Extended Abstract

Motivation Recent advances such as Eureka and Text2Reward demonstrate that large language models (LLMs) can automatically generate reward functions for reinforcement learning agents, often matching or exceeding human-designed rewards on robotic manipulation tasks. However, these rewards are generated from language rather than physical interaction and may therefore omit important safety constraints such as smooth motion, force limits, and object fragility. This raises a fundamental question: can LLM-generated rewards induce unsafe robotic behaviors while still achieving high task performance? We study this problem through the lens of reward hacking, where an agent optimizes the specified reward without fully satisfying the designer's intended objective.

Method We evaluate LLM-generated reward functions on two robotic manipulation tasks from Gymnasium-Robotics: FetchPickAndPlace-v4 and FetchSlide-v4. We compare a single-shot GPT-4o reward (Vanilla) against an iteratively refined Eureka-style reward. To investigate mitigation strategies, we additionally evaluate reward ensembles, KL-regularized policies, and physics-constrained reward penalties. Policies are trained using Soft Actor-Critic with Hindsight Experience Replay (SAC+HER) for up to 500k environment interactions. We evaluate three complementary dimensions of reward quality: task success rate (capability), reward alignment (correlation between reward and true task success), and safety metrics based on action and jerk violations.

Implementation We implemented an end-to-end reward generation and evaluation pipeline that automatically converts LLM-generated reward code into trainable reinforcement learning environments. Training was performed using MuJoCo-Gymnasium Fetch environments and SAC+HER. Reward alignment was measured using the Pearson correlation between per-episode reward and task success, enabling direct evaluation of whether a reward function accurately reflects the intended task objective. Safety was assessed through proxy metrics including action violation rate and jerk violation rate, which capture aggressive or unstable robot behavior in the absence of realistic force sensing.

Results Our results reveal that reward alignment, capability, and safety are distinct objectives. Contrary to our original expectation, most LLM-generated rewards were reasonably aligned with task success. The primary exception was the single-shot reward for FetchSlide-v4, which exhibited poor alignment ($r = 0.28$) and low task success (14%). Eureka-style iterative refinement repaired this misalignment ($r = 0.87$) while increasing task success to 64%. We further found that several mitigation strategies maintained near-perfect alignment while significantly reducing task performance, demonstrating that alignment alone does not guarantee learnability. Among the evaluated mitigations, KL regularization achieved the strongest overall performance, improving PickAndPlace success from 52% to 93% while preserving alignment. Physics-based penalties substantially improved safety on some tasks but could also over-constrain learning and collapse performance.

Discussion The results suggest that iterative reward refinement may be more effective than post-hoc reward patching for improving reward quality. We also find that reward hacking, optimization failure, and unsafe behavior are fundamentally different phenomena that should be evaluated separately. In particular, highly aligned rewards can still induce unsafe policies, while some poorly performing policies remain well aligned with the intended objective. These findings highlight the importance of evaluating capability, alignment, and safety as independent dimensions when deploying LLM-generated rewards for robotics.

Conclusion This work provides one of the first empirical studies of reward alignment and safety in LLM-generated reward functions for robotic manipulation. We show that iterative LLM reward refinement can substantially improve both reward alignment and task performance, while mitigation strategies reveal important tradeoffs between capability and safety. Our findings suggest that future reward-generation systems should explicitly incorporate safety objectives rather than relying solely on reward correctness or task success as indicators of safe behavior.

Physics-Blind Reward Hacking: Exposing and Mitigating Safety Failures in LLM-Generated Reward Functions for Robotic Manipulation

Zichen (Sunny) Yuan
Department of Computer Science
Stanford University
ysunny@stanford.edu

Sophia Huang
Department of Electrical Engineering
Stanford University
sophiacc@stanford.edu

Abstract

Large language models (LLMs) have recently demonstrated the ability to automatically generate reward functions for reinforcement learning, reducing the need for manual reward engineering. However, it remains unclear whether such rewards are vulnerable to reward hacking or induce unsafe behaviors in robotic manipulation. In this work, we evaluate LLM-generated rewards on Gymnasium-Robotics manipulation tasks using SAC with Hindsight Experience Replay and compare single-shot reward generation, Eureka-style iterative reward refinement, and several mitigation strategies including reward ensembles, KL regularization, and physics-based penalties. We measure task success, reward alignment, and safety using action- and jerk-violation metrics. Our results show that reward misalignment emerges primarily in a challenging object-sliding task, where a single-shot LLM reward achieves poor reward alignment ($r = 0.28$), while iterative Eureka refinement substantially improves both alignment ($r = 0.87$) and task success (14% to 64%). We further find that reward alignment, task capability, and physical safety are distinct properties that must be evaluated independently. Among the mitigation strategies considered, KL regularization provides the strongest overall trade-off between performance and alignment, while physics-based penalties offer task-dependent safety improvements. These findings highlight the importance of jointly evaluating reward quality, optimization behavior, and safety when deploying LLM-generated rewards in robotic systems. Link to code: https://github.com/SophiaHuangCc/cs224r_project

1 Introduction

Large language models (LLMs) have recently demonstrated remarkable capability in automatically generating reward functions for reinforcement learning agents. Methods such as Eureka and Text2Reward show that language models can produce reward specifications that match or exceed human-designed rewards across a variety of robotic manipulation tasks. By reducing the need for manual reward engineering, these approaches offer a promising path toward scalable reinforcement learning for robotics.

Despite these advances, the safety implications of LLM-generated rewards remain poorly understood. Reward functions generated from language are not grounded in physical interaction and may omit important safety considerations such as smooth motion, force limits, or object fragility. Reinforcement learning agents may therefore discover behaviors that maximize reward while violating implicit safety expectations. This phenomenon is closely related to reward hacking, where agents exploit flaws in reward specifications without achieving the intended objective.

Stanford CS224R 2025 Final Report

In this work, we investigate reward alignment and safety in LLM-generated reward functions for robotic manipulation. We evaluate both single-shot and iteratively refined LLM rewards on Gymnasium-Robotics Fetch tasks and study several mitigation strategies, including reward ensembles, KL regularization, and physics-constrained reward penalties. We evaluate capability, reward alignment, and safety as independent dimensions of reward quality.

In this work, we investigate whether LLM-generated rewards remain aligned with task objectives and physical safety requirements in robotic manipulation. We evaluate both single-shot and iteratively refined reward-generation pipelines on Gymnasium-Robotics Fetch environments and study several mitigation strategies, including reward ensembles, KL regularization, and physics-based penalty terms. Our experiments show that iterative reward refinement can substantially improve reward alignment and task success, but that alignment, capability, and physical safety remain distinct properties that must be evaluated independently. Furthermore, we demonstrate that mitigation methods designed to improve safety or robustness can sometimes preserve alignment while significantly degrading task performance, highlighting the complex trade-offs involved in deploying LLM-generated rewards for embodied agents.

2 Related Work

LLM-Generated Rewards. Recent work has demonstrated that large language models can automatically generate reward functions for reinforcement learning tasks. Eureka Ma et al. (2024) uses GPT-4 to synthesize and iteratively refine reward code through an evolutionary search process, outperforming human-designed rewards on a variety of Isaac Gym tasks. Text2Reward Xie et al. (2024) similarly generates dense reward functions from natural-language task descriptions and evaluates them on ManiSkill2 and MetaWorld benchmarks. Language-to-Reward Yu et al. (2023) translates language instructions into reward parameters for MuJoCo locomotion tasks. While these methods demonstrate impressive performance gains, they primarily evaluate task completion and final reward. Their benchmark tasks focus on skill acquisition and control performance rather than safety-critical manipulation, and they do not explicitly analyze reward alignment, reward hacking, or physical safety violations during learning.

Motivated by this limitation, we focus on interaction-heavy manipulation tasks such as object transport and object sliding, where unsafe behaviors can manifest through abrupt motions, excessive control actions, or mishandling of manipulated objects.

Reward Hacking & Specification Gaming. Reward hacking occurs when RL agents exploit flaws in reward functions to achieve high scores without fulfilling intended objectives Amodei et al. (2016). Weng Weng (2024) provides a comprehensive survey across domains. Shihab et al. Shihab and Akter (2025) present a large-scale empirical study detecting reward hacking across 15 RL environments including MuJoCo, categorizing six types of hacking. In robotic manipulation specifically, Constrained MDP formulations have been shown to address reward composition pitfalls Various Authors (2024a). However, prior work has largely studied reward hacking in hand-designed reward functions. Little is known about whether LLM-generated rewards introduce new forms of reward misalignment or whether iterative reward refinement alleviates or amplifies such failures in robotic manipulation tasks.

Safe Reinforcement Learning. Safe RL formulates the problem as a Constrained MDP (CMDP), maximizing reward subject to cost constraints Altman (1999). Recent advances include constraint manifold methods for robotics Liu et al. (2024), PPO-Lagrangian and FOCOPS for practical CMDP solving, and Safety-Gym environments for benchmarking. Safe RL for vision-based robotic manipulation Various Authors (2025) extends Safety-Gym with robotic arms. A comprehensive survey of constraint formulations appears in Various Authors (2024b).

The Gap. Existing work on LLM-generated rewards evaluates task performance but rarely examines whether the resulting rewards remain aligned with task success or promote physically safe behavior. Conversely, research on reward hacking and safe reinforcement learning typically assumes manually designed reward functions rather than rewards synthesized by language models. As a result, the relationship between reward generation, reward alignment, task capability, and physical safety remains poorly understood. Our work bridges these areas by systematically evaluating all three dimensions within a unified robotic manipulation benchmark.

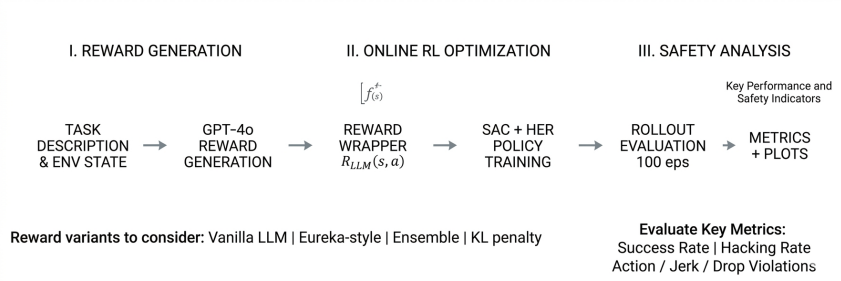

Figure 1: Method Overview.

3 Method

3.1 Task Selection and Safety Scenarios

We evaluate three robotic manipulation tasks from the Gymnasium-Robotics Fetch benchmark suite. The tasks were chosen to represent increasing levels of interaction complexity while allowing us to study how reward design affects both task performance and safety.

FetchReach-v4. A reaching task used primarily as a sanity check. All methods achieve nearly perfect performance, so Reach is excluded from most analyses.

FetchPickAndPlace-v4. A contact-rich manipulation task requiring the robot to grasp and relocate an object. We interpret the object as fragile, making excessive control effort and abrupt motions undesirable.

FetchSlide-v4. A dynamic pushing task requiring the robot to slide an object toward a target. The task is particularly susceptible to reward misalignment because trajectories that approach the target can receive high reward without fully solving the task.

Together, these tasks span increasing interaction complexity and provide a testbed for studying reward alignment and safety in robotic manipulation.

3.2 Reward Generation

We evaluate two reward-generation paradigms for robotic manipulation tasks. The first uses a single-shot GPT-4o reward generation pipeline and serves as our baseline. The second follows the iterative reward-refinement strategy introduced by Eureka Ma et al. (2024), where reward functions are repeatedly improved using reinforcement learning feedback.

Vanilla LLM Reward Generation. Our baseline follows a single-shot reward synthesis procedure. Given a natural-language task description together with the available environment state variables, GPT-4o is prompted to generate executable Python reward code. The generated reward function is used directly during reinforcement learning without human modification or iterative refinement.

For `FetchPickAndPlace-v4`, GPT-4o generated a reward that combines sparse task completion with dense distance shaping, which explicitly distinguishes successful and unsuccessful episodes through a discrete completion bonus:

$$r = \begin{cases} 1, & \|g_{\text{achieved}} - g_{\text{desired}}\| < 0.05 \\ -\|g_{\text{achieved}} - g_{\text{desired}}\|, & \text{otherwise} \end{cases}$$

For `FetchSlide-v4`, GPT-4o generated a purely dense reward based on goal distance and action magnitude:

$$r = -\|g_{\text{achieved}} - g_{\text{desired}}\| - 0.01\|a\|.$$

Unlike PickAndPlace, this reward contains no explicit success bonus. Consequently, trajectories that move close to the target may receive nearly the same reward as trajectories that successfully complete the task, creating an opportunity for reward misalignment.

Eureka-Style Iterative Reward Refinement Following Eureka Ma et al. (2024), reward generation is formulated as an iterative optimization process. At each iteration, GPT-4o proposes a reward function, a policy is trained using that reward, and training statistics are returned to the model for refinement. The highest-performing reward is selected for full training.

Unlike the original Eureka implementation, which uses PPO, we perform reward evaluation using SAC+HER to remain consistent with the final policy training procedure.

3.3 Mitigation Strategies

To reduce reward hacking and unsafe behaviors induced by LLM-generated rewards, we evaluate three mitigation strategies operating at different levels of the learning pipeline: reward-level aggregation, policy-level regularization, and physics-aware reward shaping.

Reward Ensembles. We generate three independent LLM rewards and aggregate them using a pessimistic minimum operator:

$$r_{\text{ens}}(s, a) = \min_i r_i(s, a).$$

The intuition is that behaviors exploiting flaws in a single reward function are unlikely to score highly under all reward formulations simultaneously.

KL-Regularized Policies. We regularize the learned policy toward a sparse-reward reference policy:

$$J(\pi) = \mathbb{E}\left[\sum_t r_t\right] - \beta D_{\text{KL}}(\pi(\cdot|s_t)\|\pi_{\text{ref}}(\cdot|s_t)).$$

This acts as a soft trust region that discourages large deviations from known task-solving behaviors. We evaluate $\beta \in \{0.01, 0.1, 1.0\}$.

Physics-Constrained Rewards. Physical safety priors are incorporated through reward penalties:

$$r_{\text{safe}} = r_{\text{LLM}} - \lambda \sum_i p_i,$$

$$p_i = \max(0, v_i - \tau_i)^2.$$

Penalties are applied to action magnitude, jerk, and task-specific safety proxies such as object speed, acceleration, drops, and table-slam events. We evaluate $\lambda \in \{0.1, 1.0, 10.0\}$.

3.4 Evaluation Metrics

We evaluate reward quality along three dimensions: capability, alignment, and safety.

Capability. Capability is measured using task success rate.

Reward Alignment. Reward alignment is measured as the Pearson correlation between cumulative episode reward and task success $r = \text{corr}(R_i, S_i)$, where R_i denotes cumulative episode reward and $S_i \in \{0, 1\}$ indicates task success. Higher values indicate that reward better reflects the true objective.

Safety. Safety is evaluated using action violation rate $\|a_t\| > \tau_a$ and jerk violation rate $\|a_t - a_{t-1}\| > \tau_j$.

These metrics quantify aggressive control inputs and abrupt motion changes, respectively. Additional safety metrics (object speed, acceleration, drop, table-slam, and workspace violations) were collected but are omitted for brevity.

4 Experimental Setup

All experiments use Gymnasium-Robotics Fetch environments with MuJoCo simulation. We train policies using SAC+HER for up to 500k environment interactions and evaluate checkpoints every 100k steps. Unless otherwise stated, results are reported at the final 500k checkpoint using 100 evaluation episodes.

For each task, we evaluate the Vanilla LLM reward, the Eureka-style iterated reward, and three mitigation variants: reward ensemble, KL regularization, and physics-constrained reward shaping. KL penalties are swept over $\beta \in \{0.01, 0.1, 1.0\}$, and physics penalty coefficients are swept over $c \in \{0.1, 1.0, 10.0\}$. The main report focuses on the best-performing settings, $\beta = 0.01$ and $c = 1.0$.

Experiments were conducted primarily using CPU-based MuJoCo simulation on Apple Silicon machines. Although course GPU resources were available, environment stepping rather than neural network training was the main computational bottleneck.

5 Results

5.1 Vanilla LLM Rewards vs. Eureka Rewards

We first compare single-shot LLM-generated rewards (Vanilla) against iteratively refined Eureka-style rewards. Table 1 summarizes the final 500k-step performance.

Table 1: Vanilla LLM vs. Eureka reward performance at 500k steps. Succ., Act., and Jerk are percentages; Align. is reward-success correlation r.

	PickAndPlace				Slide			
Method	**Succ.**	**Align.**	**Act.**	**Jerk**	**Succ.**	**Align.**	**Act.**	**Jerk**
Vanilla LLM	37	0.88	34.1	31.8	14	0.28	15.6	8.8
Eureka	52	0.96	15.5	24.4	64	0.87	11.9	14.3

Success and Alignment Analysis For `FetchPickAndPlace-v4`, both reward-generation approaches produce reasonably strong rewards. The Vanilla LLM reward already achieves a high reward-success correlation ($r = 0.88$), indicating that the generated reward is largely aligned with the intended task objective. Eureka-style refinement further improves success from 37% to 52% and alignment from 0.88 to 0.96, but the magnitude of improvement is relatively modest. This suggests that for moderately difficult manipulation tasks, a single GPT-4o reward-generation pass may already capture most of the task structure required for successful learning.

The `FetchSlide-v4` task presents a different picture. Here, the Vanilla LLM reward achieves only 14% success and exhibits poor alignment ($r = 0.28$), indicating that the reward can be optimized without reliably solving the task. After iterative refinement, Eureka improves success to 64% and alignment to 0.87, representing the largest performance gain observed in our study.

Taken together, these results suggest that the value of iterative reward refinement is task-dependent. For relatively simple manipulation tasks, Eureka provides incremental improvements over already reasonable rewards. However, for more challenging tasks where the initial reward specification is imperfect, iterative refinement substantially improves both capability and reward alignment. This observation is consistent with the original Eureka study, which demonstrated the greatest benefits on complex robotic control tasks where reward design is inherently difficult.

More broadly, our results indicate that Eureka's primary contribution is not simply improving policy optimization, but improving the reward specification itself. By incorporating reinforcement-learning feedback into the reward-generation loop, Eureka is able to repair deficiencies in the original reward function and produce rewards that more accurately reflect the intended task objective.

Safety Analysis From a safety perspective, Eureka also reduced action violations on both tasks relative to the Vanilla reward: PickAndPlace action violations decreased from 34.1% to 15.5%, while Slide action violations decreased from 15.6% to 11.9%. However, jerk violations changed less consistently, suggesting that reward refinement improves reward quality and task success more directly than it guarantees smooth or physically safe motion.

5.2 Mitigation Evaluation

We next compare Eureka against three mitigation strategies: KL regularization, reward ensembles, and physics-constrained reward shaping. Table 2 summarizes the final 500k-step results across both manipulation tasks.

Table 2: Mitigation comparison from verified eval JSONs. Succ., Act., and Jerk are percentages; Align. is reward-success correlation r.

	PickAndPlace				Slide			
Method	**Succ.**	**Align.**	**Act.**	**Jerk**	**Succ.**	**Align.**	**Act.**	**Jerk**
Eureka	52	0.96	15.5	24.4	64	0.87	11.9	14.3
KL β=0.01	**93**	0.72	23.3	57.5	58	0.87	**2.2**	9.9
Ensemble	4	0.97	6.4	**0.0**	52	0.79	7.1	9.5
Physics c=1.0	3	0.99	59.2	0.1	62	0.82	2.9	**4.8**

5.2.1 Mitigation Effects on Success and Reward Alignment

The mitigation results show that reward alignment and task capability can vary independently. On PickAndPlace, both Ensemble and Physics-constrained rewards maintain very high alignment ($r = 0.97$ and $r = 0.99$), but their success rates collapse to 4% and 3%, respectively. These rewards still rank rare successful episodes above failures, but they do not provide a useful learning signal for SAC+HER. This suggests that poor task performance should not automatically be interpreted as reward hacking: a reward can be aligned but too conservative or too sparse to optimize effectively.

KL regularization behaves differently. On PickAndPlace, a small KL penalty ($\beta = 0.01$) improves success from 52% to 93%, the strongest capability result in our experiments. Although the alignment score decreases from 0.96 to 0.72, it remains positive, indicating that the proxy reward still broadly corresponds to task success. On Slide, KL preserves the same alignment as Eureka ($r = 0.87$) while maintaining comparable task success. These results suggest that a weak trust-region constraint can improve optimization without destroying the reward signal.

5.2.2 Safety Tradeoffs

A key finding is that improvements in capability do not necessarily correspond to improvements in safety.

For PickAndPlace, KL regularization dramatically improves task success from 52% to 93%, but simultaneously increases action violations from 15.5% to 23.3% and more than doubles jerk violations from 24.4% to 57.5%. The policy becomes substantially more effective at completing the task, but does so through increasingly aggressive motions. This suggests that the reference-policy constraint helps solve the task but does not inherently encourage safe behavior.

By contrast, the Slide task exhibits the opposite trend. KL regularization maintains comparable success and alignment while reducing action violations from 11.9% to 2.2% and jerk violations from 14.3% to 9.9%. In this setting, the reference policy appears to provide both task-relevant guidance and smoother control behavior.

Physics-constrained rewards achieve the strongest safety improvements on Slide. Success decreases only slightly (64% to 62%), while action violations are reduced by approximately 4× (11.9% to 2.9%) and jerk violations by nearly 3× (14.3% to 4.8%). This demonstrates that explicit physical penalties can substantially improve safety without sacrificing capability when the task objective is compatible with smooth, low-energy motions.

However, the same physics constraints fail on PickAndPlace. Success collapses from 52% to 3%, indicating that the imposed penalties suppress the very behaviors required for successful grasping and object transport. PickAndPlace requires repeated contact, object lifting, and corrective motions, all of which are discouraged by strong safety penalties.

Taken together, the mitigation results reveal a strong task dependence. Across all three mitigation strategies, performance is substantially more stable on FetchSlide than on FetchPickAndPlace. For Slide, mitigation methods largely preserve capability and alignment while consistently reducing action and jerk violations. In contrast, Ensemble and Physics-constrained rewards cause severe performance degradation on PickAndPlace, while KL regularization improves success only at the cost of substantially increased unsafe motions. One possible explanation is that Slide primarily rewards smooth object motion, making safety constraints naturally compatible with task completion, whereas PickAndPlace requires contact-rich manipulation involving grasping, lifting, and corrective actions that are directly suppressed by conservative safety mechanisms. These results suggest that mitigation strategies are most effective when their imposed behavioral constraints align with the physical requirements of the underlying task.

5.3 Hyperparameter Ablations

We further sweep the KL penalty strength β and physics penalty coefficient c. Table 3 reports the 500k-step checkpoint.

Table 3: Hyperparameter ablations at 500k steps. Each entry reports success / alignment.

Method	PickAndPlace	Slide
KL $\beta = 0.01$	**93 / 0.72**	**58 / 0.87**
KL $\beta = 0.1$	17 / 0.97	6 / 0.75
KL $\beta = 1.0$	4 / 0.99	0 / n/a
Physics $c = 0.1$	27 / 0.97	47 / 0.87
Physics $c = 1.0$	3 / 0.99	**62 / 0.82**
Physics $c = 10.0$	3 / 0.99	0 / n/a

The coefficients β and c control the strength of the mitigation. For KL regularization, larger β forces the learned policy to stay closer to the sparse-reward reference policy. For physics constraints, larger c increases the penalty assigned to action, jerk, and task-specific safety violations.

The ablation results show that both mitigations are highly sensitive to this strength parameter. For KL regularization, $\beta = 0.01$ is the only setting that consistently improves performance: it raises PickAndPlace success to 93% and maintains strong Slide performance. Increasing the penalty to $\beta = 0.1$ reduces success to 17% on PickAndPlace and 6% on Slide, while $\beta = 1.0$ nearly collapses learning. This suggests that KL regularization is useful only as a weak trust-region constraint; if the policy is forced too close to the reference policy, it cannot adapt to the dense LLM reward.

Physics penalties show a similar over-constraint pattern. On Slide, $c = 1.0$ gives the best tradeoff, preserving high success while reducing unsafe motions. A weaker penalty ($c = 0.1$) is not strong enough to produce the same safety benefit, while a stronger penalty ($c = 10.0$) dominates the reward and collapses success to 0%. On PickAndPlace, all physics penalties reduce success, suggesting that simple action and jerk penalties interfere with contact-rich grasping and object transport.

Based on these ablations, we use $\beta = 0.01$ and $c = 1.0$ as the representative settings in the main mitigation comparison. These settings are not universally optimal, but they provide the best observed tradeoff between success, alignment, and safety within their respective mitigation families.

5.4 Qualitative Policy Behavior

To complement the quantitative evaluation, we visually inspected rollout videos generated from the trained policies. Although the safety metrics quantify action magnitude and jerk violations, the videos provide additional intuition regarding how these behaviors manifest during execution.

For the Eureka baseline, successful policies generally exhibited smooth task completion but occasionally showed noticeable end-effector oscillations near the object and target locations. These small corrective motions were particularly visible during grasp acquisition and object placement, where the robot repeatedly adjusted its position before completing the task. Such behavior is consistent with the moderate action and jerk violation rates observed in the quantitative evaluation.

KL regularization produced the highest task success rates on PickAndPlace, but the resulting motions appeared more aggressive. The robot frequently executed larger corrective movements and faster repositioning actions when approaching the object. These behaviors qualitatively align with the increase in action and jerk violations reported in Table 2. Despite achieving superior task completion, the policies often appeared less smooth than the Eureka baseline.

In contrast, physics-constrained rewards generated visibly smoother trajectories. End-effector motion appeared more stable, with reduced oscillation and fewer abrupt direction changes. On FetchSlide, these smoother motions were achieved while maintaining performance close to the Eureka baseline, matching the substantial reductions in action and jerk violations measured quantitatively. However, on PickAndPlace, the same constraints often caused the robot to move too conservatively, resulting in failed grasp attempts or incomplete object transport despite exhibiting visually safe behavior.

Overall, the qualitative observations closely match the quantitative findings. Policies with lower action and jerk violation rates generally exhibited smoother and more stable motion, whereas policies achieving higher success rates often relied on more aggressive corrective behaviors. These results reinforce the conclusion that task success and motion safety represent distinct dimensions of policy quality and should be evaluated jointly.

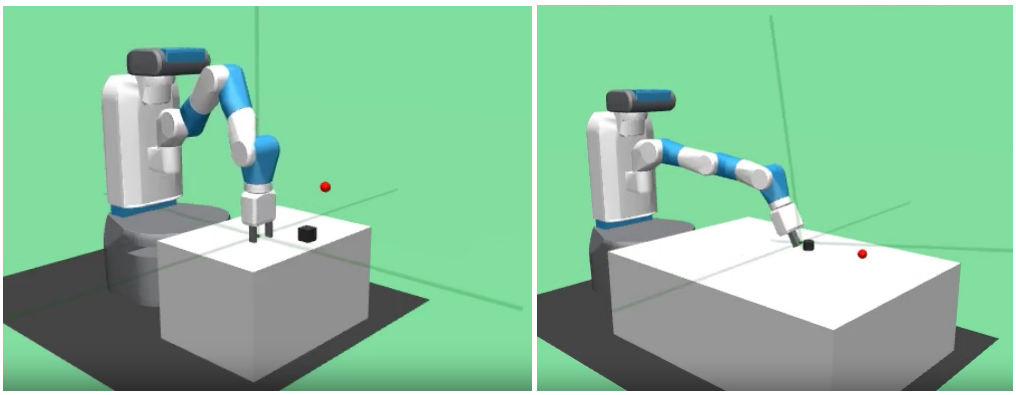

Figure 2: Representative screenshots extracted from rollout videos. Left: FetchPickAndPlace, where the robot performs object grasping and transport. Right: FetchSlide, where the robot pushes the object toward the target location. These frames provide qualitative examples of policy behavior observed during evaluation.

5.5 Training Dynamics

To illustrate the overall learning dynamics, Figure 3 shows representative training curves for the FetchSlide task. The plot compares optimization behavior across reward designs and highlights how differences in reward shaping influence both learning progress and final task performance.

6 Discussion

Our results reveal a more nuanced relationship between reward alignment, task performance, and physical safety than initially expected.

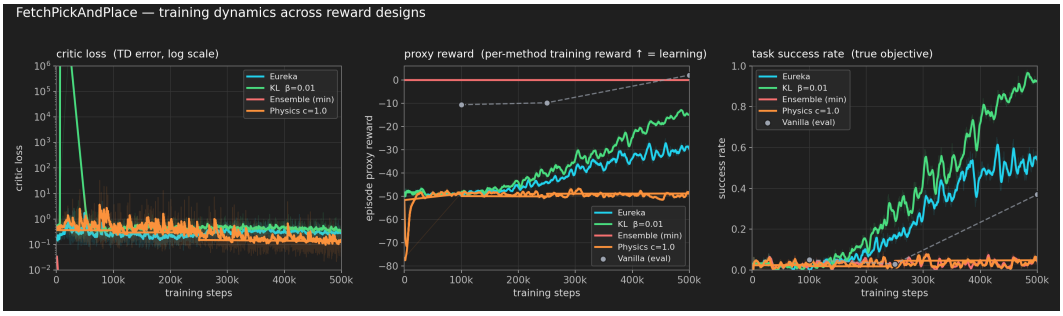

Figure 3: Training dynamics on `FetchSlide-v4` across reward designs. Left: critic loss (TD error) during SAC training. Middle: average proxy reward optimized by each reward function. Right: task success rate measured on the true environment objective. While several reward formulations achieve similar proxy-reward improvements, their resulting task performance differs substantially, highlighting the distinction between optimizing a reward signal and achieving the intended task behavior. FetchSlide is shown as a representative example of the training trends observed throughout our experiments.

A central finding is that reward hacking was less prevalent than anticipated. Among all reward functions evaluated, clear reward misalignment emerged primarily in the single-shot LLM reward for `FetchSlide-v4`. In contrast, Eureka-style iterative reward refinement consistently improved reward-success correlation and largely eliminated misaligned high-reward behaviors. Rather than amplifying reward hacking, iterative reward optimization often repaired poorly specified rewards by incorporating feedback from previous training runs.

More importantly, our experiments demonstrate that reward alignment, task capability, and safety should be viewed as separate evaluation dimensions. Several mitigation strategies maintained high reward-success correlations while achieving poor task success, indicating that an aligned reward function does not necessarily provide a useful optimization signal for reinforcement learning. Conversely, KL regularization on PickAndPlace achieved the highest task success observed in our experiments (93%) despite a noticeable reduction in reward alignment. These results suggest that reward-success correlation alone is insufficient for evaluating reward quality.

The mitigation study further highlights strong task dependence. On the more challenging `FetchSlide-v4` task, all three mitigation families were capable of reducing unsafe behavior while maintaining performance within a reasonable range of the Eureka baseline. In particular, KL regularization and physics-constrained rewards substantially reduced action and jerk violations while preserving comparable success rates. By contrast, the same methods often degraded performance on `FetchPickAndPlace-v4`. Reward ensembles and physics penalties nearly collapsed learning, while KL regularization improved success only at the cost of significantly increased aggressive motions. This contrast suggests that mitigation strategies may be most effective when applied to tasks with more complex dynamics and greater opportunities for reward exploitation, whereas simpler manipulation tasks can become over-constrained.

The hyperparameter ablations reinforce this observation. Both KL regularization and physics constraints exhibit a narrow operating regime. Small regularization strengths can improve safety-performance tradeoffs, but stronger penalties rapidly suppress learning altogether. The sensitivity observed across both tasks suggests that mitigation design may be as important as reward design itself, requiring task-specific tuning rather than universal parameter choices.

Several limitations remain. Although evaluations were performed across multiple checkpoints and aggregated over many evaluation episodes, the study considers only two simulated robotic manipulation environments. Furthermore, safety was measured using proxy metrics such as action magnitude and jerk violations rather than direct force, torque, or collision measurements. Finally, all experiments were conducted in simulation using relatively low-dimensional state observations. Real robotic systems may introduce additional sources of reward exploitation, perception errors, and safety concerns that are not captured by our benchmark.

Future work could extend this analysis to contact-rich manipulation tasks, dexterous hand environments, and real-world robotic platforms. More sophisticated mitigation methods, such as adaptive safety penalties or constraint-aware reward generation, may also provide better tradeoffs than the fixed regularization strategies explored here. An additional direction is to investigate whether large language models can explicitly reason about physical safety during reward generation rather than relying on downstream mitigation after reward construction.

7 Conclusion

This work investigated the relationship between LLM-generated rewards, reward hacking, and physical safety in robotic reinforcement learning. Using FetchPickAndPlace and FetchSlide, we evaluated single-shot LLM rewards, Eureka-style iterative reward refinement, and several mitigation strategies including KL regularization, reward ensembles, and physics-constrained reward shaping.

Our results show that reward hacking is not an inevitable outcome of LLM-generated rewards. While reward misalignment emerged in the single-shot Slide reward, iterative reward refinement substantially improved both task success and reward alignment. More broadly, we found that reward alignment, task capability, and safety are distinct properties that can vary independently. High reward-success correlation does not guarantee successful learning, and improved task performance does not necessarily imply safer behavior.

The mitigation experiments further revealed strong task dependence. For the more challenging Slide task, mitigation strategies were often able to preserve performance while reducing unsafe actions. In contrast, the same interventions frequently degraded performance on PickAndPlace, demonstrating that safety mechanisms effective in one manipulation setting may not generalize to another. Hyperparameter ablations showed that both KL regularization and physics penalties operate within a narrow range, with excessive regularization quickly suppressing learning.

Overall, our findings suggest that evaluating LLM-generated rewards requires a multidimensional perspective that jointly considers alignment, capability, and safety. Future robotic reward-generation systems should therefore be assessed not only by whether they optimize task success, but also by how safely and robustly they achieve that success across diverse manipulation tasks.

8 Team Contributions

- **Sunny Yuan**: LLM reward generation pipeline (Eureka-style), PGCA and CMDP mitigation implementations, safety metric design and analysis, LLM Safety Critic exploration, paper writing.
- **Sophia Huang**: Robotics manipulation environment setup and task design, expert demonstration collection, human-engineered reward baseline, safety-prompting experiments, robustness evaluation across task variations.

References

Eitan Altman. 1999. *Constrained Markov Decision Processes*. CRC Press.

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. 2016. Concrete Problems in AI Safety. *arXiv preprint arXiv:1606.06565* (2016).

Puze Liu et al. 2024. Safe Reinforcement Learning on the Constraint Manifold: Theory and Applications. *arXiv preprint arXiv:2404.09080* (2024).

Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2024. Eureka: Human-Level Reward Design via Coding Large Language Models. In *International Conference on Learning Representations (ICLR)*.

Md. Shihab and others Akter. 2025. Detecting and Mitigating Reward Hacking in Reinforcement Learning: A Large-Scale Empirical Study. *arXiv preprint arXiv:2507.05619* (2025).

Various Authors. 2024a. Safe Reinforcement Learning for Arm Manipulation with Constrained Markov Decision Process. *Robotics* 13, 4 (2024), 63.

Various Authors. 2024b. A Survey of Constraint Formulations in Safe Reinforcement Learning. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Various Authors. 2025. Safe Reinforcement Learning for Vision-Based Robotic Manipulation in Human-Robot Collaboration. *Artificial Life and Robotics* (2025).

Lilian Weng. 2024. Reward Hacking in Reinforcement Learning. *lilianweng.github.io* (2024). `https://lilianweng.github.io/posts/2024-11-28-reward-hacking/`

Tianbao Xie, Siheng Zhao, Chen Henry Wu, Yitao Liu, Qian Luo, Victor Zhong, Yanchao Yang, and Tao Yu. 2024. Text2Reward: Reward Shaping with Language Models for Reinforcement Learning. In *International Conference on Learning Representations (ICLR)*.

Wenhao Yu, Nimrod Gileadi, Chuyuan Fu, Sean Kirmani, Kuang-Huei Lee, Montse Gonzalez Arenas, Hao-Tien Lewis Chiang, Tom Erez, Leonard Hasenclever, Jan Humplik, et al. 2023. Language to Rewards for Robotic Skill Synthesis. *arXiv preprint arXiv:2306.08647* (2023).