Forging Agentic AI: A Comprehensive Survey on the Symbiotic Convergence of Large Language Models and Reinforcement Learning

Minsoo Kim[1], Joongheon Kim[2], and Soyi Jung[3]
[1]*Dept. Aritificial Intelligence Convergence Network, Ajou Univerity, Suwon, 16499, South Korea*
[2]*Dept. Electrical and Computer Engineering, Korea University, Seoul, 02841, South Korea*
[3]*Dept. Electrical and Computer Engineering, Ajou University, Suwon, 16499, South Korea*
E-mails: andykim000@ajou.ac.kr, joongheon@korea.ac.kr, sjung@ajou.ac.kr

***Abstract*—Reinforcement learning (RL) has shown remarkable success in solving complex decision-making problems; however, it still suffers from fundamental limitations such as low sample efficiency, sparse rewards, and poor generalization. Recent advances in large language models (LLMs) have introduced new possibilities for overcoming these challenges by enhancing multiple components of the RL pipeline. This survey provides a comprehensive overview of LLM-augmented reinforcement learning (LLM-RL), highlighting the ways in which LLMs contribute to reward design, exploration, planning, state and action representation, policy learning, and generalization. We categorize integration strategies, analyze representative frameworks, and discuss key application areas including robotics, gaming, and virtual environments. Finally, we discuss current limitations, open research challenges, and future directions to highlight the potential synergy between LLMs and RL in building more general and adaptable autonomous agents.**

***Index Terms*—Reinforcement learning, Large language models, Reinforcement learning, Reward shaping, Generalization, Artificial intelligence**

I. Introduction

In the field of artificial intelligence (AI), reinforcement learning (RL) has made remarkable progress over the past several decades, contributing significantly to solving a variety of complex problems. Notably, Google DeepMind's AlphaGo defeated a world champion in the game of Go, and the Deep Q-Network (DQN) achieved human-level performance on Atari games [1], [2]. These successes have demonstrated the potential of RL in a wide range of domains, including games, robotic control, autonomous driving, and financial trading [3], [4]. RL agents learn optimal behavioral policies by interacting with their environments via trial and error, thus acquiring goal-directed behavior without explicit programming.

Despite these successes, traditional RL methodologies still face several fundamental limitations. First, RL typically suffers from low sample efficiency, requiring massive amounts of data and interactions with the environment to learn effectively. In real-world scenarios, collecting such extensive data may be impractical or even hazardous. Second, in environments with sparse rewards, agents often struggle to find meaningful reward signals, making effective exploration extremely challenging. Third, constructing accurate environment models or reward functions is notably difficult in complex and dynamic real-world settings [5]. Lastly, policies optimized for specific training environments often fail to generalize to new or unseen environments, presenting a critical limitation. These challenges remain major obstacles to the widespread adoption of RL in practical applications.

Recently, LLMs have revolutionized not only natural language processing (NLP), but also the broader landscape of AI research. Pretrained on massive textual corpora, LLMs exhibit outstanding capabilities in language understanding and generation, reasoning, planning, and even few-shot learning, the ability to perform new tasks given only a handful of examples [6]. These advanced cognitive abilities offer promising opportunities to address the long-standing issues of traditional RL and enable the development of more powerful and flexible AI agents. LLMs can provide prior knowledge about environments, understand complex instructions, generate action plans, and even create reward signals, significantly enhancing the learning efficiency and generalization capability of RL agents by integrating at multiple stages of the RL process.

This survey provides a comprehensive review of recent advances in LLM-augmented RL. Specifically, it examines how LLMs can overcome the inherent limitations of conventional RL and pave the way for a new paradigm in AI. Section II introduces the fundamental concepts and key algorithms of RL, highlighting the primary challenges faced by traditional RL. Section III categorizes and explains various approaches for integrating LLMs into different components of RL, such as reward design, exploration, representation, and policy learning. Section IV presents major applications and real-world case studies of LLM-augmented RL, demonstrating its effectiveness. Finally, Section V discusses ongoing challenges and outlines promising directions for future research. This survey aims to provide insights into the synergy between LLMs and RL, thereby advancing understanding and fostering further research in this rapidly evolving field.

II. Fundamentals of Traditional Reinforcement Learning

RL is a machine learning paradigm in which an agent interacts with its environment and learns an optimal behavioral

policy through trial and error. In this process, the agent takes an action in a given state, receives a corresponding reward from the environment, and transitions to a next state. The ultimate objective of RL is to discover a policy that maximizes the agent's expected cumulative reward over the long term. This foundational framework underlies much of the recent interest in combining RL with LLMs, as explored in the following section, which introduces the fundamental concepts and principal algorithms of traditional RL that form the basis for understanding LLM-augmented RL. Additionally, we clearly outline the inherent limitations and core challenges faced by this field, which motivate the integration of LLMs into RL frameworks.

To formalize RL problems, the Markov Decision Process (MDP) is widely adopted. An MDP is defined by the tuple (S, A, P, R, γ) as follows: This formalism provides a structured framework for modeling sequential decision-making under uncertainty.

- State Space (S): The set of all possible environmental states perceivable by the agent.
- Action Space (A): The set of all possible actions the agent can take in a given state.
- Transition Probability Function ($P(s'|s,a)$): The probability that the environment transitions to state s' when the agent takes action a in state s.
- Reward Function ($R(s,a,s')$): The immediate reward received after transitioning from state s to state s' via action a.
- Discount Factor ($\gamma \in [0,1)$): A factor that discounts the value of future rewards, typically chosen close to 1.

The agent's behavior is governed by a policy ($\pi(a|s)$), which specifies the probability of taking action a in state s. While policies are often stochastic, in some cases they may be deterministic, directly mapping states to actions. The primary objective of RL is to find the optimal policy π^* that maximizes the expected return, i.e., the sum of discounted future rewards.

This expected return is quantitatively expressed through the value function:

- State-Value Function ($V^\pi(s)$): The expected return when starting from state s and following policy π.
- Action-Value Function ($Q^\pi(s,a)$): The expected return when starting from state s, taking action a, and subsequently following policy π.

These value functions satisfy recursive relationships that capture the principle of optimality in dynamic programming, and are recursively defined via the Bellman equation, which forms the mathematical foundation for computing optimal policies in RL [7].

$$V^\pi(s) = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s \right] \tag{1}$$

$$Q^\pi(s,a) = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s, A_t = a \right] \tag{2}$$

A. Categorization of RL Algorithms

Traditional RL algorithms can be broadly categorized into three major paradigms: value-based methods, policy-based methods, and actor-critic methods. In addition, RL algorithms can be further classified as either model-based or model-free, depending on whether an explicit model of the environment is utilized.

1) Value-based methods: Value-based methods learn the optimal value function and indirectly derive the optimal policy from it. The agent typically follows a greedy policy by selecting the action with the highest Q-value in each state.

- *Q-learning:* Q-learning is an off-policy algorithm, meaning the policy used to generate behavior may differ from the policy being optimized. It directly estimates the optimal action-value function $Q^*(s,a)$ from experience.
- *DQN:* DQN integrates deep neural networks into the Q-learning framework to effectively approximate the value function in high-dimensional state spaces. It achieved human-level performance on Atari games, marking the beginning of the deep RL era. Key techniques such as the replay buffer and target network are introduced to improve learning stability [8].

2) Policy-based methods: Policy-based methods learn the optimal policy π directly, without explicitly computing a value function. These methods are particularly suitable for environments with continuous or very large action spaces, and have the advantage of enabling the learning of stochastic policies [9].

- *REINFORCE:* REINFORCE is one of the most fundamental policy gradient algorithms, which updates policy parameters using the total reward obtained in an episode [10].
- *Trust Region Policy Optimization (TRPO):* TRPO introduces the concept of a trust region to prevent overly large policy updates and thus improves the stability of learning [11].
- *Proximal Policy Optimization (PPO):* PPO reduces the complexity of TRPO while maintaining similar performance. It is widely used in RL applications due to its simple implementation and robustness [12].

3) Actor-critic methods: Actor-critic methods combine the strengths of both value-based and policy-based approaches. The actor learns the policy to select actions, while the critic learns the value function to evaluate the actions taken by the actor and provides an advantage signal for policy updates.

- *Advantage Actor-Critic (A2C) / Asynchronous Advantage Actor-Critic (A3C):* These are representative examples of the actor-critic framework, significantly improving learning efficiency through parallel training. In A3C, multiple agents interact with the environment independently to collect experiences and asynchronously update a central neural network, thereby accelerating the learning process [13].

4) Model-based vs. Model-free methods:

- *Model-based:* Model-based methods first learn a model of the environment (i.e., P and R), and use this model for planning and predicting future outcomes. By generating simulated experiences using the learned model, these methods can improve sample efficiency. AlphaGo, which combines deep neural networks and Monte Carlo tree

search (MCTS), is a prominent example of a model-based RL system [1]. However, learning an accurate model of a complex environment remains a fundamental challenge.

- *Model-free:* Model-free methods do not explicitly learn the environment's transition probability function P or reward function R. Instead, agents learn policies or value functions directly from experience obtained through interaction with the environment (e.g., Q-learning, DQN, PPO) [14], [15]. These methods are simple to implement and easy to apply to various environments, but typically suffer from low sample efficiency.

B. Challenges in Traditional Reinforcement Learning

As discussed in the introduction, traditional RL faces several significant challenges when applied to complex real-world problems. These limitations provide the core motivation for the necessity of LLM-augmented RL.

1) Low sample efficiency: Most RL algorithms require a large number of interactions with the environment to learn an optimal policy. This becomes a major constraint in cases where learning in the real physical world is expensive—such as in robotic control—or when building high-fidelity simulation environments is difficult.

2) Sparse rewards and reward function design: In many real-world environments, agents only receive rewards upon achieving specific goals, or reward signals are extremely rare. Such sparse reward settings make it difficult for agents to discover effective learning paths. Additionally, designing reward functions for complex tasks is often very challenging or even infeasible for humans, and poorly designed rewards can lead to undesired behaviors [5].

3) Challenges in effective exploration: Agents face the exploration-exploitation dilemma—balancing the need to explore unknown state-action spaces while exploiting known optimal actions. Designing efficient exploration strategies is especially difficult in high-dimensional, complex environments [16].

4) Lack of generalization capability: Traditional RL agents are typically trained on specific environments or tasks and lack generalization capability to new or slightly modified tasks. This severely limits the applicability of RL in diverse and dynamic real-world scenarios.

5) Long horizons and hierarchical planning: Many complex tasks require planning over long sequences of actions (long horizons). Traditional RL algorithms tend to learn at the level of individual actions, making it difficult to establish hierarchical and long-term plans.

These challenges have been the focus of much research in RL. In the next section, we discuss how LLMs can potentially overcome these limitations and open new horizons for RL systems.

III. LLM-Augmented Reinforcement Learning

LLM-RL is an emerging research area that seeks to overcome the fundamental limitations of traditional RL by leveraging the powerful language understanding, reasoning, and generation capabilities of LLMs. In this section, we categorize and describe the specific approaches by which LLMs are integrated into various components of the RL pipeline to enhance learning performance.

A. Reward Function Design and Generation with LLMs

In traditional RL, manually designing reward functions is labor-intensive and prone to error. LLMs address these challenges by leveraging their strong coding abilities and extensive domain knowledge to automatically generate and refine reward functions. This shifts reward design from a manual process to an automated engineering approach. Frameworks such as R* and CARD have been proposed, in which LLMs are used to generate and improve reward function code [17], [18].

Furthermore, LLMs can implicitly learn reward signals directly from human preferences. Direct preference optimization (DPO) reparameterizes the reward model to directly compute the optimal policy, enabling end-to-end training of the LLM policy without explicitly training a separate reward model [19]. DPO optimizes the policy by directly comparing preferred and dispreferred output pairs, effectively making the language model itself implicitly act as a reward model. This approach offers advantages in stability and computational efficiency, making it particularly valuable for refining LLM behaviors in alignment with complex human values and preferences.

B. Exploration and Planning with LLMs

Traditional RL struggles with efficient exploration and long-term planning in complex environments. LLMs, with their vast pretrained knowledge and powerful reasoning capabilities, help overcome these limitations. Using LLMs, RL exploration can shift from a fixed predefined stochastic process to an adaptive, task-specific strategy, greatly improving both efficiency and performance [20]. For example, *LLM-Explorer* utilizes the analysis and reasoning abilities of LLMs to assess the agent's current policy learning state during training and adaptively generate probability distributions for future policy exploration. This approach has led to an average performance improvement of 37.27 % on Atari and MuJoCo benchmarks [21].

LLMs can leverage their knowledge of the environment to support long-term planning for goal achievement [22]. Combined with hierarchical RL (HRL) and goal-conditioned RL, LLMs can hierarchically decompose complex objectives based on natural language descriptions (hierarchical task decomposition), or provide instructions for specific states. Frameworks such as LDSC, LGRL, and SAMA explicitly utilize LLMs for subgoal generation and adaptive updating, thereby improving sample efficiency and generalization [23]–[25]. In the ACE framework, LLMs serve dual roles as both the policy actor and value critic, while DPSDP trains an actor-critic LLM system to enhance response quality [26], [27].

C. State/Action Representation and Reasoning with LLMs

LLMs enable the transition of state and action representations from low-level raw sensory data to high-level, semantic, and flexible formats, facilitating more human-like understanding and control. For example, LESR autonomously generates

TABLE I: Taxonomy of LLM Integration Roles and RL Challenges Addressed

Integration Type	Role of LLM	RL Challenge Addressed	Representative Approach/Paper
Reward Function Design/Generation	Reward code generation, critic, preference modeling, reward signal refinement	Sparse rewards, difficulty of reward function design	R* [17], DPO [19]
Exploration	Task-specific exploration strategies, learning state analysis	Low sample efficiency, exploration in complex environments	SafeGPT [20], LLM-Explorer [21]
Planning	Long-term planning, high-level goal decomposition, subgoal generation	Long horizons, planning in complex environments	LGRL [23], LDSC [25], SAMA [24]
State Representation	Summarizing states as natural language features, task-relevant code generation	High-dimensional state space, lack of generalization	LESR [28], Do LLMs Build World Representations? [29]
Action Representation/Selection	Providing prior action distributions, candidate action filtering, generating mid-level actions	High-dimensional action space, sample efficiency	Efficient RL with LLM Priors [30], NaVILA [31], Combining LLM decision and RL action selection [32]
Policy Learning/Modification	Direct policy generation, real-time policy update, co-evolution	Policy learning complexity, lack of adaptability	LLM-as-Policy [33], CORY [34], Policy as Code
Generalization	Utilizing pretrained knowledge, few-shot/zero-shot learning	Lack of generalization, adaptation to novel environments/tasks	Prompt-DT [35], TEDUO [36]

task-relevant state representation code using LLMs, which accelerates efficient training [28]. LLMs can also summarize the world state into goal-oriented abstractions, and recent studies have explored converting agent-environment interaction histories into natural language text for further reasoning [29]. In addition, approaches such as NaVILA generate mid-level natural language actions that are then executed by low-level locomotion RL policies [31].

LLMs contribute to the flexible expansion of complex action spaces and facilitate the filtering of candidate actions [26]. By leveraging pretrained knowledge, LLMs can provide prior action distributions, reducing the complexity of exploration and optimization, and thereby improving sample efficiency [32]. Moreover, LLMs enhance reasoning capabilities through multi-step inference processes and self-correction mechanisms. For instance, RRO focuses on optimizing LLM agents by emphasizing "reward escalation" during consecutive reasoning steps [37].

D. Policy Learning and Generalization with LLMs

LLMs can directly implement or significantly influence learned policies, leading to more flexible and adaptive agents."The LLM-as-Policy" paradigm treats the LLM itself as the policy, enabling it to generate actions directly [33]. LLMs can be fine-tuned via RL to adapt to specific tasks, and text-based user preferences can be utilized to update RL policies in real time. The CORY framework extends LLM fine-tuning into a sequential, collaborative multi-agent RL (MARL) framework, enabling co-evolution and the emergence of new capabilities [34]. Building on this, LLMs also leverage their vast pretrained knowledge to enhance zero-shot and few-shot generalization to new environments and tasks. For example, prompt-based decision transformer (prompt-DT) uses few-shot demonstrations as "trajectory prompts" to guide policy generation [35], while TEDUO utilizes LLMs as cost-effective data augmenters and flexible generalizers, enabling the learning of language-conditioned policies [36].

IV. Key Applications and Case Studies

A. Robotics

In robotics, LLM-RL plays a key role in enabling robot control via natural language commands, learning complex manipulation tasks, and improving adaptability across diverse environments. LLMs are used to translate human natural language instructions into low-level actions or mid-level plans that robots can understand and execute. For example, *ELEMENTAL* combines natural language instructions with visual user demonstrations to align robot behaviors with user intent [38]. *NaVILA* generates mid-level natural language actions to enhance the navigation capabilities of legged robots [31]. Furthermore, LLM-based agent orchestration architectures integrate memory-augmented task planning to autonomously manage household objects [39].

B. Gaming

In the gaming domain, LLM-RL contributes to making agent behaviors more human-like, learning complex game rules, and automating various aspects of game design. LLMs are used to generate dialogue and behaviors for non-player characters (NPCs), enriching player interaction. LLM-assisted RL methods such as *ELLM* and *Read and Reap Rewards* address challenges of sparse rewards and sample efficiency, making RL agent training more effective [40], [41].

C. Simulation and Virtual Environments

LLM-RL is utilized in simulation and virtual environments to control agent behaviors, deliver personalized user experiences, and generate complex virtual worlds. LLM agents, empowered by extensive world knowledge and reasoning

capabilities, can achieve stronger performance compared to traditional RL-based miniature agents. Frameworks such as *EnvGen* train RL agents within environments generated by LLMs, dynamically adapting the environment to incrementally improve agent skills [42]. LLM-RL enables the development of virtual agents, telepresence experiences, and multimodal content analysis tools, providing personalized and interactive user experiences.

V. Challenges and Future Research Directions

The field of LLM-augmented RL is still in its early stages; while rapid progress has been made and models like DeepSeek-R1 show impressive performance, its implementation remains complex—requiring sophisticated algorithms, reward modeling strategies, and optimization techniques—which poses difficulties for systematic research, and key challenges persist in advancing LLM capabilities, including improving multi-step reasoning, handling chained tasks, balancing structured prompting with flexibility, enhancing long-context retrieval, and integrating external tools, all of which must be addressed for sustained development and real-world adoption.

- *Reliability and Safety:* The hallucination problem of LLMs can pose significant challenges to the reliability and safety of LLM-RL systems. LLMs may generate factually incorrect or unfounded content, which can cause RL agents to learn from false information or perform unsafe actions. To address this issue, various approaches are required, including hallucination detection based on the latent space of LLMs, reconstructing the representation space using lightweight steering vectors such as the truthfulness separator vector (TSV), and developing integrated models that verify LLM responses against context and general knowledge [43], [44].
- *Data Efficiency and Bias:* While the vast pretraining data of LLMs endows them with powerful capabilities, it can also introduce bias and present challenges in terms of computational cost and data efficiency during RL training. Techniques such as difficulty-targeted online data selection and rollout replay contribute to improving the data efficiency of LLM RL fine-tuning [45].
- *Reasoning and Explainability:* Understanding and explaining the decision-making processes of LLM-based RL systems is essential for ensuring reliability and facilitating debugging. The adoption of LLMs has increased the use of natural language explanations in NLP model interpretability research; however, the faithfulness of such explanations remains an open question.
- *Computational Complexity and Cost:* The high computational cost of LLMs and the management of large-scale training data constitute major constraints for LLM-RL systems. Although LLM-based agents can achieve strong performance, frequent invocation of LLMs is slow and expensive. Standard RL algorithms for LLM fine-tuning (e.g. PPO) can also be unstable and prone to distribution collapse due to the large discrete action space and sparse rewards associated with LLMs.
- *Unified Framework Development:* The development of general frameworks that encompass various LLM-RL integration strategies can accelerate progress in this field. For example, *LAMARL* proposes an integrated framework that incorporates LLMs into MARL to improve coordination, communication, and generalization [23]. *CoRL* aims to enhance both generative and comprehension capabilities through a unified RL framework for unified multimodal LLMs (ULMs) [46], [47].

VI. Conclusion

This survey has presented a comprehensive overview of LLM-RL, a rapidly emerging paradigm at the intersection of natural language processing and decision-making. By integrating large language models into various components of the RL pipeline—such as reward modeling, exploration, planning, representation learning, and policy optimization—LLMs help address long-standing challenges in traditional RL, including sparse rewards, low sample efficiency, and limited generalization. These models bring semantic understanding, prior knowledge, and reasoning capabilities to agents, resulting in more adaptable and effective behavior in complex environments. Case studies in robotics, gaming, and simulation demonstrate how LLM-RL enables natural language interaction, flexible task execution, and personalized agent behavior.

Despite its promise, LLM-RL remains in its early stages, facing challenges related to reliability, hallucination, bias, computational overhead, and explainability. Addressing these issues will require new algorithms, scalable architectures, and unified evaluation frameworks. Nevertheless, LLM-RL represents a significant shift in autonomous agent design, offering a promising direction for building intelligent systems that are generalizable, interpretable, and capable of real-world operation.

Acknowledgment

This work was supported by the National Research Foundation of Korea (NRF) Grant funded by the Korea Government [Ministry of Science and ICT (Information and Communications Technology) (MSIT)] under Grant RS-2024-00358662.

References

[1] D. Silver, A. Huang, C. J. Maddison *et al.*, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, pp. 484–489, Oct. 2017.

[2] V. Mnih, K. Kavukcuoglu, D. Silver *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.

[3] R. S. Sutton and A. G. Barto, "Reinforcement learning: An introduction," *MIT Press*, vol. 2, 2018.

[4] J. Kim, S. Park, S. Jung, and C. Cordeiro, "Cooperative multi-uav positioning for aerial internet service management: A multi-agent deep reinforcement learning approach," *IEEE Transactions on Network and Service Management*, vol. 21, no. 4, pp. 3797–3812, Aug. 2024.

[5] P. Abbeel, A. Coates, M. Quigley, and A. Ng, "An application of reinforcement learning to aerobatic helicopter flight," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, vol. 19, Vancouver, Canada, Dec. 2006, pp. 1–8.

[6] T. B. Brown, B. Mann, N. Ryder *et al.*, "Language models are few-shot learners," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, no. 159, Dec 2020, pp. 1877–1901.

[7] S. Park, G. S. Kim, S. Jung, and J. Kim, "Markov decision policies for distributed angular routing in LEO mobile satellite constellation networks," *IEEE Internet of Things Journal*, vol. 11, no. 23, pp. 38 744–38 754, Aug. 2024.

[8] J. Jang, J. Kim, J. Kim, and S. Jung, "Joint interference approximation and guard-band management for spectrum-efficient integrated ntn–tn networks," *IEEE Internet of Things Journal*, vol. 12, no. 15, pp. 32 220–32 236, Aug 2025.

[9] H. Lee, S. Jung, and S. Park, "Situation-aware deep reinforcement learning for autonomous nonlinear mobility control in cyber-physical loitering munition systems," *IEEE/KICS Journal of Communications and Networks*, vol. 27, no. 1, pp. 10–22, Feb. 2025.

[10] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine Learning*, vol. 8, no. 3-4, pp. 229–256, May 1992.

[11] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *Proc. 32nd International Conference on Machine Learning*, vol. 37, Lille, France, July 2015, pp. 1889–1897.

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

[13] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Proc. 33rd International Conference on Machine Learning (ICML)*, vol. 48, New York, NY, USA, Jun 2016, pp. 1928–1937.

[14] C. Park, G. S. Kim, S. Park, S. Jung, and J. Kim, "Multi-agent reinforcement learning for cooperative air transportation services in city-wide autonomous urban air mobility," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 8, pp. 4016–4030, Aug. 2023.

[15] S. Jung, W. J. Yun, M. Shin, J. Kim, and J.-H. Kim, "Orchestrated scheduling and multi-agent deep reinforcement learning for cloud-assisted multi-uav charging systems," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 6, pp. 5362–5377, Jun. 2021.

[16] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine Learning*, vol. 47, pp. 235–256, May 2002.

[17] P. Li, J. Hao, H. Tang, Y. Yuan, J. Qiao, Z. Dong, and Y. Zheng, "R*: Efficient reward design via reward structure evolution and parameter alignment optimization with large language models," in *Proc. 42nd International Conference on Machine Learning (ICML)*, Vienna, Austria, Jul. 2025.

[18] J. Adamczyk, V. Makarenko, S. Tiomkin, and R. V. Kulkarni, "Bootstrapped reward shaping," in *Proc. 39th AAAI Conference on Artificial Intelligence (AAAI-25)*, Philadelphia, Pennsylvania, USA, Feb.–Mar. 2025, pp. 15 302–15 310.

[19] R. Rafailov, A. Sharma, E. Mitchell, S. Ermon, C. D. Manning, and C. Finn, "Direct preference optimization: Your language model is secretly a reward model," in *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, no. 2338, New Orleans, LA, USA, Dec. 2023, pp. 53 728–53 741.

[20] H. Ahn, S. Oh, G. Kim, S. Jung, S. Park, and J. Kim, "Hallucination-aware generative pretrained transformer for cooperative aerial mobility control," 2025. [Online]. Available: https://arxiv.org/abs/2504.10831

[21] Q. Hao, Y. Song, Q. Liao, J. Yuan, and Y. Li, "LLM-Explorer: A plug-in reinforcement learning policy exploration enhancement driven by large language models," arXiv preprint arXiv:2505.15293, 2025.

[22] W. Shi, X. He, Y. Zhang, C. Gao, X. Li, J. Zhang, Q. Wang, and F. Feng, "Large language models are learnable planners for long-term recommendation," in *Proc. 47th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, Washington, DC, USA, Jul. 2024, pp. 1893–1903.

[23] F. Yang, J. Liu, and K. Li, "LLM-guided reinforcement learning for interactive environments," *Mathematics*, vol. 13, no. 12, p. 1932, Jun 2025.

[24] W. Li, D. Qiao, B. Wang, X. Wang, B. Jin, and H. Zha, "Semantically aligned task decomposition in multi-agent reinforcement learning," arXiv preprint arXiv:2305.10865, 2023.

[25] C. L. Shek and P. Tokekar, "Option discovery using LLM-guided semantic hierarchical reinforcement learning," arXiv preprint arXiv:2503.19007, 2025.

[26] X. Wan, W. Xu, C. Yang, and M. Sun, "Think twice, act once: A co-evolution framework of LLM and RL for large-scale decision making," in *Proc. 42nd International Conference on Machine Learning (ICML)*, Vancouver, Canada, Jul. 2025.

[27] Y. Yuan and T. Xie, "Reinforce LLM reasoning through multi-agent reflection," in *Proc. 42nd International Conference on Machine Learning (ICML)*, Vancouver, Canada, Jul. 2025.

[28] B. Wang, Y. Qu, Y. Jiang, J. Shao, C. Liu, W. Yang, and X. Ji, "LLM-empowered state representation for reinforcement learning," in *Proc. 41st International Conference on Machine Learning (ICML)*, no. 2106, Vienna, Austria, Jul. 2024, pp. 51 348–51 375.

[29] Z. Li, Y. Cao, and J. C. K. Cheung, "Do LLMs build world representations? Probing through the lens of state abstraction," in *Proc. 38th Conference on Neural Information Processing Systems (NeurIPS)*, no. 3110, Vancouver, Canada, Jun. 2024, pp. 98 009–98 032.

[30] X. Yan, Y. Song, X. Feng, M. Yang, H. Zhang, H. B. Ammar, and J. Wang, "Efficient reinforcement learning with large language model priors," in *Proc. 13th International Conference on Learning Representations (ICLR)*, Singapore, Apr. 2025.

[31] A. Cheng, Y. Ji, Z. Yang, Z. Gongye, X. Zou, J. Kautz, E. Bıyık, H. Yin, S. Liu, and X. Wang, "NaVILA: Legged robot vision-language-action model for navigation," in *Proc. Robotics: Science and Systems (RSS)*, Jun. 2025.

[32] K. Karine and B. M. Marlin, "Combining LLM decision and RL action selection to improve rl policy for adaptive interventions," arXiv preprint arXiv:2501.06980, 2025.

[33] Z. Zhou, B. Hu, C. Zhao, P. Zhang, and B. Liu, "Large language model as a policy teacher for training reinforcement learning agents," in *Proc. Thirty-Third International Joint Conference on Artificial Intelligence (IJCAI-24)*, no. 627, Jeju, South Korea, Aug. 2024.

[34] H. Ma, T. Hu, Z. Pu, B. Liu, X. Ai, Y. Liang, and M. Chen, "Coevolving with the other you: Fine-tuning LLM with sequential cooperative multi-agent reinforcement learning," in *Proc. 38th Conference on Neural Information Processing Systems (NeurIPS)*, vol. 37, no. 495, Vancouver, Canada, Dec. 2024.

[35] M. Xu, Y. Shen, S. Zhang, Y. Lu, D. Zhao, J. B. Tenenbaum, and C. Gan, "Prompting decision transformer for few-shot policy generalization," in *Proc. 39th International Conference on Machine Learning (ICML)*, vol. 162, Baltimore, USA, Jul. 2022, pp. 24 631–24 645.

[36] T. Pouplin, K. Kobalczyk, H. Sun, and M. van der Schaar, "The synergy of LLMs & RL unlocks offline learning of generalizable language-conditioned policies with low-fidelity data," in *Proc. 42nd International Conference on Machine Learning (ICML)*, Jul. 2025.

[37] Z. Wang, J. Yang, S. Nag, S. Varshney, X. Tang, H. Jiang, J. Shang, and S. M. Sarwar, "RRO: LLM agent optimization through rising reward trajectories," arXiv preprint arXiv:2505.20737, 2025.

[38] L. Chen, N. Moorman, and M. Gombolay, "ELEMENTAL: Interactive learning from demonstrations and vision-language models for reward design in robotics," in *Proc. 42nd International Conference on Machine Learning (ICML)*, Jul. 2025.

[39] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David *et al.*, "Do as i can, not as i say: Grounding language in robotic affordances," arXiv preprint arXiv:2204.01691, 2022.

[40] Y. Du, O. Watkins, Z. Wang, C. Colas, T. Darrell, P. Abbeel, A. Gupta, and J. Andreas, "Guiding pretraining in reinforcement learning with large language models," in *Proc. 40th International Conference on Machine Learning (ICML)*, vol. 202, no. 346, Honolulu, Hawaii, USA, 2023, pp. 8657–8677.

[41] Y. Wu, Y. Fan, P. P. Liang, A. Azaria, Y. Li, and T. M. Mitchell, "Read and reap the rewards: Learning to play atari with the help of instruction manuals," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, vol. 36, no. 48, New Orleans, LA, USA, Dec 2023, pp. 1009–1023.

[42] A. Zala, J. Cho, H. Lin, J. Yoon, and M. Bansal, "Envgen: Generating and adapting environments via LLMs for training embodied agents," in *Proc. Conference on Language Modeling (COLM)*, Philadelphia, PA, USA, Oct. 2024.

[43] S. Park, X. Du, M.-H. Yeh, H. Wang, and S. Li, "Steer LLM latents for hallucination detection," in *Proc. 42nd International Conference on Machine Learning (ICML)*, Vancouver, Canada, Jul. 2025.

[44] B. Paudel, A. Lyzhov, P. Joshi, and P. Anand, "Hallucinot: Hallucination detection through context and common knowledge verification," arXiv preprint arXiv:2504.07069, 2025.

[45] Y. Sun, J. Shen, Y. Wang, T. Chen, Z. Wang, M. Zhou *et al.*, "Improving data efficiency for LLM reinforcement fine-tuning through difficulty-targeted online data selection and rollout replay," arXiv preprint arXiv:2506.05316, 2025.

[46] G. Zhu, R. Zhou, W. Ji, and S. Zhao, "LAMARL: LLM-aided multi-agent reinforcement learning for cooperative policy generation," *IEEE Robotics and Automation Letters*, vol. 10, no. 7, pp. 7476–7483, Jul. 2025.

[47] J. Jiang, C. Si, J. Luo, H. Zhang, and C. Ma, "Co-reinforcement learning for unified multimodal understanding and generation," arXiv preprint arXiv:2505.17534, 2025.