ICPL: Few-shot In-Context Preference Learning via LLMs

Chao Yu, Qixin Tan, Hong Lu, Jiaxuan Gao, Xinting Yang, Yu Wang, Yi Wu, Eugene Vinitsky

Keywords: Large Language Models, Preference-based RL, Reinforcement Learning from Human Feedback

Summary

Preference-based reinforcement learning is an effective way to handle tasks where rewards are hard to specify but can be exceedingly inefficient as preference learning is often tabula rasa. We propose In-Context Preference Learning (ICPL), which uses the in-context learning capabilities of LLMs to reduce human query inefficiency. ICPL uses the task description and basic environment code to create sets of reward functions, which are iteratively refined by placing human feedback over videos of the resultant policies into the context of an LLM and requesting better rewards. We demonstrate ICPL's effectiveness through a synthetic preference study, showing that it significantly outperforms baseline preference-based methods in performance and efficiency, and that improvements come from both LLM grounding and reward refinement via preference feedback. We also perform real human preference-learning trials and show that ICPL extends to humans-in-the-loop and to challenging, subjective tasks such as making a humanoid jump like a human.

Contribution(s)

1. We propose ICPL, an LLM-based online preference learning algorithm that uses in-context learning to refine reward functions from human preference feedback over videos of agent behaviors.
   Context: Prior preference-based RL methods learn reward models tabula rasa and often require thousands of queries; we show that leveraging LLM coding and in-context reasoning significantly reduces query count while improving performance.
2. We demonstrate that ICPL sharply outperforms tabula-rasa PbRL methods (PrefPPO (Lee et al., 2021b), PEBBLE (Lee et al., 2021a), SURF (Park et al., 2022)) in query efficiency and final task score, and is competitive with reward-generation methods that use ground-truth sparse rewards (Eureka (Ma et al., 2023)).
   Context: Experiments on nine IsaacGym tasks with proxy human preferences show over 30$\times$ reduction in queries at equivalent or better performance; ablations confirm the gain is from preference-driven refinement rather than zero-shot task memorization.
3. Through human-in-the-loop trials on five IsaacGym tasks and a novel HumanoidJump task, we show that ICPL works effectively with real (noisy) human preferences and can guide agents toward nuanced, human-like behaviors (e.g., jumping with both legs).
   Context: Human-in-the-loop trials across five IsaacGym tasks produce results consistent with the proxy-preference metric. On the subjective HumanoidJump task, human participants show a clear majority preference for ICPL over an open-loop baseline.

ICPL: Few-shot In-Context Preference Learning via LLMs

Chao Yu[1,†], **Qixin Tan**[1,†], **Hong Lu**[1], **Jiaxuan Gao**[1], **Xinting Yang**[1], **Yu Wang**[1], **Yi Wu**[1,2], **Eugene Vinitsky**[3]
vinitsky.eugene@gmail.com

[1]**Tsinghua University, China**
[2]**Shanghai Qi Zhi Institute, China**
[3]**NYU Tandon School of Engineering, USA**

[†] These authors contributed equally.
Correspondence to: Eugene Vinitsky <vinitsky.eugene@gmail.com>

Abstract

Preference-based reinforcement learning is an effective way to handle tasks where rewards are hard to specify but can be exceedingly inefficient as preference learning is often tabula rasa. To address this challenge, we propose In-Context Preference Learning (ICPL), which uses in-context learning capabilities of LLMs to reduce human query inefficiency. ICPL uses the task description and basic environment code to create sets of reward functions which are iteratively refined by placing human feedback over videos of the resultant policies into the context of an LLM and then requesting better rewards. We first demonstrate ICPL's effectiveness through a synthetic preference study, providing quantitative evidence that it significantly outperforms baseline preference-based methods with much higher performance and orders of magnitude greater efficiency. We observe that these improvements are not solely coming from LLM grounding in the task but also the quality of the rewards improvement via preference feedback. Additionally, we perform a series of real human preference-learning trials and observe that ICPL extends beyond synthetic settings and can work effectively with humans-in-the-loop.

1 Introduction

Defining desired agent behaviors explicitly is often challenging—how does one design a reward function for "following instructions" or "behaving in a non-toxic manner"? Preference-based reinforcement learning (PbRL) offers a solution by learning rewards through comparisons of trajectory orderings, bypassing the need for explicit reward specification. While directly encoding behaviors like "following instructions" is difficult, determining whether an instruction has been followed is much more straightforward. PbRL, in which we simultaneously learn a policy and reward that satisfy a set of preferences, has shown success in various tasks (Christiano et al., 2017; Ibarz et al., 2018; Liu et al., 2020; Wu et al., 2021; Lee et al., 2021a). However, its deployment and scalability remain limited by the costly data collection process, which serves as a major bottleneck. Even for simple tasks, such as learning a reward for pressing a button, it requires over 10k labeled comparisons (Lee et al., 2021b) to achieve good performance as PbRL is often tabula rasa.

We investigate whether in-context learning (ICL), the ability of LLMs to modify their behavior based on a few provided examples, can be used to make the preference-learning step of PbRL sample efficient. Specifically, we focus on *online* PbRL, where data collection and reward function training occur in an iterative loop. In each round, we collect preference data, use it to learn a reward function, learn a policy under that reward, and then gather new preference data under the updated policy.

To incorporate ICL into the PbRL loop, we put into context the collected series of preferences and programmatic reward functions and ask the LLM to compare across the context to infer a new reward that better explains the preferences. Asking this of an LLM goes beyond previously observed ICL capabilities, requiring not just imitation of provided examples that have been stored in context but also reasoning about the relationships between different rewards and their alignment with preferences. Moreover, determining the optimal way to structure this information to maximize preference learning performance remains an open question.

As a step towards LLMs functioning as effective guides of the PbRL loop, we introduce a new method, **I**n-**C**ontext **P**reference **L**earning (**ICPL**), which significantly enhances the human query efficiency of PbRL. Our approach is to harness the coding capabilities of LLMs to autonomously generate improved reward functions, represented as code, that take into context prior preferences and generated rewards. Specifically, ICPL leverages an LLM to generate executable, diverse reward functions based on the task description and environment source code. We acquire preferences by evaluating the agent behaviors resulting from these reward functions, selecting the most and least preferred behaviors. The selected functions, along with historical data such as reward traces of the generated reward functions from RL training, are then fed back into the LLM which attempts to generate an improved reward function.

To study the effectiveness of ICPL, we perform experiments on a diverse set of RL tasks. For scalability of experimentation, we first study tasks with proxy human preferences where a ground-truth reward function is used to assign preference labels. We observe that compared to traditional PbRL algorithms, ICPL achieves over a 30 times reduction in the required number of preference queries to achieve equivalent or superior performance. Moreover, given only preference feedback, ICPL achieves performance comparable to reward-generation methods that require a ground-truth sparse reward as feedback (Ma et al., 2023). Additionally, we perform a series of real human preference learning trials and observe that ICPL extends beyond synthetic settings and can work effectively with humans-in-the-loop. Finally, we test ICPL on a particularly challenging task, "making a humanoid jump like a real human," where designing a reward is difficult. By using real human feedback, our method successfully trained an agent capable of bending both legs and performing stable, human-like jumps, showcasing the potential of ICPL in tasks where human intuition plays a critical role.

In summary, the contributions of the paper are the following:

- We propose ICPL, an LLM-based online preference learning algorithm. Over a synthetic set of preferences, we demonstrate that ICPL's effectiveness is not solely coming from LLM grounding but also the quality of the rewards improvement via preference feedback.
- We demonstrate that ICPL sharply outperforms tabula-rasa PbRL methods in terms of query efficiency and performance and is also competitive with reward-generation methods that rely on access to a ground-truth sparse reward.
- Through human-in-the-loop trials, we demonstrate that ICPL performs effectively even when dealing with significantly noisier preference labels.

2 Related Work

Feedback from humans has been proven to be effective in training RL agents that better match human preferences (Retzlaff et al., 2024; Mosqueira-Rey et al., 2023; Kwon et al., 2023). Previous works have investigated human feedback in various forms, such as trajectory comparisons, preferences, demonstrations, and corrections (Wirth et al., 2017; Ng et al., 2000; Jeon et al., 2020; Peng et al., 2024). Among these various methods, PbRL learns both a reward model and policy based on human preferences across different trajectories (Liu et al., 2020; Wu et al., 2021) and has been successfully scaled to train large foundation models for hard tasks like dialogue, e.g. ChatGPT (Ouyang et al., 2022). In LLM-based applications, prompting is a simple way to provide human feedback in order to align LLMs with human preferences (Giray, 2023; White et al., 2023; Chen et al., 2023). Iteratively refining the prompts with feedback from the environment or human users has shown promise in

improving the output of the LLMs (Wu et al., 2021; Nasiriany et al., 2024). This work builds on these demonstrations of the ability to control LLM behavior via in-context prompts. We use interactive rounds of preference feedback between the LLM and humans to guide the LLM to generate reward functions that gradually elicit behaviors that align with human preferences.

Our investigation into using LLMs to generate novel reward functions via in-context learning builds on several previously demonstrated capabilities of LLMs. First, it has been previously observed that LLMs can translate textual descriptions of desired behaviors into reward functions, expressed as code, that approximate those behaviors (Ma et al., 2022; Du et al., 2023; Karamcheti et al., 2023; Kwon et al., 2023; Wang et al., 2024; Ma et al., 2024; Holk et al., 2024). Second, prior works have demonstrated that LLMs can use a variety of signals such as a history of rewards to generate new programs that improve on the history of rewards (Ma et al., 2023; Romera-Paredes et al., 2024). This suggests that LLMs can function as approximate optimizers, a capability that forms the foundation of our approach to preference learning. Finally, LLMs have been demonstrated to be able to perform complex functional transformations on data stored in their context, as is required for preference learning, being able to perform linear regression (Tang et al., 2024), no-regret learning (Park et al., 2024), and combinatorial problems (Liu et al., 2024).

In concurrent work (Clark et al., 2025), LLMs generate task parameterizations for quadruped locomotion, with humans ranking trajectories to identify optimal configurations via preference learning. Both papers noted that LLMs can be used to perform in-context preference learning; their approach focuses on generating task parameterizations as reward function vectors for gradient-based optimization, while our ICPL framework shows that LLMs can directly act as few-shot preference learners to generate and optimize reward functions.

3 Problem Definition

Our goal is to design a reward function that can be used to train RL agents that demonstrate human-preferred behaviors. It is usually hard to design proper reward functions in reinforcement learning that induce policies that align well with human preferences.

Markov Decision Process with Preferences (Wirth et al., 2017). A *Markov Decision Process with Preferences* (MDPP) is defined as a tuple $M = \langle \mathcal{S}, A, \mu, \sigma, \gamma, \rho \rangle$ where $\mathcal{S}$ denotes the state space, A denotes the action space, μ is the distribution of initial states, σ is the state transition model, $\gamma \in [0, 1)$ is the discount factor. ρ is the preference relation over trajectories, i.e. $\rho(\tau_i \succ \tau_j)$ denotes the probability with which trajectory τ_i is preferred over τ_j. Given a set of preferences ζ, the goal in an MDPP is to find a policy π^* that maximally complies with ζ. A preference $\tau_1 \succ \tau_2$ is satisfied by π if and only if $\Pr_\pi(\tau_1) > \Pr_\pi(\tau_2)$ where $\Pr_\pi(\tau) = \mu(s_0) \prod_{t=0}^{|\tau|} \pi(a_t|s_t)\sigma(s_{t+1}|s_t, a_t)$. This can be viewed as finding a π^* that minimizes a preference loss $L(\pi_\zeta) = \sum_i L(\pi, \zeta_i)$, where $L(\pi, \tau_1 \succ \tau_2) = -(\Pr_\pi(\tau_1) - \Pr_\pi(\tau_2))$.

Reward Design Problem with Preferences. A *reward design problem with preferences (RDPP)* is a tuple $P = \langle M, \mathcal{R}, A_M, \zeta \rangle$, where M is a Markov Decision Process with Preferences, $\mathcal{R}$ is the space of reward functions, $A_M(\cdot) : \mathcal{R} \to \Pi$ is a learning algorithm that outputs a policy π that optimizes a reward $R \in \mathcal{R}$ in the MDPP. $\zeta = \{(\tau_1, \tau_2)\}$ is the set of preferences. In an RDPP, the goal is to find a reward function $R \in \mathcal{R}$ such that the policy $\pi = A_M(R)$ that optimizes R maximally complies with the preference set ζ. In PbRL, the learning algorithms usually involve multiple iterations, and the preference set ζ is constructed in every iteration by sampling trajectories from the policy or policy population.

4 Method

Our proposed method, In-Context Preference Learning (ICPL), integrates LLMs with human preferences to synthesize reward functions. The LLM receives environmental context and a task description to generate an initial set of K executable reward functions. ICPL then iteratively refines these

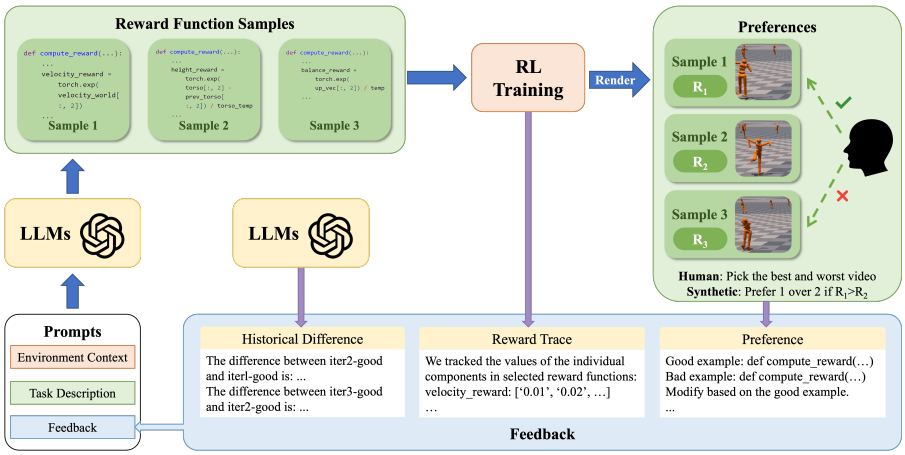

Figure 1: ICPL employs the LLM to generate initial K executable reward functions based on the task description and environment context. Using RL, agents are trained with these reward functions. Videos are generated of the resultant agent behavior from which human evaluators select their most and least preferred. These selections serve as examples of positive and negative preferences. The preferences, along with additional contextual information, are provided as feedback prompts to the LLM, which is then requested to synthesize a new set of reward functions. For experiments simulating human evaluators, task scores are used to determine the best and worst reward functions.

functions. In each iteration, the LLM-generated reward functions are trained within the environment, producing a set of agents; we use these agents to generate videos of their behavior. A ranking is formed over the videos, from which we retrieve the best and worst reward functions corresponding to the top and bottom videos in the ranking. These selections serve as examples of positive and negative preferences. The preferences, along with additional contextual information, such as reward traces and differences from previous good reward functions, are provided as feedback prompts to the LLM. The LLM takes in this context and is asked to generate a new set of rewards. Fig. 1 illustrates the overall process of ICPL.

4.1 Reward Function Initialization

To enable the LLM to synthesize effective reward functions, it is essential to provide task-specific information, which consists of two key components: a description of the environment, including the observation and action space, and a description of the task objectives. At each iteration, ICPL ensures that K executable reward functions are generated by resampling until there are K executable reward functions.

4.2 Search Reward Functions by Human Preferences

For tasks without reward functions, the traditional PbRL typically involves constructing a reward model, which often demands substantial human feedback. Our approach, ICPL, aims to enhance efficiency by leveraging LLMs to directly search for optimal reward functions without the need to learn a reward model.

To expedite this search process, we use an LLM-guided search to find well-performing reward functions. Specifically, we generate $K = 6$ executable reward functions per iteration across $N = 5$ iterations. In each iteration, humans select the most preferred and least preferred videos, resulting in a good reward function and a bad one. These are used as a context for the LLM to use to synthesize a new set of K reward functions. These reward functions are then used in a PPO (Schulman et al., 2017) training loop, and videos are rendered of the final trained agents.

4.3 Automatic Feedback

In each iteration, the LLM not only incorporates human preferences but also receives automatically synthesized feedback. This feedback is composed of three elements: the evaluation of selected reward functions, the differences between historical good reward functions, and the reward trace of these historical reward functions.

Evaluation of reward functions: The component values that make up the good and bad reward functions are obtained from the environment during training and provided to the LLM. This helps the LLM assess the usefulness of different parts of the reward function by comparing the two.

Differences between historical reward functions: The best reward functions selected by humans from each iteration are taken out, and for any two consecutive good reward functions, their differences are analyzed by another LLM. These differences are supplied to the primary LLM to assist in adjusting the reward function.

Reward trace of historical reward functions: The reward trace, consisting of the values of the good reward functions during training from all prior iterations, is provided to the LLM. This reward trace enables the LLM to evaluate how well the agent is actually able to optimize those reward components.

5 Experiments

In this section, we conducted two sets of experiments to evaluate the effectiveness of our method: one using proxy human preferences and the other using real human preferences.

1) **Proxy Human Preference**: We follow the standard experiment setting of PbRL where human-designed rewards were used as proxies of human preferences. It enables rapid and quantitative evaluation of our approach. Proxy human preference corresponds to a noise-free case that is likely easier than human trials; if ICPL performed poorly here it would be unlikely to work in human trials. Importantly, human-designed rewards were only used to automate the selection of samples and were not included in the prompts sent to the LLM; the LLM **never observes the functional form of the ground truth rewards nor does it ever receive any values from them**. Since proxy human preferences are free from noise, they offer a reliable comparison to evaluate our approach efficiently. However, as discussed later in the limitations section, these proxies may not correctly measure challenges in human feedback such as the inability to rank samples, intransitive preferences, or other biases.

2) **Human-in-the-loop Preference**: To further validate our method, we conducted a second set of experiments with human participants. These participants repeated the tasks from the Proxy Human Preferences and engaged in an additional task that lacked a clear reward function.

5.1 Baselines

We consider three PbRL methods as baselines, which update reward models during training. B-Pref (Lee et al., 2021b), a benchmark specifically designed for PbRL, provides two of our baseline algorithms: **PrefPPO** and **PEBBLE**. PrefPPO is based on the on-policy RL algorithm PPO, while PEBBLE builds upon the off-policy RL algorithm SAC. Additionally, we include **SURF** (Park et al., 2022), which enhances PEBBLE by utilizing unlabeled samples with data augmentation to improve feedback efficiency. For each task, we use the default hyperparameters of PPO and SAC provided by IsaacGym, which were fine-tuned for high performance. This ensures a fair comparison across methods.

Definition of Human Query Q. To evaluate the human effort required for ICPL and baseline methods, we track the number of human queries Q, which quantifies the amount of human effort involved in a human-in-the-loop experiment—a crucial factor for these methods. Specifically, we define a single human query as a human comparing two trajectories or videos and providing a preference.

In ICPL, each iteration generates K reward function samples, resulting in K corresponding videos. The human compares these videos, first selecting the best one, then picking the worst from the remaining $K-1$ videos. After $N=5$ iterations, the best video of each iteration is compared to select the overall best. The number of human queries Q can be calculated as $Q=(K-1)\times 2N-1$. For ICPL, with $K=6$ and $N=5$, this results in $Q=49$. In baseline methods, humans compare two sampled trajectories and provide a preference label to update the reward model. To ensure a fair comparison, we set the maximum number of queries to $Q=49$, matching ICPL. Additionally, we evaluate larger query budgets of $Q=150, 1.5k, 15k$, denoted as Baseline-#Q.

5.2 Testbed

Tasks. We first adopt several tasks from the GPU-based IsaacGym with human-designed rewards for quantitative comparison (Ma et al., 2023), covering diverse environments: *Cartpole, BallBalance, Quadcopter, Anymal, Humanoid, Ant, FrankaCabinet, ShadowHand,* and *AllegroHand*. To ensure fair evaluation, we strictly follow the original task configurations, including observation space, action space, and reward computation. We refer to these tasks collectively as *IsaacGym Tasks* in the following discussion. Additionally, we introduce a new task, HumanoidJump, defined as "making a humanoid jump like a real human." Defining a clear reward for this task is challenging, as human-like jumping lacks easily quantifiable criteria.

Task Metric. The task metric is the average of the sparse rewards across parallel environments. To assess the generated reward function or the learned reward model for each RL run, we take the maximum task metric value sampled at fixed intervals, marked as *task score of reward function/model* (RTS). In each iteration, ICPL generates 6 RL runs and selects the highest RTS as the result for that iteration. ICPL performs 5 iterations and then selects the highest RTS from these iterations as the *task score* (TS) for each experiment. Due to the inherent randomness of LLMs, we run 5 experiments for all methods, and report the highest TS as the *final task score* (FTS) for each approach. A higher FTS indicates better performance across all tasks.

5.3 Training Details

We trained policies and rendered videos on a single A100 GPU machine. The total time for a full experiment was less than one day of wall clock time. We utilized GPT-4, specifically GPT-4-0613, as the backbone LLM in the Proxy Human Preference experiment. For the Human-in-the-loop Preference experiment, we employ GPT-4o.

5.4 Results of Proxy Human Preference

Experiment Setup. In ICPL, we use human-designed sparse rewards as proxies to simulate ideal human preferences. Specifically, in each iteration, we select the reward function with the highest RTS as the good example and the reward function with the lowest RTS as the bad example for feedback. All baseline methods leverage dense rewards to simulate proxy human preference, offering a stronger and more informative signal for labeling preferences. If the cumulative reward of trajectory 1 is greater than that of trajectory 2, then trajectory 1 is preferred over trajectory 2.

Main Results. Table 1 shows the final task score (FTS) for ICPL and baseline methods with $Q=49, 150, 1.5k, 15k$ across IsaacGym tasks. As shown in Table 1, for the simpler tasks like *Cartpole* and *BallBalance*, all methods achieve equal performance. Notably, we observe that for these particularly simple tasks, ICPL can generate correct reward functions in a zero-shot manner, without requiring feedback. As a result, ICPL only requires querying the human 5 times, while baseline methods, after 5 queries, fail to train a reasonable reward model with the preference-labeled data. For relatively more challenging tasks, Baseline-49 performs significantly worse than ICPL when using the same number of human queries. In fact, Baseline-49 fails in most tasks. As the number of human queries increases, baselines' performance improves across most tasks, but it still falls noticeably short compared to ICPL. This demonstrates that ICPL, with the integration of LLMs, can

Table 1: The final task score of all methods across different tasks in IsaacGym. The top result and those within one standard deviation are highlighted in bold.

	Cart.	Ball.	Quad.	Anymal	Ant	Human.	Franka	Shadow	Allegro
PrefPPO-49	**499**	**499**	-1.066	-1.861	0.743	0.457	0.0044	0.0746	0.0125
PEBBLE-49	**499**	**499**	-0.901	-1.3357	5.9891	3.67	0.0453	0.2627	0.1467
SURF-49	**499**	**499**	-1.202	-1.35	0.874	2.406	0.0345	0.2338	0.2002
PrefPPO-150	**499**	**499**	-0.959	-1.818	0.171	0.607	0.0179	0.0617	0.0153
PEBBLE-150	**499**	**499**	-1.059	-1.394	7.257	4.1417	0.0532	0.269	0.2811
SURF-150	**499**	**499**	-1.114	-1.383	7.878	4.312	0.5285	0.2512	0.2727
PrefPPO-1.5k	**499**	**499**	-0.486	-1.417	4.458	1.329	0.3248	0.0488	0.0284
PEBBLE-1.5k	**499**	**499**	-0.529	-1.213	9.364	4.075	0.1966	0.2538	0.2664
SURF-1.5k	**499**	**499**	-0.308	-1.278	7.921	3.577	0.8032	0.2575	0.2283
PrefPPO-15k	**499**	**499**	-0.250	-1.357	4.626	1.317	0.0399	0.0468	0.0157
PEBBLE-15k	**499**	**499**	-0.231	-0.73	8.543	6.162	0.8613	0.246	0.2755
SURF-15k	**499**	**499**	-0.266	-0.76	7.859	3.532	0.5466	0.3199	0.2352
ICPL(Ours)	**499**	**499**	**-0.0195**	**-0.007**	**12.04**	**9.227**	**0.9999**	**13.231**	**25.030**
Eureka	499	499	-0.023	-0.003	10.86	9.059	0.9999	11.532	25.250

reduce human effort in preference-based learning by at least 30 times. To more comprehensively evaluate the performance beyond the final best selection, we also report the mean TS across all iterations in Appendix Table 5.

Performance Analysis. We further report the performance of reward-generation methods that utilize ground-truth sparse rewards, which serve as an approximate upper bound on the expected performance ICPL could achieve. For this, we use Eureka (Ma et al., 2023), a state-of-the-art LLM-powered reward design method that leverages sparse rewards as fitness scores. Specifically, in Eureka, the reward function with the highest RTS is selected as the candidate reward function for feedback in each iteration. Additionally, RTS is incorporated as the "task score" in the reward reflection prompt sent to the LLM. Original Eureka generates 16 reward functions in each iteration without checking their executability, assuming at least one will typically work across all considered environments in the first iteration. To ensure a fair comparison, we modified Eureka to generate a fixed number of executable reward functions, specifically $K = 6$ per iteration, the same as ICPL. This adjustment improves Eureka's performance in more challenging tasks, where it often generates fewer executable reward functions. As shown in Table 1, ICPL surprisingly achieves comparable performance, indicating that ICPL's use of LLMs for preference learning is effective.

5.5 Method Analysis

To validate the effectiveness of ICPL's module design, we conducted ablation studies. We aim to answer several questions that could undermine the results presented here:

1. Are components such as the reward trace or the reward difference helpful?
2. Can the reward functions improve via preference feedback? Or is it simply zero-shot outputting the correct reward function due to the task being in the training data?

5.5.1 Ablations

The results of the ablations are shown in Table 2. In these studies, "ICPL w/o RT" refers to removing the reward trace from the prompts sent to the LLMs. "ICPL w/o RTD" indicates the removal of both the reward trace and the differences between historical reward functions from the prompts. "ICPL w/o RTDB" removes the reward trace, differences between historical reward functions, and bad reward functions, leaving only the good reward functions and their evaluation in the prompts. The "OpenLoop" configuration samples $K \times N$ reward functions without any feedback, corresponding to the ability of the LLM to zero-shot accomplish the task.

Table 2: Ablation studies on ICPL modules. The values in parentheses represent the standard deviation.

	Cart.	Ball.	Quad.	Anymal	Ant	Human.	Franka	Shadow	Allegro
ICPL w/o RT	**499**(0)	**499**(0)	-0.0340(0.05)	-0.387(0.26)	10.50(0.45)	8.337(0.60)	**0.9999**(0.25)	10.769(2.30)	**25.641**(9.46)
ICPL w/o RTD	**499**(0)	**499**(0)	-0.0216(0.14)	**-0.009**(0.38)	10.53(0.39)	**9.419**(2.10)	**1.0000**(0.18)	11.633(1.25)	23.744(8.80)
ICPL w/o RTDB	**499**(0)	**499**(0)	**-0.0136**(0.03)	-0.014(0.42)	**11.97**(0.71)	8.214(2.88)	0.5129(0.06)	**13.663**(1.83)	**25.386**(3.42)
OpenLoop	**499**(0)	**499**(0)	-0.0410(0.32)	-0.016(0.50)	9.350(2.34)	8.306(1.63)	**0.9999**(0.22)	9.476(2.44)	23.876(7.91)
ICPL(Ours)	**499**(0)	**499**(0)	**-0.0195**(0.09)	**-0.007**(0.35)	**12.04**(1.69)	**9.227**(0.93)	**0.9999**(0.24)	**13.231**(1.88)	25.030(3.721)

Due to the large variance of the experiments, we mark the top two results in bold. As shown, ICPL achieves top 2 results in 8 out of 9 tasks and is comparable on the *Allegro* task. The "OpenLoop" configuration performs the worst, indicating that our method does not solely rely on GPT-4's either having randomly produced the right reward function or having memorized the reward function during its training. Additionally, "ICPL w/o RT" underperforms on multiple tasks, highlighting the importance of incorporating the reward trace of historical reward functions into the prompts.

5.5.2 Improvement Analysis

Table 1 presents the performance achieved by ICPL. The main synthetic metric uses a best-of-N style selection across generated reward functions, iterations, and repeated runs, so it intentionally reflects the strongest candidate under the protocol rather than a plain average-run score. To make this dependence explicit, we also report the mean TS across all iterations in Appendix 5, which remains consistently better than the baselines. Rather than relying only on absolute scores, we emphasize whether ICPL improves reward quality through iterative preference incorporation. We calculated the average RTS improvement compared to the first iteration for the two tasks with the largest improvements compared with "OpenLoop", *Ant* and *ShadowHand*. As shown in Fig. 2, RTS improves after multiple iterations (e.g., 5 vs. 1), highlighting that preference feedback refines reward functions over time.

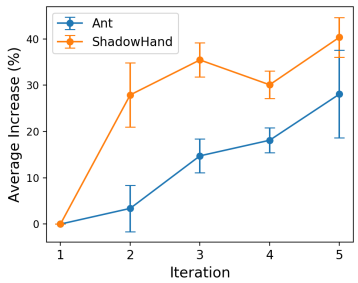

Figure 2: Average improvement of the Reward Task Score (RTS) compared with the first iteration in ICPL for the Ant and ShadowHand tasks, demonstrating the method's effectiveness in refining reward functions.

Qualitatively, the reward functions also evolve in a way that is consistent with the quantitative improvement. In representative running-task examples, early iterations mainly emphasize a small number of progress terms such as forward velocity, while later iterations introduce explicit stabilizers, for example uprightness or smoother action penalties, and rebalance their weights. In the human-in-the-loop jump task, early candidates can satisfy the minimal "jump" criterion with a leg-lift style behavior, but later iterations shift toward more human-like jumping by using both legs and lowering the upper body. This progression suggests that preference feedback is not only selecting among samples, but also reshaping the reward specification toward a richer objective.

5.6 Results of Human-in-the-loop Preference

To address the limitations of proxy human preferences, which simulate idealized human preference and may not fully capture the challenges humans may face in providing preferences, we conducted

Table 3: The final task score of human-in-the-loop preference across 5 IsaacGym tasks. The values in parentheses represent the standard deviation. "Impro." indicates the relative improvement over "OpenLoop", and "Average" reports the mean improvement across all 5 tasks.

	Quadcopter	Ant	Humanoid	Shadow	Allegro	Average
OpenLoop	-0.0410 (0.32)	9.350 (2.35)	8.306 (1.63)	9.476 (2.44)	23.876 (7.91)	–
ICPL-proxy	-0.0195 (0.09)	12.040 (1.69)	9.227 (0.93)	13.231 (1.88)	25.030 (3.72)	–
Impro.	52.4%	28.8%	11.1%	39.6%	4.83%	27.4%
ICPL-real	-0.0183 (0.29)	11.142 (0.37)	8.392 (0.53)	10.740 (0.92)	24.134 (6.52)	–
Impro.	55.4%	19.2%	1.04%	13.3%	1.08%	18.4%

experiments with real human participants. We recruited 7 volunteers for human-in-the-loop experiments, with 5 assigned to IsaacGym tasks and 2 to a newly designed task. Additionally, 20 volunteers were recruited to evaluate the performance of different methods. None of the volunteers had prior experience with these tasks, ensuring an unbiased evaluation based on their preferences.

Before the experiment, each volunteer was provided with a detailed explanation of the experiment's purpose and process. Additionally, volunteers were fully informed of their rights, and written consent was obtained from each participant. The experimental procedure was approved by the department's ethics committee to ensure compliance with institutional guidelines on human subject research.

5.6.1 IsaacGym Tasks

Due to the simplicity of the *Cartpole*, *BallBalance*, *Franka* tasks, where LLMs were able to zero-shot generate correct reward functions without any feedback, these tasks were excluded from the human trials. The *Anymal* task, which involved commanding a robotic dog to follow random commands, was also excluded as it was difficult for humans to evaluate whether the commands were followed based solely on the videos.

Table 3 presents the FTS for the human-in-the-loop preference experiments conducted across 5 suitable IsaacGym tasks, labeled as "ICPL-real". The results of the proxy human preference experiment are labeled as "ICPL-proxy". As observed, on average, "ICPL-proxy" achieves a 27.4% improvement over "OpenLoop". The performance of "ICPL-real" is comparable or slightly lower than that of "ICPL-proxy" in all 5 tasks, yet it still outperforms the "OpenLoop" results in 3 out of 5 tasks, with an average improvement of 18.4%. We can also observe that ICPL exhibits lower variance than "OpenLoop", indicating more stable reward learning behavior. This indicates that while humans may have difficulty providing consistent preferences from videos as proxies, their feedback can still be effective in improving performance when combined with LLMs.

5.6.2 HumanoidJump Task

Figure 3: A common behavior.

The most common behavior observed in this task, as illustrated in Fig. 3, is what we refer to as the "leg-lift jump." This behavior involves initially lifting one leg to raise the center of mass, followed by the opposite leg pushing off the ground to achieve lift. The previously lifted leg is then lowered

to extend airtime. Various adjustments of the center of mass with the lifted leg were also noted. This behavior meets the minimal metric of a jump: achieving a certain distance off the ground. If feedback were provided based solely on this minimal metric, the "leg-lift jump" would likely be selected as a candidate reward function. However, such candidates show limited improvement in subsequent iterations, failing to evolve into more human-like jumping behaviors.

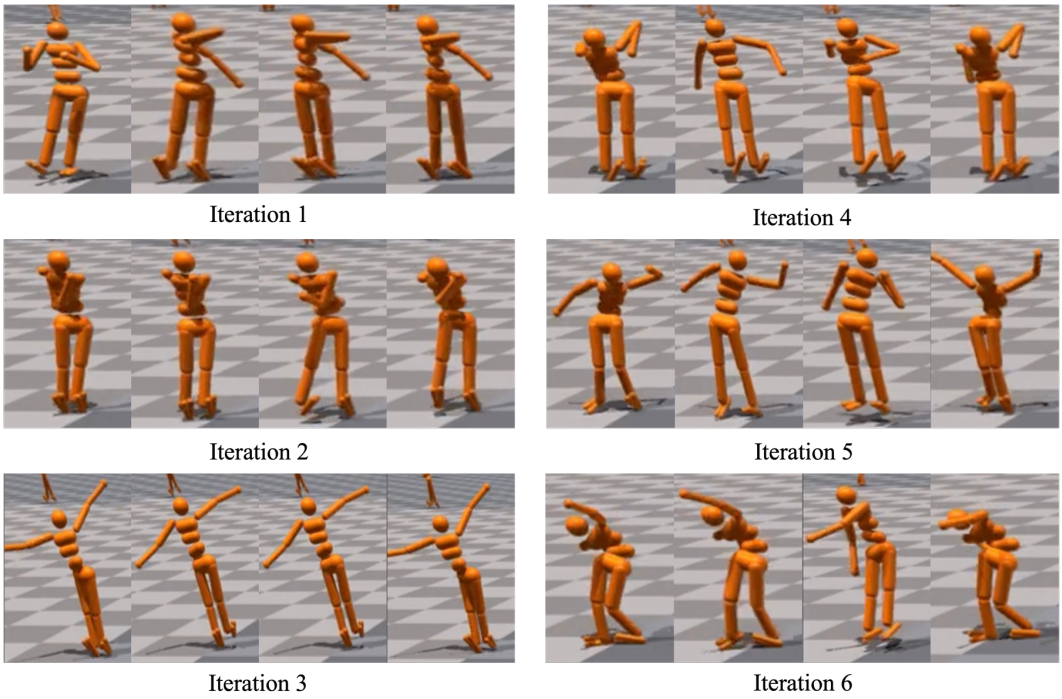

Figure 4: The humanoid learns a human-like jump by bending legs and lowering the upper body to shift the center of mass in a trial of human-in-the-loop experiments. Note that both legs are used to jump and the agent bends at the hips.

Conversely, when real human preferences were used to guide the task, the results were notably different. The volunteer judged the overall quality of the humanoid's jump behavior instead of just the metric of leaving the ground. Fig. 4 illustrates that the volunteer successfully guided the humanoid towards a more human-like jump by selecting behaviors that, while initially not optimal, displayed promising movement patterns. In the first iteration, "leg-lift jump" was not selected despite the humanoid jumping off the ground. Instead, a video where the humanoid appears to attempt a jump using both legs, without leaving the ground, was chosen. By the fifth and sixth iterations, the humanoid demonstrated more sophisticated behaviors, such as bending both legs and lowering the upper body to shift the center of mass, behaviors that are much more akin to a real human jump. The videos can be found at our website.

Table 4: Human Preferences over different behaviors.

Method	**Vote**
OpenLoop	3/20
ICPL	17/20

Quantitative Evaluation. Since designing a reward metric for the *HumanoidJump* task is challenging, we adopt human votes for quantitative evaluation instead. As a baseline, we use the "OpenLoop" configuration, which generates $K \times N$ reward functions without any feedback, on the HumanoidJump task. In this configuration, we performed 5 independent experiments, each comprising 6 iter-

Table 5: The average task score of all methods across different tasks in IssacGym. The values in parentheses represent the standard deviation.

	Cart.	Ball.	Quad.	Anymal	Ant	Human.	Franka	Shadow	Allegro
PrefPPO-49	**499**(0)	**499**(0)	-1.282(0.16)	-1.906(0.03)	0.351(0.20)	0.261(0.09)	0.0017(0.00)	0.0551(0.02)	0.0059(0.003)
PEBBLE-49	**499**(0)	**499**(0)	-1.190(0.14)	-1.424(0.06)	2.424(2.47)	2.196(1.32)	0.0237(0.01)	0.2341(0.03)	0.1120(0.03)
SURF-49	**499**(0)	**499**(0)	-1.238(0.03)	-1.444(0.09)	0.702(0.18)	1.630(0.53)	0.0177(0.01)	0.1925(0.03)	0.1314(0.03)
PrefPPO-150	**499**(0)	**499**(0)	-1.181(0.15)	-1.936(0.07)	0.112(0.05)	0.237(0.02)	0.0044(0.01)	0.0443(0.01)	0.0111(0.004)
PEBBLE-150	**499**(0)	**499**(0)	-1.209(0.07)	-1.440(0.03)	4.935(2.34)	2.643(1.11)	0.0329(0.02)	0.2490(0.02)	0.1691(0.06)
SURF-150	**499**(0)	**499**(0)	-1.220(0.06)	-1.414(0.03)	4.672(1.64)	2.732(1.18)	0.1553(0.18)	0.2287(0.01)	0.1863(0.05)
PrefPPO-1.5k	**499**(0)	**499**(0)	-0.689(0.11)	-1.835(0.21)	3.005(1.30)	0.856(0.33)	0.0775(0.12)	0.0383(0.01)	0.0211(0.005)
PEBBLE-1.5k	**499**(0)	**499**(0)	-0.694(0.14)	-1.338(0.07)	8.240(0.71)	3.528(0.44)	0.0894(0.07)	0.2233(0.03)	0.1851(0.07)
SURF-1.5k	**499**(0)	**499**(0)	-0.415(0.06)	-1.362(0.06)	6.044(1.93)	3.255(0.24)	0.3015(0.27)	0.2327(0.02)	0.1774(0.05)
PrefPPO-15k	**499**(0)	**499**(0)	-0.355(0.06)	-1.373(0.02)	3.862(0.57)	0.899(0.34)	0.0096(0.02)	0.0414(0.00)	0.0137(0.003)
PEBBLE-15k	**499**(0)	**499**(0)	-0.283(0.04)	-1.139(0.21)	7.887(0.56)	4.539(0.97)	0.6767(0.16)	0.2343(0.02)	0.1913(0.07)
SURF-15k	**499**(0)	**499**(0)	-0.302(0.02)	-1.138(0.20)	6.051(1.45)	2.614(0.82)	0.3937(0.08)	0.2301(0.05)	0.1475(0.07)
ICPL(Ours)	**499**(0)	**499**(0)	**-0.068**(0.09)	**-0.380**(0.35)	**9.856**(1.69)	**7.706**(0.93)	**0.7240**(0.24)	**10.541**(1.88)	**20.647**(3.721)
Eureka	499(0)	499(0)	-0.068(0.07)	-0.270(0.38)	9.749(0.85)	7.558(0.83)	0.7790(0.23)	9.619(1.38)	18.519(9.583)

ations with 6 samples per iteration. A volunteer selected the most preferred video as the final result. 20 additional volunteers were recruited to compare the performance of ICPL and OpenLoop. Each volunteer indicated their preference between two videos presented in random order—one generated by ICPL and the other by OpenLoop. As shown in Table 4, 17 out of 20 participants preferred the ICPL agent, demonstrating that ICPL produces behaviors more aligned with human preferences.

6 Conclusion

By leveraging the generative capabilities of LLMs to autonomously produce reward functions, and iteratively refining them using human preference feedback, ICPL reduces the complexity and human effort typically associated with PbRL. Our experimental results, both in proxy human preference and human-in-the-loop settings, show that ICPL not only surpasses traditional PbRL baselines in human query efficiency but also competes effectively with methods utilizing ground-truth rewards instead of preferences. Furthermore, the success of ICPL in complex, subjective tasks like humanoid jumping highlights its versatility in capturing nuanced human intentions, opening new possibilities for future applications in complex real-world scenarios where traditional reward functions are difficult to define.

Limitations. While ICPL demonstrates significant potential, it faces limitations in tasks where human evaluators struggle to assess performance from video alone, such as *Anymal*'s "follow random commands." In such cases, subjective human preferences may not provide adequate guidance. Future work will explore integrating partially defined metrics with human preferences. Another area for improvement is incorporating text feedback, where participants explain their preferences, potentially guiding the LLM more efficiently. Additionally, we observe that the performance of the task is qualitatively dependent on the diversity of the initial reward functions that seed the search. Relying on the LLM to provide this initial diversity is a current limitation. Furthermore, the limited number of participants in human-in-the-loop experiments may restrict the generalizability of our findings, as it might not fully capture the broad range of human preferences. Another limitation of ICPL is that each iteration involves training new RL policies, resulting in a waiting period of several hours for participants before they can provide additional feedback. This could be addressed by continuously training an RL agent under non-stationary reward functions, which presents a promising direction for future work.

Appendix

To more comprehensively evaluate the performance beyond the final best selection, we also report the mean task score (TS) across all iterations, as shown in Table 5.

The separate Supplementary Materials PDF submitted alongside this paper provides a fuller collection of auxiliary material, including prompt templates, experimental data summaries, representative reward-function evolution examples, and code-level comparisons across iterations.

For additional details, demonstrations, and videos of ICPL in action, we suggest visiting https://sites.google.com/view/few-shot-icpl/home.

References

Banghao Chen, Zhaofeng Zhang, Nicolas Langrené, and Shengxin Zhu. Unleashing the potential of prompt engineering in large language models: a comprehensive review. *arXiv preprint arXiv:2310.14735*, 2023.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

Jaden Clark, Joey Hejna, and Dorsa Sadigh. Efficiently generating expressive quadruped behaviors via language-guided preference learning, 2025. URL https://arxiv.org/abs/2502.03717.

Yuqing Du, Ksenia Konyushkova, Misha Denil, Akhil Raju, Jessica Landon, Felix Hill, Nando de Freitas, and Serkan Cabi. Vision-language models as success detectors. *arXiv preprint arXiv:2303.07280*, 2023.

Louie Giray. Prompt engineering with chatgpt: a guide for academic writers. *Annals of biomedical engineering*, 51(12):2629–2633, 2023.

Simon Holk, Daniel Marta, and Iolanda Leite. Predilect: Preferences delineated with zero-shot language-based reasoning in reinforcement learning. In *2024 19th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pp. 259–268, 2024.

Borja Ibarz, Jan Leike, Tobias Pohlen, Geoffrey Irving, Shane Legg, and Dario Amodei. Reward learning from human preferences and demonstrations in atari. *Advances in neural information processing systems*, 31, 2018.

Hong Jun Jeon, Smitha Milli, and Anca Dragan. Reward-rational (implicit) choice: A unifying formalism for reward learning. *Advances in Neural Information Processing Systems*, 33:4415–4426, 2020.

Siddharth Karamcheti, Suraj Nair, Annie S Chen, Thomas Kollar, Chelsea Finn, Dorsa Sadigh, and Percy Liang. Language-driven representation learning for robotics. *arXiv preprint arXiv:2302.12766*, 2023.

Minae Kwon, Sang Michael Xie, Kalesha Bullard, and Dorsa Sadigh. Reward design with language models. *arXiv preprint arXiv:2303.00001*, 2023.

Kimin Lee, Laura Smith, and Pieter Abbeel. Pebble: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training. *arXiv preprint arXiv:2106.05091*, 2021a.

Kimin Lee, Laura Smith, Anca Dragan, and Pieter Abbeel. B-pref: Benchmarking preference-based reinforcement learning. *arXiv preprint arXiv:2111.03026*, 2021b.

Fei Liu et al. Learning to summarize from human feedback. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.

Shengcai Liu, Caishun Chen, Xinghua Qu, Ke Tang, and Yew-Soon Ong. Large language models as evolutionary optimizers. In *2024 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–8, 2024. DOI: 10.1109/CEC60901.2024.10611913.

Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. Vip: Towards universal visual reward and representation via value-implicit pre-training. *arXiv preprint arXiv:2210.00030*, 2022.

Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-level reward design via coding large language models. *arXiv preprint arXiv:2310.12931*, 2023.

Yecheng Jason Ma, William Liang, Hung-Ju Wang, Sam Wang, Yuke Zhu, Linxi Fan, Osbert Bastani, and Dinesh Jayaraman. Dreureka: Language model guided sim-to-real transfer. *arXiv preprint arXiv:2406.01967*, 2024.

Eduardo Mosqueira-Rey, Elena Hernández-Pereira, David Alonso-Ríos, José Bobes-Bascarán, and Ángel Fernández-Leal. Human-in-the-loop machine learning: a state of the art. *Artificial Intelligence Review*, 56(4):3005–3054, 2023.

Soroush Nasiriany, Fei Xia, Wenhao Yu, Ted Xiao, Jacky Liang, Ishita Dasgupta, Annie Xie, Danny Driess, Ayzaan Wahid, Zhuo Xu, et al. Pivot: Iterative visual prompting elicits actionable knowledge for vlms. *arXiv preprint arXiv:2402.07872*, 2024.

Andrew Y Ng, Stuart Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, pp. 2, 2000.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744, 2022.

Chanwoo Park, Xiangyu Liu, Asuman Ozdaglar, and Kaiqing Zhang. Do llm agents have regret? a case study in online learning and games. *arXiv preprint arXiv:2403.16843*, 2024.

Jongjin Park, Younggyo Seo, Jinwoo Shin, Honglak Lee, Pieter Abbeel, and Kimin Lee. SURF: Semi-supervised reward learning with data augmentation for feedback-efficient preference-based reinforcement learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=TfhfZLQ2EJO.

Zhenghao Mark Peng, Wenjie Mo, Chenda Duan, Quanyi Li, and Bolei Zhou. Learning from active human involvement through proxy value propagation. *Advances in neural information processing systems*, 36, 2024.

Carl Orge Retzlaff, Srijita Das, Christabel Wayllace, Payam Mousavi, Mohammad Afshari, Tianpei Yang, Anna Saranti, Alessa Angerschmid, Matthew E Taylor, and Andreas Holzinger. Human-in-the-loop reinforcement learning: A survey and position on requirements, challenges, and opportunities. *Journal of Artificial Intelligence Research*, 79:359–415, 2024.

Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M Pawan Kumar, Emilien Dupont, Francisco JR Ruiz, Jordan S Ellenberg, Pengming Wang, Omar Fawzi, et al. Mathematical discoveries from program search with large language models. *Nature*, 625(7995):468–475, 2024.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

Eric Tang, Bangding Yang, and Xingyou Song. Understanding llm embeddings for regression. *arXiv preprint arXiv:2411.14708*, 2024.

Yufei Wang, Zhanyi Sun, Jesse Zhang, Zhou Xian, Erdem Biyik, David Held, and Zackory Erickson. Rl-vlm-f: Reinforcement learning from vision language foundation model feedback. *arXiv preprint arXiv:2402.03681*, 2024.

Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, and Douglas C Schmidt. A prompt pattern catalog to enhance prompt engineering with chatgpt. *arXiv preprint arXiv:2302.11382*, 2023.

Christian Wirth, Riad Akrour, Gerhard Neumann, and Johannes Fürnkranz. A survey of preference-based reinforcement learning methods. *Journal of Machine Learning Research*, 18(136):1–46, 2017.

Jeff Wu, Long Ouyang, Daniel M Ziegler, Nisan Stiennon, Ryan Lowe, Jan Leike, and Paul Christiano. Recursively summarizing books with human feedback. *arXiv preprint arXiv:2109.10862*, 2021.

Supplementary Materials

The following content was not necessarily subject to peer review.

The supplementary submission includes a detailed appendix PDF and the accompanying codebase. The PDF contains the complete prompts, algorithm and baseline details, environment specifications, additional experimental results, human-study procedures, and representative reward-function evolution examples. The codebase provides the implementation, experiment scripts, configurations, prompt templates, and reward-synthesis resources used in the study.