

Multi-Round Reinforcement Learning with Feedback Reflection for Large Language Models Post-Training

Rundong Wang^{*}, Xiaobo Yang[†], Zhanfeng Mo^{*}, Shihong Deng[†], Shuchang Zhou[‡]

Nanyang Technological University^{*}, Peking University[†], StepFun AI[‡]
rundong001@e.ntu.edu.sg, 1800010745@pku.edu.cn, zhanfeng001@e.ntu.edu.sg, dshnightmare@gmail.com
shuchang.zhou@gmail.com

Abstract

While Large language models (LLMs) have demonstrated strong reasoning capabilities, solving complex tasks in interactive environments requires more than single-round responses; they must learn to act strategically, reflect on feedback, and iteratively refine their responses. In this paper, we propose Multi-round Reinforcement Learning (MRRL) for LLMs, a novel post-training framework to complete tasks through strategic interactions with environments while learning to revise their responses by reflecting on and integrating the environment feedback. Beyond environment rewards, our MRRL leverages text-based feedback from the environment, which provides richer and more explicit guidance toward task completion. We formalize the LLM interaction process as a Markov Decision Process (MDP) and derive the proximal policy gradient update for MRRL. To help the LLM reflect on and leverage environmental feedback, we propose the Feedback Reflection Imitation Learning algorithm, which uses an LLM to generate feedback reflection and alleviate the distribution gap for imitation learning from successful trajectories. We conduct extensive experiments across diverse tasks, including text-based games, mathematical problems, search tasks, and logical puzzles, validating that our MRRL achieves stable multi-round RL training for LLMs in various environments. Furthermore, MRRL reduces the number of interaction rounds needed to complete tasks, unleashing the full potential of LLMs in multi-round reasoning. Our code is available at <https://github.com/xiaobo-yang/TinyAgentZero>.

Introduction

Large language models (LLMs) (Achiam et al. 2023; DeepSeek-AI et al. 2024; Team et al. 2024; Grattafiori et al. 2024) have recently achieved remarkable performance across a wide range of reasoning tasks (Huang and Chang 2023; Hao et al. 2025), particularly through techniques such as chain-of-thought (CoT) prompting (Wei et al. 2022; Chu et al. 2024), which enables structured, step-by-step reasoning. After being pretrained on massive and diverse text corpora via next-token-prediction (Brown et al. 2020), these models are capable of generating coherent, contextually appropriate responses, often matching or surpassing human-level performance on complex benchmarks (Cobbe et al. 2021; Jain et al. 2025; MAA 2024). Such results highlight the promise

of LLMs as general-purpose reasoning engines, particularly when tasks can be framed as few-shot language prompts.

While existing literature primarily focuses on the reasoning capabilities of LLMs within a single response, many complicated sequential decision-making tasks, such as text-based games, searching, and planning, require multi-round interactions. In these settings, LLMs must learn to reflect on the resulting feedback or state transition and generate successive responses accordingly. While LLMs have exhibited strong capability in single-shot reasoning tasks, they are not inherently able to learn from feedback or to improve their performance over multiple rounds of interactions, leading to limited performance in tasks that require exploration, revision, and learning from failure.

To address this limitation, we propose Multi-round Reinforcement Learning (MRRL), a post-training framework to enable LLMs to act strategically with interactive environments. Beyond environment rewards, our MRRL leverages text-based feedback from the environment, which provides richer and more explicit guidance toward task completion. We formalize the LLM-environment interaction as a Markov Decision Process (MDP), in which each generated response corresponds to an action, and environment feedback is treated as part of the state transition and reward signals. We derive the proximal policy gradient update for MRRL, which can be simply implemented using feedback masking. Furthermore, we introduce the Feedback Reflection Imitation Learning algorithm, which encourages LLM to learn from successful LLM-environment interaction trajectories that consist of both high-quality responses and feedback reflections. Specifically, the feedback reflections are generated by instructing a reference LLM to rephrase the raw environmental feedback. This approach reduces the distribution gap between the raw feedback and the LLM’s generation policy, thereby stabilizing the imitation learning process.

We conduct extensive experiments across a diverse set of interactive tasks that require reasoning, planning, and feedback-driven adaptation. Our results show that MRRL enables stable and effective multi-round reinforcement learning for LLMs. Our proposed MRRL consistently outperforms multiple baselines in success rate. Moreover, our MRRL effectively reduces the number of interaction rounds across various scenarios, demonstrating the potential of LLMs for multi-round reasoning tasks in interactive environments.

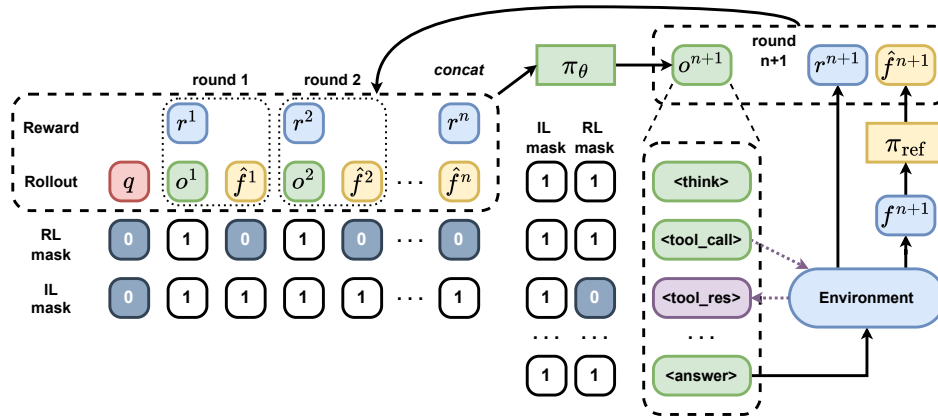


Figure 1: An illustration of our proposed MRRL framework. At each round, the LLM policy π_θ generates a response conditioned on the interaction history. The environment evaluates this response and returns text-based feedback, such as a comment on its correctness, along with a scalar reward. A reference policy π_{ref} then produces a reflection based on the feedback to guide subsequent learning. During training, feedback masking disables gradients from environment feedback tokens in the RL loss, while enabling gradient flow across all tokens in successful rollouts for the imitation loss. Mask values equaling 1 or 0 indicate whether gradient backpropagation is enabled or disabled.

Related Works

LLM Reasoning with RL

The integration of reinforcement learning into LLMs began with Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al. 2022), which fine-tunes models using preference data collected from human annotators to align outputs with human values. This approach has become foundational for aligning generative models with intended behavior. Building on this foundation, reinforcement learning has also been applied to enhance chain-of-thought (CoT) prompting (Wei et al. 2023), enabling LLMs to perform more structured, multi-step reasoning. More recently, large-scale RL-based post-training has led to state-of-the-art performance in complex reasoning tasks (Jaech et al. 2024), demonstrating the scalability and effectiveness of RL for reasoning-oriented capabilities. This paradigm has also been extended to open-weight models, broadening access and reproducibility in the community (Guo et al. 2025). A notable recent trend is the development of zero-setting approaches, which apply RL directly to base LLMs without requiring task-specific fine-tuning. This strategy supports generalization and simplifies deployment pipelines, inspiring a wave of reproducible, open-source efforts such as DAPO (Yu et al. 2025) and Open Reasoner Zero (Hu et al. 2025). These works collectively reflect a growing interest in leveraging reinforcement learning to unlock and stabilize complex reasoning behaviors in LLMs.

LLM Agent with Multi-round RL

Recent work on language agents has explored how LLMs interact with tools and environments over extended decision sequences. ReAct (Yao et al. 2023) enhances the capabilities of Large Language Models (LLMs) by enabling them to interact with external environments and make decisions in a more dynamic and informed way. The Berkeley Function-Calling Leaderboard (Yan et al. 2024) categorizes these interactions into two paradigms: multi-step tool use and multi-round en-

vironment interaction. In multi-step settings, LLMs invoke tools, such as code interpreters (Feng et al. 2025) or search engines (Jin et al. 2025), multiple times to solve a single query, a process known as tool-integrated reasoning (Gou et al. 2023). Recent studies have applied reinforcement learning to improve tool-use strategies (Liu et al. 2023; Song et al. 2025; Feng et al. 2025; Qian et al. 2025), enabling models to generalize across tasks. Multi-round settings, in contrast, involve extended interactions between LLMs, users, and environments, posing challenges in maintaining context, adapting to feedback, and planning. Approaches in this space range from training LLMs for human-AI collaboration (Zhou et al. 2025) to grounding them in interactive environments (Wang et al. 2025). RL-based approaches have also treated LLMs as decision-making agents for long-horizon tasks (Feng et al. 2024; Zhai et al. 2024; Zhou et al. 2024; Liu et al. 2024). In this work, we enable LLM post-training in a ReAct style and use the term multi-round interaction to refer to scenarios where an LLM generates a sequence of responses, receives feedback from the environment after each round, and strategically revises its behavior based on that feedback.

Preliminaries

Markov Decision Process. A Markov Decision Process (MDP) provides a formal framework for modeling multi-round decision-making, where an agent interacts iteratively with the environment across multiple rounds to accomplish a specific task. An MDP is defined by the tuple $(\mathcal{S}, \mathcal{A}, \rho, P, r, \gamma, T)$, where \mathcal{S} is the set of states; \mathcal{A} is the set of valid actions; $\rho(\cdot)$ is the probability measure of the initial state $s_0 \in \mathcal{S}$ in the environment; for any state-action pair $(s, o) \in \mathcal{S} \times \mathcal{A}$, $P(\cdot|s, o)$ represents a probability measure over \mathcal{S} , where $P(s'|s, o)$ indicates the probability of transitioning from the current state s to the next state s' upon taking action o ; $R(s, o) : \mathcal{S} \times \mathcal{A} \mapsto [0, 1]$ is reward function, where a larger value indicates a higher degree of task accomplishment and a value of 1 signifies successful task

completion; $\gamma \in [0, 1]$ is a discount factor; T is the largest possible interaction round. Suppose π_θ is a policy parameterized by θ in this MDP, such that for any $s \in \mathcal{S}$ and $o \in \mathcal{A}$, $\pi_\theta(\cdot|s, o)$ is a probability measure over \mathcal{S} . At the t -th interaction step, the agent selects an action o^{t+1} in state s^t according to the policy, i.e., $o^{t+1} \sim \pi_\theta(\cdot|s^t)$. The agent then transitions to the next state s^{t+1} following the state transition probability $s^{t+1} \sim P(\cdot|s^t, o^{t+1})$, and receives a reward $r^{t+1} \triangleq R(s^{t+1}, o^{t+1})$ based on the chosen action. The objective of the agent is to learn an optimal policy that maximizes the expected return, that is, the expected sum of discounted rewards over time:

$$\max_{\theta} J(\pi_\theta) \triangleq \mathbb{E}_{\substack{s^0 \sim \rho(\cdot), \\ o^t \sim \pi_\theta(\cdot|s^{t-1}), \\ s^t \sim P(\cdot|s^{t-1}, o^t)}} \left[\sum_{t=1}^T \gamma^t R(s^t, o^t) \right]. \quad (1)$$

RL-based LLM Post-Training. RL-based LLM post-training methods focus on refining a pre-trained LLM π_θ by adapting it to specific tasks or domains (e.g., text-based games, tool usage) using reinforcement learning algorithms. Suppose V is a finite text token vocabulary, and we denote the space of all token sequences as $V^\infty \triangleq \cup_{L=1}^\infty V^L$. In these scenarios, the training data consists of query-answer pairs $(q, a) \in V^\infty \times V^\infty$ sampled from an underlying distribution $(q, a) \sim \mathcal{D}$. For each query q , the LLM π_θ aims to generate text-based response $o \sim \pi_\theta(\cdot|q)$ that leads to the ground truth answer a . Therefore, this solution generation process of the LLM can be modeled as a one-round MDP $(\mathcal{S}, \mathcal{A}, \rho, P, R, \gamma, T)$ with $T = 1$. The state space $\mathcal{S} \subset V^\infty$ represents the current context, which is either the query alone or the query concatenated with the LLM’s generated answer. The action space $\mathcal{A} \subset V^\infty$ consists of all possible generated sequences. For any query q and LLM-generated answer o , the next state is defined as the concatenation of q and o , denoted by $[q, o]$. As a result, the transition probability $P(\cdot|q, o)$ is deterministic and is equivalent to $\delta_{[q, o]}(\cdot)$, the Dirac delta measure centered at $[q, o]$. Let a denote the ground-truth answer for query q . The reward is defined as $R(o, a)$, a $[0, 1]$ -valued verification function, where a higher value indicates better alignment between o and a , and a value of 1 denotes a correct answer. The objective of LLM RL post-training is:

$$\max_{\theta} J(\pi_\theta) \triangleq \mathbb{E}_{(q, a) \sim \mathcal{D}, o \sim \pi_\theta(\cdot|q)} [R(o, a)]. \quad (2)$$

Group Relative Policy Optimization. Group Relative Policy Optimization (GRPO) (Shao et al. 2024) has become one of the most popular policy gradient algorithms (Kaelbling, Littman, and Moore 1996; Schulman et al. 2017; Sutton et al. 1999) for optimizing (2). At each optimization step, GRPO updates the parameters of the LLM policy using gradient ascent on a surrogate training objective $J_{\text{GRPO}}(\pi_\theta)$. To compute this objective, we first generate a group of G rollout samples from a frozen LLM π_{old} , denoted by $\{o_1, o_2, \dots, o_G\}$, where each rollout is sampled from $o_i \sim \pi_{\text{old}}(\cdot|q)$. This LLM is typically a past snapshot of π_θ . We then evaluate the

following expression:

$$J_{\text{GRPO}}(\theta) = \mathbb{E}_{\substack{(q, a) \sim \mathcal{D} \\ \{o_i\}_{i=1}^G \sim \pi_{\text{old}}(\cdot|q)}} \left[\frac{1}{G} \sum_{i=1}^G \hat{r}_i(\theta) - \beta \text{KL}_i(\theta) \right],$$

$$\hat{r}_i(\theta) \triangleq \min(r_i(\theta) A_i, \text{clip}_\epsilon(r_i(\theta)) A_i),$$

$$\text{KL}_i(\theta) \triangleq \frac{\pi_{\text{ref}}(o_i|q)}{\pi_\theta(o_i|q)} - \log\left(\frac{\pi_{\text{ref}}(o_i|q)}{\pi_\theta(o_i|q)}\right) - 1,$$

Specifically, $r_i(\theta) \triangleq \pi_\theta(o_i|q)/\pi_{\text{old}}(o_i|q)$ denotes the probability ratio, $\text{clip}_\epsilon(\cdot)$ is the clipping function that truncates any real-value into the range $[1 - \epsilon, 1 + \epsilon]$, $A_i \triangleq (R(o_i, a) - \text{mean}(\{R(o_i, a)\}_{i=1}^G))/\text{std}(\{R(o_i, a)\}_{i=1}^G)$ denotes the group normalized reward and each $R(o_i, a)$ denotes the reward on the i -th rollout sample. $\epsilon \in (0, 1)$ is the clipping threshold hyperparameter, $\beta > 0$ is the penalty weight, and $\text{KL}_i(\theta)$ is the KL-divergence between π_θ and π_{ref} , where π_{ref} is another frozen LLM that is usually set as the initialization of π_θ (e.g. the SFT model or pretrained base model) (Shao et al. 2024). Intuitively, GRPO optimizes the target LLM policy π_θ to increase the likelihood of generating high-quality answers, prioritizing samples o_i with large group-normalized rewards A_i . The clipping operator and the KL-divergence penalty are applied to prevent excessively large updates during training, thereby stabilizing the optimization process.

Methodology

In this section, we introduce **Multi-Round Reinforcement Learning (MRRL)**, a reinforcement learning framework that enables LLMs to complete tasks through strategic interactions with the environment, while also learning to revise their responses by reflecting on and integrating the feedback. We first derive the proximal policy gradient update for MRRL and demonstrate how it can be efficiently implemented using *feedback masking*. In Section Preliminaries, we extend MRRL with *feedback reflection imitation*, enabling LLMs to learn task-specific knowledge from environmental feedback and to reflect on their interactions with the environment.

Multi-Round RL for LLM

MRRL Objective. MRRL considers using LLM to interact with an environment for multiple rounds to accomplish the specific task. Beyond environment rewards, our MRRL leverages text-based feedback from the environment, which provides richer and more explicit guidance toward task completion. For instance, in a game-playing scenario, we aim to solve a text-based puzzle in multiple rounds via LLM: at each round, the LLM generates the next move and receives execution feedback from the puzzle verifier, which indicates whether the current move is valid or if the final goal has been achieved. Suppose the target task involves finding the answer a to each query q , where each query-answer pair is sampled from an underlying distribution $(q, a) \sim \mathcal{D}$. The multi-round interaction can be formalized as an MDP $(\mathcal{S}, \mathcal{A}, \rho, P, R, \gamma, T)$ with $T > 1$. Specifically, $\mathcal{A} \subset V^\infty$ is the action space, consisting of all possible answers generated by the LLM. $\mathcal{S} \subset V^\infty$ represents the state space, where each state $s \in \mathcal{S}$ consists of the full LLM context,

Algorithm 1: Multi-Round RL with Feedback Reflection Imitation Learning

```

# main loop of MRRL training
for query in dataloader:
    traj, mask, rewards = [query], [zeros_like(query)], []
    for t in range(T):
        resp = llm.generate(traj) # Generate rollout samples for each query
        feedback_raw, reward = env.interact(resp) # receive environment feedback
        reflection_prompt = traj + [resp, feedback_raw, reflection_inst]

        feedback_reflection = llm_ref.generate(reflection_prompt) # feedback reflection

        traj = traj + [resp, feedback_reflection]
        mask = mask + [ones_like(resp), zeros_like(feedback_reflection)] # feedback masking
        rewards.append(reward)

    loss = ppo_loss(traj, mask, rewards) + lam * imitation_loss(traj, rewards)
    loss.backward() # update LLM
    optimizer.step()

```

including the query, all previous answer attempts, and the environment feedback. For any query-answer pair (q, a) , and an LLM-generated answer o , $R(o, a, f)$ is the reward function defined above. The key distinction compared to conventional RL-based LLM post-training is that the transition probability $P(\cdot|s, o)$ is no longer deterministic, as it depends on the distribution of the environment feedback.

For each query-answer pair (q, a) , the LLM interacts with the MDP to address q within T rounds. Let o^t be the LLM-generated answer and f^t be the environment feedback at the t -th round. The state is defined as the temporary context, i.e., $s^t \triangleq [q, o^1, f^1, \dots, o^t, f^t]$. At round t , the LLM generates a new response based on the current context, i.e., $o^t \sim \pi_\theta(\cdot|s^{t-1})$, which is then sent to the environment to receive corresponding textual feedback, denoted by $f^t \sim P(\cdot|s^{t-1}, o^t)$. The state is then updated by appending the latest answer and feedback to the existing context, i.e., $s^t \triangleq [s^{t-1}, o^t, f^t] = [q, o^1, f^1, \dots, o^t, f^t]$, and the initial state is defined as $s^0 \triangleq q$. Meanwhile, a reward $R(o^t, a, f^t)$ is computed, and the LLM proceeds to the next round to make another attempt at solving the task. If the LLM generates the correct answer at round t , the interaction is terminated, and the remainder of the trajectory is padded with placeholders. This iterative process produces an answer-feedback trajectory for query q , denoted by $\tau \triangleq [o^1, f^1, \dots, f^{T-1}, o^T] \sim (\pi_\theta(\cdot|q), P(\cdot|\cdot, \cdot))$, where π_θ generates answers and P returns environment feedback. MRRL aims to maximize the expected return, formally defined as:

$$J_{\text{MRRL}}(\pi_\theta) \triangleq \mathbb{E}_{\substack{(q,a) \sim \mathcal{D} \\ \tau \sim (\pi_\theta(\cdot|q), P(\cdot|\cdot, \cdot))}} \left[\sum_{t=1}^T \gamma^t R(o^t, a, f^t) \right]. \quad (3)$$

The key distinction between conventional single-round RL-based LLM post-training and MRRL is that, while each rollout trajectory in single-round RL contains only LLM-generated tokens, each MRRL trajectory includes both LLM-generated tokens o_i^t and environment feedback tokens f_i^t . This requires a carefully designed policy gradient estimator to ensure the LLM effectively learns to generate high-quality answers throughout the multi-round interaction trajectory.

MRRL Optimization. Proposition 1 presents the proximal policy gradient estimator for (3), which enables approximate

updates of π_θ in the direction of the MRRL policy gradient $\nabla_\theta J_{\text{MRRL}}(\pi_\theta)$.

Proposition 1 (Proximal Policy Gradient Estimator of MRRL). *Under the previous conditions, a proximal policy gradient estimation is given by $\nabla_\theta \hat{J}_{\text{MRRL}}(\pi_\theta)$, where the MRRL training objective is*

$$\hat{J}_{\text{MRRL}}(\pi_\theta) \triangleq \mathbb{E}_{\substack{(q,a) \sim \mathcal{D} \\ \{\tau_i\}_{i=1}^G \sim (\pi_{\text{old}}(\cdot|q), P(\cdot|\cdot, \cdot))}} \left[\frac{1}{G} \sum_{i=1}^G \sum_{t=1}^T \hat{r}_{i,t}(\theta) \right], \quad (4)$$

where $\hat{r}_{i,t}(\theta) \triangleq \min \left(r_i^t(\theta) \hat{A}_i^t, \text{clip}_\epsilon(r_i^t(\theta)) \hat{A}_i^t \right)$, $r_i^t(\theta) \triangleq \pi_\theta(o_i^t|s_i^{t-1})/\pi_{\text{old}}(o_i^t|s_i^{t-1})$ denotes the ratio factor. \hat{A}_i^t is implemented as one of the generalized advantage estimators (Schulman et al. 2016), such as the group normalized trajectory return $\hat{A}_i^t \triangleq (R_i - \text{mean}(\{R_i\}_{i=1}^G))/\text{std}(\{R_i\}_{i=1}^G)$ with $R_i \triangleq \sum_{t=1}^T \gamma^t R(o_i^t, a, f_i^t)$ denoting the return of τ_i .

The proof of Proposition 1 is detailed in Appendix. To compute the MRRL training objective (4), at each optimization step, we first sample a batch of query-answer pairs from the underlying dataset $(q, a) \sim \mathcal{D}$. For each query q , we sample G trajectories $\{\tau_i\}_{i=1}^G \sim (\pi_{\text{old}}(\cdot|q), P(\cdot|\cdot, \cdot))$ from a T -round interaction with the MDP, using the rollout policy π_{old} . The parameters of π_{old} are frozen during sampling and periodically synchronized with those of π_θ . After calculating the return of each trajectory R_i , we obtain \hat{A}_i^t via group-level reward normalization. Finally, we calculate the ratio terms $r_i^t(\theta)$ over the LLM-generated answer tokens and plug them into (4).

MRRL with Feedback Masking. Unlike conventional single-round RL optimization methods such as GRPO (Shao et al. 2024) or PPO (Schulman et al. 2017), which compute the policy gradient over all tokens in the rollout trajectories, Proposition 1 explicitly excludes terms associated with environment feedback tokens by zeroing out their corresponding likelihood ratios: the MRRL training objective (4) does not contain any $\pi_\theta(f_i^t|s_i^{t-1}, o_i^t)/\pi_{\text{old}}(f_i^t|s_i^{t-1}, o_i^t)$ factors. As outlined in Algorithm 1, we implement this using a **feedback masking** strategy: after computing the trajectory probability $\pi_\theta(\tau|q)$ with a single forward pass of the target

LLM π_θ , the output probabilities corresponding to environment feedback tokens f_i^t are masked as zero before stepping into the standard proximal policy gradient update steps. The rationale behind feedback masking is that the sampling of environment feedback is externally determined and independent of the LLM’s parameters. As such, it does not contribute to the policy gradient. Masking these tokens is essential to ensure that the LLM focuses on optimizing its own generated outputs while mitigating the risk of overfitting to environment feedback, which may be noisy and non-stationary. This principle is validated by our derivation in the Appendix and the results presented in the Experiments Section.

MRRL with Feedback Reflection Imitation

While Proposition 1 facilitates the optimization of the MRRL objective, the feedback masking mechanism prevents the LLM from directly utilizing the information contained in the environment feedback. In many multi-round task-solving scenarios, such feedback is crucial for refining responses, and ignoring it can hinder the model’s ability to identify and correct errors in previous answers.

MRRL with Imitation Learning. To address this limitation, we propose training the model to simultaneously learn how to interact with the environment and how to incorporate environment feedback into the LLM policy. This involves enabling the LLM to interpret feedback, reflect on it, and identify potential errors and opportunities for improvement. Specifically, we propose the **MRRL with Imitation Learning** (MRRL w/ IL) framework, which enables the LLM to selectively imitate the positive interaction trajectories, namely, complete interactions that lead to successful outcomes. The modified training objective is defined as follows:

$$\hat{J}_{\text{MRRL}}(\pi_\theta) \triangleq \mathbb{E}_{(q,a) \sim \mathcal{D}} \left[\underbrace{\frac{1}{G} \sum_{i=1}^G \sum_{t=1}^T \hat{r}_{i,t}(\theta)}_{\text{Reinforcement Learning Loss}} - \underbrace{\frac{\lambda}{|\mathcal{T}|} \sum_{\tau_i \in \mathcal{T}} \frac{1}{|\tau_i|} \log \pi_\theta(\tau_i|q)}_{\text{Imitation Learning Loss}} \right], \quad (5)$$

where \mathcal{T} denotes the success trajectories among the G roll-out samples, $\lambda > 0$ is the penalty weight of the imitation loss. The difference compared to (4) are highlighted in blue. Intuitively, (5) encourages the model to maximize the likelihood of successful trajectories, effectively imitating both high-quality responses and the associated environment feedback. By maximizing the likelihood of generating the entire trajectory, including both LLM-generated and feedback tokens, the model learns to produce better answers and interpret and incorporate feedback for iterative refinement.

MRRL with Feedback Reflection Imitation. However, directly adding imitation loss to MRRL can be problematic, as it may cause the model to overfit to the environment feedback. This is particularly risky when the feedback is irregular or noisy, for example, text describing partial progress on a puzzle, or raw output from an external tool. As the underlying distribution of the environment feedback may differ significantly from the LLM’s generation distribution, the model may struggle to learn meaningful patterns from the feedback.

To mitigate this issue, we propose a novel feedback reflection mechanism that leverages a reference LLM to reflect on the environment feedback and generate a more informative version of it, which can be used to guide the subsequent interactions.

At the t -th round of interaction, after generating an answer o^t and receiving raw environment feedback f^t , the model samples a feedback reflection $\hat{f}^t \sim \pi_{\text{ref}}(\cdot | [s^{t-1}, o^t, \text{inst}, f^t])$. Here, π_{ref} is a reference LLM with frozen parameters, which can be instantiated as either an SFT model or the initial checkpoint of π_θ . $\text{inst} \in V^\infty$ is a fixed instruction prompting the model to reflect on the feedback and its prior response. The reflection \hat{f}^t is then appended to the context to guide the next round of interaction. We denote the resulting trajectory as $\tau_+ \triangleq [o^1, \hat{f}^1, \dots, \hat{f}^{T-1}, o^T] \sim (\pi_{\text{old}}(\cdot | q), P(\cdot | \cdot, \cdot), \pi_{\text{ref}}(\cdot | \cdot))$, where the notation compactly represents the sampling process: π_{old} generates answers, P provides environment feedback, and π_{ref} produces feedback reflections. The state is redefined as $s^t \triangleq [q, o^1, \hat{f}^1, \dots, o^t, \hat{f}^t]$. This defines the **MRRL with Feedback Reflection Imitation** (MRRL w/ FRI) training objective, expressed as:

$$\hat{J}_{\text{MRRL}}(\pi_\theta) \triangleq \mathbb{E}_{(q,a) \sim \mathcal{D}} \left[\underbrace{\frac{1}{G} \sum_{i=1}^G \sum_{t=1}^T \hat{r}_{i,t}(\theta)}_{\text{Reinforcement Learning Loss}} - \underbrace{\frac{\lambda}{|\mathcal{T}|} \sum_{\tau_i \in \mathcal{T}} \frac{1}{|\tau_{+,i}|} \log \pi_\theta(\tau_{+,i}|q)}_{\text{Imitation Learning Loss}} \right], \quad (6)$$

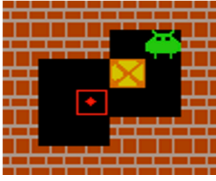
The key distinction from the original objective in (5) is that the raw feedback f^t is replaced by its reflection \hat{f}^t in the trajectory, and both the MRRL loss and imitation loss are computed over the modified trajectory $\tau_{+,i}$ instead of the original τ_i . This has two advantages: 1. The reflected feedback, generated by the reference LLM, filters out irrelevant or noisy details (e.g., irregular formatting from tool outputs or puzzle interpreters), helping the target policy better extract essential information from the environment feedback; and 2. The distribution of the LLM-generated feedback reflections are closer to that of the target model’s outputs, which makes imitation learning more effective. This setup encourages the model to learn how to reflect on feedback and iteratively improve its responses, mimicking a multi-round reasoning process. A Pytorch-style pseudo-code of our proposed method is outlined in Algorithm 1.

Experiments

Datasets or Environments

We evaluate MRRL across four benchmark domains designed to stress-test multi-round reasoning under distinct challenge regimes:

Sokoban: Sokoban (Schrader 2018) is a grid-based puzzle game requiring agents to push crates to target positions while avoiding irreversible deadlocks. This environment evaluates spatial planning, recovery from erroneous actions, and long-horizon strategy adaptation through incremental environment feedback (e.g., invalid move notifications, crate positions).



```

# # # # #
# # # _ P #
# _ _ X _ #
# _ O _ _ #
# _ _ # # #
# # # # #

```

(a) An example of Sokoban game.

There are 4 houses, numbered from 1 to 4. Each house has a different owner, and each person has 3 characteristics. These possible characteristics are as follows:

- Pet: **Goldfish, Hamster, Rat, Fish**
- Favorite Music Genre: **Funk, Pop, Indie, Jazz**
- Favorite Sport: **Skateboarding, Soccer, Skiing, Swimming**

Background

Clues

1. **Goldfish** is nearby **Rat**
2. **Rat** is nearby **Pop**
3. **Fish** is between **Swimming** and **Skiing**
4. **Jazz** is on the right of **Soccer**
5. **Pop** is between **Skiing** and **Indie**
6. **Skiing** is either on the far left or far right

Solution

Sports Pet Music

1	Skateboarding	Goldfish	Funk
2	Swimming	Rat	Indie
3	Soccer	Fish	Pop
4	Skiing	Hamster	Jazz

(b) An example of Zebra Logic puzzle with 4 houses, 3 attributes, and 6 clues. One should use logical deduction to get the solution based on the background and the clues.

Figure 2: An illustration of environments. All environments are text-based and implemented with the standard APIs of Gym (Brockman et al. 2016).

Question Answering (QA): Using Natural Questions (NQ) (Kwiatkowski et al. 2019), we simulate a search-augmented QA setup where the agent iteratively refines queries, retrieves passages, and synthesizes answers. The environment provides feedback via retrieved text relevance scores and answer correctness signals, testing open-domain reasoning and evidence integration.

Mathematical Reasoning: We employ the Deepscaler dataset (Luo et al. 2025), featuring 40K problems spanning multiple data sources. Each problem is paired with a stepwise verifier that returns partial credit for intermediate results, enabling the model to revise solutions iteratively.

Zebra Logic Puzzles: Zebra (Lin et al. 2025) is a constrained satisfaction benchmark with natural language descriptions of entity-attribute relationships. The environment validates proposed solutions against implicit rules, returning where the error happens in the solution to guide iterative deduction.

For QA, math, and Zebra tasks, we implement interactive environments where the agent submits answers and receives structured feedback (e.g., retrieval scores, partial correctness, and constraint violations).

Experimental Setup

We measure performance using the following metrics: success rate, response length, and average environment interactions. To explore how model capacity influences MRRL’s performance, we experiment with three models of varying sizes: DeepSeek-R1-Distill-Qwen-1.5B (Guo et al. 2025), Qwen2.5-7B, and Qwen2.5-14B-Instruct (Yang et al. 2024). We extend the VERL framework (Sheng et al. 2024), a reinforcement learning platform, by adding a modular patch to support multi-round interactions. This approach preserves compatibility with VERL’s latest updates without altering its core structure. Each model is trained on a single node with 8× NVIDIA 80G H800 GPUs. More details on experimental settings can be found in the Appendix.

Main result

The first two rows in Figure 3 present normalized rewards for MRRL variants during training and inference across four environments: Sokoban, Mathematical Reasoning, Search Tasks, and Zebra Logic Puzzles. During training, MRRL w/ FRI consistently achieves higher asymptotic rewards than the baseline MRRL model, with pronounced gains in Sokoban and Zebra puzzles. This advantage stems from FRI’s ability to leverage multi-round interactions to refine reasoning chains, enabling more effective policy optimization.

In inference, MRRL w/ FRI demonstrates superior generalization, outperforming baselines in Mathematical Reasoning and Search Tasks. Structured feedback in these domains facilitates precise refinement of reasoning trajectories, yielding higher-quality solutions with fewer steps. These findings highlight that FRI not only accelerates training convergence but also enhances robustness during deployment, enabling large language models (LLMs) to produce high-quality solutions across diverse interactive settings. More results can be seen in Appendix.

Response Length

The third row in Figure 3 illustrates the evolution of response length during MRRL training across four environments. In Sokoban, MRRL exhibits a steady reduction in response length, reflecting improved spatial planning efficiency through iterative feedback. Mathematical Reasoning shows rapid early-stage reductions, indicating the framework’s ability to internalize error patterns and streamline derivations. Search Tasks demonstrate early stabilization of response lengths, driven by rapid adaptation to structured tool feedback. Zebra Logic Puzzles, however, require prolonged refinement, as the model distills complex constraint propagation into concise logical deductions.

Across all environments, MRRL w/ FRI achieves the shortest final response lengths, requiring fewer tokens than the baseline by integrating paraphrased feedback into the agent’s behavioral distribution. This reduction correlates with fewer redundant reasoning steps and faster task resolution, partic-

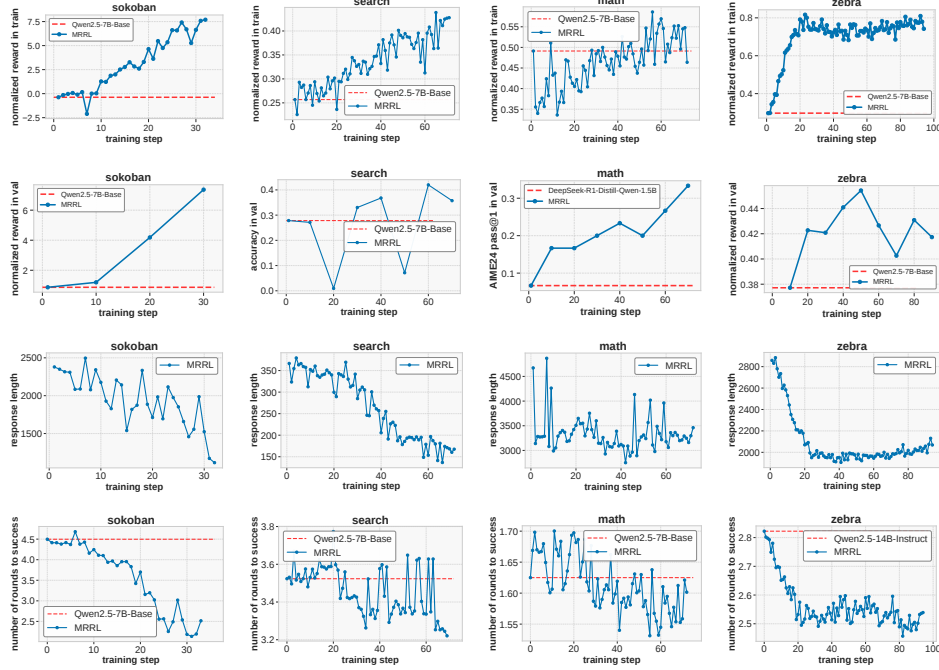


Figure 3: Performance during training, including normalized training reward, normalized validation reward, AIME24 accuracy pass@1, response length, and the number of rounds to success in different environments.

ularly in noisy feedback settings (e.g., partial puzzle states in Sokoban). These trends underscore MRRL’s ability to balance response quality with computational efficiency, a critical advancement for real-time LLM applications.

Environment Interaction Analysis

We also analyze the number of environment interaction steps in the last row in Figure 3. MRRL w/ FRI achieves the lowest average interaction steps per successful task, with significant reductions in Zebra Puzzles, Sokoban, and Mathematical Reasoning compared to the baseline. This efficiency arises from FRI’s mechanism to paraphrase raw feedback into the LLM’s intrinsic behavioral distribution, filtering task-specific noise (e.g., partial puzzle states or unstructured tool outputs) and accelerating the internalization of actionable heuristics. For example, in Zebra Puzzles, FRI agents streamline multi-round constraint propagation into structured deductions, minimizing redundant exploration.

These results demonstrate MRRL’s ability to optimize interaction efficiency while maintaining robustness, a key contribution for real-world deployment where computational and API costs are critical. By reducing interaction steps and response lengths, MRRL w/ FRI advances the scalability of LLMs in interactive, feedback-driven environments.

Ablation Study

We perform ablation studies on Zebra Logic Puzzles by varying λ , the weight of the feedback reflection imitation loss in MRRL. As shown in Figure 4, increasing λ leads to higher and more stable training rewards. Notably, even with $\lambda = 0.3$, where training is less stable, the final performance surpasses the baseline ($\lambda = 0$), indicating that the feedback reflection

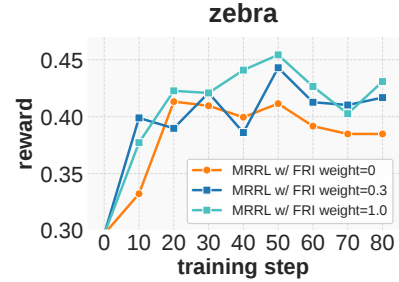


Figure 4: Ablation study on the weight of feedback reflection imitation loss.

imitation loss helps the LLM learn more effectively from the full trajectory.

Conclusion

We propose Multi-round Reinforcement Learning with Feedback Reflection Imitation (MRRL w/ FRI), a framework that enhances LLMs’ strategic reasoning in interactive settings. In addition to environment rewards, MRRL incorporates text-based feedback to provide richer task guidance. A reference LLM generates feedback reflections, enabling the target policy to learn from successful interaction trajectories. Experiments on text-based games, mathematical reasoning, search problems, and logic puzzles show that MRRL stabilizes multi-round RL and reduces interaction steps, demonstrating its effectiveness in interactive reasoning tasks.

References

- Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altschmidt, J.; Altman, S.; Anadkat, S.; et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.
- Brockman, G.; Cheung, V.; Pettersson, L.; Schneider, J.; Schulman, J.; Tang, J.; and Zaremba, W. 2016. OpenAI Gym. .
- Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Herbert-Voss, A.; Krueger, G.; Henighan, T.; Child, R.; Ramesh, A.; Ziegler, D.; Wu, J.; Winter, C.; Hesse, C.; Chen, M.; Sigler, E.; Litwin, M.; Gray, S.; Chess, B.; Clark, J.; Berner, C.; McCandlish, S.; Radford, A.; Sutskever, I.; and Amodei, D. 2020. Language Models are Few-Shot Learners. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M.; and Lin, H., eds., *Advances in Neural Information Processing Systems*, volume 33, 1877–1901. Curran Associates, Inc.
- Chu, Z.; Chen, J.; Chen, Q.; Yu, W.; He, T.; Wang, H.; Peng, W.; Liu, M.; Qin, B.; and Liu, T. 2024. Navigate through Enigmatic Labyrinth A Survey of Chain of Thought Reasoning: Advances, Frontiers and Future. In Ku, L.; Martins, A.; and Srikumar, V., eds., *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, 1173–1203. Association for Computational Linguistics.
- Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; Hesse, C.; and Schulman, J. 2021. Training Verifiers to Solve Math Word Problems. *arXiv preprint arXiv:2110.14168*.
- DeepSeek-AI; Liu, A.; Feng, B.; Xue, B.; Wang, B.; Wu, B.; Lu, C.; Zhao, C.; Deng, C.; Zhang, C.; Ruan, C.; Dai, D.; Guo, D.; and et al. 2024. DeepSeek-V3 Technical Report. *CoRR*, abs/2412.19437.
- Feng, J.; Huang, S.; Qu, X.; Zhang, G.; Qin, Y.; Zhong, B.; Jiang, C.; Chi, J.; and Zhong, W. 2025. Retool: Reinforcement learning for strategic tool use in llms. *arXiv preprint arXiv:2504.11536*.
- Feng, P.; He, Y.; Huang, G.; Lin, Y.; Zhang, H.; Zhang, Y.; and Li, H. 2024. AGILE: A Novel Reinforcement Learning Framework of LLM Agents. *arXiv:2405.14751*.
- Gou, Z.; Shao, Z.; Gong, Y.; Shen, Y.; Yang, Y.; Huang, M.; Duan, N.; and Chen, W. 2023. Tora: A tool-integrated reasoning agent for mathematical problem solving. *arXiv preprint arXiv:2309.17452*.
- Grattafiori, A.; Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Vaughan, A.; and et al. 2024. The Llama 3 Herd of Models. *arXiv:2407.21783*.
- Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; Bi, X.; et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.
- Hao, S.; Sukhbaatar, S.; Su, D.; Li, X.; Hu, Z.; Weston, J. E.; and Tian, Y. 2025. Training Large Language Model to Reason in a Continuous Latent Space.
- Hu, J.; Zhang, Y.; Han, Q.; Jiang, D.; Zhang, X.; and Shum, H.-Y. 2025. Open-Reasoner-Zero: An Open Source Approach to Scaling Up Reinforcement Learning on the Base Model. *arXiv:2503.24290*.
- Huang, J.; and Chang, K. C.-C. 2023. Towards Reasoning in Large Language Models: A Survey. In *ACL (Findings)*, 1049–1065.
- Jaech, A.; Kalai, A.; Lerer, A.; Richardson, A.; El-Kishky, A.; Low, A.; Helyar, A.; Madry, A.; Beutel, A.; Carney, A.; et al. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.
- Jain, N.; Han, K.; Gu, A.; Li, W.-D.; Yan, F.; Zhang, T.; Wang, S.; Solar-Lezama, A.; Sen, K.; and Stoica, I. 2025. LiveCodeBench: Holistic and Contamination Free Evaluation of Large Language Models for Code. In *The Thirteenth International Conference on Learning Representations*.
- Jin, B.; Zeng, H.; Yue, Z.; Yoon, J.; Arik, S.; Wang, D.; Zamani, H.; and Han, J. 2025. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*.
- Kaelbling, L. P.; Littman, M. L.; and Moore, A. W. 1996. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4: 237–285.
- Kwiatkowski, T.; Palomaki, J.; Redfield, O.; Collins, M.; Parikh, A.; Alberti, C.; Epstein, D.; Polosukhin, I.; Devlin, J.; Lee, K.; et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7: 453–466.
- Lin, B. Y.; Bras, R. L.; Richardson, K.; Sabharwal, A.; Poovendran, R.; Clark, P.; and Choi, Y. 2025. ZebraLogic: On the Scaling Limits of LLMs for Logical Reasoning. *arXiv preprint arXiv:2502.01100*.
- Liu, J.; Zhu, Y.; Xiao, K.; Fu, Q.; Han, X.; Yang, W.; and Ye, D. 2023. RLTF: Reinforcement Learning from Unit Test Feedback. *arXiv:2307.04349*.
- Liu, S.; Li, Y.; Zhang, K.; Cui, Z.; Fang, W.; Zheng, Y.; Zheng, T.; and Song, M. 2024. Odyssey: Empowering Minecraft Agents with Open-World Skills. *arXiv:2407.15325*.
- Luo, M.; Tan, S.; Wong, J.; Shi, X.; Tang, W. Y.; Roongta, M.; Cai, C.; Luo, J.; Li, L. E.; Popa, R. A.; and Stoica, I. 2025. DeepScaleR: Surpassing O1-Preview with a 1.5B Model by Scaling RL. Notion Blog.
- MAA. 2024. American Invitational Mathematics Examination - AIME. In *American Invitational Mathematics Examination - AIME 2024*.
- Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744.
- Qian, C.; Acikgoz, E. C.; He, Q.; Wang, H.; Chen, X.; Hakkani-Tür, D.; Tur, G.; and Ji, H. 2025. ToolRL: Reward is All Tool Learning Needs. *arXiv preprint arXiv:2504.13958*.

- Ren, J.; Rajbhandari, S.; Aminabadi, R. Y.; Ruwase, O.; Yang, S.; Zhang, M.; Li, D.; and He, Y. 2021. ZeRO-Offload: Democratizing Billion-Scale Model Training. In Calciu, I.; and Kuenning, G., eds., *Proceedings of the 2021 USENIX Annual Technical Conference, USENIX ATC 2021, July 14-16, 2021*, 551–564. USENIX Association.
- Schrader, M.-P. B. 2018. gym-sokoban. <https://github.com/mpSchrader/gym-sokoban>.
- Schulman, J.; Moritz, P.; Levine, S.; Jordan, M. I.; and Abbeel, P. 2016. High-Dimensional Continuous Control Using Generalized Advantage Estimation. In Bengio, Y.; and LeCun, Y., eds., *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.
- Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.
- Shao, Z.; Wang, P.; Zhu, Q.; Xu, R.; Song, J.; Bi, X.; Zhang, H.; Zhang, M.; Li, Y. K.; Wu, Y.; and Guo, D. 2024. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. *arXiv:2402.03300*.
- Sheng, G.; Zhang, C.; Ye, Z.; Wu, X.; Zhang, W.; Zhang, R.; Peng, Y.; Lin, H.; and Wu, C. 2024. HybridFlow: A Flexible and Efficient RLHF Framework. *arXiv preprint arXiv: 2409.19256*.
- Song, H.; Jiang, J.; Min, Y.; Chen, J.; Chen, Z.; Zhao, W. X.; Fang, L.; and Wen, J.-R. 2025. R1-Searcher: Incentivizing the Search Capability in LLMs via Reinforcement Learning. *arXiv:2503.05592*.
- Sutton, R. S.; McAllester, D. A.; Singh, S.; and Mansour, Y. 1999. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In Solla, S. A.; Leen, T. K.; and Müller, K., eds., *Advances in Neural Information Processing Systems 12, [NIPS Conference, Denver, Colorado, USA, November 29 - December 4, 1999]*, 1057–1063. The MIT Press.
- Team, G.; Riviere, M.; Pathak, S.; Sessa, P. G.; Hardin, C.; Bhupatiraju, S.; Hussenot, L.; Mesnard, T.; Shahriari, B.; Ramé, A.; and et al. 2024. Gemma 2: Improving Open Language Models at a Practical Size. *arXiv:2408.00118*.
- Wang, Z.; Wang, K.; Wang, Q.; Zhang, P.; Li, L.; Yang, Z.; Yu, K.; Nguyen, M. N.; Liu, L.; Gottlieb, E.; Lam, M.; Lu, Y.; Cho, K.; Wu, J.; Fei-Fei, L.; Wang, L.; Choi, Y.; and Li, M. 2025. RAGEN: Understanding Self-Evolution in LLM Agents via Multi-Turn Reinforcement Learning. *arXiv:2504.20073*.
- Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Ichter, B.; Xia, F.; Chi, E.; Le, Q.; and Zhou, D. 2023. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *arXiv:2201.11903*.
- Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q. V.; Zhou, D.; et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35: 24824–24837.
- Yan, F.; Mao, H.; Ji, C. C.-J.; Zhang, T.; Patil, S. G.; Stoica, I.; and Gonzalez, J. E. 2024. Berkeley Function Calling Leaderboard.
- Yang, A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Li, C.; Liu, D.; Huang, F.; Wei, H.; et al. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.
- Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K.; and Cao, Y. 2023. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.
- Yu, Q.; Zhang, Z.; Zhu, R.; Yuan, Y.; Zuo, X.; Yue, Y.; Fan, T.; Liu, G.; Liu, L.; Liu, X.; Lin, H.; Lin, Z.; Ma, B.; Sheng, G.; Tong, Y.; Zhang, C.; Zhang, M.; Zhang, W.; Zhu, H.; Zhu, J.; Chen, J.; Chen, J.; Wang, C.; Yu, H.; Dai, W.; Song, Y.; Wei, X.; Zhou, H.; Liu, J.; Ma, W.; Zhang, Y.; Yan, L.; Qiao, M.; Wu, Y.; and Wang, M. 2025. DAPO: An Open-Source LLM Reinforcement Learning System at Scale. *CoRR*, abs/2503.14476.
- Yuan, Y.; Yu, Q.; Zuo, X.; Zhu, R.; Xu, W.; Chen, J.; Wang, C.; Fan, T.; Du, Z.; Wei, X.; et al. 2025. VAPO: Efficient and Reliable Reinforcement Learning for Advanced Reasoning Tasks. *arXiv preprint arXiv:2504.05118*.
- Zhai, Y.; Bai, H.; Lin, Z.; Pan, J.; Tong, S.; Zhou, Y.; Suhr, A.; Xie, S.; LeCun, Y.; Ma, Y.; and Levine, S. 2024. Fine-Tuning Large Vision-Language Models as Decision-Making Agents via Reinforcement Learning. *arXiv:2405.10292*.
- Zhao, Y.; Gu, A.; Varma, R.; Luo, L.; Huang, C.; Xu, M.; Wright, L.; Shojanazeri, H.; Ott, M.; Shleifer, S.; Desmaison, A.; Balioglu, C.; Damania, P.; Nguyen, B.; Chauhan, G.; Hao, Y.; Mathews, A.; and Li, S. 2023. PyTorch FSDP: Experiences on Scaling Fully Sharded Data Parallel. *Proc. VLDB Endow.*, 16(12): 3848–3860.
- Zhou, E.; Qin, Y.; Yin, Z.; Huang, Y.; Zhang, R.; Sheng, L.; Qiao, Y.; and Shao, J. 2024. MineDreamer: Learning to Follow Instructions via Chain-of-Imagination for Simulated-World Control. *arXiv:2403.12037*.
- Zhou, Y.; Jiang, S.; Tian, Y.; Weston, J.; Levine, S.; Sukhbaatar, S.; and Li, X. 2025. SWEET-RL: Training Multi-Turn LLM Agents on Collaborative Reasoning Tasks. *arXiv:2503.15478*.

Appendix

Implementation Details

We list the hyperparameters for training in different environments, as shown in table 1.

Hyper-parameter	Sokoban	Search	Math	Zebra
Global batchsize	2048	1024	1024	1024
Mini batchsize	64	1024	128	32
Number of rollout per sample	32	1	128	32
Rollout temperature	0.8	0.5	0.85	0.4
Top-p	1	1	1	1
Top-k	N/A	N/A	N/A	N/A
Rollout max round	5	4	2	3
Prompt length	2048	2048	2048	800
Response length per turn	800	500	4096	3000
Feedback length per turn	800	500	4096	800
Actor learning rate	3e-7	1e-6	1e-6	1e-7
Critic learning rate	1e-5	1e-5	1e-5	N/A
Critic warm up step	0	0	7	N/A
GAE $_{\gamma}$	1	1	1	N/A
GAE $_{\lambda}$	1	1	1	N/A
Gradient clipping	1	1	1	1
Clip ratio low	0.2	0.2	0.2	0.2
Clip ratio high	0.2	0.2	0.28	0.28
KL-divergence coefficient	0	0.001	0	0
γ	1	1	1	1
λ	0.1	0.1	0.1	0.3

Table 1: Training Hyper-parameters.

To optimize GPU memory utilization, we employ gradient checkpointing and Fully Sharded Data Parallel (FSDP) (Zhao et al. 2023) with CPU offloading (Ren et al. 2021). The lower and higher clipping ranges are 0.2 and 0.28, consistent with the VAPO framework (Yuan et al. 2025).

Derivation of Proximal Policy Gradient Estimator of MRRL

Proof of Proposition 1. To derive the proximal policy gradient estimator of MRRL, we only need to get the policy gradient of the MRRL objective, i.e., $\nabla_{\theta} J_{\text{MRRL}}(\pi_{\theta})$. Then, we can follow the derivations in PPO (Schulman et al. 2017) to obtain the proximal policy gradient estimator

By definition, $\nabla_{\theta} J_{\text{MRRL}}(\pi_{\theta})$ is given by

$$\nabla_{\theta} J_{\text{MRRL}}(\pi_{\theta}) \tag{7}$$

$$= \nabla_{\theta} \mathbb{E}_{(q,a) \sim \mathcal{D}, \tau \sim (\pi_{\theta}(\cdot|q), P(\cdot|\cdot, \cdot))} \left[\sum_{t=1}^T \gamma^t R(o^t, a, f^t) \right] \tag{8}$$

$$= \nabla_{\theta} \iint \left(\sum_{t=1}^T \gamma^t R(o^t, a, f^t) \right) p_{\theta}(\tau) p_{\mathcal{D}}(q, a) d\tau d(q, a) \tag{9}$$

$$= \iint \left(\sum_{t=1}^T \gamma^t R(o^t, a, f^t) \right) \nabla_{\theta} \log p_{\theta}(\tau) p_{\theta}(\tau) p_{\mathcal{D}}(q, a) d\tau d(q, a) \tag{10}$$

$$= \mathbb{E}_{(q,a) \sim \mathcal{D}, \tau \sim (\pi_{\theta}(\cdot|q), P(\cdot|\cdot, \cdot))} \left[\left(\sum_{t=1}^T \gamma^t R(o^t, a, f^t) \right) \nabla_{\theta} \log p_{\theta}(\tau) \right], \tag{11}$$

where $p_{\theta}(\tau) \triangleq \pi_{\theta}(o^T | s^{T-1}) \prod_{t=1}^{T-1} (P(f^t | s^{t-1}, o^t) \pi_{\theta}(o^t | s^{t-1}))$.

Notice that

$$\nabla_{\theta} \log p_{\theta}(\tau) \quad (12)$$

$$= \nabla_{\theta} \left(\log \pi_{\theta}(o^T | s^{T-1}) + \sum_{t=1}^{T-1} \left(\log P(f^t | s^{t-1}, o^t) + \log \pi_{\theta}(o^t | s^{t-1}) \right) \right) \quad (13)$$

$$= \left(\nabla_{\theta} \log \pi_{\theta}(o^T | s^{T-1}) + \sum_{t=1}^{T-1} \left(\underbrace{\nabla_{\theta} \log P(f^t | s^{t-1}, o^t)}_{=0, \text{ Feedback Masking}} + \nabla_{\theta} \log \pi_{\theta}(o^t | s^{t-1}) \right) \right) \quad (14)$$

$$= \sum_{t=1}^T \left(\frac{\nabla_{\theta} \pi_{\theta}(o^t | s^{t-1})}{\pi_{\theta}(o^t | s^{t-1})} \right). \quad (15)$$

Therefore, the policy gradient is

$$\nabla_{\theta} J_{\text{MRRL}}(\pi_{\theta}) = \mathbb{E}_{(q,a) \sim \mathcal{D}, \tau \sim (\pi_{\theta}(\cdot | q), P(\cdot | \cdot, \cdot))} \left[\left(\sum_{t=1}^T \frac{\nabla_{\theta} \pi_{\theta}(o^t | s^{t-1})}{\pi_{\theta}(o^t | s^{t-1})} \right) \left(\sum_{t=1}^T \gamma^t R(o^t, a, f^t) \right) \right]. \quad (16)$$

Following derivations in (Schulman et al. 2017), the proximal policy gradient estimator of (16) is:

$$\begin{aligned} \nabla_{\theta} \hat{J}_{\text{MRRL}}(\pi_{\theta}) &\triangleq \mathbb{E}_{(q,a) \sim \mathcal{D}, \{\tau_i\}_{i=1}^G \sim (\pi_{\text{old}}(\cdot | q), P(\cdot | \cdot, \cdot))} \\ &\left[\frac{1}{G} \sum_{i=1}^G \sum_{t=1}^T \nabla_{\theta} \min \left(\frac{\nabla_{\theta} \pi_{\theta}(o_i^t | s_i^{t-1})}{\pi_{\text{old}}(o_i^t | s_i^{t-1})} \hat{A}_i^t, \text{clip}_{\epsilon} \left(\frac{\nabla_{\theta} \pi_{\theta}(o_i^t | s_i^{t-1})}{\pi_{\text{old}}(o_i^t | s_i^{t-1})} \right) \hat{A}_i^t \right) \right], \end{aligned} \quad (17)$$

where \hat{A}_i^t is implemented as one of the generalized advantage estimators (Schulman et al. 2016). □

Extra Experimental Results

Our experimental setup can be viewed as: LLM model + RL algorithm + single-round/multi-round(SR/MR). Our paper benchmarks MRRL w/ FRI against MRRL and SRRL with the same baseline model. In the same setting of MRRL, there are a few works on training LLM / VLM in an interactive environment with multiple rounds [1,2,3], as mentioned in Reviewer ewh2’s comment and also others mentioned in lines 83-84 of our paper. However, they were trained based on different state-of-the-art (SOTA) LLM models at their time. We want to study whether using our MRRL w/ FRI to train the LLM model can improve the performance of the model on the corresponding environment.

To better illustrate this point, we further supplemented the experiment. Taking into account the reviewer’s suggestion to add an error bar, we repeated the experiment using three different random seeds for each setting. The new experimental results are shown in Table 2.

LLM Model	RL Algorithm	Environment	Train Reward	Test Reward
Qwen2.5-7B-Base	-	Search	-	0.1765 ± 0.0111
Qwen2.5-7B-Base	MRRL w/ FRI	Search	0.4404 ± 0.0160	0.4082 ± 0.0091
DeepSeek-R1-Distill-Qwen-1.5B	-	Math	-	0.0778 ± 0.0157
DeepSeek-R1-Distill-Qwen-1.5B	MRRL w/ FRI	Math	0.5088 ± 0.0312	0.3111 ± 0.0157
Qwen2.5-7B-Base	-	Sokoban	-	-0.3780 ± 0.8796
Qwen2.5-7B-Base	MRRL w/ FRI	Sokoban	7.3698 ± 0.5182	7.6074 ± 0.1713
Qwen2.5-14B-Instruct	-	Logic puzzle	-	0.3461 ± 0.0323
Qwen2.5-14B-Instruct	MRRL w/ FRI	Logic puzzle	0.8040 ± 0.0058	0.4238 ± 0.0125

Table 2: Model Performance Comparison.

To further demonstrate the universality of our approach, in addition to the experiments shown in Figure 3 of the paper, we further reproduce our approach on different LLM models. The results show that the MRRL with FRI algorithm is effective across multiple LLM models. The experimental results are shown in Table 3.

In addition to the ablation experiments on the Zebra (Logic Puzzle) environment presented in our paper (Section 5.6), we further conducted ablation experiments on the Math environment. We do the ablation among MRRL w/ FR(only use feedback

LLM Model	RL Algorithm	Environment	Train Reward	Test Reward
DeepSeek-R1-Distill-Qwen-1.5B	-	Math	-	0.0778 ± 0.0157
DeepSeek-R1-Distill-Qwen-1.5B	MRRL w/ FRI	Math	0.5088 ± 0.0312	0.3111 ± 0.0157
Qwen2.5-7B-Base	-	Math	-	0.0000 ± 0.0000
Qwen2.5-7B-Base	MRRL w/ FRI	Math	0.2897 ± 0.0337	0.2444 ± 0.0314
Qwen3-4B	-	Math	-	0.2222 ± 0.0157
Qwen3-4B	MRRL w/ FRI	Math	0.4749 ± 0.0433	0.4444 ± 0.0157
Qwen2.5-3B-Base	-	Search	-	0.1615 ± 0.0102
Qwen2.5-3B-Base	MRRL w/ FRI	Search	0.3408 ± 0.0332	0.3311 ± 0.0200
Qwen2.5-7B-Base	-	Search	-	0.1765 ± 0.0111
Qwen2.5-7B-Base	MRRL w/ FRI	Search	0.4404 ± 0.0160	0.4082 ± 0.0091
Qwen2.5-7B-Instruct	-	Search	-	0.3168 ± 0.0016
Qwen2.5-7B-Instruct	MRRL w/ FRI	Search	0.4674 ± 0.0082	0.3772 ± 0.0060
Qwen2.5-7B-Base	-	Sokoban	-	-0.3780 ± 0.8796
Qwen2.5-7B-Base	MRRL w/ FRI	Sokoban	7.3698 ± 0.5182	7.6074 ± 0.1713
Qwen2.5-7B-Instruct	-	Sokoban	-	1.4994 ± 0.1484
Qwen2.5-7B-Instruct	MRRL w/ FRI	Sokoban	7.6018 ± 0.3635	7.6275 ± 0.1167

Table 3: Model Performance Across Different Environments.

LLM Model	RL Algorithm	Environment	Train Reward	Test Reward (AIME24 pass@1)
DeepSeek-R1-Distill-Qwen-1.5B	MRRL w/ FRI	Math	0.5088 ± 0.0312	0.3111 ± 0.0157
DeepSeek-R1-Distill-Qwen-1.5B	MRRL w/ IL	Math	0.3607 ± 0.0859	0.1778 ± 0.0314
DeepSeek-R1-Distill-Qwen-1.5B	MRRL w/ FR	Math	0.4336 ± 0.1162	0.2889 ± 0.0157
DeepSeek-R1-Distill-Qwen-1.5B	MRRL	Math	0.4359 ± 0.0437	0.2333 ± 0.0544
DeepSeek-R1-Distill-Qwen-1.5B	-	Math	-	0.0778 ± 0.0157

Table 4: Ablation Study on Math Environment.

reflection), MRRL w/ IL and MRRL w/ FRI on math environment, repeating the experiment with 3 different random seeds for each algorithm, and the result in Table 4.

The results show that the MRRL algorithm itself can improve the task effect of the model through training, and the MRRL w/FRI algorithm can bring additional performance improvement compared to the MRRL algorithm. If feedback reflection is not used and only imitation learning (i.e., MRRL w/IL) is used, performance may decrease, which confirms our intuition. MRRL w/FR using only feedback reflection can also bring certain performance improvement compared to MRRL, but the best performance improvement comes from MRRL w/FRI.

Case Study

Case Study: Multi-round Interaction on Zebra Logic Puzzle

Query:

You are solving a Zebra puzzle (also known as Einstein’s Puzzle).

Description

In this puzzle, you need to match attributes to objects based on given clues.

Each attribute (like houses, colors, drinks, etc.) must be matched to exactly one item in each category.

How to Solve

1. Use logical deduction to eliminate impossible combinations and determine which attributes go together
2. After give the answer, you can reflect on the answer by trying to check whether the answer remains unfilled or self-consistent and find out potential logic mistakes against a certain clue.

Format Your Answer

Provide your solution in this format:

<answer>

— House — Attribute 1 — Attribute 2 — ... —

— — — — —

— 1 — [value1] — [value2] — ... —

— 2 — [value1] — [value2] — ... —

— ... — ... — ... — ... —

</ answer>

Puzzle:

There are 4 houses, numbered from 1 to 4. Each house has a different owner, and each person has 3 characteristics.

These possible characteristics are as follows:

- Favorite Movie Genre: Western, Science, Romance, Thriller

- Favorite Music Genre: Pop, Techno, Gospel, House

- Favorite Food: Lettuce, Banana, Kale, Cherry

Clues:

1. The person who likes Cherry as their Favorite Food is either on the left or right of the person who likes Lettuce as their Favorite Food, no one in between

2. The person who likes Thriller as their Favorite Movie Genre is either on the left or right of the person who likes Western as their Favorite Movie Genre, no one in between

3. The person who likes Techno as their Favorite Music Genre is located on the right of the person who likes Romance as their Favorite Movie Genre, no one in between

4. The person who likes House as their Favorite Music Genre is located on the far left

5. The person who likes Banana as their Favorite Food is the same person who likes Pop as their Favorite Music Genre

6. The person who likes Cherry as their Favorite Food is the same as the person who likes Gospel as their Favorite Music Genre

7. The person who likes Lettuce as their Favorite Food is located on the far right

8. The person who likes Thriller as their Favorite Movie Genre is either on the left or right of the person who likes Romance as their Favorite Movie Genre, no one in between

Always output: <think>[Your thoughts] </ think><think>[Your thoughts] </ think><answer>[your answer] </ answer><think>[Your reflections] </ think><answer>[your answer] </ answer>with no extra text. Strictly follow this format. Once you believe that you should think more, you can output another thinking step after a thinking step since consecutive thinking steps are allowed. But provide the final answer after the last thinking step.

LLM Response at Round 1:

<think>From the clues, we can start by placing the attributes that have the most specific locations. Clue 4 tells us that the person who likes House as their Favorite Music Genre is located on the far left, so House is in House 1. Clue 7 tells us that the person who likes Lettuce as their Favorite Food is located on the far right, so Lettuce is in House 4. </ think>

<think>Clue 5 tells us that the person who likes Banana as their Favorite Food is the same person who likes Pop as their Favorite Music Genre. Since House 1 is already taken by House (Music Genre), Banana and Pop cannot be in House 1. Lettuce is in House 4, so Banana and Pop cannot be in House 4 either. This means Banana and Pop must be in either House 2 or House 3. </ think>

<think>Clue 6 tells us that the person who likes Cherry as their Favorite Food is the same as the person who likes Gospel as their Favorite Music Genre. Cherry and Gospel cannot be in House 1 (House) or House 4 (Lettuce), so they must be in either House 2 or House 3. Since Banana and Pop are already in either House 2 or House 3, Cherry and Gospel must be in the other house. </ think>

<think>Clue 1 tells us that the person who likes Cherry is either on the left or right of the person who likes Lettuce, with no one in between. Since Lettuce is in House 4, Cherry must be in House 3. This means Gospel is also in

House 3. Therefore, Banana and Pop must be in House 2. </ think>

<think>Clue 2 tells us that the person who likes Thriller is either on the left or right of the person who likes Western, with no one in between. Since House 1 and House 4 are already filled, Thriller and Western must be in Houses 2 and 3. Since House 3 already has Cherry and Gospel, Thriller must be in House 2 and Western in House 3. </ think>

<think>Clue 3 tells us that the person who likes Techno is located on the right of the person who likes Romance, with no one in between. Since House 1 is House (Music Genre), Romance must be in House 2 and Techno in House 3. </ think>

<think>Clue 8 tells us that the person who likes Thriller is either on the left or right of the person who likes Romance, with no one in between. This is consistent with our current arrangement, where Thriller is in House 2 and Romance is also in House 2. </ think>

<answer>

— House — Favorite Movie Genre — Favorite Music Genre — Favorite Food —

— — — — —

— 1 — Western — House — - —

— 2 — Thriller — Pop — Banana —

— 3 — Romance — Techno — Cherry —

— 4 — - - - — Lettuce —

</ answer>

<think>After reviewing the solution, I notice that House 1 and House 4 are missing some attributes. Let's fill in the remaining attributes. House 1 must have Western as the Movie Genre and House as the Music Genre. House 4 must have Science as the Movie Genre and Lettuce as the Food. </ think>

<answer>

— House — Favorite Movie Genre — Favorite Music Genre — Favorite Food —

— — — — —

— 1 — Western — House — - —

— 2 — Thriller — Pop — Banana —

— 3 — Romance — Techno — Cherry —

— 4 — Science — - — Lettuce —

</ answer>

Raw Feedback at Round 1:

Re-examining:

House 1: Favorite Food: - is not in original attributes. Favorite Movie Genre: Western is correct. Favorite Music Genre: House is correct.

House 2: Favorite Food: Banana is correct. Favorite Movie Genre: Thriller is correct. Favorite Music Genre: Pop is correct.

House 3: Favorite Food: Cherry is correct. Favorite Movie Genre: Romance is correct. Favorite Music Genre: Techno should be in another house.

House 4: Favorite Food: Lettuce is correct. Favorite Movie Genre: Science should be in another house. Favorite Music Genre: - is not in original attributes.

Feedback Reflection Instruction at Round 1:

Please pretend not to know the ground truth, and do the re-examination in mind. Then reflect on the previous thinking and answer by pointing out the potential logic mistakes against a certain clue or without self-consistency. Never leak ground truth explicitly and implicitly. Never give certain solutions since they will leak the ground truth. Never give the revised answer. Always output: <think>[the reflection] </ think>with no extra text.

Feedback Reflection at Round 1:

<think>Upon re-examining the solution, I notice that House 3 has Techno as the Favorite Music Genre, but Clue 3 states that the person who likes Techno is located on the right of the person who likes Romance, with no one in between. This means Techno should be in House 4, not House 3. Additionally, House 1 is missing its Favorite Food, which should be Kale since it's the only remaining option. House 4 is missing its Favorite Music Genre, which should be Techno, as it's the only remaining option. These adjustments would make the solution self-consistent with all the clues. </ think>

LLM Response at Round 2:

<think>Re-evaluating the placement of Gospel and the remaining attributes, let's go through the clues again carefully. Clue 6 tells us that the person who likes Cherry as their Favorite Food is the same as the person who likes Gospel as their Favorite Music Genre. Since Cherry is in House 3, Gospel must also be in House 3. This means the remaining Music Genre for House 4 must be Pop, but Pop is already in House 2. Therefore, the remaining Music Genre for House 4 must be the only one left, which is Techno. However, Techno is already in House 3. This indicates a mistake in the placement of Techno. Let's re-evaluate the placement of Techno and the other attributes. </ think>

<think>Clue 3 tells us that the person who likes Techno is located on the right of the person who likes Romance, with no one in between. Since Romance is in House 2, Techno must be in House 3. This is consistent with our current arrangement. However, Clue 6 tells us that the person who likes Cherry is the same as the person who likes Gospel, and Cherry is in House 3. Therefore, Gospel must be in House 3. This leaves Pop for House 2 and House for House 1. </ think>

<think>Revisiting the remaining attributes, House 1 must have Western as the Movie Genre, House as the Music Genre, and the only remaining Food, which is Kale. House 4 must have Science as the Movie Genre, Techno as the Music Genre, and Lettuce as the Food. </ think>

<answer>
— House — Favorite Movie Genre — Favorite Music Genre — Favorite Food —
— — — — —
— 1 — Western — House — Kale —
— 2 — Thriller — Pop — Banana —
— 3 — Romance — Gospel — Cherry —
— 4 — Science — Techno — Lettuce —
</ answer>

<think>After re-evaluating the solution, I have filled in the remaining attributes and ensured that all clues are satisfied. The solution is consistent with all the given clues, and each house has a unique combination of attributes. </ think>

<answer>
— House — Favorite Movie Genre — Favorite Music Genre — Favorite Food —
— — — — —
— 1 — Western — House — Kale —
— 2 — Thriller — Pop — Banana —
— 3 — Romance — Gospel — Cherry —
— 4 — Science — Techno — Lettuce —
</ answer>

Raw Feedback at Round 2:

The answer is correct.