Proceedings of CONF-MLA 2025 Symposium: Intelligent Systems and Automation: AI Models, IoT, and Robotic Algorithms
DOI: 10.54254/2755-2721/2025.LD28451

Research on Robotic Arm Motion Path Based on Reinforcement Learning and LLM

Shihan Zhe

Sino-German College of Applied Sciences, Tongji University, Shanghai, China
shihan.zhe@outlook.com

Abstract. Robot control is a current research hotspot, with robotic arm trajectory planning being a key direction. However, traditional methods exhibit insufficient adaptability and flexibility, making it difficult to meet the demands of complex tasks and dynamic environments. This paper proposes a path planning system based on the collaboration of reinforcement learning (RL) and large language models (LLM). The system consists of three modules: environment perception and LLM-based task decomposition and scheduling, RL-based trajectory planning, and motion command generation. By integrating the cognitive capabilities of LLM with the optimization capabilities of RL, the system enables task-driven robotic arm trajectory planning. In terms of design, the upper layer employs LLM for task analysis and high-level command generation, while the lower layer uses RL for trajectory optimization, forming a hierarchical collaborative mechanism. To verify its effectiveness, experiments were conducted on both simulated and real COBOT platforms for a static block-grabbing task, comparing three schemes: pure RL, pure LLM, and the proposed LLM-RL fusion. Results show that the LLM-RL approach outperforms the baselines in terms of average path length and execution time, while also significantly improving RL training efficiency.

Keywords: robotic arm, reinforcement learning, large language model, path planning

1. Introduction

In China, research on manipulator trajectory planning has undergone a transition from traditional algorithms to the gradual integration of intelligent approaches. Traditional path planning methods have played an indispensable role in early studies. Zong Chengxing et al. [1] proposed a spatial multi-DOF manipulator trajectory planning method based on the A* algorithm, which employed a customized heuristic function to improve search efficiency in high-dimensional spaces. Liu Yaqiu et al. [2] improved the RRT algorithm for industrial robot obstacle-avoidance by optimizing sampling distribution and extension strategies, significantly enhancing performance in cluttered environments. Similarly, Zou Yuxing et al. [3] improved the PRM for harvesting manipulators by refining sampling and connection strategies, enhancing adaptability in agricultural environments. Li Yang et al. [4] proposed a gravity-adaptive step-size RRT for dual-arm coordination, effectively overcoming the low efficiency of traditional RRT in cooperative scenarios. More recently, Wang Yu et al. [5]

DOI: 10.54254/2755-2721/2025.LD28451

integrated reinforcement learning into the RRT framework to optimize sampling, thereby accelerating convergence and improving path quality in manipulator trajectory planning.

Internationally, significant progress has been made in integrating reinforcement learning (RL) with large language models (LLMs) for trajectory planning, leading to the emergence of diversified technical pathways. For instance, Ma et al. [6] introduced the ExplorLLM framework, where LLMs provide linguistically informed exploration strategies to RL agents, thereby significantly enhancing learning efficiency in complex environments. Building on the need for handling more complex tasks, Kheirandish et al. tackled the challenge of decomposing high-level objectives into learnable subtasks [7]. They proposed a hybrid framework that combines LLMs with symbolic reinforcement learning (Symbolic RL), allowing for more structured task representations. Extending this idea, Shukla et al. proposed the LgTS framework, in which LLMs generate subgoal DAGs to guide teacher–student learning [8], enabling RL agents to efficiently acquire strategies in dynamic conditions with fewer environment interactions. Shek et al. [9] advanced the concept by designing an LLM-guided hierarchical RL framework, where subgoal sequences and relation trees are employed to construct option hierarchies within a three-level policy structure. Finally, addressing collaborative scenarios, Siedler et al. [10] explored the integration of LLM-based interventions into multi-agent reinforcement learning frameworks, thereby extending the applicability of LLM-RL integration to multi-manipulator systems.

Building upon these advancements, this paper proposes a unified framework for robotic arm trajectory planning that integrates the semantic reasoning and task decomposition capabilities of LLMs with the adaptive optimization strengths of RL. The system is designed in a layered structure: the LLM module (DeepSeek) interprets natural language instructions and decomposes them into structured sub-goals, while the RL module, based on the Proximal Policy Optimization (PPO) algorithm, focuses on learning optimized motion policies for each sub-goal. The RL problem is carefully formulated with a tailored state representation, action space, and a composite reward function, which collectively ensure efficiency, stability, and task success.

The significance of this study lies in demonstrating that such an LLM-RL collaborative framework can outperform pure LLM or pure RL approaches. Through simulation and real-world experiments, the proposed method achieves shorter paths, faster execution, higher task success rates, and—crucially—improved sample efficiency in RL training. These results highlight the potential of combining LLM-driven semantic guidance with RL-based adaptive optimization to enhance robotic autonomy. Ultimately, this research provides a feasible approach to developing more intelligent, flexible, and human-compatible robotic systems, offering promising implications for intelligent manufacturing and human-robot collaboration in dynamic industrial environments.

2. System design

2.1. Overall architecture and workflow

The proposed system consists of three core modules, as illustrated in Figure 1:

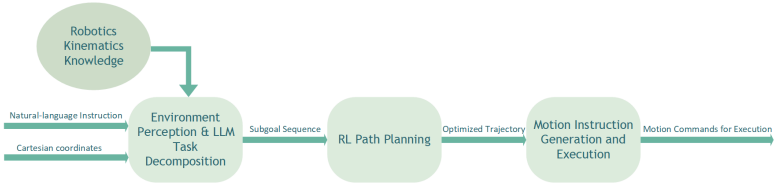

Figure 1. Workflow

DOI: 10.54254/2755-2721/2025.LD28451

(1) Environment Perception & LLM Task Decomposition Module: This module is responsible for acquiring basic scene information (primarily object and target bin coordinates provided as input or from perception) and processing the user's natural language command. The LLM component parses the instruction and decomposes the overall task into a logical sequence of executable sub-goals ("move to pre-grasp pose", "close gripper", "move to pre-place pose", "open gripper").

(2) RL Path Planning Module: This serves as the core decision-making unit. For each sub-goal received from the LLM module, the RL agent (based on PPO) learns and executes an optimal motion path. It interacts with the simulation environment, utilizing a defined state representation, action space, and reward function to learn a policy that maximizes cumulative reward for the sub-task.

(3) Motion Instruction Generation and Execution Module: This module converts the planned path (typically a sequence of joint velocities or target poses from the RL module) into low-level motion commands compatible with the specific robotic arm controller (using the Agilebot SDK). It handles necessary conversions and ensures safe execution.

The integrated workflow is as follows: The user provides a natural language instruction and relevant coordinates. The LLM module processes this input and generates a sub-goal sequence. This sequence is passed sequentially to the RL module. For each sub-goal, the RL module plans and executes the corresponding motion trajectory. The Motion Instruction module converts these plans into actionable commands for the robot. This loop continues until all sub-goals are completed or a failure is detected.

2.2. Core module details

2.2.1. LLM task decomposition module

In this framework, the task decomposition module is powered by DeepSeek-V3, which was selected as the core LLM due to its outstanding ability to process and understand Chinese natural language instructions—an essential requirement for the intended application scenario. Beyond language comprehension, DeepSeek-V3 also exhibits strong logical reasoning and planning capabilities, making it well-suited for breaking down complex tasks into structured sub-goals.

To ensure reliable and machine-parsable outputs, prompt engineering strategies were carefully designed. First, the role of the LLM was explicitly defined as a “robotic task planning assistant” to anchor its behavior. Second, the output format was strictly specified in JSON, with clearly defined fields such as action_type, target_description, and parameters (e.g., x, y, z for move_cart actions)such as action_type, target_description, and parameters (e.g., x, y, z for move_cart actions). Third, environment context—including workspace coordinates and safety height constraints—was injected into the prompts to provide situational grounding. Additionally, few-shot examples were incorporated, illustrating input-instruction to output-subgoal sequences to guide the model’s reasoning process. Finally, safety constraints were embedded within the prompt logic, such as minimum Z-axis limits and approach heights, ensuring that generated sub-goals adhered to operational safety requirements.

2.2.2. RL path planning module

• Algorithm: Proximal Policy Optimization (PPO) was selected due to its stability, efficiency, and suitability for continuous control problems like robotic manipulation.

DOI: 10.54254/2755-2721/2025.LD28451

• State Space (s_t): The state observation vector combines proprioceptive and goal-oriented information:

$$s_t = [q_1, q_2, \ldots, q_6, \dot{q}_1, \dot{q}_2, \ldots, \dot{q}_6, x, y, z, \ x_{target}, y_{target}, z_{target}] \tag{1}$$

where q_i are joint angles, $\dot{q}_i$ are joint velocities, $[x, y, z]$ is the end-effector Cartesian position, and $[x_{target}, y_{target}, z_{target}]$ is the current target position (from the sub-goal).

• Action Space (a_t): The action is defined as normalized target joint velocities for the 6 joints:

• $a_t = [\dot{q}_{cmd,1}, \ldots, \dot{q}_{cmd,6}]$, where each component is constrained to [-1, 1].

• Reward Function (r_t): A composite reward function was designed to guide learning:

$$r_t = 10.0\, r_{target} + 1.0\, r_{dist} + 10.0\, r_{place} \tag{2}$$

r_{target} : A sparse reward (+100) granted only when the end-effector successfully reaches the current target position (distance < 0.05m). This strongly signals the achievement of the sub-goal task.

r_{dist} : A dense, negative reward proportional to the Euclidean distance between the end-effector and the current target ($\| p_{ee} - p_{goal} \|_2$). This provides continuous guidance throughout the motion.

r_{place} : A reward term specific to the final placement action, providing positive reward for success and negative reward based on distance during placement attempts. Phase switches (grasp vs. place) were determined by gripper state and proximity.

3. Experiments and results

3.1. Experimental setup

• Simulation Platform: Experiments were conducted using PyBullet physics simulation A URDF model of the Agilebot GBT-C5A 6-DOF collaborative robotic arm was adopted, with accurate kinematic parameters configured.. Figure 2(a) shows the simulation environment running on a PC, and Figure 2(b) shows the actual physical scene.

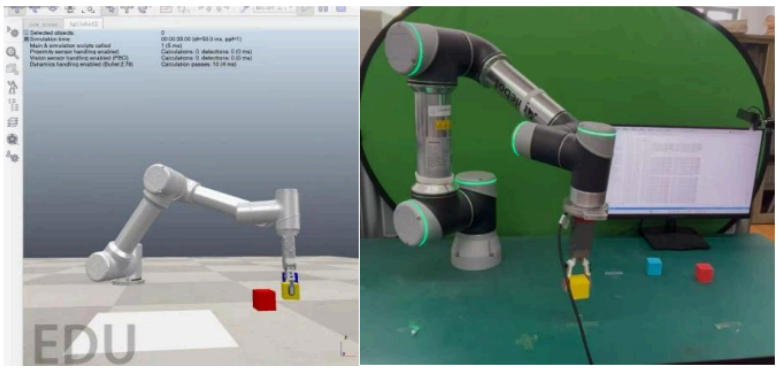

Figure 2. (a) Simulation environment on PyBullet, (b) actual physical scene

• Task: The benchmark task involved picking a colored block from a specified coordinate on a workspace and placing it accurately into a designated bin location. The environment contained static obstacles.

DOI: 10.54254/2755-2721/2025.LD28451

• Compared Methods:

• Pure LLM: The LLM generated a sequence of Cartesian coordinate waypoints based on the instruction and embedded logic/constraints. These waypoints were directly converted to robot commands via inverse kinematics (using SDK functions). This method lacked online optimization or adaptation.

• Pure RL: A PPO agent was trained end-to-end to perform the entire pick-and-place task from scratch based on a reward function designed for the overall objective. This required learning the complete task sequence and long-horizon planning.

• LLM-RL Fusion (Proposed): The LLM decomposed the task into sub-goals. A separate instance of the PPO agent (with the same core architecture as pure RL) was trained (or executed, if pre-trained) for each type of sub-goal ('move to point', 'grasp').

• Evaluation Metrics:

• Average Path Length (APL): The total Cartesian distance travelled by the end-effector in successful tasks.

• Success Rate: Percentage of tasks completed successfully without errors (dropping object, collision).

• Training Efficiency: The average number of training steps required for the RL policy to converge to a stable performance level.

3.2. Results and analysis

• Task Performance: The LLM-RL fusion method achieved the best performance across both primary metrics (Table 1), which demonstrates the effectiveness of the combined approach. Pure RL performed better than pure LLM but worse than the fusion method. Pure LLM generated longer paths due to lack of motion optimization.

Table 1. Performance comparison (lower is better)

Metric	Pure LLM	Pure RL	LLM-RL Fusion
Avg. Path Length (m)	1.92	1.55	1.38
Avg. Exec. Time (s)	25.5	20.1	16.5

The pure LLM method produced longer, less optimized paths as it relied on geometric reasoning without motion optimization. Although the pure RL method found better paths than the pure LLM method, it often got trapped in local optima or required more time to learn the full task sequence—resulting in worse performance than the fusion approach. The fusion method benefits from the LLM providing a logically correct skeleton of the task and the RL optimizing the motion between these defined points.

DOI: 10.54254/2755-2721/2025.LD28451

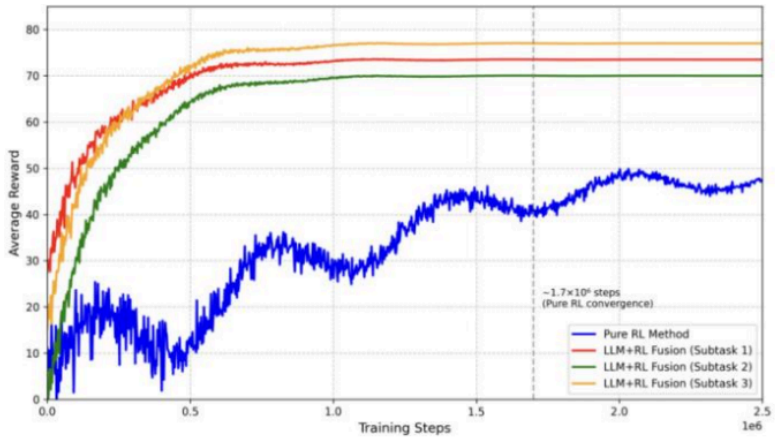

Figure 2. Comparison of pure RL method and LLM+RL fusion method

• Training Efficiency: A significant advantage of the fusion approach was observed in the sample efficiency of the RL component. The RL agent focused on learning policies for individual sub-goals converged much faster (requiring approximately 380,000 steps per sub-task type on average) compared to the pure RL agent which needed to learn the entire long-horizon task end-to-end (requiring approximately 1,700,000 steps), which can be seen in Figure 3. This is because each sub-task represents presents a simpler, shorter-horizon problem with more immediate and dense rewards, thereby facilitating faster learning.

• Success Rate: The fusion method also achieved a higher task success rate (93%) compared to the pure LLM method (82%), benefiting from the RL's ability to adapt to physics and optimize locally.

4. Conclusion

This paper proposed an integrated framework for robotic arm path planning that leverages the complementary strengths of large language models (LLMs) and reinforcement learning (RL). Specifically, the system architecture enables the LLM (DeepSeek) to perform high-level semantic parsing and decompose tasks into manageable sub-goals, while the RL agent (PPO) is dedicated to learning optimized motion policies for each sub-goal. Within this framework, we formulated the RL problem in detail by defining the state representation, action space, and a composite reward function that jointly ensures task success, efficiency, and smoothness of motion. Experimental validation in simulation demonstrated that the proposed LLM-RL fusion method consistently outperformed pure LLM and pure RL baselines in terms of path efficiency, execution speed, and task success rate, while also significantly improving the sample efficiency of RL training. These results confirm the potential of integrating semantic reasoning from LLMs with the adaptive optimization capabilities of RL, highlighting a viable direction for addressing complex robotic manipulation tasks in future research.

References

[1] Zong, C., Lu, L., Lei, X., & Zhao, P. (2017). A path planning approach for multi-dof spatial manipulator via A* algorithm. Journal of Hefei University of Technology (Natural Science), 40(2), 164-168.

[2] Yaqiu, L., Hanchen, Z., Xun, L., & Yan, X. (2021). An Improved RRT Based Obstacle Avoidance Path Planning Algorithm for Industrial Robot. Information and Control, 50(2), 235-246.

DOI: 10.54254/2755-2721/2025.LD28451

[3] Zou, Y., Li, L., & Gao, Z. (2019). Obstacle avoidance path planning for harvesting robot arm based on improved PRM. Transducer and Microsystem Technologies, 38(1), 52-56.
[4] Li, Y., & Xu, D. (2020). Cooperative path planning of dual-arm robot based on attractive force self-adaptive step size RRT. Robot, 42(5), 606-616.
[5] Wang, Y., Liu, Y., Jia, H., & Xue, G. (2022). Path planning of mechanical arm based on intensified RRT algorithm. J. Shandong Univ.(Eng. Sci.), 52, 123-130.
[6] Ma, R., Luijkx, J., Ajanovic, Z., & Kober, J. (2024). Explorllm: Guiding exploration in reinforcement learning with large language models. arXiv preprint arXiv: 2403.09583.
[7] Kheirandish, A., Xu, D., & Fekri, F. (2024). LLM-Augmented Symbolic Reinforcement Learning with Landmark-Based Task Decomposition. arXiv preprint arXiv: 2410.01929.
[8] Shukla, Y., Gao, W., Sarathy, V., Velasquez, A., Wright, R., & Sinapov, J. (2023). Lgts: Dynamic task sampling using llm-generated sub-goals for reinforcement learning agents. arXiv preprint arXiv: 2310.09454.
[9] Shek, C. L., & Tokekar, P. (2025). Option Discovery Using LLM-guided Semantic Hierarchical Reinforcement Learning. arXiv preprint arXiv: 2503.19007.
[10] Siedler, P. D., & Gemp, I. (2025). LLM-Mediated Guidance of MARL Systems. arXiv preprint arXiv: 2503.13553.