semantic engine to evolve complex, context-aware risk policies.

Recent studies on LLM-assisted multi-robot navigation often adopt a system-level decoupling paradigm. In these architectures, the LLM serves as a high-level planner that can decompose goals (Bai et al. 2025; Zhou et al. 2025), coordinate multiple agents (Yu et al. 2025; Rajvanshi et al. 2025; Shen et al. 2025) or perform replanning (Skreta et al. 2024; Zhou, Zhou, and Liu 2024), whereas a separate fast controller generates low-level actions. However, after the LLM completes inference, the environment may have undergone significant changes, making it difficult to cope with scenarios that require rapid response. Some works add feedback (Wang et al. 2023) or perceptual grounding (Skreta et al. 2024) to reduce unreliable LLM output, but they do not solve the core problem of converting knowledge from a potentially unreliable source into a rapid and dependable policy. These observations underscore the need for a framework that translates an LLM's knowledge-driven capability

*Corresponding author.

directly into a real-time policy and ensures the reliability.

To address the above issues and obtain a real-time compatible and reliable policy, we propose **GUIDER**, an RL framework based on a two-stage offline methodology, which is expected to effectively transfer the knowledge-driven capabilities of LLMs for multi-robot navigation. In contrast to conventional approaches (Figure 1a), the first stage of GUIDER (Figure 1b) treats the LLM as an **offline knowledge source** and distills its expertise into a compact model via incorporating dynamic confidence into the loss function during training. To further ensure reliability at execution time, a secondary confidence filter is applied during inference and uses the obtained model to provide efficient guidance for RL agents when their value estimation is uncertain. In the second stage, GUIDER employs the LLM as a **semantic operator**. The LLM merges and refines candidate risk policies based on their underlying logic and high-level objectives, allowing it to automatically discover a complex, state-dependent risk function. With our two-stage GUIDER approach, we can obtain a real-time navigation policy that integrates the LLM's knowledge for multi-robot navigation without requiring any online calls to the LLM during deployment. The contributions of this work are as follows:

- We propose GUIDER, a novel two-stage offline framework that first distills the LLM's expertise into a compact model for efficient guidance, while the second repurposes the LLM to evolve a risk-adaptive function.
- We introduce a mechanism that guidance is requested only when the agent is uncertain about its value estimates, and then vetted by a dual-confidence filter that rejects low-quality advice. Furthermore, we leverage the LLM to perform semantic evolution, where it acts as a logical operator to combine and mutate risk policies.
- We provide extensive validation in diverse environments, using a comprehensive suite of metrics that assess safety, task completion, and efficiency. In the most challenging maritime scenarios, GUIDER improves the task-success rate and reduces the collision rate significantly against state-of-the-art RL baselines.

Related Work

Multi-robot navigation is a critical technology that underpins a wide array of applications, from automated warehouse logistics to search-and-rescue operations. Reactive planners such as Artificial Potential Fields (APF) (Fan et al. 2020) and Reciprocal Velocity Obstacles (RVO) (Van den Berg et al. 2008) perform well in static scenes but degrade in dynamic, cluttered environments. DRL has therefore been adopted for multi-robot navigation, enabling agents to learn adaptable policies directly from interaction (Feng et al. 2024; Chang et al. 2023; Escudie, Matignon, and Saraydaryan 2024) and even to switch among multiple sub-policies online (Paudel, Xiao, and Stein 2024). Yet, like classical planners, these DRL approaches usually lack a channel for injecting common-sense knowledge that could guide reasoning in unfamiliar, unstructured situations.

Most recent multi-robot systems place the LLM at the high level for task decomposition and allocation while leaving low-level controllers to execute the issued commands (Zu et al. 2024; Yu et al. 2025; Rajvanshi et al. 2025; Shen et al. 2025; Zhou et al. 2025). Although this decoupled architecture benefits from the LLM's semantic reasoning, every control cycle must wait for a fresh LLM reply, which substantially inflates latency. An alternative line embeds the LLM directly in the control loop of a *single* agent; with Chain-of-Thought prompting and self-reflection, such methods match or surpass long-trained RL policies (Fu et al. 2024; Wen et al. 2024). However, each LLM inference takes on the order of tens of seconds, making these schemes unsuitable for real-time navigation.

Risk-sensitive control modulates behaviour via metrics such as Conditional Value-at-Risk (CVaR) (Lin, McConnell, and Englot 2023). Existing adaptations employ handcrafted heuristics (Lin et al. 2024), statistical estimates of value distributions (Liu et al. 2023), or control-theoretic CVaR barrier functions (Wang et al. 2025); which operate on low-level numerical features without semantic understanding.

The GUIDER Framework

Problem Formulation

Preliminary We consider a multi-robot navigation scenario where a team of N agents is tasked with reaching their individual goals without collisions in a finite time. The task is inherently a decentralized partially observable Markov decision process (Dec-POMDP). For brevity, we use a single-agent MDP approximation per agent, where s denotes that agent's information state built from its observations. At each time step t, an agent observes a state $s \in \mathcal{S}$ and selects an action $a \in \mathcal{A}$ according to its policy π. After executing the action, the agent receives a scalar reward r and transitions to a new state. The standard objective in RL is to learn a policy π that maximizes the expected cumulative discounted return, $\mathbb{E}[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1}]$, where k is the future step index and the discount factor $\gamma \in [0, 1)$ determines the present value of future rewards. This expectation is represented by the function $Q^\pi(s, a)$, which denotes the expected return for taking action a in state s and following policy π thereafter. However, relying solely on this expectation can be perilous, as it obscures underlying risks where an action might lead to either a high reward or a failure.

Formulation To address this, our framework builds upon Distributional RL (Bellemare, Dabney, and Munos 2017), which moves beyond the expectation to model the full distribution of returns, $Z(s, a)$. This allows us to learn a policy that optimizes for a risk-sensitive measure of this distribution. Specifically, we learn the distribution's quantile function $F^{-1}_{Z(s,a)}(\tau)$, inspired by Implicit Quantile Networks (IQN) (Dabney et al. 2018), *i.e.*, the inverse of the cumulative distribution function for a quantile fraction $\tau \in [0, 1]$. This enables explicit risk-sensitive control. We also take the Conditional Value at Risk (CVaR) in (Dabney et al. 2018) as our objective. For a given state-action pair, CVaR_α measures the expected return within the worst α-fraction of outcomes:

$$\text{CVaR}_\alpha(Z(s, a)) = \mathbb{E}[Z(s, a) | Z(s, a) \leq F^{-1}_{Z(s,a)}(\alpha)], \quad (1)$$

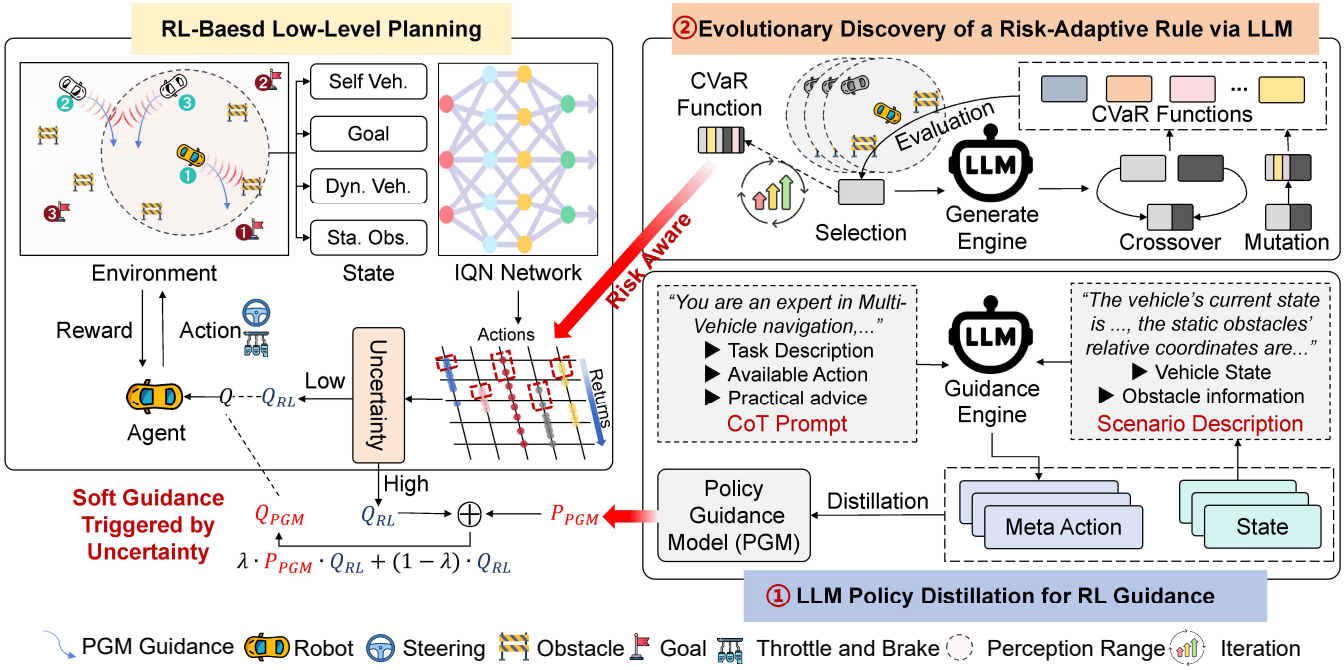

Figure 2: Overview of **GUIDER**: (1) LLM expertise is distilled into a compact model that provides guidance only when the learning agent is uncertain; (2) Offline, the LLM is repurposed to drive an evolutionary process, automatically discovering a risk-adaptive policy that maps the current situation to an appropriate level of risk.

where the risk-level parameter $\alpha \in (0, 1]$ controls the agent's risk appetite. A smaller α value directs the policy to be more risk-averse by optimizing the average over a more extreme set of aforementioned negative outcomes.

State Observation. Each robot' policy operates on a local, ego-centric observation. The state vector S_i for agent i is an aggregation of three components: $S_i = \{S_{\text{internal}}, S_{\text{static}}, S_{\text{dynamic}}\}$. $S_{\text{internal}} = [p_{\text{goal}} - p_{\text{self}}, v_{\text{self}}]$ is represented with the robot's velocity and the relative vector to its goal; S_{static} describes the information on the N_s nearest static obstacles, where each obstacle j is represented by its relative position and radius, $[p_{o_j} - p_{\text{self}}, r_{o_j}]$; S_{dynamic} describes the information on the N_d nearest dynamic robots, where each neighbor k is described by its relative position and absolute velocity, $[p_{d_k} - p_{\text{self}}, v_{d_k}]$.

Action Space. Each robot's policy outputs a discrete action from a set of 9 choices. These actions combine three levels of longitudinal acceleration ($a_v \in \{-0.4, 0, 0.4\}$ m/s^2) and three levels of angular acceleration ($a_\omega \in \{-\frac{\pi}{6}, 0, \frac{\pi}{6}\}$ rad/s^2).

Reward Function. The total reward R is the sum of four components, $R = R_{\text{goal}} + R_{\text{coll}} + R_{\text{dist}} + R_{\text{time}}$, where $R_{\text{goal}} = +50$ is a sparse reward for reaching the destination, $R_{\text{coll}} = -100$ is a large penalty for any collision, $R_{\text{dist}} = d_{t-1} - d_t$ is a potential-based shaping reward based on the distance of current step d_t and previous step d_{t-1} to the goal, and $R_{\text{time}} = -0.1$ is a constant per-step time penalty.

Knowledge Distillation from an LLM for Selective Policy Guidance

In this phase, we train a navigation policy with sparse guidance distilled from an LLM. The Guidance is applied only at states where the agent is uncertain, and it is filtered for reliability before influencing agent's behavior.

Uncertainty Trigger We utilize the spread of the learned return distribution as a proxy for epistemic uncertainty. Our agent adopts IQN (Dabney et al. 2018) to predict K quantile values $Q_{\tau_i}(s, a)$ for each action, setting K=32 with sorted fractions $\tau_1 < \cdots < \tau_K$. For state s_t, we compute the uncertainty metric $\mathcal{U}(s_t)$ by measuring the inter-quantile gap, taking the maximum across all actions:

$$\mathcal{U}(s_t) = \max_a \left[\frac{1}{m} \sum_{i=K-m+1}^{K} Q_{\tau_i}(s_t, a) - \frac{1}{m} \sum_{i=1}^{m} Q_{\tau_i}(s_t, a) \right], \tag{2}$$

where $m = \lfloor 0.1K \rfloor$. To determine when to trigger an intervention, each agent maintains a rolling window W_t of the most recent W=100 uncertainty values. A dynamic threshold is then derived via a percentile parameter P_{thres}: $\tau_{\text{thres}} = \text{Percentile}(W_t, P_{\text{thres}})$. The intervention is triggered solely when $\mathcal{U}(s_t) > \tau_{\text{thres}}$.

LLM Distillation and Dual-Confidence PGM We distill guidance from an LLM into a compact *Policy Guidance Model* (PGM). Each snapshot is passed to the LLM under

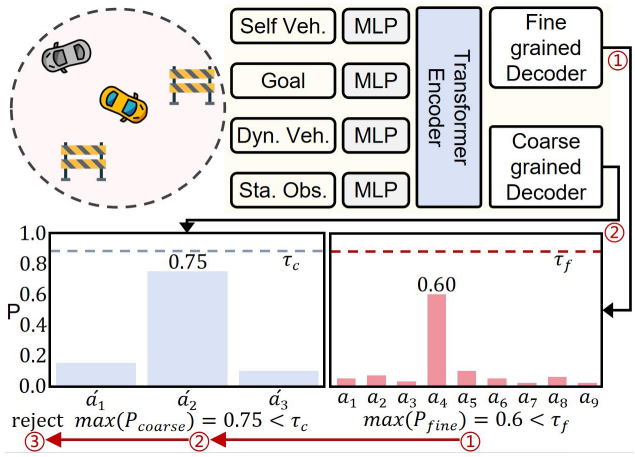

Figure 3: Policy Guidance Model (PGM). A shared transformer encoder feeds a fine head (9 actions) and a coarse head (3 lateral classes). Dual-confidence gating accepts fine, falls back to coarse, or rejects guidance.

a structured prompt that supplies local geometry and task context and asks for an action recommendation using a CoT format. The PGM is trained on the final action label. Prompt templates are given in the supplement file.

The discrete action space contains 9 fine actions that couple lateral direction and speed change and are grouped into 3 lateral classes (**L**eft, **S**traight, **R**ight) in our setup. The PGM uses a shared transformer backbone with two softmax heads: a fine head over the 9 actions (p_{fine}) and a coarse head over the 3 classes (p_{coarse}). A dual-confidence cascade gates guidance (Fig. 3). If $\max(p_{\text{fine}}) \geq \tau_f$, we accept the fine distribution. Otherwise, if $\max(p_{\text{coarse}}) \geq \tau_c$, we accept the coarse distribution and broadcast its class probabilities uniformly to the member fine actions:

$$\tilde{p}_{\text{PGM}}(a \mid s_t) = \frac{p_{\text{coarse}}(c \mid s_t)}{|\mathcal{G}(c)|}, \quad \forall a \in \mathcal{G}(c), \tag{3}$$

where $\mathcal{G}(c)$ is the set of fine actions grouped under lateral class $c, c \in \{\text{Left, Straight, Right}\}$. If neither head passes its threshold we set $\tilde{p}_{\text{PGM}}{=}0$ and revert to standard RL. The $\tilde{p}_{\text{PGM}}$ is used to modify the policy. Compared to directly covering RL actions, we adopt a soft guidance approach via:

$$\hat{Q}(s_t, a) = Q(s_t, a)\Big[(1 - \lambda_t) + \lambda_t \tilde{p}_{\text{PGM}}(a \mid s_t)\Big], \tag{4}$$

where the mixing weight λ_t is linearly annealed from 1.0 to 0.1 over training so that early behavior can benefit from conservative external knowledge while later behavior reflects the agent's own estimates.

LLM-Semantic Evolution of Risk Adaptive Policies

An individual is a function that outputs a continuous $\alpha \in [0, 1]$. Its fitness is evaluated by deploying the fixed Phase 1 agent in a held-out, the highest-density scenario of a given environment (terrestrial or maritime) and computed by:

$$\mathcal{F} = \lambda_1 \cdot \text{TSR} - \lambda_2 \cdot \text{CR} - \lambda_3 \cdot \frac{\text{ACT}}{T_{\text{norm}}}, \tag{5}$$

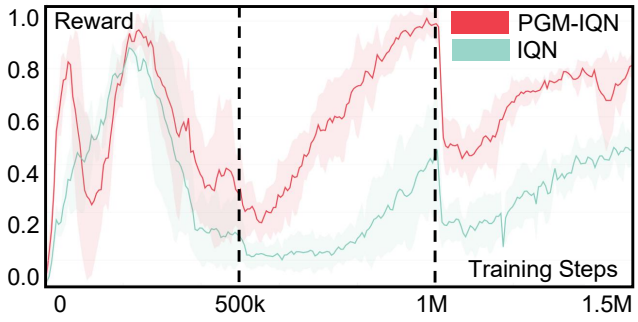

Figure 4: Training performance comparison of our PGM-guided approach (PGM-IQN) vs. the Standard IQN baseline. Results are averaged over 5 random seeds, with solid lines denoting the mean and shaded regions representing the variance. Vertical dashed lines indicate curriculum stages where the environmental difficulty was increased at the 500k and 1M timestep marks.

where TSR, CR, and ACT denote the Team Success Rate, Collision Rate, and Average Completion Time (in seconds), respectively. The coefficients λ_1, λ_2, and λ_3 represent the weights for each objective, and T_{norm} is a normalization factor for time. In our experiments, we set $\lambda_1 = 1.0$ to reward task completion, $\lambda_2 = 2.0$ to impose a strict penalty on collisions for safety, and $\lambda_3 = 0.5$ with $T_{\text{norm}} = 60$ to encourage efficiency within a minute-scale time horizon. For each environment, we randomly sample one Phase 1 checkpoint from several training runs and keep it fixed throughout evolution; the evaluation environment is disjoint from all test environments used later.

Figure 2 (upper right) illustrates our offline search loop. We initialize the population by prompting an LLM to generate diverse seed functions given the physical context and state encoding. Each generation evaluates every individual, preserves the top elites, and fills the next population through tournament selection followed by *semantic* crossover or mutation driven by the LLM: **Semantic Crossover.** The LLM receives the code and performance summaries of two parent functions and produces an offspring that combines favorable clauses from both parents. **Semantic Mutation.** Given one parent and a high-level instruction (e.g., "increase caution near fast head-on obstacles"), the LLM edits the parent's logic in a small, context-aware manner.

The generated code is syntax-checked and unit-tested on random states, where programs that fail to execute or violate the output range are discarded. The process runs for a fixed number of generations ($G_{\max}$) and returns the best individual f^* for the corresponding environment. Search is completely offline; at deployment f^* is evaluated in microseconds, and no LLM query is required.

Experiments

Experimental Setup

We benchmarked GUIDER in a standard **Terrestrial** environment and a **Maritime** environment with unpredictable

disturbances (Acheson 1990; Lin et al. 2024). We evaluate our GUIDER against a range of baselines spanning classical planners and DRL agents. For classical methods, we include Reciprocal Velocity Obstacles (RVO) (Van den Berg et al. 2008) and an Improved Artificial Potential Field (IAPF) (Fan et al. 2020). In the DRL category, we use IQN as a foundational DRL baseline, and also include Improved Expectile-Quantile Networks (IEQN) (Jullien et al. 2025) in the comparison list. To specifically evaluate our evolved risk policy, we compare it against two distinct risk-adaptive baselines. The first is Ada-IQN (Lin et al. 2024), which adjusts risk using a hand-crafted heuristic. The second is ART-IQN (Liu et al. 2023), a method that autonomously adapts its risk-tendency based on an internal uncertainty estimate derived from the learned return distribution. Performance is evaluated on a suite of metrics that assess safety (Collision Rate, **CR**), task completion (Team/Individual Success Rate, **TSR/ISR**), efficiency (Average Completion Time, **ACT**) and energy consumption (**EC**). Finally, we report the average computational overhead per step (**Time**), measured in milliseconds (10^{-3}s). We employ a curriculum learning (CL) policy during training, detailed in Table 1.

Validation of the Policy Guidance Model We first validate the performance of the distilled PGM. Our evaluation demonstrates the effectiveness of the dual-confidence mechanism. In a majority of test cases (**56.94%** coverage), the PGM provides highly accurate (**97.73%**) fine-grained guidance. When less certain, it falls back to provide still-reliable (**96.98%** accuracy) coarse-grained guidance for an additional **17.91%** of cases. Crucially, the model responsibly rejects providing advice in the remaining **25.15%** of low-confidence states, preventing the injection of poor-quality knowledge. Furthermore, PGM is highly efficient with an average inference time of just **5.29ms** on a single GPU, and it achieves a speed-up of over **2000x** compared to the multi-second latency of the original LLM.

As shown in Figure 4, our PGM-guided training (PGM-IQN) is significantly more sample-efficient and converges to a superior policy compared to the standard IQN baseline. This is most evident in the challenging Maritime environment, where our agent achieves nearly double the final mean reward (0.8 vs. 0.45) and over twice the cumulative successes (50k vs. 20k). The PGM's guidance also stabilizes training in this environment by reducing variance.

Parameter	Training Timesteps (millions)		
	0.0-0.5	0.5-1.0	1.0-1.5
N_{robot}	3	5	7
N_{static}	2	4	6
$\min(d_{goal})$(m)	30	35	40
N_{vortex}	2	4	6

Table 1: CL settings: the number of robots (N_{robot}), static obstacles (N_{static}), vortices in the marine environment (N_{vortex}), and the minimum start-to-goal distance ($\min(d_{\text{goal}})$).

Analysis of Evolved Risk-Adaptive Policies We run the semantic genetic algorithm *offline* and independently for the terrestrial and maritime environments. For each environment, we sample one Phase 1 checkpoint and evaluate all candidate functions in the corresponding *highest-density* scenario (12 robots, 4 static obstacles, 6 vortices in the maritime case). These evaluation environments are independent of the testing environment used later. Fitness is computed with Eq. (5), and we use the following hyper-parameters *identical in both environments*: population size $N_p = 10$, number of generations $G_{\max} = 10$, elitism count $N_e = 1$, crossover probability $p_c = 0.7$, mutation probability $p_m = 0.2$, tournament size $T_s = 2$. Each evolution run consumes about 4 GPU-hours of NVIDIA RTX 4090 time, counting only LLM inference and code filtering. All generated programs are syntax-checked and sandboxed before evaluation. Only 3.5% fail these checks and are discarded.

Trigger Condition	Terr. α_{evo}	Mari. α_{evo}
Predicted collision	≈ 0.12	$= 0.0$
High-risk state	$\approx 0.1 - 0.4$	$= 0.0$
Nominal navigation	$\approx 0.12 - 0.88$	$\approx 0.0 - 0.8$

Table 2: Learned piece-wise CVaR mapping for the two environments, with their corresponding numeric α_{evo} values. The complete evolved functions are provided in Appendix.

The evolutionary process converges on the final risk-adaptive policies summarized in Table 2. A striking contrast emerges between two types of environments, revealing how the evolved policies adapt to the environment characteristics.

The two evolved risk–adaptation functions exhibit distinct design choices that correspond to the characteristics of their respective environments. For the *terrestrial* environment, the policy employs a comparatively simple risk model based on linear distance terms and a single–step forward prediction; as the robot approaches its goal the estimated obstacle risk decays, which results in a faster terminal phase of the trajectory. In the *maritime* environment, where motion uncertainty is higher, the policy fuses metrics such as Time to Collision, multistep forward prediction and relative heading checks, and it includes rule-based overrides that force the conservative action $\alpha_{\text{evo}} = 0$ whenever an imminent collision is detected. Hence, the evolutionary search produced a streamlined, goal-oriented policy in the more predictable terrestrial setting, whereas it produced a more cautious, redundancy-based policy in the less predictable maritime setting.

Quantitative Analysis

We now present the quantitative analysis, and the comprehensive results are presented in Table 3.

Safety & Task Completion. CR remains as our primary metric. GUIDER attains the lowest average CR in the Maritime environment (**5.43%**) and the second-best in the Terrestrial environment (1.67%). The slight gap to Ada-IQN (**0.66%**) is offset by Ada-IQN's much lower TSR. In fact, GUIDER delivers the highest TSR in both environments, improving upon the strongest baseline (ART-IQN) by **+13.9 pp** on Maritime and **+13.0 pp** on Terrestrial. These results prove that our GUIDER converts safety gains directly

Env.	Method	TSR↑	ISR↑	CR↓	ACT(s)↓	EC↓	Time↓
Mari.	IAPF	25.70%	79.77%	11.15%	54.46	176.35	**0.15**
	RVO	12.00%	60.33%	31.14%	50.27	138.34	4.96
	IEQN	27.50%±6.93%	76.23%±4.12%	20.45%±2.95%	44.93±9.90	130.10±23.50	2.39±0.08
	ART-IQN	58.32%±5.47%	90.13%±1.64%	9.18%±1.32%	39.49±2.92	125.93±16.42	3.35±0.02
	Ada-IQN	55.70%±8.27%	89.63%±2.56%	9.43%±2.17%	44.41±4.12	137.18±13.93	1.96±0.11
	GUIDER	**72.20%±4.98%**	**94.40%±1.30%**	**5.43%±1.31%**	**34.86±2.56**	**112.83±12.29**	2.46±0.34
Terr.	IAPF	69.60%	93.65%	3.11%	39.15	124.56	**0.31**
	RVO	64.50%	90.03%	9.93%	**29.30**	**37.92**	5.50
	IEQN	53.80%±23.12%	87.64%±8.37%	8.62%±2.46%	34.65±5.88	93.31±22.57	2.70±0.09
	ART-IQN	76.50%±14.92%	94.62%±3.96%	3.27%±1.46%	32.29±3.36	92.16±13.21	3.96±0.39
	Ada-IQN	62.36%±31.72%	84.23%±17.46%	**0.66%±0.28%**	36.28±2.05	93.47±15.04	1.91±0.25
	GUIDER	**89.54%±2.79%**	**98.25%±0.48%**	1.67%±0.39%	33.05±2.24	100.33±18.70	2.17±0.02

Table 3: Quantatitive results of IAPF (Fan et al. 2020), RVO (Van den Berg et al. 2008), IEQN (Jullien et al. 2025), ART-IQN (Liu et al. 2023), Ada-IQN (Lin et al. 2024) and our GUIDER, where all results are averaged over 100 trials for each condition (from 3 to 12 robots). RL algorithms report mean ± SD across 5 seeds. **Bold**: The best. Underline: The second-best.

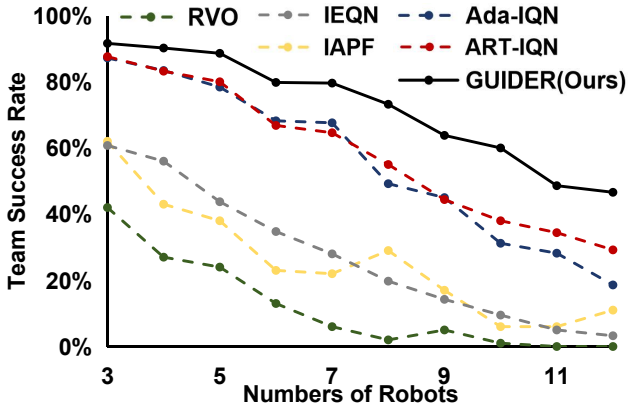

Figure 5: Performance analysis, *w.r.t.*, the number of robots in Maritime Environment.

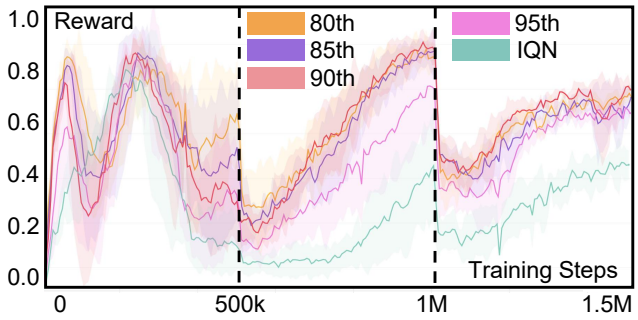

Figure 6: Sensitivity analysis of P_{thres}.

into task completion.

Efficiency. ACT and EC capture the efficiency. On Maritime setting, GUIDER is simultaneously the *safest*, the *most successful*, and the *fastest* (ACT 34.9s vs. the next best 39.5s of ART-IQN). On Terrestrial, RVO records the lowest ACT (29.3s) but pays a ten-fold increase in CR; Ada-IQN achieves extreme safety by almost stopping (ACT 36.3s, TSR 62.4%). GUIDER strikes a superior balance, reaching goals nearly as fast as ART-IQN (33.1s) while preserving a state-of-the-art **TSR of 89.5%**.

Inference Cost. IAPF remains by far the fastest method, requiring less than 0.3 ms per control step on average, whereas the geometric baseline RVO needs roughly 5 ms. Among learning-based methods, GUIDER runs in 2.46 ms which is well under the 10 ms control-loop budget of our simulator and typical real robots.

Ablation Studies

Sensitivity to Percentile Parameter P_{thres}. Figure 6 investigates how the choice of the percentile parameter P_{thres} for PGM guidance influences learning in the Maritime environment. Four settings of P_{thres} (80^{th}, 85^{th}, 90^{th}, and 95^{th}) are trained with five random seeds and compared to the vanilla IQN baseline. As shown in Figure 6, the results display a clear rise-then-fall pattern: the policy with $P_{\text{thres}} = 90^{\text{th}}$ achieves the highest average reward (about 0.80), followed by the 80^{th} and 85^{th} settings (both about 0.76). Pushing P_{thres} to 95^{th} makes the guidance overly conservative, dropping the reward to about 0.69, whereas IQN without any guidance trails far behind (about 0.46). We select $P_{\text{thres}} = 90^{\text{th}}$ for subsequent experiments because this setting delivers the highest average reward and exhibits the narrowest variance band among all evaluated configurations.

Component-Wise Impact Figure 8 illustrates the TSR in the Maritime environment under varying robot densities. Five variants are compared: vanilla IQN, our PGM-guided IQN without risk adaptation GUIDER(w/o RA), and three risk-adaptive versions built on PGM-IQN: GUIDER-Ada, GUIDER-ART, and GUIDER (LLM-evolved policy).

Effect of PGM guidance. As shown in Figure 8, GUIDER(w/o RA) consistently outperforms vanilla IQN regardless of robot density. The visible improvements in TSR confirm that the *offline* PGM distillation alone supplies valuable priors, yielding significantly safer and more reliable

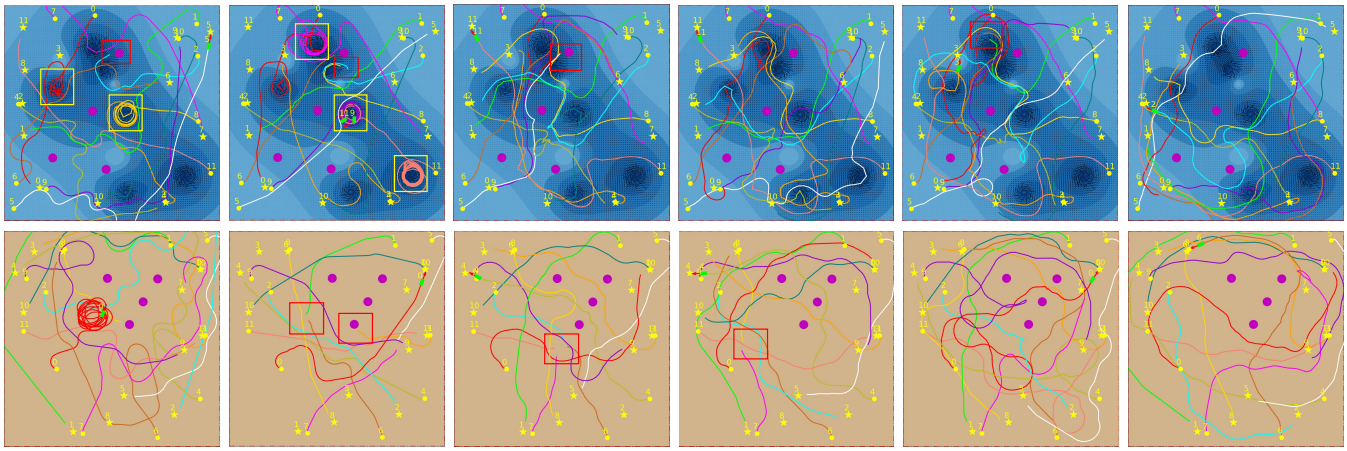

Figure 7: The trajectories of different methods (From left to right: IAPF (Fan et al. 2020), RVO (Van den Berg et al. 2008), IEQN (Jullien et al. 2025), ART-IQN (Liu et al. 2023), Ada-IQN (Lin et al. 2024) and our GUIDER) in Maritime Environment (top row) and Terrestrial Environment (bottom row).

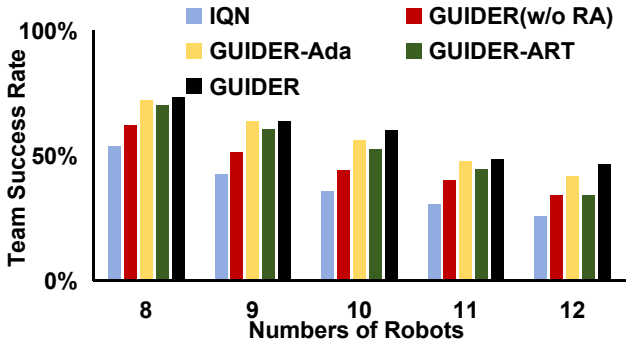

Figure 8: The ablation study of different metrics on RL-Based methods in Maritime Environment.

navigation compared to learning from scratch.

Effect of risk-adaptive policies. When the three risk-adaptive variants built on PGM-IQN are compared, GUIDER yields the greatest TSR in both environments and generally registers the lowest CR, a record that only in isolated cases is fractionally surpassed by GUIDER-Ada. In addition, GUIDER-ART favours speed and economy at the expense of a modest increase in risk, GUIDER-Ada adopts a more cautious yet slower stance, and the LLM-evolved policy embodied by GUIDER secures the most balanced compromise between safety and efficiency, thereby translating that balance into the highest overall task performance.

Qualitative Analysis of Trajectories The trajectory visualizations in Figure 7 provide qualitative results into the behavior of different navigation policies. The classical baselines, IAPF and RVO, still demonstrate frequent failures (highlighted by yellow and red boxes) due to their reactive nature: frequently captured by vortices in the Maritime environment or debilitating oscillations in the dense Terrestrial environment. IEQN avoids deadlocks and provides smoother paths, though it still faces challenges in highly dynamic scenarios. ART-IQN excels in terms of efficiency, achieving faster task completion, but occasionally sacrifices safety, leading to collisions. Ada-IQN offers more cautious behavior, preventing collisions but at the expense of speed and efficiency. In stark contrast, GUIDER exhibits superior planning intelligence, navigating both environments with fluid and coordinated maneuvers.

Conclusion

We presented GUIDER, a two-phase offline framework that leverages an LLM to generate policies for multi-robot navigation, which addresses the challenges of high inference latency and reliability in online navigation systems. In the highly uncertain marine environment, GUIDER improved the task success rate and reduced the collision rate significantly against state-of-the-art RL baselines. The results showed that distilling LLM knowledge into a compact model is highly effective for accelerating DRL training and enhancing final policy performance. Furthermore, the LLM-driven evolution can act as a semantic operator to automatically discover environment-specific risk policies. However, the current risk-adaptive function is evolved offline and remains fixed during deployment, which may limit adaptability to drastic environmental shifts. Additionally, relying solely on textual state descriptions may constrain the semantic richness available to the LLM. Future work will focus on deploying the GUIDER framework on physical robots and exploring Vision-Language Models (VLMs) to process visual inputs for more robust real-world adaptation.

Acknowledgements

This work is supported by the NSFC (Grants No. 62036008 and 62273057) and Outstanding Youth Foundation of Shaanxi Province (Grant No. 2025JC-JCQN-092).

References

Acheson, D. J. 1990. *Elementary fluid dynamics*. Oxford University Press.

Bai, D.; Singh, I.; Traum, D.; and Thomason, J. 2025. TwoStep: Multi-agent Task Planning using Classical Planners and Large Language Models. arXiv:2403.17246.

Bellemare, M. G.; Dabney, W.; and Munos, R. 2017. A Distributional Perspective on Reinforcement Learning. In *ICML*, volume 70, 449–458.

Chang, L.; Shan, L.; Zhang, W.; and Dai, Y. 2023. Hierarchical multi-robot navigation and formation in unknown environments via deep reinforcement learning and distributed optimization. *Robotics Comput. Integr. Manuf.*, 83: 102570.

Dabney, W.; Ostrovski, G.; Silver, D.; and Munos, R. 2018. Implicit Quantile Networks for Distributional Reinforcement Learning. In *ICML*, volume 80, 1104–1113.

Escudie, E.; Matignon, L.; and Saraydaryan, J. 2024. Attention Graph for Multi-Robot Social Navigation with Deep Reinforcement Learning. In *AAMAS*, 2252–2254.

Fan, X.; Guo, Y.; Liu, H.; Wei, B.; and Lyu, W. 2020. Improved artificial potential field method applied for AUV path planning. *Mathematical Problems in Engineering*, 2020(1): 6523158.

Feng, P.; Liang, J.; Wang, S.; Yu, X.; Ji, X.; Chen, Y.; Zhang, K.; Shi, R.; and Wu, W. 2024. Hierarchical Consensus-Based Multi-Agent Reinforcement Learning for Multi-Robot Cooperation Tasks. In *IROS*, 642–649.

Fu, D.; Li, X.; Wen, L.; Dou, M.; Cai, P.; Shi, B.; and Qiao, Y. 2024. Drive Like a Human: Rethinking Autonomous Driving with Large Language Models. In *WACVW*, 910–919.

Garg, K.; Zhang, S.; So, O.; Dawson, C.; and Fan, C. 2024. Learning safe control for multi-robot systems: Methods, verification, and open challenges. *Annu. Rev. Control.*, 57: 100948.

Guo, T.; Chen, X.; Wang, Y.; Chang, R.; Pei, S.; Chawla, N. V.; Wiest, O.; and Zhang, X. 2024. Large Language Model Based Multi-agents: A Survey of Progress and Challenges. In *IJCAI 2024*, 8048–8057.

Jullien, S.; Deffayet, R.; Renders, J.-M.; Groth, P.; and de Rijke, M. 2025. Distributional Reinforcement Learning with Dual Expectile-Quantile Regression. In *UAI*.

Li, W.; Wang, Z.; Mai, R.; Ren, P.; Zhang, Q.; Zhou, Y.; Xu, N.; Zhuang, J.; Xin, B.; Gao, L.; Hao, Z.; and Fan, Z. 2023. Modular design automation of the morphologies, controllers, and vision systems for intelligent robots: a survey. *Vis. Intell.*, 1(1).

Lin, X.; Huang, Y.; Chen, F.; and Englot, B. J. 2024. Decentralized Multi-Robot Navigation for Autonomous Surface Vehicles with Distributional Reinforcement Learning. In *ICRA 2024*, 8327–8333.

Lin, X.; McConnell, J.; and Englot, B. J. 2023. Robust Unmanned Surface Vehicle Navigation with Distributional Reinforcement Learning. In *IROS*, 6185–6191.

Liu, C.; et al. 2023. Adaptive Risk-Tendency: Nano Drone Navigation in Cluttered Environments with Distributional Reinforcement Learning. In *ICRA*, 7198–7204.

Paudel, A.; Xiao, X.; and Stein, G. J. 2024. Multi-Strategy Deployment-Time Learning and Adaptation for Navigation under Uncertainty. In *CoRL*, volume 270, 3908–3923.

Rajvanshi, A.; Sahu, P.; Shan, T.; Sikka, K.; and Chiu, H.-P. 2025. SayCoNav: Utilizing Large Language Models for Adaptive Collaboration in Decentralized Multi-Robot Navigation. arXiv:2505.13729.

Rybczak, M.; Popowniak, N.; and Lazarowska, A. 2024. A survey of machine learning approaches for mobile robot control. *Robotics*, 13(1): 12.

Shen, Z.; Luo, H.; Chen, K.; Lv, F.; and Li, T. 2025. Enhancing Multi-Robot Semantic Navigation Through Multimodal Chain-of-Thought Score Collaboration. *AAAI*, 39(14): 14664–14672.

Skreta, M.; Zhou, Z.; Yuan, J. L.; Darvish, K.; Aspuru-Guzik, A.; and Garg, A. 2024. RePLan: Robotic Replanning with Perception and Language Models. arXiv:2401.04157.

Van den Berg, J.; et al. 2008. Reciprocal velocity obstacles for real-time multi-agent navigation. In *ICRA*, 1928–1935.

Wang, W.; Mao, L.; Wang, R.; and Min, B.-C. 2024. Multi-robot cooperative socially-aware navigation using multi-agent reinforcement learning. In *ICRA*, 12353–12360.

Wang, X.; Kim, T.; Hoxha, B.; Fainekos, G.; and Panagou, D. 2025. Safe Navigation in Uncertain Crowded Environments Using Risk Adaptive CVaR Barrier Functions. arXiv:2504.06513.

Wang, Z.; Cai, S.; Chen, G.; Liu, A.; Ma, X.; and Liang, Y. 2023. Describe, Explain, Plan and Select: Interactive Planning with LLMs Enables Open-World Multi-Task Agents. In *NeurIPS*.

Wen, L.; Fu, D.; Li, X.; Cai, X.; Ma, T.; Cai, P.; Dou, M.; Shi, B.; He, L.; and Qiao, Y. 2024. DiLu: A Knowledge-Driven Approach to Autonomous Driving with Large Language Models. In *ICLR*.

Yu, B.; Yuan, Q.; Li, K.; Kasaei, H.; and Cao, M. 2025. CoNavGPT: Multi-Robot Cooperative Visual Semantic Navigation Using Vision Language Models. arXiv:2310.07937.

Zhou, K.; Mu, Y.; Song, H.; Zeng, Y.; Wu, P.; Gao, H.; and Liu, C. 2025. Adaptive Interactive Navigation of Quadruped Robots using Large Language Models. arXiv:2503.22942.

Zhou, Z.; Zhou, B.; and Liu, H. 2024. DynamicRouteGPT: A Real-Time Multi-Vehicle Dynamic Navigation Framework Based on Large Language Models. arXiv:2408.14185.

Zu, W.; Song, W.; Chen, R.; Guo, Z.; Sun, F.; Tian, Z.; Pan, W.; and Wang, J. 2024. Language and Sketching: An LLM-driven Interactive Multimodal Multitask Robot Navigation Framework. In *ICRA*, 1019–1025.