FuRL: Visual-Language Models as Fuzzy Rewards for Reinforcement Learning

Yuwei Fu [1 2 †] **Haichao Zhang** [2] **Di Wu** [1] **Wei Xu** [2] **Benoit Boulet** [1]

Abstract

In this work, we investigate how to leverage pre-trained visual-language models (VLM) for online Reinforcement Learning (RL). In particular, we focus on sparse reward tasks with pre-defined textual task descriptions. We first identify the problem of reward misalignment when applying VLM as a reward in RL tasks. To address this issue, we introduce a lightweight fine-tuning method, named Fuzzy VLM reward-aided RL (FuRL), based on reward alignment and relay RL. Specifically, we enhance the performance of SAC/DrQ baseline agents on sparse reward tasks by fine-tuning VLM representations and using relay RL to avoid local minima. Extensive experiments on the Meta-world benchmark tasks demonstrate the efficacy of the proposed method. Code is available at: https://github.com/fuyw/FuRL.

1. Introduction

Deep reinforcement learning (RL) has achieved great success in many different domains, including games, robotic control, and graphics (Mnih et al., 2015; Silver et al., 2016; Haarnoja et al., 2018; Agostinelli et al., 2019; Akkaya et al., 2019; Berner et al., 2019; Kalashnikov et al., 2018; Peng et al., 2021). However, despite these great achievements, one well-known issue of RL is the large number of environmental interactions required for policy learning (Wang et al., 2017; Espeholt et al., 2018).

How to improve the sample efficiency is one of the most important topics in RL (Du et al., 2019; Zhang et al., 2020). A large body of work has been done in the community from different aspects, including better exploration strategy (Pathak et al., 2017; Zhang et al., 2022), leveraging in-house behavior data (Singh et al., 2021), using transfer learning and (or) meta-learning (Rakelly et al., 2019; Mehta et al., 2020; Agarwal et al., 2023; Beck et al., 2023), *etc.*

Recent progress on the large foundation models shows impressive results in many applications (Nair et al., 2023; Ma et al., 2023a;c; Rocamonde et al., 2023a; Chan et al., 2023). These models are useful in the sense that they contain a large amount of common knowledge, which can be used in diverse downstream tasks. One promising downstream application is to use the VLM to generate dense rewards for RL tasks with sparse rewards.

Based on these observations, we investigate how to leverage a pre-trained VLM in online RL. The topic of leveraging VLM in the form of reward in RL is an emerging field, with a few recent work on this (Mahmoudieh et al., 2022; Rocamonde et al., 2023b; Adeniji et al., 2023; Rocamonde et al., 2023a; Chan et al., 2023). We follow this line of research and study the issue of reward misalignment when using VLM-based rewards in RL, where inaccurate VLM rewards could trap the agent in local minima. To mitigate this issue, we introduce a VLM-representation fine-tuning loss and adopt relay RL (Lan et al., 2023) to improve exploration. The primary contributions of this work are as follows:

- We investigate some practical challenges of using pre-trained VLM in online RL and highlight the issue of reward misalignment.
- We introduce the Fuzzy VLM reward-aided RL (FuRL), a simple yet effective method to address the challenge brought by reward misalignment.
- We compare FuRL against different baselines and provide ablation studies to reveal the importance of addressing the fuzzy reward issue.

2. Background

2.1. Markov Decision Process (MDP) and RL

An MDP (Sutton & Barto, 2018) is commonly defined by a tuple $(\mathcal{S}, \mathcal{A}, P, r, \gamma)$, where $\mathcal{S}$, $\mathcal{A}$ denote the state space and action space. $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ denotes the transition probability between states. $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function. $\gamma \in [0, 1]$ denotes the discount factor. In the standard RL formulation, our goal is to learn a policy that

† Work done during an internship at Horizon Robotics. [1]McGill University [2]Horizon Robotics. Correspondence to: Yuwei Fu <yuwei.fu@mail.mcgill.ca>, Haichao Zhang <hczhang1@gmail.com>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

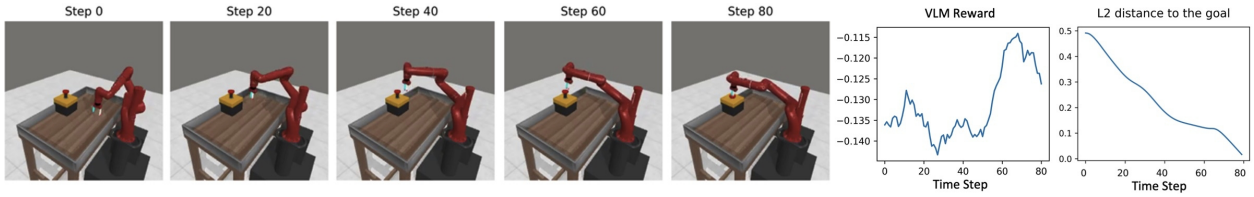

Figure 1. **Raw VLM reward is sub-optimal to teach RL agents.** In this example, the text instruction l is "press a button from the top". We plot the cosine similarity-based VLM reward with language embedding $\Phi_L(l)$ and image embedding $\Phi_I(o_t)$ and also show the distance between the end-effector and the goal. It can be observed that the cosine similarity between $\Phi_L(l)$ and $\Phi_I(o_t)$ can reflect some aspects of the task but is not always well aligned with the task progress, reflecting the fuzzy aspects of the VLM reward.

maximizes the expected accumulated discounted return. In practice, the policy is typically modelled using a neural network $\pi_\theta(a_t|s_t)$ with learnable parameters θ (Mnih et al., 2013), taking observation s_t as input and generating the action from policy $\pi_\theta(a_t|s_t)$ for each time step t.

2.2. Vision Language Models (VLM)

Vision Language Models have advanced rapidly in the past few years (Radford et al., 2021; Alayrac et al., 2022; Ma et al., 2023a). One representative work is CLIP (Radford et al., 2021), which trains the VLM by aligning image and text embedding in the latent space. CLIP has been shown to be effective in downstream tasks such as classification and also shows zero-shot transfer ability. While CLIP is a generic model motivated by vision tasks, there is also recent work on specially designed VLM for RL tasks (Ma et al., 2023a). VLM is an attractive type of models for aiding RL from different perspectives, such as reward shaping (Mahmoudieh et al., 2022; Rocamonde et al., 2023b; Adeniji et al., 2023; Rocamonde et al., 2023a; Chan et al., 2023; Lubana et al., 2023; Dang et al., 2023; Klissarov et al., 2023), task specification and success detection (Du et al., 2023) and representation (Chen et al., 2023).

3. Method

In this section, we first revisit some existing work on utilizing VLM as a reward model in RL and pinpoint the challenges therein. We then introduce the main idea and formulation of the proposed method.

3.1. VLM as Rewards Revisited

Leveraging VLM as a source of reward in RL is a popular and active emerging trend (Mahmoudieh et al., 2022; Rocamonde et al., 2023b; Adeniji et al., 2023; Rocamonde et al., 2023a; Chan et al., 2023; Lubana et al., 2023; Dang et al., 2023; Klissarov et al., 2023; Nam et al., 2023), either as a way of reward-based task specification (Mahmoudieh et al., 2022; Rocamonde et al., 2023b; Adeniji et al., 2023), or generating VLM-based reward as an additional source of supervision apart from the original task reward for RL (Rocamonde et al., 2023a; Chan et al., 2023; Lubana et al., 2023; Dang et al., 2023; Klissarov et al., 2023).

Given an observation s_t received at timestep t, the RL agent generates an action $a_t \sim \pi_\theta(a_t|s_t)$ and receives a sparse task reward r_t^{task} after a_t is executed. r_t^{task} is typically defined as $r_t^{\text{task}} = \delta_{\text{success}}$, meaning a reward of 1 is received only upon task success and otherwise the reward is 0.

This is a type of task setting commonly encountered in practice. The sparse reward makes the RL training more challenging. (Rocamonde et al., 2023a; Chan et al., 2023) propose to use *VLM as reward*, *i.e.*, augmenting the sparse task reward with another VLM reward r_t^{VLM}:

$$r_t = r_t^{\text{task}} + \rho \cdot r_t^{\text{VLM}}, \tag{1}$$

where ρ is a scalar weight parameter for balancing the VLM reward with the task reward.

Simply, these methods (Rocamonde et al., 2023a; Chan et al., 2023) add the CLIP reward, *i.e.* the cosine similarity between the language goal with an image of the latest state:

$$r_t^{\text{VLM}} \triangleq r_t^{\text{CLIP}} = \frac{\langle \Phi_L(l), \Phi_I(o_t) \rangle}{\|\Phi_L(l)\| \cdot \|\Phi_I(o_t)\|}, \tag{2}$$

to the sparse task reward. o_t is the image observation received at step t. l is the language-based task instruction issued at the beginning of the episode. Φ_L and Φ_I denote the language embedding network and image embedding network of the pre-trained CLIP model (Radford et al., 2021).

Another related set of work is using VLM as a success detector (Du et al., 2023), *i.e.*, as a sparse task reward. Apart from RL-based policy training, some recent work also proposed to use VLM-based reward for model-based planning (Ma et al., 2023a). In this case, the task reward is omitted (*i.e.* $r_t = r_t^{\text{VLM}}$) and MPC type of online planning methods are used to obtain the next action by maximizing future return.

In this work, we follow the line of research on VLM-as-reward. Same as Rocamonde et al. (2023a); Chan et al. (2023), we focus on sparse-reward tasks, and assume the access to the task instruction l (Table 4) and a goal image o_g at

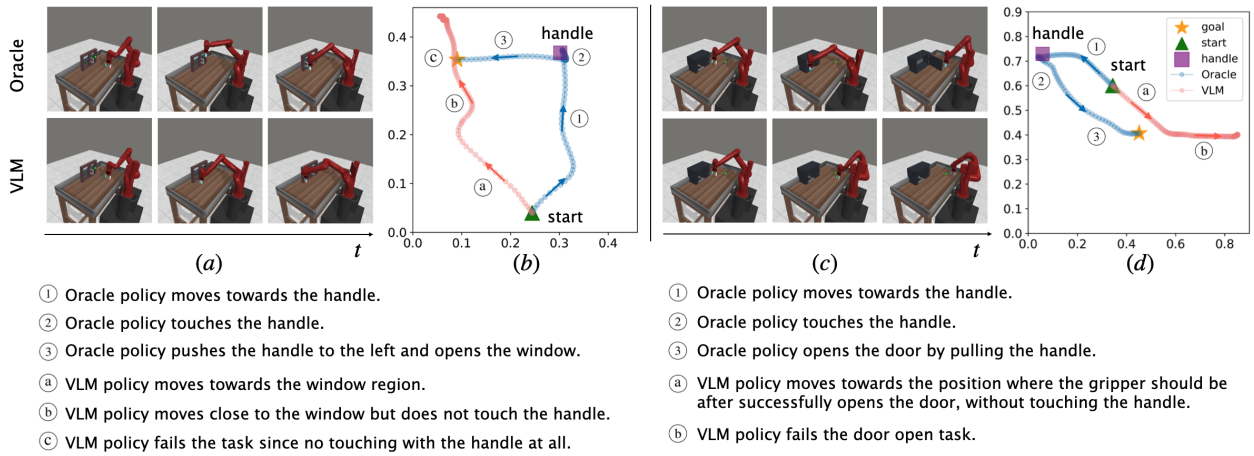

Figure 2. **Fuzzy VLM reward effect.** Visualization of end-effector trajectory in terms of (x, y) positions. *Oracle* denotes an expert policy. *VLM* denotes the policy trained using sparse task reward together with VLM reward.

the beginning of the episode. Without loss of generality, we first present results using state-based observations as policy input to disentangle the impacts of feature learning. We then increase the complexity by using pixel-based observations as policy input. For computing the VLM feature, a visual image is provided at each time step.

3.2. VLM as *Fuzzy* Rewards

Many previous methods on *VLM-as-rewards* (Rocamonde et al., 2023a; Chan et al., 2023; Mahmoudieh et al., 2022; Ma et al., 2023a) have a shared assumption that VLM-based rewards are accurate in order to achieve good policy optimization to avoid undesired solutions.

In this work, we demonstrate that *zero-shot VLM rewards are fuzzy*: meaningful in capturing the coarse semantics but inaccurate in characterizing some details. Therefore, this fuzziness in the VLM-based rewards could potentially mislead the policy optimization in the *VLM-as-reward* framework (Chan et al., 2023; Mahmoudieh et al., 2022; Ma et al., 2023a). While the degree of the reward's fuzziness can be reduced by changing different aspects of the VLM model, such as increasing its capacity, the reward's fuzziness is not likely to be eliminated due to the zero-shot nature of the VLM-as-reward framework. We carry out two sets of studies from complementary aspects to illustrate this point.

Rewards along Expert Trajectory. Figure 1 shows the VLM reward curve for an expert trajectory, where we compute the VLM reward r_t^{VLM} using Eqn. 2. In the ideal case, the reward curve should be aligned with the expert's progress, *i.e.*, higher reward when the state is closer to the task completion. However, as can be observed in Figure 1, the reward curve can reflect some aspects of task progress but is not well aligned with the task progress, reflecting the fuzzy aspect of the VLM reward.

VLM Reward Only Policy Behavior. We also trained a VLM policy only with r^{VLM}. Figure 2 illustrates the end-effector (gripper) trajectories of the robot arm in the *window-close* and *door-open* tasks from the Meta-world environment (Yu et al., 2020). We compared the trajectories generated by an oracle policy with those of a VLM policy. As depicted in Figure 2 (a-b) and Figure 2 (c-d), we can observe that the gripper is close to the window in the *window-close* task and the gripper is far away from the door in the *door-open* task at the last step, also exemplifying the effect of fuzzy VLM reward on policy behavior.

These two set of case studies show that VLM reward sometimes could provide meaningful information, *i.e.*, identifying the window in *window-close* task. However, when pre-trained VLM representations fail to capture crucial information in the target RL tasks, inaccurate VLM rewards can hinder efficient exploration. For instance, as seen in Figure 2(c-d), the robot arm got stuck at the right corner in the *door-open* task and failed to collect any successful trajectories during the training. Such inaccurate VLM rewards are mainly due to the domain shift between VLM's training dataset and the downstream target RL task (Sankaranarayanan et al., 2018; Zhang et al., 2021).

All these results indicate that the VLM-based rewards are fuzzy, *i.e.*, meaningful in some cases but could also be misleading due to their inaccuracy. This fuzzy reward issue has caught some attentions very recently (Mahmoudieh et al., 2022; Adeniji et al., 2023; Rocamonde et al., 2023a). Adeniji et al. (2023) mitigate this issue by using VLM-based reward for behavior pre-training only. Mahmoudieh et al. (2022) retrain the VLM model by using a specially tailored dataset. We instead focus on how to leverage a pre-trained VLM model in online RL and strategies to mitigate the challenges therein, as presented in the sequel.

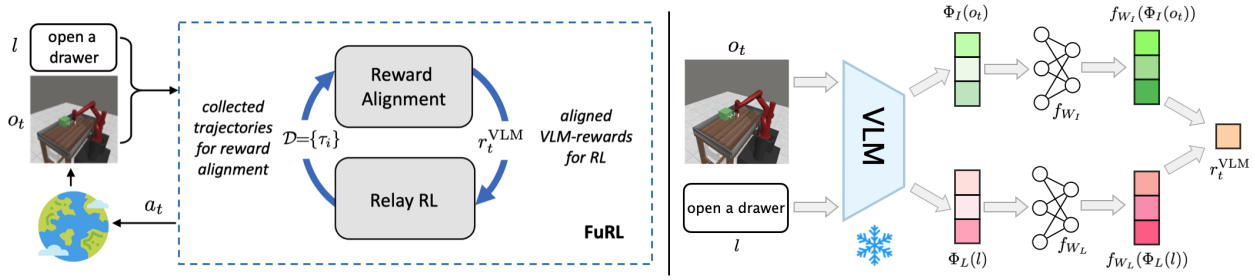

Figure 3. **Illustration of the proposed method:** (left) the overall pipeline of FuRL. (right) FuRL freezes the pre-trained VLM and only fine-tunes two MLP-based projection heads f_{W_L}, f_{W_I}.

3.3. FuRL: Fuzzy VLM rewards-aided RL

In this subsection, we introduce the *Fuzzy VLM rewards-aided RL* (FuRL), a framework that utilizes VLM rewards to facilitate learning in sparse reward tasks while addressing the inherent fuzziness of these rewards through two mechanisms: (1) reward alignment and (2) relay RL, as depicted in Figure 3 (left). These two components interact with each other in terms of exploration and learning in FuRL: ***(i)* Reward Alignment**: which fine-tunes VLM representations (generated embeddings) in a lightweight form to improve the VLM rewards, which helps exploration and policy learning; ***(ii)* Relay RL**: which helps to escape the local minima due to the fuzzy VLM rewards during exploration, and it also helps to collect more diverse data to improve the reward alignment and policy learning. We will detail these components in the following subsections respectively.

3.3.1. Reward Alignment

It is natural to understand that a VLM-based reward function, as an instance of learning-based reward function, is hard to be accurate under all kinds of input variations. Reward inaccuracy is undesirable since it could be misleading to the policy learning (Skalse et al., 2022).

With cosine similarity, inaccurate VLM rewards as defined in Eqn. 2 can be attributed to the misalignment between image and text embedding from pre-trained VLM representations. To address this issue, we introduced a lightweight alignment method as illustrated in Figure 3 (right). In particular, we freeze the pre-trained VLM and only append two small learnable networks f_{W_L} and f_{W_I} to VLM's text embedding and image embedding, respectively. In our experiments, f_{W_L} and f_{W_I} are two simple two-layer MLPs (Tolstikhin et al., 2021). Therefore, compared with fine-tuning the whole VLM model, the number of parameters to be learned in our method is much smaller.

We define the VLM reward via the cosine-similarity following (Rocamonde et al., 2023a; Chan et al., 2023) but with our projected image embedding $f_{W_I}(\Phi_I(o_t))$ and the projected text embedding $f_{W_L}(\Phi_L(l))$:

$$r_t^{\text{VLM}} \triangleq \frac{\langle f_{W_L}(\Phi_L(l)), f_{W_I}(\Phi_I(o_t))\rangle}{\|f_{W_L}(\Phi_L(l))\| \cdot \|f_{W_I}(\Phi_I(o_t))\|}. \tag{3}$$

Next, we introduce the following definition:

Definition 3.1. (*Reward Alignment*) Given a target task $\mathcal{T}$, and a reward function r, we define the process of adjusting the initially inaccurate reward function r to be more accurate in characterizing the target task $\mathcal{T}$ as reward alignment.

Since the sparse reward function r^{task} in the MDP characterizes the target task to a great extent, we will leverage the information from r^{task} for reward alignment, *i.e.*, optimizing the projection network f_{W_L} and f_{W_I}. We denote the samples from the successful trajectories τ^p as *positive* samples o^p, and the samples from the unsuccessful trajectories τ^n as *negative* samples o^n. We propose the following loss for reward alignment:

$$\mathcal{L} = \underbrace{\mathop{\mathbb{E}}_{\{o^p\in\tau^p, o^n\in\tau^n\}} \ell_\delta(o^p, o^n)}_{\mathcal{L}_{\text{pos-neg}}} + \underbrace{\mathop{\mathbb{E}}_{\{o^p_{i-k}, o^p_i\in\tau^p\}} \ell_\delta(o^p_i, o^p_{i-k})}_{\mathcal{L}_{\text{pos-pos}}}, \tag{4}$$

where $\ell_\delta(o_p, o_n) \triangleq \max(0, r^{\text{VLM}}(o_n) - r^{\text{VLM}}(o_p) + \delta)$ is a ranking loss with a margin of $\delta \in \mathbb{R}^+$, generating a loss if $r^{\text{VLM}}(o_n) + \delta$ is larger than $r^{\text{VLM}}(o_p)$. $\mathcal{L}_{\text{pos-neg}}$ learns to generate a higher VLM reward for a positive sample than that of a negative sample. $\mathcal{L}_{\text{pos-pos}}$ learns to rank two samples from the same successful trajectory, giving samples later in time a higher rank (larger score) since it is closer to the task success. Here, k is a window size parameter.

Moreover, since successful trajectories are unavailable in the beginning of the training, an optional step can be used when an additional goal image o_g is available before encountering any successful trajectories, *i.e.*, learning from all *negative* samples with zero task rewards:

$$\mathcal{L}_{\text{neg-neg}} = \mathop{\mathbb{E}}_{\{o^n_i, o^n_j\in\tau^n L_2(o^n_i, o_g) < L_2(o^n_j, o_g) - \delta'\}} \ell_\delta(o^n_i, o^n_j), \tag{5}$$

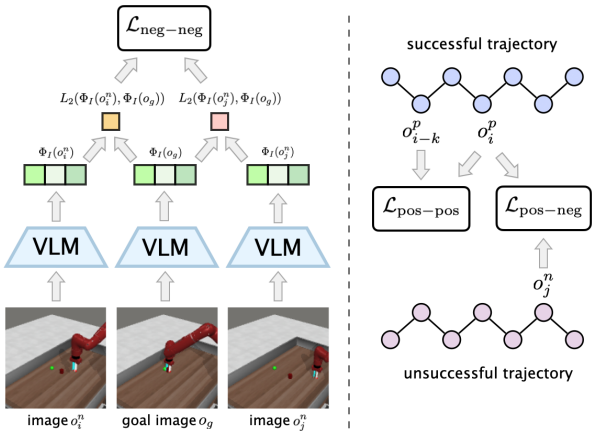

Figure 4. **Contrastive learning loss**: (left) without any successful trajectories, we can use L2 distance w.r.t. an goal image to rank the goodness of two negative samples; (right) when we collected some successful trajectories, the contrastive loss learns to distinguish samples from both of the successful and unsuccessful trajectories.

with $L_2(o, o_g) \triangleq \|\Phi_I(o) - \Phi_I(o_g)\|_2$. This essentially uses the distance w.r.t. the goal image o_g in the image embedding space to rank the goodness of samples, ranking samples with smaller distance higher (Figure 4). The main purpose of Eqn. 5 is to accelerate the learning to find the first successful trajectory earlier. In practice, we can also replace $\mathcal{L}_{\text{neg-neg}}$ by using parallel agents or exploration intrinsic reward to search for the first successful trajectory. We leave the exploration of this as a future work.

Algorithm 1 Fuzzy VLM rewards aided RL (FuRL)

Input: pre-trained VLM Φ_I and Φ_L, goal image o_g, task language goal l, relay steps $T_s = [T_1, \cdots, T_n]$, shared replay buffer $\mathcal{D}_{\text{shared}}$, total trajectory number N.
Output: trained VLM agent π_{VLM}.
Initialize: image projection head f_{W_I}, language projection head f_{W_L}, VLM agent π_{VLM}, SAC agent π_{SAC}.
for $i = 1$ **to** N **do**
  if Collected positive samples **then**
    Unroll VLM policy π_{VLM}.
    Update f_{W_I} and f_{W_L} using Eqn. 4.
    Update π_{VLM} using $r^{\text{task}} + \rho r^{\text{VLM}}$.
  else
    Sample a replay step T_i from T_s.
    Iteratively unroll π_{VLM} and π_{SAC} for T_i steps.
    Update f_{W_I} and f_{W_L} using Eqn. 5.
    Update π_{SAC} using r^{task}.
    Update π_{VLM} using $r^{\text{task}} + \rho r^{\text{VLM}}$.
  end if
end for

3.3.2. Relay RL

As previously mentioned, one notable challenge in VLM reward alignment is how to find the first successful trajectory earlier. When the current VLM policy is trapped in local minima due to the inaccurate VLM rewards, as shown in Figure 2, it is likely that the agent fails to collect any successful trajectories. Under such circumstances, Eqn. 4 is never triggered as we have no positive samples.

Given this observation, we introduce a simple exploration strategy based on the relay RL (Gupta et al., 2020; Lan et al., 2023) to mitigate this representative issue caused by fuzzy VLM reward. More specifically, we maintained an extra SAC agent π_{SAC} besides the current VLM agent π_{VLM}. At the beginning of an episode τ_i, we first randomly select a relay step T_i from some pre-defined values or a specified range. We then iteratively unroll π_{SAC} and π_{VLM} for T_i steps until the end of the trajectory, as shown in Figure 5. The collected samples are added to a shared buffer, which we use to train π_{VLM} with $r_t^{\text{task}} + \rho r_t^{\text{VLM}}$ and π_{SAC} with r_t^{task}. The evaluation is done on the VLM agent.

The motivation of the relay RL is to let the SAC agent help to escape the local minima once the VLM agent gets stuck. On the other hand, relay RL also helps to increase the data diversity by starting π_{VLM} and π_{SAC} from different initial states. Generally, starting with π_{VLM} forms a curriculum learning for the SAC agent as in Jump-start RL (Uchendu et al., 2023). Once we have collected some successful trajectories, we can turn off the relay RL and focus on collecting samples with the VLM policy π_{VLM}. The pseudo-code of FuRL is summarized in the Algorithm 1.

4. Related Work

4.1. RL with VLM

As one specific type of foundation models, VLM connects language with visual signals and has been playing an important role in the fields that involves both modalities such as visual question answering (Antol et al., 2015; Das et al., 2018; Zhang et al., 2023a). VLM has been used in RL in various ways. It has been used as a reward function (Mahmoudieh et al., 2022; Rocamonde et al., 2023b; Adeniji et al., 2023; Rocamonde et al., 2023a; Chan et al., 2023; Lubana et al., 2023; Dang et al., 2023; Klissarov et al., 2023; Nam et al., 2023), as revisited in detail in Section 3.1. Sontakke et al. (2023) also used VLM for reward computation but requires additional expert demonstrations. Sumers et al. (2023) used a generative VLM for hindsight relabeling-based data augmentation to improve dataset diversity.

Apart from this, VLM has also been used in other ways such as a promptable representation learner (Chen et al., 2023). In this work, we focus on the VLM-as-reward setting. In

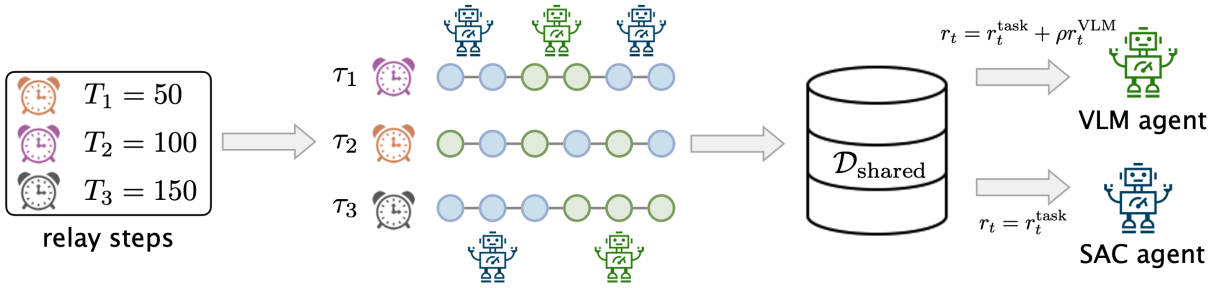

Figure 5. **Illustration of the relay RL based exploration:** at the beginning of an episode τ_i, we randomly select a relay step T_i. We iteratively unroll a VLM agent and a SAC agent for T_i steps and save the collected samples in a shared buffer. We turn off the relay exploration when we collected 2500 positive samples from the successful trajectories.

Table 1. **Experiment results on the MT10 benchmark with sparse reward and fixed goal.** We report the average success rate P (%) in the evaluation at the last timestep across 5 random seeds after training.

Environment		SAC	VLM-RMs	VLM-RMs-GB	LIV	LIV-Proj	Relay	FuRL w/o goal-image	FuRL
	r^{VLM}	✗	✓	✓	✓	✓	✓	✓	✓
	r^{task}	✓	✗	✗	✓	✓	✓	✓	✓
button-press-topdown-v2		0	0	0	0	0	60	80	100
door-open-v2		50	0	0	0	0	80	100	100
drawer-close-v2		100	0	0	100	100	100	100	100
drawer-open-v2		20	0	0	0	0	40	80	80
peg-insert-side-v2		0	0	0	0	0	0	0	0
pick-place-v2		0	0	0	0	0	0	0	0
push-v2		0	0	0	0	0	0	40	80
reach-v2		60	0	0	80	80	100	100	100
window-close-v2		60	0	0	60	40	80	100	100
window-open-v2		80	0	0	40	20	80	100	100
average		37.0	0.0	0.0	28.0	24.0	54.0	70.0	**76.0**

terms of training, VLM can be trained in a general way, without being tailored to the downstream tasks. Recently, robotics/RL oriented VLM are emerging (Gao et al., 2023; Ma et al., 2023a). The backbone VLM of this work is base on one of such a model (Ma et al., 2023a).

4.2. RL with Foundation Models

Our work falls within the broader category of leveraging foundation models in RL, which is an active field with many exciting advances (Lubana et al., 2023; Nam et al., 2023; Wang et al., 2023; Hu et al., 2023). Apart from VLM, other forms of foundation models such as large language models (LLM) have also been used in many different ways, including planning (Huang et al., 2022a;b), task decomposing with grounding (Ahn et al., 2022; Huang et al., 2023; Zhang et al., 2023b), generating code as policy/skill (Liang et al., 2023), reward design (Yu et al., 2023; Ma et al., 2023b) *etc.*. In this paper, we focus on a complementary perspective by highlighting the potential issues of using a pre-trained VLM in RL and proposing practical remedies.

5. Experiments

In this section, we focus on the following questions: (1) How does the proposed FuRL perform compared to other baselines? (2) Is FuRL effective with pixel-based observations? (3) Can FuRL generalize to other VLM backbone models? (4) Are both the reward alignment and relay RL components useful? (5) What is the influence of the VLM reward weight parameter ρ?

5.1. Baselines

To validate the efficacy of the proposed method, we compare the proposed FuRL to the following baselines:

1. SAC: a state-based SAC agent (Haarnoja et al., 2018) using the sparse binary task reward r_t^{task}.
2. VLM-RMs (Rocamonde et al., 2023a): a recent baseline which only uses the cosine similarity based VLM reward without the task reward.
3. VLM-RMs-GB (Rocamonde et al., 2023a): a variant of VLM-RMs which adds a goal-baseline regularization.

4. LIV (Ma et al., 2023a): a state-based SAC agent trained with task reward and dense LIV reward as in Eqn. 1.

5. LIV-Proj: similar to LIV baseline but with the LIV reward computed as in Eqn. 3 using randomly initialized and fixed f_{W_L} and f_{W_I}.

6. Relay (Lan et al., 2023): a simplified version of FuRL where we incorporated relay into LIV baseline.

7. FuRL-without-image: a variant of FuRL that does not use the goal image and starts to fine-tune only after collecting the first successful trajectory.

5.2. Experiment Settings

We use ten robotics tasks from the Meta-world MT10 environment (Yu et al., 2020) with state-based observations and sparse rewards (referred to as *Sparse Meta-world Tasks*). In each task, the RL agent only receives reward 1 when it reaches the goal and otherwise the reward is 0. We use SAC (Haarnoja et al., 2018) as the backbone RL agent, and the total environmental step is 1e6. We use the Adam optimizer with a learning rate of 0.0001. The VLM reward weight ρ is 0.05. For the VLM model, we use the pre-trained LIV (Ma et al., 2023a) from the official implementation. We provide more detailed information in the Appendix B. More results and resources are available on the project page. [1]

5.3. Results on Sparse MT10

We first validate the effectiveness of FuRL on the fixed-goal MT10 benchmark (Yu et al., 2020). Experiment results are shown in the Table 1 and Figure 6. We report the average and standard deviation of the success rate in the evaluation across five random seeds. We can observe that FuRL and FuRL-without-image generally outperform the other baselines in most tasks. In addition, none of these methods is able to solve the *peg-insert-side* and *pick-place* tasks. The main reason is that these two tasks require the agent to master multiple subtasks, *i.e.*, grabbing an item and then moving to a target position, which is highly challenging under a sparse reward setting. All the methods struggle on these tasks including those with VLM rewards. This is a common weakness within the existing VLM as reward framework and how to go beyond to address this issue is an interesting future direction.

From Table 1, we can also observe that the VLM-RMs and VLM-RMs-GB fail to solve any tasks. This is not surprising since no task reward is used in VLM-RMs and VLM-RMs-GB, and purely the VLM reward is used. This exemplifies that it is hard to only rely on the zero-shot VLM rewards since there is no guarantee on the reward alignment. Moreover, the lower performance of LIV compared to SAC further illustrates the fuzzy reward effect, showing that naively using VLM rewards in online RL can perform worse than the SAC baseline, leading to policies getting stuck in local minima as shown in Figure 2. The better performance of FuRL compared to Relay proves the benefits of reward alignment in mitigating the issue of fuzzy VLM rewards.

[1] https://sites.google.com/site/hczhang1/projects/furl

Table 2. **Experiment on Sparse MT10 with random goals.**

Environment	SAC	Relay	FuRL
button-press-topdown-v2	16.0 (32.0)	56.0 (38.3)	64.0 (32.6)
door-open-v2	78.0 (39.2)	80.0 (30.3)	96.0 (8.0)
drawer-close-v2	100.0 (0.0)	100.0 (0.0)	100.0 (0.0)
drawer-open-v2	40.0 (49.0)	50.0 (42.0)	84.0 (27.3)
pick-place-v2	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)
peg-insert-side-v2	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)
push-v2	0.0 (0.0)	0.0 (0.0)	6.0 (8.0)
reach-v2	100.0 (0.0)	100.0 (0.0)	100.0 (0.0)
window-close-v2	86.0 (28.0)	96.0 (4.9)	100.0 (0.0)
window-open-v2	78.0 (39.2)	92.0 (7.5)	96.0 (4.9)
average	49.8 (7.9)	57.4 (7.0)	**64.6 (5.0)**

We also evaluate the proposed FuRL on the random-goal MT10 benchmark, where the goal position changes in each trajectory. We compare FuRL with the SAC and Relay baselines in the Table 2. We can observe that FuRL also outperforms other baselines, which indicates that FuRL is able to generalize well with different goal positions.

5.4. Results with Pixel-based Observations

We further evaluate the proposed method with pixel-based observations. More specifically, we adopt the DrQ (Yarats et al., 2021; 2022) as the backbone RL agent. We mainly follow the settings from the DrQ to use an image size of 84, stacking frame of 3 and action repeat of 2. Figure 7 shows the results on the *window-open* tasks with random goals. We can observe that the evaluation curve of FuRL is quite similar to the result in the state-based experiments, which learns much faster than the DrQ baseline.

5.5. Ablation Studies

We conducted ablation studies to validate the efficacy of the different components in FuRL. We used the average success rate and the average Area Under the Curve (AUC) as evaluation metrics. To facilitate comparison, we normalized both metrics with respect to the FuRL results.

The Impact of Reward Alignment. We first validate the effectiveness of the proposed reward alignment loss functions in Figure 8. The *no Reward-align* baseline is a variant of FuRL without reward alignment. We can observe that *no Reward-align* baseline consistently underperforms FuRL,

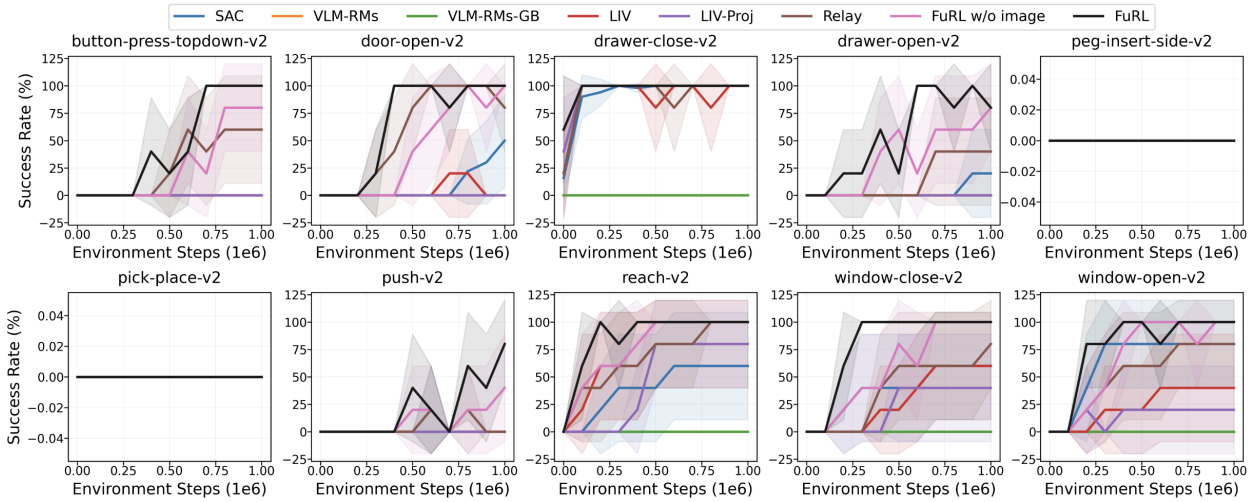

Figure 6. **Evaluation curves on the Sparse Meta-world tasks**: FuRL generally outperforms the other baselines.

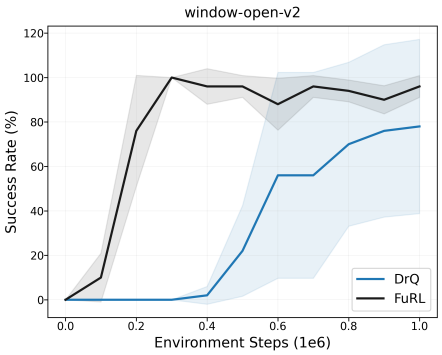

Figure 7. **Results on the window-open task with pixel-based observations**: FuRL is also effective with pixel-based inputs.

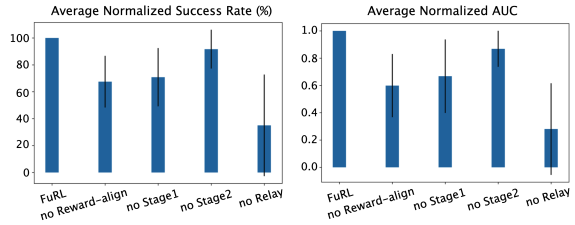

Figure 8. **Ablations.** Reward alignment is important and leads to better performance (*FuRL* v.s. *no Reward-align*). Within the reward alignment part, both Stage 1 (Eqn. 5) and Stage 2 (Eqn. 4) have contributions. Relay RL is crucial for tasks where the exploration is hard, *i.e.*, the VLM policy is prone to get stuck in local minima due to inaccurate VLM rewards.

which demonstrates the efficacy of the reward alignment component. Furthermore, we refer *no Stage1* to fine-tuning with only Eqn. 4, and we refer *no Stage2* to fine-tuning with only Eqn. 5. We can observe that removing any of Stage1 or Stage2 will lead to performance degradation, *i.e.*, slower convergence and (or) higher variance. This shows that both loss functions in the reward alignment component are effective, where the Eqn. 4 helps to mitigate the representation misalignment issue, and Eqn. 5 helps to find the first successful trajectory earlier.

The Impact of Relay RL. From Figure 8, we can also observe that the relay RL plays a crucial role in FuRL. Without relay RL, the agent completely failed in the some tasks, *i.e.*, *button-press-topdown* and *door-open* tasks. This is because the reward alignment loss function Eqn. 4 is triggered after we collected the first successful trajectory. The relay RL addresses the challenge of getting stuck in local minima when we only have negative samples.

The Impact of VLM Embedding. Figure 9 shows the results of training a FuRL agent without the pre-trained VLM-representation. We can observe that the variant without VLM-representation performs worse than FuRL and performs better than the SAC baseline. This verifies the benefits of using the VLM model and also the effectiveness of the proposed reward alignment objective.

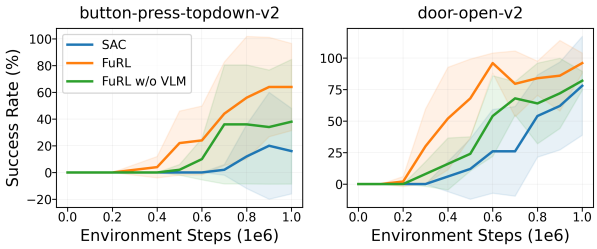

Figure 9. **Impact of VLM-representation.**

Using CLIP as the VLM model. In the previous experiments, we instantiate the VLM model as a pre-trained LIV

Table 3. **Experiment of using CLIP as the VLM.**

Environment	CLIP	CLIP-FuRL
button-press-topdown-v2	0 (0)	80 (40)
door-open-v2	60 (49)	100 (0)
drawer-close-v2	100 (0)	100 (0)
drawer-open-v2	0 (0)	80 (40)
peg-insert-side-v2	0 (0)	0 (0)
pick-place-v2	0 (0)	0 (0)
push-v2	0 (0)	60 (49)
reach-v2	80 (40)	100 (0)
window-close-v2	60 (49)	100 (0)
window-open-v2	80 (40)	80 (40)
average	38 (9.8)	**70 (6.3)**

model (Ma et al., 2023a). In this subsection, we evaluate the proposed FuRL by instantiating the VLM model as a pre-trained CLIP (Radford et al., 2021). Results on MT10 with fixed-goal are shown in Table 3. The CLIP baseline is a SAC agent that learns with task reward and dense CLIP reward. We can observe that the CLIP-based FuRL also outperforms the CLIP baseline, which shows that the proposed method can generalize to different VLM base models.

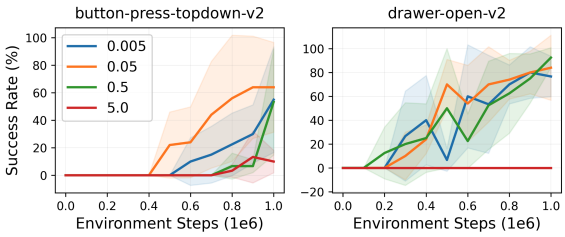

Figure 10. **Impact of parameter** ρ.

The Impact of ρ**.** Figure 10 shows the results of using different ρ values on the *drawer-open* and *button-press-topdown* tasks with random goals. We can observe that a large value usually performs poorly, where the inaccurate VLM reward distracts the agent from learning from the task reward information. On the other hand, a small value might lead to slow learning or larger variance.

5.6. Visualization of VLM Rewards

Figure 11 shows the VLM rewards before (LIV) and after alignment (FuRL) for the trajectory shown in Figure 1. Compared with the pre-trained LIV reward curve, the FuRL VLM reward curve generally has a larger value when it is closer to the goal. The reason that the reward does not reach 1 after alignment is because a ranking loss is used in Eqn. 4, which focuses on the relative ranking instead of absolute reward values. The accurate relative trend in the aligned reward from FuRL is already effective in aiding the RL agent in exploration and learning.

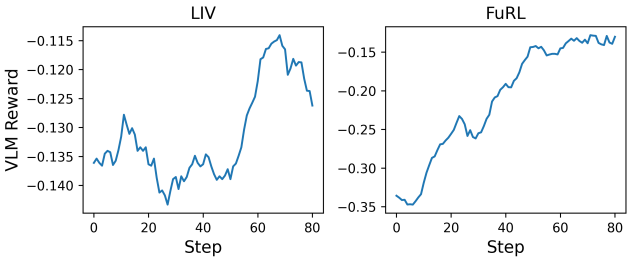

Figure 11. **VLM reward curves** before (left) and after (right) fine-tuning, along the same expert trajectory. Some frames along the trajectory are shown in Figure 1. FuRL reward has a larger value when it is closer to the goal.

6. Conclusion and Future Work

In this work, we highlighted the fuzzy reward issue in applying VLM reward in online RL and further proposed the FuRL approach, which mitigated the fuzzy reward issue with reward alignment and relay RL collaboratively. Various experiments demonstrated the effectiveness of the proposed method. In the current work, we have demonstrated the approach on a number of tasks, with the reward alignment networks trained with data from a single task. An interesting future direction is to train the reward alignment module across multiple task jointly. Another point is that we have used simple language descriptions following Rocamonde et al. (2023a); Ma et al. (2023a). Applying the proposed approach to more complex compositional language instructions is also an interesting direction to pursue in the future. From a broader perspective, the hallucination issue of VLMs and large language models (LLMs) (Li et al., 2023; Chakraborty et al., 2024) has greatly impacted their applicability in downstream tasks. It is an interesting future work to generalize some of the ideas in this work together with other techniques such as adversarial learning (Ilyas et al., 2019; Zhang & Wang, 2019; Yao et al., 2023) to a broader context with pre-trained foundations models.

Impact Statement

Since our work is a combination of VLM and RL, one potential negative societal impact could be the improper language instructions. When we apply the proposed method to real-world applications, the usage of some language instructions might lead to dangerous behaviours. To mitigate this issue, we could adopt some rule-based keyword blacklists to filter dangerous language instructions, or we could further fine-tune the trained policy to learn some safety knowledge.

Acknowledgements

We would like to thank the anonymous reviewers for their efforts in reviewing our work and their constructive comments that help to further improve the quality of this paper.

References

Adeniji, A., Xie, A., Sferrazza, C., Seo, Y., James, S., and Abbeel, P. Language reward modulation for pretraining reinforcement learning. *arXiv:2308.12270*, 2023.

Agarwal, A., Song, Y., Sun, W., Wang, K., Wang, M., and Zhang, X. Provable benefits of representational transfer in reinforcement learning. In *The Thirty Sixth Annual Conference on Learning Theory*, pp. 2114–2187. PMLR, 2023.

Agostinelli, F., McAleer, S., Shmakov, A., and Baldi, P. Solving the rubik's cube with deep reinforcement learning and search. *Nature Machine Intelligence*, 1:356–363, 2019.

Ahn, M., Brohan, A., Brown, N., Chebotar, Y., Cortes, O., David, B., Finn, C., Fu, C., Gopalakrishnan, K., Hausman, K., Herzog, A., Ho, D., Hsu, J., Ibarz, J., Ichter, B., Irpan, A., Jang, E., Ruano, R. J., Jeffrey, K., Jesmonth, S., Joshi, N. J., Julian, R., Kalashnikov, D., Kuang, Y., Lee, K.-H., Levine, S., Lu, Y., Luu, L., Parada, C., Pastor, P., Quiambao, J., Rao, K., Rettinghouse, J., Reyes, D., Sermanet, P., Sievers, N., Tan, C., Toshev, A., Vanhoucke, V., Xia, F., Xiao, T., Xu, P., Xu, S., Yan, M., and Zeng, A. Do as I can, not as I say: Grounding language in robotic affordances. In *Conference on Robot Learning*, 2022.

Akkaya, I., Andrychowicz, M., Chociej, M., Litwin, M., McGrew, B., Petron, A., Paino, A., Plappert, M., Powell, G., Ribas, R., Schneider, J., Tezak, N., Tworek, J., Welinder, P., Weng, L., Yuan, Q., Zaremba, W., and Zhang, L. Solving rubik's cube with a robot hand. *CoRR*, abs/1910.07113, 2019.

Alayrac, J.-B., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., Lenc, K., Mensch, A., Millican, K., Reynolds, M., Ring, R., Rutherford, E., Cabi, S., Han, T., Gong, Z., Samangooei, S., Monteiro, M., Menick, J. L., Borgeaud, S., Brock, A., Nematzadeh, A., Sharifzadeh, S., Bińkowski, M. a., Barreira, R., Vinyals, O., Zisserman, A., and Simonyan, K. Flamingo: a visual language model for few-shot learning. In *Advances in Neural Information Processing Systems*, 2022.

Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Zitnick, C. L., and Parikh, D. VQA: Visual question answering. In *IEEE International Conference on Computer Vision*, 2015.

Beck, J., Jackson, M. T., Vuorio, R., and Whiteson, S. Hypernetworks in meta-reinforcement learning. In *Conference on Robot Learning*, pp. 1478–1487. PMLR, 2023.

Berner, C., Brockman, G., Chan, B., Cheung, V., Debiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C., Józefowicz, R., Gray, S., Olsson, C., Pachocki, J. W., Petrov, M., de Oliveira Pinto, H. P., Raiman, J., Salimans, T., Schlatter, J., Schneider, J., Sidor, S., Sutskever, I., Tang, J., Wolski, F., and Zhang, S. Dota 2 with large scale deep reinforcement learning. *ArXiv*, 2019.

Chakraborty, N., Ornik, M., and Driggs-Campbell, K. Hallucination detection in foundation models for decision-making: A flexible definition and review of the state of the art. *arXiv:2403.16527*, 2024.

Chan, H., Mnih, V., Behbahani, F., Laskin, M., Wang, L., Pardo, F., Gazeau, M., Sahni, H., Horgan, D., Baumli, K., Schroecker, Y., Spencer, S., Steigerwald, R., Quan, J., Comanici, G., Flennerhag, S., Neitz, A., Zhang, L. M., Schaul, T., Singh, S., Lyle, C., Rocktäschel, T., Parker-Holder, J., and Holsheimer, K. Vision-language models as a source of rewards. In *Second Agent Learning in Open-Endedness Workshop*, 2023.

Chen, W., Mees, O., Kumar, A., and Levine, S. Vision-language models provide promptable representations for reinforcement learning. In *NeurIPS 2023 Foundation Models for Decision Making Workshop*, 2023.

Dang, X., Edelkamp, S., and Ribault, N. CLIP-Motion: Learning reward functions for robotic actions using consecutive observations. *arXiv:2311.03485*, 2023.

Das, A., Datta, S., Gkioxari, G., Lee, S., Parikh, D., and Batra, D. Embodied question answering. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

Du, S., Krishnamurthy, A., Jiang, N., Agarwal, A., Dudik, M., and Langford, J. Provably efficient RL with rich observations via latent state decoding. In *International Conference on Machine Learning*, 2019.

Du, Y., Konyushkova, K., Denil, M., Raju, A., Landon, J., Hill, F., de Freitas, N., and Cabi, S. Vision-language models as success detectors. *arXiv:2303.07280*, 2023.

Espeholt, L., Soyer, H., Munos, R., Simonyan, K., Mnih, V., Ward, T., Doron, Y., Firoiu, V., Harley, T., Dunning, I., et al. IMPALA: Scalable distributed deep-RL with importance weighted actor-learner architectures. In *International conference on machine learning*, 2018.

Frostig, R., Johnson, M. J., and Leary, C. Compiling machine learning programs via high-level tracing. *Systems for Machine Learning*, pp. 23–24, 2018.

Gao, J., Sarkar, B., Xia, F., Xiao, T., Wu, J., Ichter, B., Majumdar, A., and Sadigh, D. Physically grounded vision-language models for robotic manipulation. *arXiv:2309.02561*, 2023.

Gupta, A., Kumar, V., Lynch, C., Levine, S., and Hausman, K. Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning. In *Conference on Robot Learning*, 2020.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, 2018.

Hu, Y., Xie, Q., Jain, V., Francis, J., Patrikar, J., Keetha, N., Kim, S., Xie, Y., Zhang, T., Zhao, S., Chong, Y. Q., Wang, C., Sycara, K., Johnson-Roberson, M., Batra, D., Wang, X., Scherer, S., Kira, Z., Xia, F., and Bisk, Y. Toward general-purpose robots via foundation models: A survey and meta-analysis. *arXiv:2312.08782*, 2023.

Huang, W., Abbeel, P., Pathak, D., and Mordatch, I. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International Conference on Machine Learning*, 2022a.

Huang, W., Xia, F., Xiao, T., Chan, H., Liang, J., Florence, P., Zeng, A., Tompson, J., Mordatch, I., Chebotar, Y., Sermanet, P., Brown, N., Jackson, T., Luu, L., Levine, S., Hausman, K., and Ichter, B. Inner monologue: Embodied reasoning through planning with language models, 2022b.

Huang, W., Xia, F., Shah, D., Driess, D., Zeng, A., Lu, Y., Florence, P., Mordatch, I., Levine, S., Hausman, K., and brian ichter. Grounded decoding: Guiding text generation with grounded models for embodied agents. In *Advances in Neural Information Processing Systems*, 2023.

Ilyas, A., Santurkar, S., Tsipras, D., Engstrom, L., Tran, B., and Madry, A. Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems*, 2019.

Imambi, S., Prakash, K. B., and Kanagachidambaresan, G. Pytorch. *Programming with TensorFlow: Solution for Edge Computing Applications*, pp. 87–104, 2021.

Kalashnikov, D., Irpan, A., Pastor, P., Ibarz, J., Herzog, A., Jang, E., Quillen, D., Holly, E., Kalakrishnan, M., Vanhoucke, V., and Levine, S. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning*, 2018.

Klissarov, M., D'Oro, P., Sodhani, S., Raileanu, R., Bacon, P.-L., Vincent, P., Zhang, A., and Henaff, M. Motif: Intrinsic motivation from artificial intelligence feedback. *arXiv:2310.00166*, 2023.

Kostrikov, I., Nair, A., and Levine, S. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2022.

Lan, L.-C., Zhang, H., and Hsieh, C.-J. Can agents run relay race with strangers? generalization of RL to out-of-distribution trajectories. In *International Conference on Learning Representations*, 2023.

Li, Y., Du, Y., Zhou, K., Wang, J., Zhao, W. X., and Wen, J.-R. Evaluating object hallucination in large vision-language models. *arXiv:2305.10355*, 2023.

Liang, J., Huang, W., Xia, F., Xu, P., Hausman, K., Ichter, B., Florence, P., and Zeng, A. Code as policies: Language model programs for embodied control. *arXiv:2209.07753*, 2023.

Lubana, E. S., Brehmer, J., de Haan, P., and Cohen, T. FoMo rewards: Can we cast foundation models as reward functions? *arXiv:2312.03881*, 2023.

Ma, Y. J., Liang, W., Som, V., Kumar, V., Zhang, A., Bastani, O., and Jayaraman, D. LIV: Language-image representations and rewards for robotic control. *arXiv:2306.00958*, 2023a.

Ma, Y. J., Liang, W., Wang, G., Huang, D.-A., Bastani, O., Jayaraman, D., Zhu, Y., Fan, L., and Anandkumar, A. Eureka: Human-level reward design via coding large language models. *arXiv:2310.12931*, 2023b.

Ma, Y. J., Sodhani, S., Jayaraman, D., Bastani, O., Kumar, V., and Zhang, A. VIP: Towards universal visual reward and representation via value-implicit pre-training. In *International Conference on Learning Representations*, 2023c.

Mahmoudieh, P., Pathak, D., and Darrell, T. Zero-shot reward specification via grounded natural language. In *International Conference on Machine Learning*, 2022.

Mehta, B., Diaz, M., Golemo, F., Pal, C. J., and Paull, L. Active domain randomization. In *Conference on Robot Learning*, 2020.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D. Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529–533, 2015.

Nair, S., Rajeswaran, A., Kumar, V., Finn, C., and Gupta, A. R3M: A universal visual representation for robot manipulation. In *Conference on Robot Learning*, 2023.

Nam, T., Lee, J., Zhang, J., Hwang, S. J., Lim, J. J., and Pertsch, K. Lift: Unsupervised reinforcement learning with foundation models as teachers. *arXiv:2312.08958*, 2023.

Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning*, 2017.

Peng, X. B., Ma, Z., Abbeel, P., Levine, S., and Kanazawa, A. AMP: adversarial motion priors for stylized physics-based character control. *CoRR*, abs/2104.02180, 2021.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021.

Rakelly, K., Zhou, A., Finn, C., Levine, S., and Quillen, D. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *International conference on machine learning*, 2019.

Rocamonde, J., Montesinos, V., Nava, E., Perez, E., and Lindner, D. Vision-language models are zero-shot reward models for reinforcement learning. *arXiv:2310.12921*, 2023a.

Rocamonde, J., Montesinos, V., Nava, E., Perez, E., and Lindner, D. Vision-language models are zero-shot reward models for reinforcement learning. *arXiv:2310.12921*, 2023b.

Sankaranarayanan, S., Balaji, Y., Jain, A., Lim, S. N., and Chellappa, R. Learning from synthetic data: Addressing domain shift for semantic segmentation. In *IEEE Conference on Computer Vision and pPttern Recognition*, 2018.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

Singh, A., Liu, H., Zhou, G., Yu, A., Rhinehart, N., and Levine, S. Parrot: Data-driven behavioral priors for reinforcement learning. In *International Conference on Learning Representations*, 2021.

Skalse, J., Howe, N. H. R., Krasheninnikov, D., and Krueger, D. Defining and characterizing reward hacking. *arXiv:2209.13085*, 2022.

Sodhani, S., Zhang, A., and Pineau, J. Multi-task reinforcement learning with context-based representations. In *International Conference on Machine Learning*, 2021.

Sontakke, S. A., Arnold, S., Zhang, J., Pertsch, K., Biyik, E., Sadigh, D., Finn, C., and Itti, L. Roboclip: One demonstration is enough to learn robot policies. In *Advances in Neural Information Processing Systems*, 2023.

Sumers, T., Marino, K., Ahuja, A., Fergus, R., and Dasgupta, I. Distilling internet-scale vision-language models into embodied agents. In *International Conference on Machine Learning*, 2023.

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*. MIT press, 2018.

Tolstikhin, I. O., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Yung, J., Steiner, A., Keysers, D., Uszkoreit, J., et al. Mlp-mixer: An all-mlp architecture for vision. *Advances in neural information processing systems*, 34:24261–24272, 2021.

Uchendu, I., Xiao, T., Lu, Y., Zhu, B., Yan, M., Simon, J., Bennice, M., Fu, C., Ma, C., Jiao, J., et al. Jump-start reinforcement learning. In *International Conference on Machine Learning*, 2023.

Wang, G., Xie, Y., Jiang, Y., Mandlekar, A., Xiao, C., Zhu, Y., Fan, L., and Anandkumar, A. Voyager: An open-ended embodied agent with large language models. *arXiv:2305.16291*, 2023.

Wang, Z., Bapst, V., Heess, N., Mnih, V., Munos, R., Kavukcuoglu, K., and de Freitas, N. Sample efficient actor-critic with experience replay. In *International Conference on Learning Representations*, 2017.

Yao, J.-Y., Ning, K.-P., Liu, Z.-H., Ning, M.-N., and Yuan, L. LLM lies: Hallucinations are not bugs, but features as adversarial examples. *arXiv:2310.01469*, 2023.

Yarats, D., Kostrikov, I., and Fergus, R. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *International Conference on Learning Representations*, 2021.

Yarats, D., Fergus, R., Lazaric, A., and Pinto, L. Mastering visual continuous control: Improved data-augmented reinforcement learning. In *International Conference on Learning Representations*, 2022.

Yu, T., Quillen, D., He, Z., Julian, R., Hausman, K., Finn, C., and Levine, S. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning*, 2020.

Yu, W., Gileadi, N., Fu, C., Kirmani, S., Lee, K.-H., Arenas, M. G., Chiang, H.-T. L., Erez, T., Hasenclever, L., Humplik, J., Ichter, B., Xiao, T., Xu, P., Zeng, A., Zhang, T., Heess, N., Sadigh, D., Tan, J., Tassa, Y., and Xia, F. Language to rewards for robotic skill synthesis. In *Conference on Robot Learning*, 2023.

Zhang, H. and Wang, J. Defense against adversarial attacks using feature scattering-based adversarial training. In *Advances in Neural Information Processing Systems*, 2019.

Zhang, H., Xu, W., and Yu, H. Generative planning for temporally coordinated exploration in reinforcement learning. In *International Conference on Learning Representations*, 2022.

Zhang, J., Huang, J., Jin, S., and Lu, S. Vision-language models for vision tasks: A survey. *arXiv:2304.00685*, 2023a.

Zhang, J., Zhang, J., Pertsch, K., Liu, Z., Ren, X., Chang, M., Sun, S.-H., and Lim, J. J. Bootstrap your own skills: Learning to solve new tasks with large language model guidance. In *Conference on Robot Learning*, 2023b.

Zhang, K., Kakade, S., Basar, T., and Yang, L. Model-based multi-agent rl in zero-sum markov games with near-optimal sample complexity. *Advances in Neural Information Processing Systems*, 33:1166–1178, 2020.

Zhang, M., Marklund, H., Dhawan, N., Gupta, A., Levine, S., and Finn, C. Adaptive risk minimization: Learning to adapt to domain shift. In *Advances in Neural Information Processing Systems*, 2021.

A. Limitations, Potential Broader Impact, Ethical Aspects and Societal Impacts

In this work, we mainly focus on illustrating the *Fuzzy-reward* effect and why it could be problematic to directly apply the pre-trained VLM rewards in online RL tasks. One limitation is that our current experiments are all in simulated environments. We plan to validate the effectiveness of the proposed method in real-world robotics in future work. Another limitation is that we maintained a second policy for Relay RL to escape the local minima during the online exploration. Though we turn off the Relay RL when we collect some successful trajectories, the extra policy still increases the computation complexity. An interesting future direction is to replace the replay policy by some lightweight exploration intrinsic reward to mitigate the issue of getting stuck in local minima.

Since our work is a combination of VLM and RL, one potential negative societal impact could be the improper language instructions. When we apply the proposed method to real-world applications, the usage of some language instructions might lead to dangerous behaviours. To mitigate this issue, we could adopt some rule-based keyword blacklists to filter dangerous language instructions, or we could further fine-tune the trained policy to learn some safety knowledge.

B. Experimental Setup

B.1. Implementation Details

In the experiment, we re-implement the SAC (Haarnoja et al., 2018) and DrQ (Yarats et al., 2021) baseline RL agents in JAX (Frostig et al., 2018). For the VLM model, we use the provided PyTorch code (Imambi et al., 2021) and checkpoint for both of LIV and CLIP from the official LIV codebase[2]. In the experiments, we use the latest Meta-world environment [3]. For the other main softwares, we use the following versions:

- Python 3.9
- jax 0.4.16
- jaxlib-0.4.16+cuda12.cudnn89-cp39
- flax 0.7.4
- gymnasium 0.29.1
- imageio 2.33.1
- optax 0.1.7
- torch 2.1.2
- torchvision 0.16.2
- numpy 1.26.2

B.2. Meta-world MT10 Benchmark

In the experiments, we use a constant reward shaping r^{task} - 1 for the sparse task reward as in some previous work (Kostrikov et al., 2022). For the task description for each environment, we followed the setting from the CARE (Sodhani et al., 2021).

In the experiments of Table 1, the goal is hidden and fixed for each random seed. Since we always use the mean action of the SAC policy in the evaluation and the Meta-world environment is deterministic, the evaluation success rate for each task is either 1 or 0 for each random seed. For the results of random goal setting in Table 2, we report the average evaluation success rate over ten trajectories. For the average results across all tasks in the Table 1, Table 2, and Table 3, we first group experiments for one algorithm into five runs across ten tasks, and get an ten-task average success rate, and compute the standard deviation based on the five numbers. For the goal image, we simply use a fixed goal image for both fixed goal and random goal tasks. The main idea of the goal image is to provide some useful information to distinguish two negative samples. Therefore, we adopt this simple setting without the loss of generality.

[2]https://github.com/penn-pal-lab/LIV
[3]https://github.com/Farama-Foundation/Metaworld

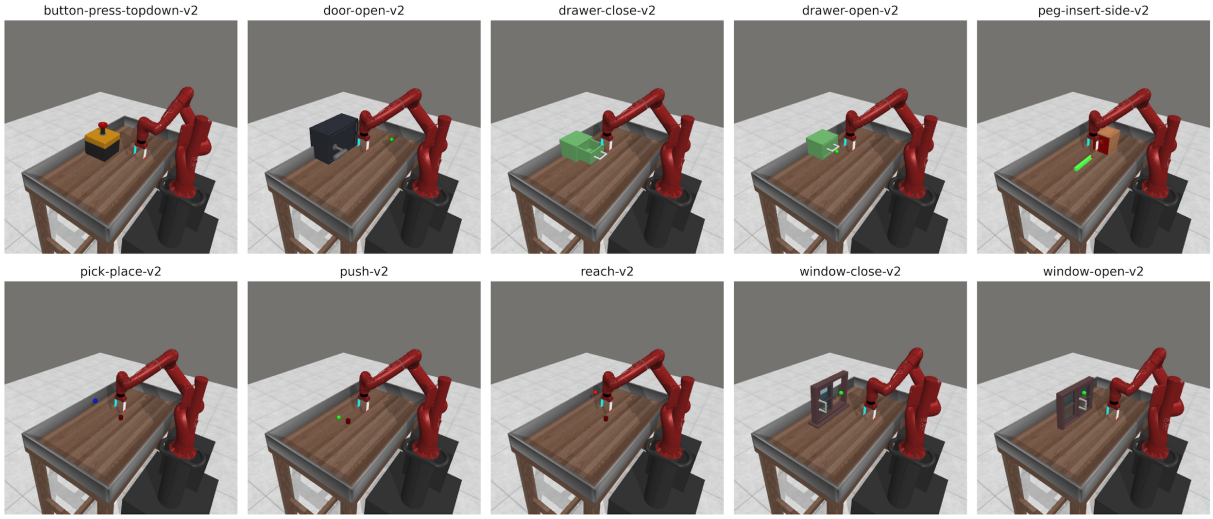

Figure 12. Meta-world MT10 benchmark tasks.

Table 4. Text instruction for each environment in the experiments.

Environment	Text instruction
button-press-topdown-v2	Press a button from the top.
door-open-v2	Open a door with a revolving joint.
drawer-close-v2	Push and close a drawer.
drawer-open-v2	Open a drawer.
peg-insert-side-v2	Insert a peg sideways.
pick-place-v2	Pick and place a puck to a goal.
push-v2	Push the puck to a goal.
reach-v2	Reach a goal position.
window-close-v2	Push and close a window.
window-open-v2	Push and open a window.

B.3. Computation Complexity

We run our experiments on a workstation with NVIDIA GeForce RTX 3090 GPU and a 12th Gen Intel(R) Core(TM) i9-12900KF CPU. The average wall-clock running time for the FuRL on the state-based experiment and pixel-based experiment are 3 hours and 6.5 hours, respectively.

B.4. Parameter Settings

Some key parameters are summarized in the Table 5. We mainly followed the parameter settings from some prior work (Ma et al., 2023c;a; Yarats et al., 2021; 2022). For the relay steps, we select Ts to be a set of four discrete values [50, 100, 150, 200] out of simplicity. Some other choices, *i.e.*, using a uniform distribution U[50, 250], are also acceptable. Since the main focus of this work is to illustrate the Fuzzy reward issue and showcase the effectiveness of the proposed method, therefore, we did not tune much of these hyper-parameters. It is likely to achieve better performances with further parameter tuning.

Table 5. Summarization of hyper-parameters.

Parameter	Value
Total environment step	1e6
Adam learning rate	1e-4
Batch size	256
Camera Id	2
VLM reward weight ρ	0.05
Target network τ	0.01
Discount factor γ	0.99
FuRL language projection network	(256, 64)
FuRL image projection network	(256, 64)
FuRL window size k	10
FuRL reward margin	0.1
FuRL L2 distance margin	0.2
SAC buffer size	1e6
SAC actor network	(256, 256)
SAC critic network	(256, 256)
SAC target entropy	-$\|\mathcal{A}\|$/2
DrQ buffer size	2e5
DrQ action repeat	2
DrQ frame stack	3
DrQ image size	(84, 84, 3)
DrQ embedding dimension	50
DrQ CNN features	(32, 32, 32, 32)
DrQ CNN kernels	(3, 3, 3, 3)
DrQ CNN strides	(2, 1, 1, 1)
DrQ CNN padding	VALID
DrQ actor network	(256, 256, 256)
DrQ critic network	(256, 256, 256)

C. Additional Experiment Results

C.1. Experiment on Sparse MT10 with random goals

Figure 13 shows the results of SAC, Relay and FuRL on Sparse MT10 with random goals. Similar to the conclusion as in Figure 6, FuRL generally outperforms the SAC and Relay baselines. Compared with the fixed goal setting, we can observe that FuRL generally achieved similar performances except for the *push-v2* task. Figure 14 shows two successful trajectories for the oracle policy and the FuRL policy. We can observe that the oracle policy first grasps the red cylinder and then moves to the green goal position. On the other hand, the FuRL policy didn't learn how grasp the cylinder and just simply pushes the cylinder to the goal position. Compared with the oracle policy, the FuRL policy is less robust without grasping the object, especially when the goal position is far away from the initial object position and the cylinder falls down and starts rolling. Moreover, another challenge in the *push-v2* task is that the arm sometimes blocks the goal and (or) the cylinder in the camera due to their small sizes. This makes the random goal setting much more difficult because the current VLM reward only relies on the input image, and an image without the goal provides less informative VLM reward.

C.2. The impact of language instructions

In the previous experiments, we all used the same language instructions. Here, we try to investigate how the input language instruction affects the final performance. We compare three different language instructions for the *button-press-topdown* and *drawer-open* tasks. The first language instruction is a dummy input text of "None". The other two language instructions are summarized in Table 6. From Figure 15, we can observe that using a more detailed language instruction can help to learn faster, while using a meaningless language instruction usually performs poorly. These results indicate that a more detailed language instruction sometimes could provide more accurate VLM rewards that help the agent to find the first successful trajectory earlier, and a misleading language instruction could provide inaccurate VLM rewards, which are more likely to trap the agent in local minima. Since the focus of this work is to illustrate the issue of Fuzzy VLM rewards and how to

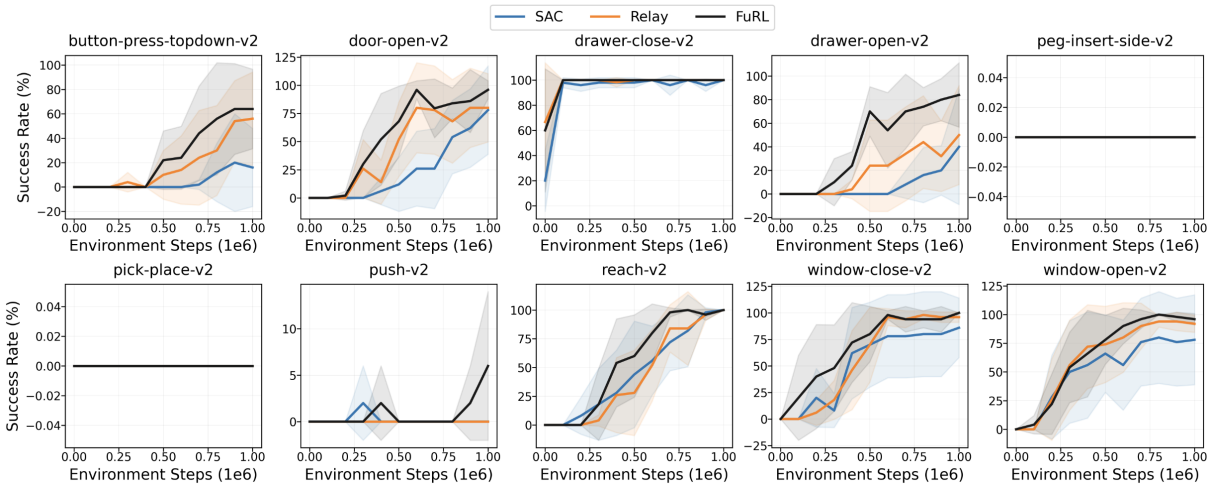

Figure 13. **Evaluation curves for tasks with random goal.**

address this issue, we leave the exploration of how to generate better text instructions for future work.

Table 6. Different text instructions.

Environment	Length	Text instruction
button-press-topdown-v2	Short	Press a button from the top.
	Long	Move close to the orange box and press down the red button.
drawer-open-v2	Short	Open a drawer.
	Long	Grab the white handle and open the green drawer.

C.3. Visualization of Inaccurate VLM Reward

In Figure 16, we plot three trajectories for an oracle policy π_o, a sub-optimal policy π_s and a random policy π_r. The sub-optimal policy is a policy been trained for small number of steps (1e5), which can move the arm around but is not able to complete the task yet (task success rate is 0%).

In Figure 17, we plot the VLM-rewards at different steps for the oracle policy π_o, random policy π_r and sub-optimal policy π_s for 50 trajectories, showing both individual curves as well as mean-std curves. We can observe that the random policy π_r is not very informative in this case since a random policy mostly causes the robot arm to jitter around its initial position and can barely move the arm. In addition, we can observe that the VLM-reward is very noisy. For example, there is a significant overlap between the curves from oracle and sub-optimal policies, although their level of expertise is drastically different, meaning the set of states covered by them are very different. Moreover, for the sub-optimal trajectories, there are cases where the VLM assigns very high reward to some sub-optimal states (e.g. around step 40, and step 80), with values comparable to (sometime even higher than) the VLM-reward for expert trajectory's success state (expert's trajectory around step 80). All these illustrate the fuzzy VLM-reward issue. Learning using this type of reward could mislead or trap RL agent in a local minimum and leading to undesired behaviors as shown in Figure 2 of the main paper.

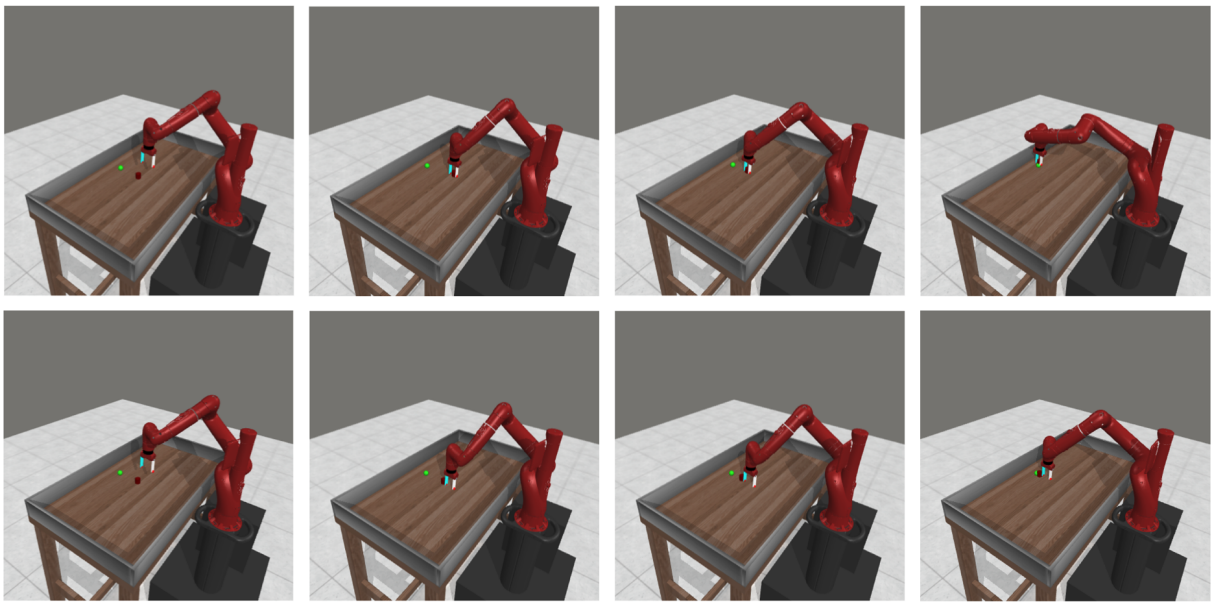

Figure 14. **Comparison of the oracle policy and the FuRL policy.** The oracle policy first grasps the red cylinder and then move to the green goal position (top row). On the other hand, the learned FuRL policy simply pushes the cylinder to the goal position (bottom row).

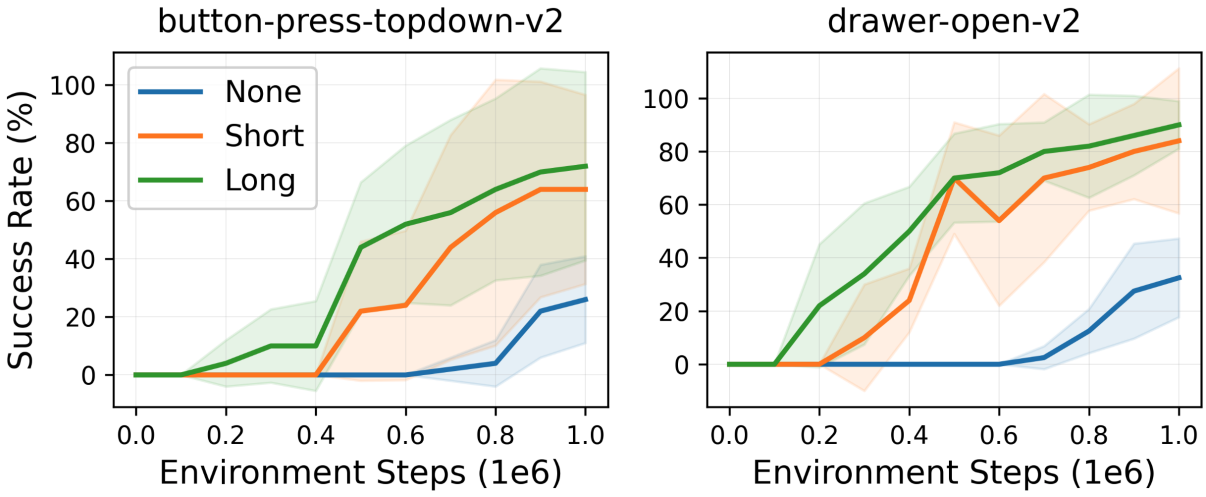

Figure 15. **Ablation for different language instructions.**

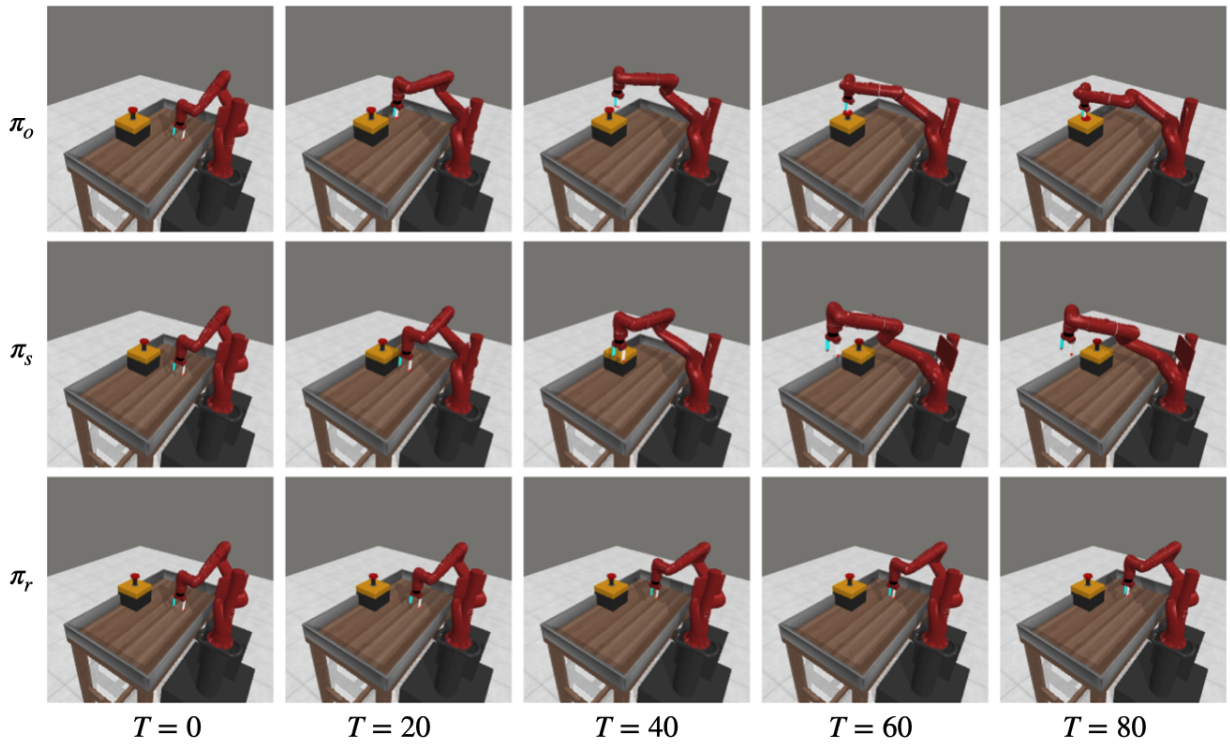

Figure 16. **More trajectory visualizations:** (top) an oracle policy π_o, (middle) a sub-optimal policy π_s, and (bottom) a random policy π_r.

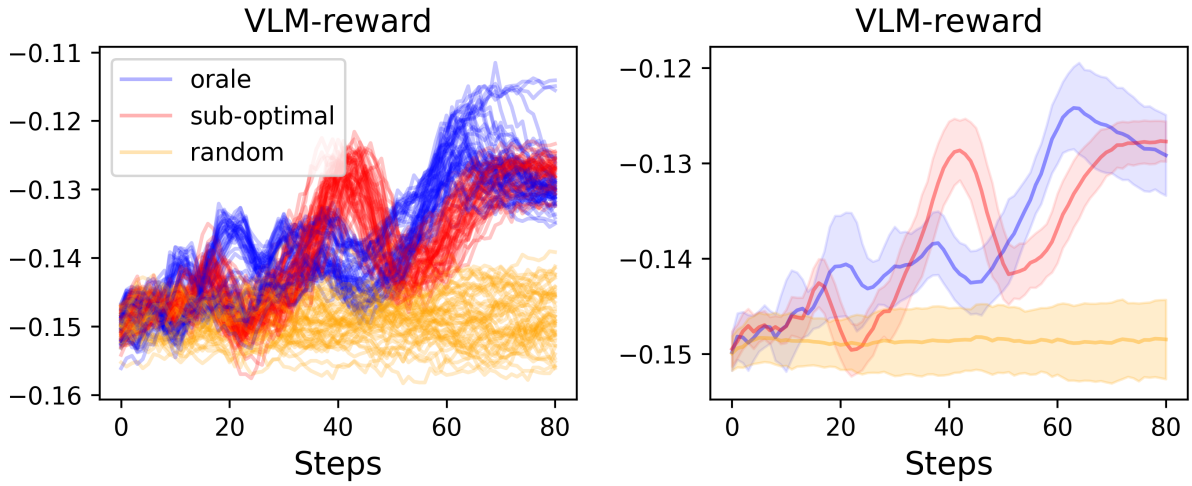

Figure 17. **Visualization of noisy VLM-rewards:** there is a significant overlap between the curves from oracle and sub-optimal policies.