RL-VLM-F: Reinforcement Learning from Vision Language Foundation Model Feedback

Yufei Wang [*1] **Zhanyi Sun** [*1] **Jesse Zhang** [2] **Zhou Xian** [1] **Erdem Bıyık** [2] **David Held** [†1] **Zackory Erickson** [†1]

Abstract

Reward engineering has long been a challenge in Reinforcement Learning (RL) research, as it often requires extensive human effort and iterative processes of trial-and-error to design effective reward functions. In this paper, we propose RL-VLM-F, a method that *automatically* generates reward functions for agents to learn new tasks, using only *a text description of the task goal and the agent's visual observations*, by leveraging feedbacks from vision language foundation models (VLMs). The key to our approach is to query these models to give *preferences over pairs of the agent's image observations* based on the text description of the task goal, and then learn a reward function from the preference labels, rather than directly prompting these models to output a raw reward score, which can be noisy and inconsistent. We demonstrate that RL-VLM-F successfully produces effective rewards and policies across various domains — including classic control, as well as manipulation of rigid, articulated, and deformable objects — without the need for human supervision, outperforming prior methods that use large pretrained models for reward generation under the same assumptions. Videos can be found on our project website: https://rlvlmf2024.github.io/.

1. Introduction

One of the key challenges of applying reinforcement learning (RL) is designing an appropriate reward function that will lead to the desired behavior. This procedure, known as reward engineering, demands considerable human effort and trial-and-error iterations, but is often required for good results (Laud, 2004; Silver et al., 2016; OpenAI et al., 2019; Gupta et al., 2022). In this work, we aim to develop a fully automated system that can generate a reward function and use it to teach agents to perform a task with RL *by using only a language description of the task*, eliminating the extensive human effort required to craft reward functions manually.

Prior work has studied replacing human supervision by prompting large language models (LLMs) to write code-based reward functions (Xie et al., 2023; Ma et al., 2023b; Wang et al., 2023). However, these methods usually assume access to the environment code, rely on the low-level ground-truth state information for reward generation, and face challenges with scaling up to high-dimensional environments and observations, such as manipulating complex deformable objects. Others (Klissarov et al., 2023; Chu et al., 2023) extract an intrinsic reward and combine it with the task reward using preference labels generated by an LLM comparing text descriptions of two agent states. However, text descriptions of the states can be non-trivial for certain tasks, such as manipulating deformable objects, as the exact states are hard to describe accurately using language. Further, these works rely on the ground-truth low-level state information to generate the text descriptions of the states, which may not be easily accessible.

Another related line of work obtains rewards from visual observations by using contrastively trained vision language models, such as CLIP (Radford et al., 2021), to align image or video observations with task descriptions in a learned latent space (Cui et al., 2022b; Mahmoudieh et al., 2022; Ma et al., 2023a; Sontakke et al., 2023; Adeniji et al., 2023; Rocamonde et al., 2023). However, the reward signals produced in these works are often of high variance and noisy (Sontakke et al., 2023; Mahmoudieh et al., 2022). As a result, prior work often has to fine-tune these CLIP-style models for their specific tasks at hand (Ma et al., 2023a; Mahmoudieh et al., 2022).

To this end, we present RL-VLM-F, a method that *automatically* generates reward functions for agents to learn new task. RL-VLM-F (Figure 1) requires only a single text description of the task goal and the agent's visual observations,

[*]Equal contribution. [†]Equal advising. [1]Robotics Institute, Carnegie Mellon University [2]Department of Computer Science, University of Southern California. Correspondence to: Yufei Wang <yufeiw2@andrew.cmu.edu>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

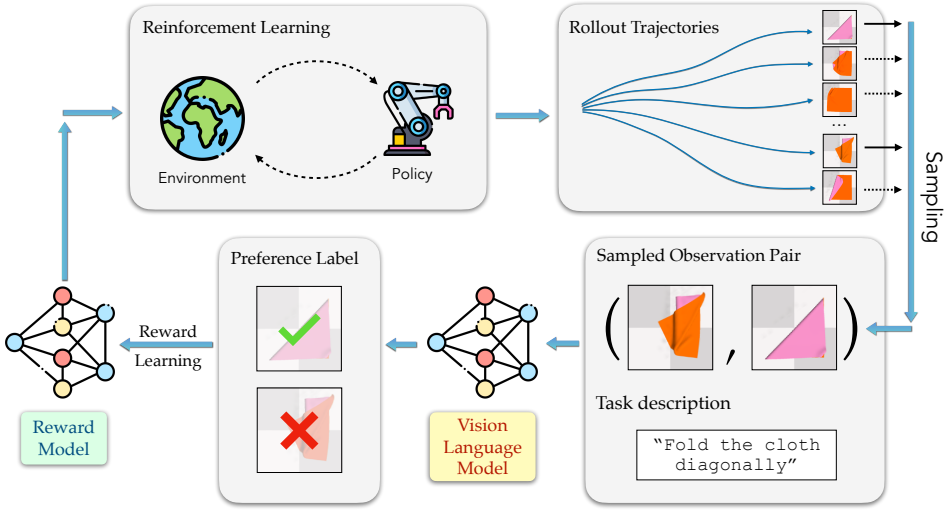

Figure 1. RL-VLM-F automatically generates reward functions for policy learning on new tasks, using only a text description of the task goal and the agent's visual observations. The key to RL-VLM-F is to query VLMs to give preferences over pairs of the agent's image observations based on the text description of the task goal, and then learn a reward function from the preference labels.

leveraging vision language foundation models (VLMs) that are trained on diverse, general text and image corpora (e.g., GPT-4V (OpenAI, 2023), Gemini (Team et al., 2023)). The key to our approach is to query these models to give *preferences over pairs of the agent's image observations* based on the text description of the task goal and then learn a reward function from the preference labels, rather than directly prompting these models to output a raw reward score, which can be noisy and inconsistent (Sontakke et al., 2023; Rocamonde et al., 2023). This allows us to draw from the rich literature on reinforcement learning from human preferences (Christiano et al., 2017; Wirth et al., 2017; Lee et al., 2021a), without requiring actual humans, to train reward functions automatically for new tasks. Furthermore, by using a VLM to compare image observations instead of text descriptions of the states, RL-VLM-F does not need access to the low-level ground-truth states for reward generation and can be applied to complex tasks involving deformable objects where accurate text description of the states are non-trivial. We test our method on 7 tasks involving classic control, rigid, articulated, and deformable object manipulation. We show that our approach can produce reward functions that lead to policies that solve diverse tasks, and our approach substantially outperforms prior methods and alternative ways to use VLMs to generate rewards. We also perform extensive analysis and ablation studies to provide insights into RL-VLM-F's learning procedure and performance gains.

In summary, we make the following contributions:

- We propose RL-VLM-F, a method that *automatically* generates reward functions for agents to learn new tasks, using only *a text description of the task goal and the agent's visual observations*, eliminating the extensive human effort involved in manually crafting reward functions.
- We show that RL-VLM-F can be used to generate reward functions and learn policies that can solve a series of rigid, articulated, and deformable object manipulation tasks, and it greatly outperforms prior methods.
- We perform extensive analysis and ablation studies to provide insights into RL-VLM-F's learning procedure and performance gains.

2. Related Works

Inverse Reinforcement Learning. Similar to our work, inverse reinforcement learning (IRL) aims to learn a reward function that can be used to train a policy to solve tasks. IRL methods usually learn a reward function from expert demonstrations (Ng & Russell, 2000; Abbeel & Ng, 2004; Ziebart et al., 2008; Ho & Ermon, 2016; Fu et al., 2018; Ni et al., 2021). In contrast, while RL-VLM-F also learns a reward function to train a policy, it only requires a text description of the task goal and does not require collecting expert demonstrations.

Learning from Human Feedback. Another line of work directly learns a reward function from human feedback, in the form of pairwise trajectory preference or ranking comparisons, to train a reward function (Christiano et al., 2017; Wirth et al., 2017; Ibarz et al., 2018; Leike et al., 2018; Biyik et al., 2019; 2020; Lee et al., 2021a; Myers et al., 2021; Bıyık et al., 2022). In most cases, human preferences and rankings of robot trajectories are easier to collect than demonstrations of robot trajectories. However, because each comparison conveys little information on its own, many preference queries are needed before the reward

function is well-trained enough to train an agent to perform the task. RL-VLM-F instead queries a VLM to perform the comparison to train a reward function, removing the need for extensive human labor in giving preference labels.

Large Pre-trained Models as Reward Functions. Kwon et al. (2023) first demonstrated that large pre-trained models—large language models (LLM) specifically—can generate rewards for RL agents in text-based tasks. Other works followed by demonstrating that LLMs can write structured code for training robots (Yu et al., 2023) or directly write Python code for training many kinds of agents (Xie et al., 2023; Ma et al., 2023b; Wang et al., 2023). However, many tasks are challenging to write reward functions for. For example, cloth folding requires tracking the locations of many individual cloth keypoints, which can change from one folding task to another. In these instances, visual reasoning is better suited for understanding how to reward the agent. RL-VLM-F queries a VLM to compare agent observation *images* so that it can use visual observations to reason about how well the agent is progressing in a task. In addition, prior methods usually assume access to the environment source code when writing the reward functions, whereas our method does not require such assumptions.

Another line of prior works rewards agents from image observations by aligning agent trajectory images with task language descriptions or demonstrations with contrastively trained visual language models (Cui et al., 2022a; Fan et al., 2022; Nottingham et al., 2023; Ma et al., 2023a; Sontakke et al., 2023; Rocamonde et al., 2023; Nam et al., 2023). However, experiments from these papers directly demonstrate that contrastive alignment is noisy and its accuracy relies heavily on the input task specification and how well-aligned the agent observations are to the pre-training data (Ma et al., 2023a; Sontakke et al., 2023; Rocamonde et al., 2023; Nam et al., 2023). Further, CLIP-style models have thus far been limited to outputting noisy raw scores. We demonstrate that using preferences results in superior performance to outputting raw scores, shown in our experiments in Section 6. Finally, our work shares a similar idea to RLAIF (Bai et al., 2022), which proposed to mix preference labels generated by an LLM and a human in the context of fine-tuning LLMs, and Motif (Klissarov et al., 2023), which proposed to generate intrinsic rewards using preference feedback from an LLM in the game of NetHack based on ground-truth text descriptions of the game state. In contrast, we use a VLM to generate the preference labels without any human labeling and learn the reward function from visual image observations without the need to access ground-truth states, focus on the domain of robotics control and manipulation, and directly generate task rewards instead of intrinsic rewards.

3. Background

We consider the standard Markov Decision Process and reinforcement learning setup (Sutton & Barto, 2018). At every timestep t, the agent receives a state s_t from the environment and chooses an action a_t based on a policy $\pi(a_t \mid s_t)$. The environment gives a reward r_t after the agents executes action a_t and transitions to s_{t+1}. The goal of the agent is to maximize the return, which is defined as discounted sum of rewards $R = \sum_{k=0}^{\infty} \gamma^k r(s_k, a_k)$ with discount factor γ.

Preference-based reinforcement learning. Our work builds upon preference-based RL, in which an agent learns a reward function from preference labels over its behaviors (Christiano et al., 2017; Ibarz et al., 2018; Lee et al., 2021a;b). Formally, a segment σ is a sequence of states $\{s_1, ..., s_H\}$, $H \geq 1$. In this paper we consider the case where the segment is represented using a single image, i.e., $H = 1$. Given a pair of segments (σ^0, σ^1), an annotator gives a feedback label y indicating which segment is preferred: $y \in \{-1, 0, 1\}$, where 0 indicates the first segment σ^1 is preferred, 1 indicates the second segment is preferred, and -1 indicates they are incomparable or equally preferable. Given a parameterized reward function r_ψ over the states, we follow the standard Bradley-Terry model (Bradley & Terry, 1952) to compute the preference probability of a pair of segments:

$$P_\psi[\sigma^1 \succ \sigma^0] = \frac{\exp\left(\sum_{t=1}^{H} r_\psi(s_t^1)\right)}{\sum_{i\in\{0,1\}} \exp\left(\sum_{t=1}^{H} r_\psi(s_t^i)\right)}, \tag{1}$$

where $\sigma^i \succ \sigma^j$ denotes segment i is preferred to segment j. Given a dataset of preferences $D = \{(\sigma_i^0, \sigma_i^1, y_i)\}$, preference-based RL algorithms optimize the reward function r_ψ by minimizing the following loss:

$$\mathcal{L}_{\text{Reward}} = -\,\mathbb{E}_{(\sigma^0,\sigma^1,y)\sim\mathcal{D}}\Big[\mathbb{I}\{y = (\sigma^0 \succ \sigma^1)\} \log P_\psi[\sigma^0 \succ \sigma^1] + \mathbb{I}\{y = (\sigma^1 \succ \sigma^0)\} \log P_\psi[\sigma^1 \succ \sigma^0]\Big]. \tag{2}$$

In preference-based RL algorithms, a policy π_θ and reward function r_ψ are updated alternatively: the reward function is updated with a dataset of preferences as described above, and the policy is updated with respect to this learned reward function using standard reinforcement learning algorithms. Specifically, we use PEBBLE (Lee et al., 2021a), a preference-based RL method with unsupervised pre-training and off-policy learning, as the underlying preference-based RL algorithm.

4. Assumptions

We make the following assumptions on the VLMs to be used in this paper: 1) We assume that the VLMs have been

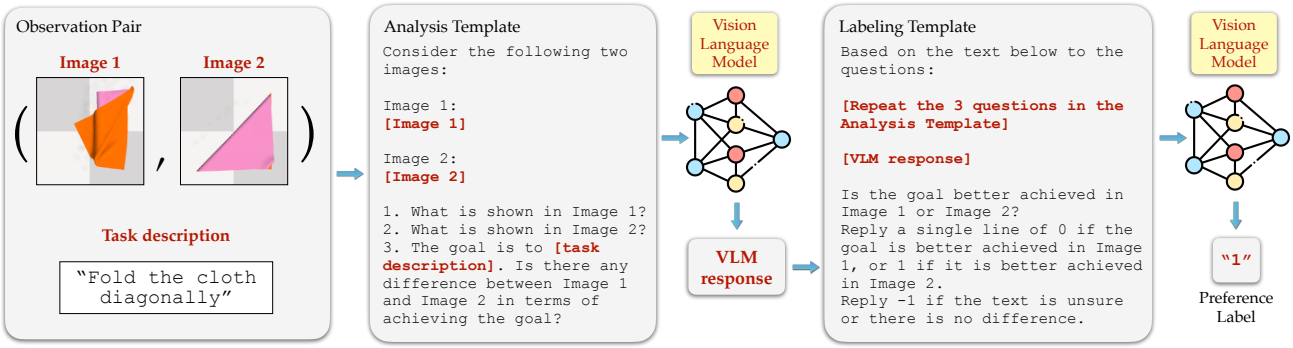

Figure 2. We use a two-stage VLM-querying process for generating preference labels to train the reward function. In the analysis stage, we query the VLM to generate free-form responses describing and comparing how well each of the two image observations achieves the task goal. Then, in the labeling stage, we prompt the VLM with the VLM-generated text responses from the first stage to extract a preference label between the two image observations. The template shown here is the actual entire template we use for all experiments.

Algorithm 1 RL-VLM-F

input Text description of task goal l
Initialize policy π_θ and reward r_ψ
Initialize the preference buffer $\mathcal{D} \leftarrow \emptyset$, RL replay buffer $\mathcal{B} \leftarrow \emptyset$, image observation buffer $\mathcal{I} \leftarrow \emptyset$, policy gradient update steps $\mathcal{N}_\pi$, reward gradient update steps $\mathcal{N}_r$, VLM query frequency K, number of preference queries per time M
for each iteration $iter$ **do**
  // POLICY LEARNING AND DATA COLLECTION
  for $t = 1$ to T **do**
    Collect state s_{t+1}, image I_{t+1} by taking $a_t \sim \pi_\theta(a_t|s_t)$
    Add transition $\mathcal{B} \leftarrow \mathcal{B} \cup \{(s_t, a_t, s_{t+1}, r_\psi(s_t))\}$
    Add image observation $\mathcal{I} \leftarrow \mathcal{I} \cup \{I_{t+1}\}$
  end for
  for $n = 1$ to $\mathcal{N}_\pi$ **do**
    Sample random batch $\{(s_t, a_t, s_{t+1}, r_\psi(s_t))_j\}_{j=1}^{B} \sim \mathcal{B}$
    Optimize policy π_θ using the sampled batch with any off-policy RL algorithm
  end for
  // PREFERENCE BY VLM AND REWARD LEARNING
  if $iter \ \% \ K == 0$ **then**
    for $m = 1$ to M **do**
      Randomly sample two images (σ^0, σ^1) from buffer $\mathcal{I}$
      Query VLM with (σ^0, σ^1) and task goal l for label y
      Store preference $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\sigma^0, \sigma^1, y)\}$
    end for
    for $n = 1$ to $\mathcal{N}_r$ **do**
      Sample minibatch $\{(\sigma^0, \sigma^1, y)_j\}_{j=1}^{D} \sim \mathcal{D}$
      Optimize r_ψ in Equation (2) with respect to ψ
    end for
    Relabel entire replay buffer $\mathcal{B}$ using updated r_ψ
  end if
end for

trained on diverse text and image corpora, enabling them to generalize well and reason across various environments and tasks. 2) The VLMs should be capable of processing multiple images simultaneously and performing comparative analyses on pairs of images as this is crucial for generating preference labels. 3) RL-VLM-F is designed to operate on tasks for which the quality or success of a state can be discerned from a single image or a sequence of images. We consider large pretrained vision-language foundation models, such as Gemini (Team et al., 2023) and GPT-4 Vision (OpenAI, 2023), to satisfy these assumptions.

5. Method

Figure 1 provides an overview of RL-VLM-F. Unlike previous preference-based RL algorithms that require a human annotator to give the preference labels, RL-VLM-F leverages a VLM to do so based solely on a text description of the task's goal, thus automating preference-based RL and mitigating the time-intensive human supervision required in writing reward functions or providing preference labels. RL-VLM-F works as follows: first, the policy π_θ and the reward function r_ψ are randomly initialized. Given a task goal description, our method then iterates through the following cycle: (1) The policy π_θ is updated using RL with the reward function r_ψ, interacts with the environment, and stores image observations into a buffer; (2) A batch of image pairs is randomly sampled from the stored buffer and sent to a VLM. The VLM is queried to produce preference labels for these image pairs in terms of which one better performs the task based on the text description of the task goal; (3) The reward model is updated with the loss in Equation (2) using the preference labels produced by the VLM. The full detailed procedure of RL-VLM-F can be found in Algorithm 1.

5.1. Prompting VLMs to Generate Preference labels for Reward Learning

To train the reward model r_ψ, we first need to generate preference labels from the VLM. To do this, we sample two images from the "image observation buffer" $\mathcal{I}$, which stores image observations of the policy during learning, and then query the VLM for which of the two images better performs

the task according to the text goal description (Algorithm 1 lines 17-18).

The querying process is illustrated in Figure 2. It consists of two stages: an **analysis** stage and then a **labeling** stage. In the **analysis** stage, we query the VLM to generate free-form responses describing and comparing how well each of the two images achieves the task goal. Then, in the **labeling** stage, we prompt the VLM with the VLM-generated *text responses* from the first stage to extract a preference label between the two images.[1] Specifically, the labeling stage prompt repeats the questions in the analysis prompt, fills in the VLM's response from the analysis stage, and then asks the VLM to generate a preference label $y \in \{-1, 0, 1\}$. We specify in the prompt that 0 or 1 indicates that the first or second image is better, respectively, and -1 indicates no discernible differences. We do not use the image pairs to train the reward model if the VLM returns -1 as the preference label. Finally, as shown at line 19 of Algorithm 1, we store the preference labels produced by the VLM into the preference label buffer $\mathcal{D}$ during the training process. Standard preference-based reward learning can then be performed (as detailed in Section 3) to train the reward function with Equation 2 using the preference buffer $\mathcal{D}$. Reward learning corresponds to lines 21-24 in Algorithm 1.

To minimize prompt engineering effort, we use a unified template *across all environments* (the exact entire template is shown in Figure 2). Therefore, to train a policy for a new environment with RL-VLM-F, one only needs to provide the task goal description; the labels and subsequently the reward function will then be automatically trained with the above process.

5.2. Implementation Details

For policy training, we use SAC (Haarnoja et al., 2018) as the underlying RL algorithm. As in PEBBLE (Lee et al., 2021a), we relabel all the transitions stored in the SAC replay buffer once the reward function r_ψ is updated (line 25 in Algorithm 1). We set the policy gradient update step $\mathcal{N}_\pi$ to be 1. The values of all other parameters in Alg. 1 can be found in Appendix B.

6. Experiments

6.1. Setup

We evaluate RL-VLM-F on a set of tasks, spanning from straightforward classic control tasks to complex manipulation tasks involving rigid, articulated, and deformable objects. The tasks are as follows.

[1]We can also use an LLM in this stage as it only requires text inputs, but for simplicity, we use the same model as for the first stage of the querying process (a VLM).

- One task from OpenAI Gym (Brockman et al., 2016):
  - *CartPole* where the goal is to balance a pole on a moving cart.
- Three rigid and articulated object manipulation tasks from MetaWorld (Yu et al., 2020) with a simulated Sawyer robot:
  - *Open Drawer*, where the robot needs to pull out a drawer;
  - *Soccer*, where the robot needs to push a soccer ball into a goal; and
  - *Sweep Into*, where the robot needs to sweep a green cube into a hole on the table.
- Three deformable object manipulation tasks from SoftGym (Lin et al., 2021):
  - *Fold Cloth*, where the goal is to diagonally fold a cloth from the top left corner to the bottom right corner;
  - *Straighten Rope*, where the goal is to straighten a rope from a random configuration; and
  - *Pass Water*, where the goal is to pass a glass of water to a target location without water being spilled out.

See Figure 3 for visualizations of these tasks. Further details about the tasks can be found in Appendix A.

We compare to the following baselines that make similar assumptions to us when generating the reward function, i.e., those requiring only a text description and image observations from the agents (without access to environment code). Below is a brief description of each baseline:

- **VLM Score**. Instead of querying the VLM to give preference labels over two images, this baseline directly asks the VLM to give a raw score between 0 to 1 for a given image based on the task goal description. We inform the VLM in the prompt that the score should be 1 if the task goal is perfectly achieved in the image. A reward model is then learned to regress to the scores given by the VLM.
- **CLIP Score** (Rocamonde et al., 2023). Given an image, the reward is computed as the cosine similarity score between the embedding of the image and the text description of the task goal using the CLIP model (Radford et al., 2021). Such a reward computation method has also been explored in several other prior works (Cui et al., 2022b; Mahmoudieh et al., 2022; Adeniji et al., 2023).
- **BLIP-2 Score**. Similar to the **CLIP Score** baseline but uses BLIP-2 (Li et al., 2023) instead of CLIP to compute the cosine similarity score.
- **RoboCLIP** (Sontakke et al., 2023). This baseline uses a pre-trained video-language model, S3D (Xie et al., 2018), to compute the reward as the similarity score between the embedding of the video of the policy trajectories and a demonstration video. Since we do not assume to have access to demonstrations of the task in our method, we use the text version of RoboCLIP for a fair comparison. RoboCLIP-Text uses the pre-trained video-language

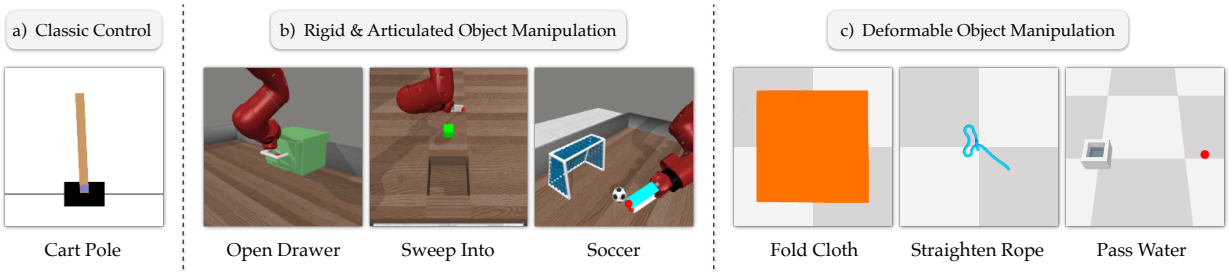

Figure 3. We evaluate RL-VLM-F on 7 tasks including classic control, rigid and articulated object manipulation, as well as deformable object manipulation. For *Pass Water*, the red dot represents the target location.

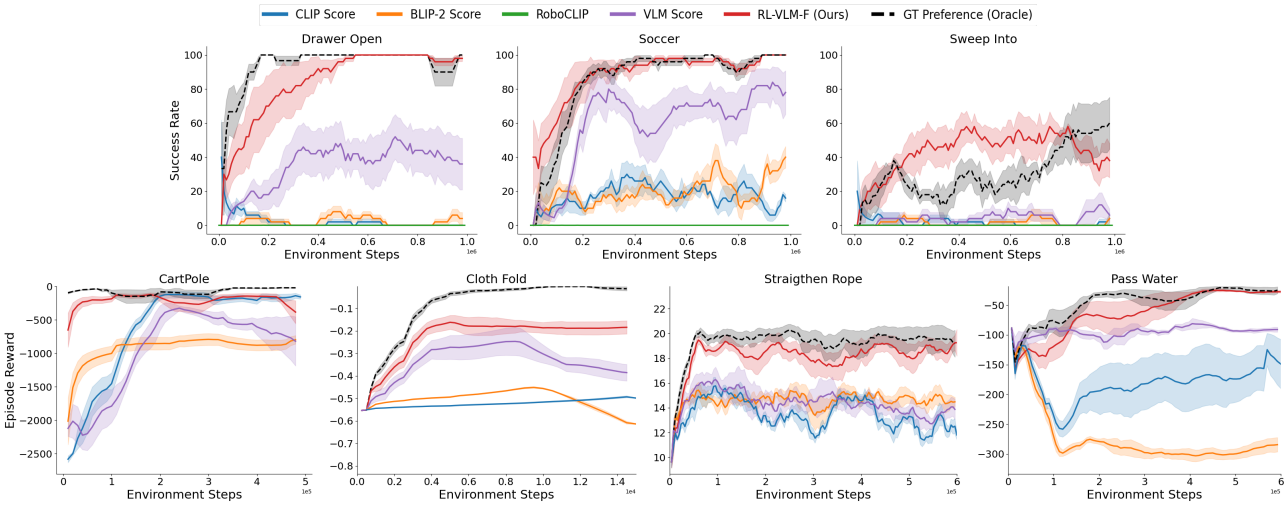

Figure 4. Learning curves of all compared methods on 7 tasks. RL-VLM-F outperforms all baselines in all tasks, and matches or surpasses the performance of GT preference on 6 of the 7 tasks. Results are averaged over 5 seeds, and shaded regions represent standard error. RoboCLIP is only evaluated on the MetaWorld tasks, as this is the set of tasks where the original method is evaluated.

model to generate rewards as the similarity score between the video embedding of the trajectory and the text embedding of the task description.

- **GT Preference**. We use the original ground-truth reward function (provided by the authors of each benchmark) to give the preference label. This should in theory serve as an oracle and upper bound on the learning performance.

Further details on the baselines, including all the text prompts we use, can be found in Appendices C and D.

For MetaWorld tasks, we use the author-defined task success rate of the policy as the evaluation metric (Yu et al., 2020). For all other tasks, we report the episode return of the learned policy. For all methods, the policy is learned with state observations, and we use the same policy learning hyper-parameters for all methods, i.e., the only difference between all compared methods is the reward function. For methods where a reward function needs to be learned (RL-VLM-F and VLM Score), the reward function is learned using image observations. For RL-VLM-F and the VLM Score baseline, we use Gemini-Pro (Team et al., 2023) as the VLM for all tasks except *Fold Cloth*. We find Gemini-Pro to perform poorly on *Fold Cloth*, so we instead use GPT-4V (OpenAI, 2023) as the VLM for this task for these two methods (see Appendix E.2 for a comparison of Gemini-Pro and GPT-4V on this specific task). We did not run GPT-4V on all tasks due to its quota limitations. For all methods except RoboCLIP, we remove the robot from the image for the MetaWorld tasks, as these tasks are all object-centric and removing the robot allows the VLM to focus on the target object when analyzing the images. Since these tasks are simulated, we conveniently use the simulator to make the robot transparent when rendering the images. For real-world applications, techniques such as inpainting can be used to remove the robot from image observations as done in prior work (Bahl et al., 2022; Bharadhwaj et al., 2023). We keep the robot within the image for RoboCLIP following the original paper's setup. We test RoboCLIP only on the MetaWorld tasks, as this is the set of tasks where the original method is evaluated.

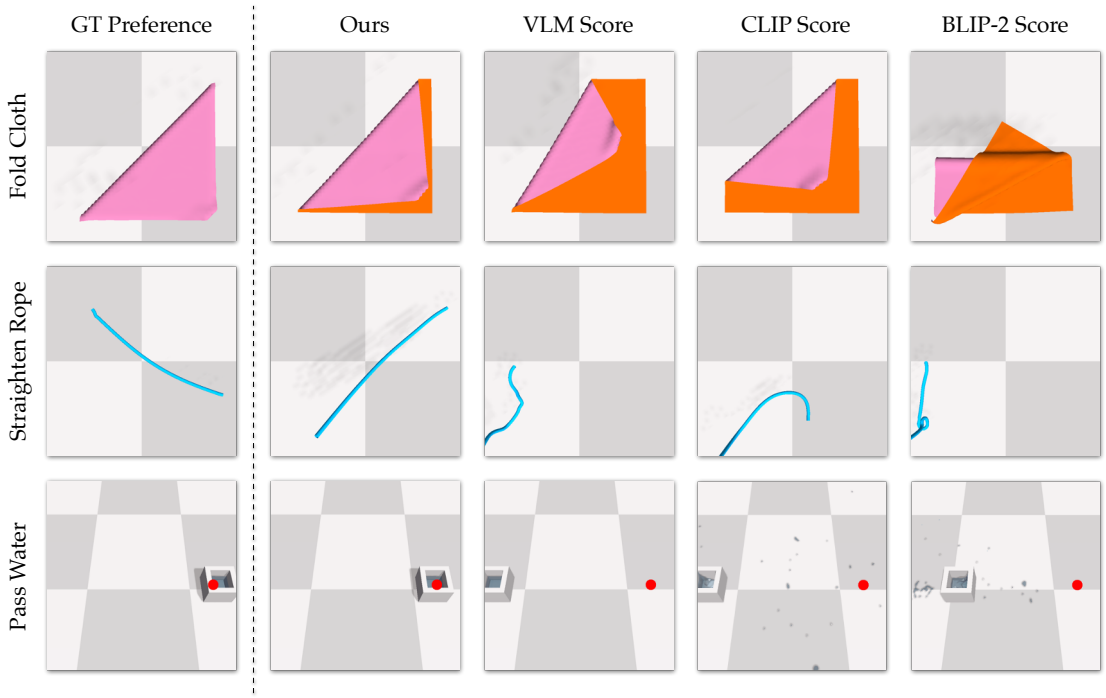

Figure 5. Comparison of the achieved final state of different methods on SoftGym deformable object manipuation tasks: *Fold Cloth* (Top), *Straighten Rope* (Middle), and *Pass Water* (Bottom). RL-VLM-F achieves better final states compared to all the baselines.

6.2. Does RL-VLM-F Learn Effective Rewards and Policies?

We first examine if RL-VLM-F leads to useful rewards and policies that can solve the tasks. The learning curves of all compared methods on all tasks are shown in Figure 4. As shown, RL-VLM-F outperforms all other baselines in all tasks. We find that prior approaches using CLIP or BLIP-2 score can only solve the easiest task – *CartPole*, and struggle for more complex environments, such as the rigid object manipulation tasks in MetaWorld and the deformable object manipulation tasks in SoftGym. The text version of RoboCLIP performs poorly on all three MetaWorld tasks, aligning with the original paper's results, as RoboCLIP works the best with video demonstrations available. RL-VLM-F also outperforms VLM Score in all tasks, which indicates that prompting VLMs to output a preference label for reward learning results in better task performance in contrast to treating the VLM as a reward function that outputs raw reward scores. We also observe that RL-VLM-F is able to match the performance of using GT preference in all tasks except Cloth Fold, which suggests we can use a single text description with RL-VLM-F to mitigate human efforts in writing complex reward functions for these tasks.

Interestingly, for the task of *Sweep Into*, the performance of RL-VLM-F actually surpasses that of using GT preference. We suspect the reason could be as follows: the ground-truth reward function written by the authors for this task includes terms that are not directly correlated to task success. This includes a reward term for grasping the cube, which is not critical for pushing the cube into the hole. In contrary, RL-VLM-F simply uses a text description of the task goal as "minimize the distance between the cube and the hole", thus the learned reward is less prone to bias in human-written reward functions and may better reflect the true task goal, leading to better performance.

We show the final states achieved by the policies learned with different methods on the three SoftGym deformable object manipulation tasks in Figure 5. As shown, for all three tasks, RL-VLM-F achieves a final state that is quantifiably better than the baselines. For *Fold Cloth*, RL-VLM-F is closest to a diagonal fold. For *Straighten Rope*, RL-VLM-F is able to fully straighten the rope and match the performance of GT preference, where all other baselines failed to fully straighten it. For *Pass Water*, RL-VLM-F is able to transport the water to the target location without any water being spilled, and the baselines either do not move the glass, or move it in a way that spills large amounts of water.

6.3. What is the Accuracy of VLM Preference Labeling?

Given that RL-VLM-F can learn effective rewards and policies that solve the tasks, we perform further analysis on the accuracy of the preference labels generated by a VLM. To compute accuracy, the VLM outputs $\{-1, 0, 1\}$ (no preference, first image preferred, second image preferred) which we compare to a ground truth preference label defined according to the environment's reward function. Note that we

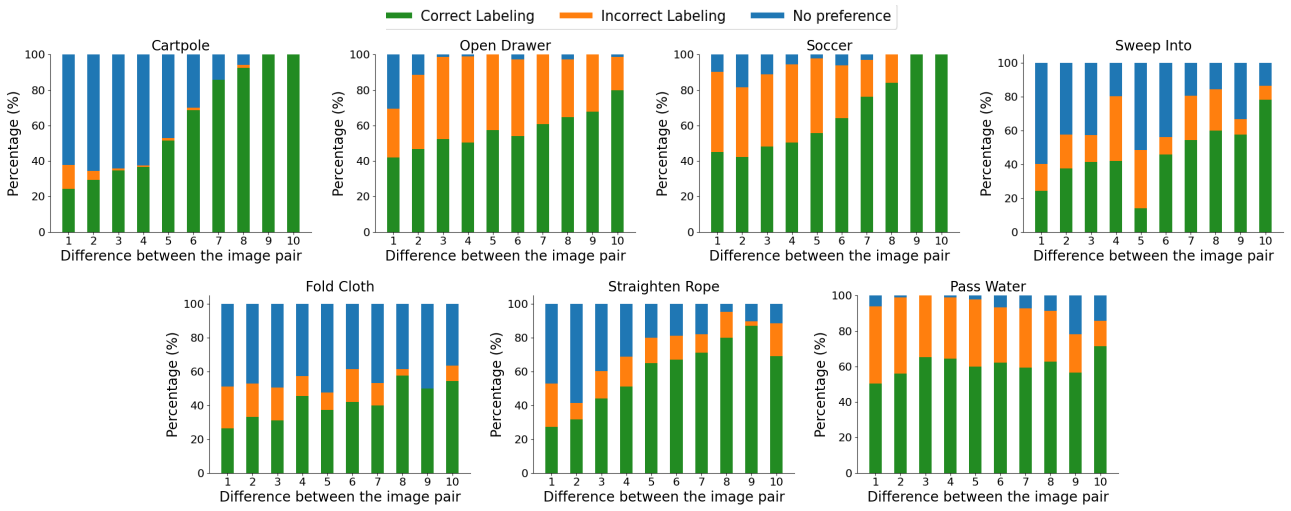

Figure 6. We provide analysis of the accuracy of the VLM preference labels, compared to ground-truth preference labels defined according to the environment's reward function. The x-axis represents different levels of differences between the image pairs, discretized into 10 bins, where the difference is measured as the difference between the ground-truth task progress associated with the image pairs. The y-axis shows the percentage where the VLM preference labels are correct, incorrect, or when it does not have a preference over the image pairs.

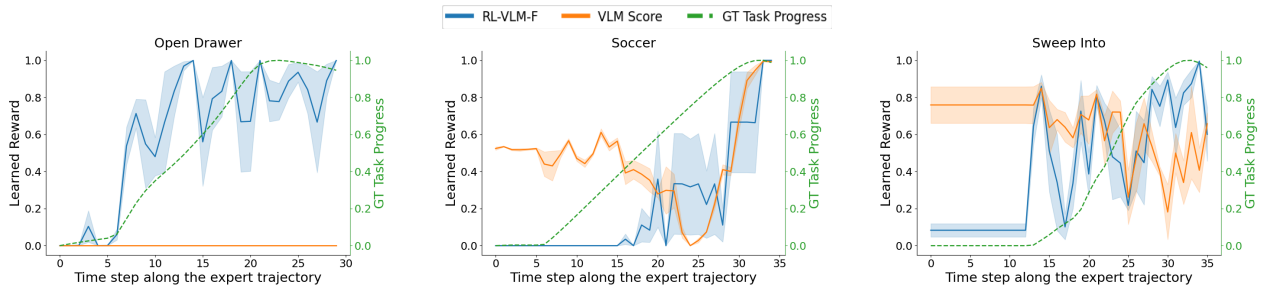

Figure 7. We compare how well the learned reward by RL-VLM-F and VLM Score align with the ground-truth task progress on 3 MetaWorld tasks along an expert trajectory. As shown, RL-VLM-F generates rewards that align better with the ground-truth task progress. The learned rewards are averaged over 3 trained reward models with different seeds, and the shaded region represents the standard error.

discard the image pairs with a label -1 (no preference) when training the reward model.

Our intuition is that, like humans, it would be hard for the VLM to give correct preference labels when comparing two similar images, and easier to produce correct preference labels when the two images are noticeably dissimilar in terms of achieving the goal. Figure 6 presents the accuracy of the VLM at various levels of differences between the two images. The "difference" between two images is measured as the difference between the ground-truth task progress associated with the images. We discretize the differences into 10 bins along the x axis in Figure 6, where a larger number indicates a greater difference between two images in terms of task progress. On the y axis, the green, orange, and blue bars represent the percentage where the VLM preference label is correct, incorrect, or when there is no preference. For all tasks, we observe a general trend of increasing accuracy, decreasing uncertainty, and decreasing error as the differences between the images increase, which aligns with intuition. This trend is most clear and consistent for the *CartPole*, *Open Drawer* and *Soccer* tasks. Overall, for all tasks, we find that the VLM is able to generate more correct preference labels than incorrect ones, and as shown in Figure 4, the accuracy of VLM-generated preference labels is sufficient for learning a good reward function and policy.

6.4. How Does the Learned Reward Align With the Task Progress?

Figure 7 plots the learned rewards (averaged over 3 trained reward models with different random seeds) as well as the true task progress on three MetaWorld environments along an expert trajectory that fully solves the task. Note the ground-truth task progress is not the same as the author-provided reward function: the author provided reward is a shaped version of the task progress. For *Open Drawer*, the task progress is measured as the distance the drawer has been pulled out; For *Soccer*, it is measured as the negative

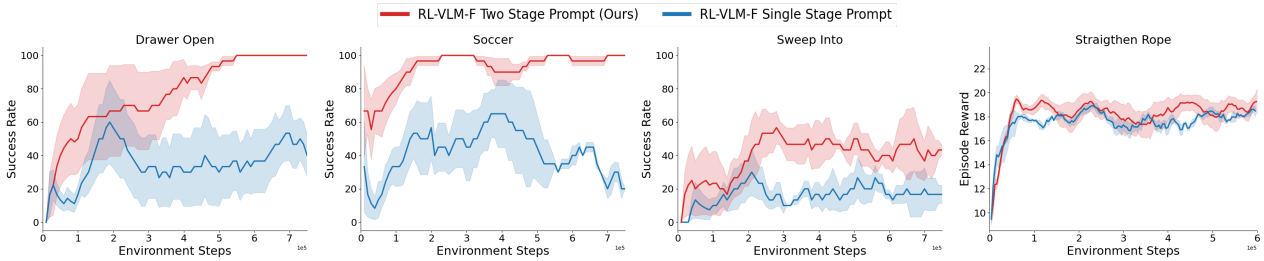

Figure 8. We compare RL-VLM-F using the proposed two-stage prompting strategy, and an ablated version of using a single-stage prompting strategy. The performance of the single-stage prompting is lower on 3 of the 4 tasks.

distance between the soccer ball and the goal; For *Sweep Into*, it is measured as the negative distance between the cube and the hole. We normalize both the ground-truth task progress and the learned reward into the range of [0, 1] for a better comparison between them. An ideal learned reward should increase as the time step increases along the expert trajectory, as like the ground-truth task progress. As shown, the reward learned by RL-VLM-F aligns better with the ground-truth task progress compared with the VLM Score baseline. We do observe that the learned reward tends to be noisy and includes many local minima. Despite this, the learned reward still achieves the highest value when the task progresses the most (except for the task of Sweep Into). As shown in Figure 4, the learned reward is sufficient for learning successful policies. For *Open drawer*, we notice that the reward produced by VLM Score remains zero. This is likely because, during training, most of the scores given by the VLM are 0, and the model learns to predict 0 at all time steps to minimize the regression loss. We find the CLIP and BLIP-2 scores on these environments are generally noisy; the corresponding plots can be found in Appendix E.3.

6.5. Ablation on the Prompt Strategy

We used a two-stage prompting strategy for RL-VLM-F, where the VLM is first asked to analyze the pair of images in the analysis stage, and then output the preference label in the labeling stage. Here we compare it with a single-stage prompting strategy where we query the VLM to directly output a preference label over the two image observations in a single stage. The detailed single-stage prompt can be found in Appendix D.4. Figure 8 presents the comparison on 4 tasks: Open Drawer, Soccer, Sweep Into and Straighten Rope. As shown, the success rate of using the VLM with the single-stage prompt is lower than that of using the two-stage prompt on 3 out of the 4 tasks.

7. Conclusion and Future Work

In this work, we present RL-VLM-F, a method that automatically generates reward functions via querying VLMs with preferences given a task descriptions and image observations for a wide range of tasks. We demonstrate our proposed method's effectiveness on rigid, articulated, and deformable object manipulation tasks.

Future work could extend RL-VLM-F to an active learning context, exploring both easy and informative VLM queries for more efficient reward learning. The adaptable nature of our method allows for the integration of more advanced VLMs when they become available, potentially addressing more complex tasks. It would also be interesting to test RL-VLM-F on tasks with a longer horizon. One could first decompose the tasks into subtasks with shorter horizons, either via manual decomposition or foundation models (Ahn et al., 2022). Then, RL-VLM-F can be used to solve each subtask. Additionally, our approach offers a practical pathway to applying RL in real-world settings, where obtaining reward functions is often difficult.

Acknowledgements

This work is supported by the National Science Foundation under Grant No. IIS-2046491. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Impact Statement

As we use pre-trained Vision Language Models for generating the reward functions, the bias presented in the VLMs might be inherited into the reward function and subsequently the learned policy. As a result, one might want to examine the behavior of the learned policy before deploying it to safety critical applications. Other than this point, we do not anticipate any societal consequences of our work that must be specifically highlighted here.

References

Abbeel, P. and Ng, A. Y. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the Twenty-First International Conference on Machine Learning*, ICML '04, pp. 1, New York, NY, USA, 2004. Asso-

ciation for Computing Machinery. ISBN 1581138385. doi: 10.1145/1015330.1015430.

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

Ahn, M., Brohan, A., Brown, N., Chebotar, Y., Cortes, O., David, B., Finn, C., Fu, C., Gopalakrishnan, K., Hausman, K., et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

Bahl, S., Gupta, A., and Pathak, D. Human-to-robot imitation in the wild. *arXiv preprint arXiv:2207.09450*, 2022.

Bai, Y., Kadavath, S., Kundu, S., Askell, A., Kernion, J., Jones, A., Chen, A., Goldie, A., Mirhoseini, A., McKinnon, C., et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022.

Bharadhwaj, H., Gupta, A., Kumar, V., and Tulsiani, S. Towards generalizable zero-shot manipulation via translating human interaction plans. *arXiv preprint arXiv:2312.00775*, 2023.

Biyik, E., Palan, M., Landolfi, N. C., Losey, D. P., and Sadigh, D. Asking easy questions: A user-friendly approach to active reward learning. In *Proceedings of the 3rd Conference on Robot Learning (CoRL)*, 2019.

Biyik, E., Huynh, N., Kochenderfer, M. J., and Sadigh, D. Active preference-based gaussian process regression for reward learning. In *Proceedings of Robotics: Science and Systems (RSS)*, July 2020. doi: 10.15607/rss.2020.xvi.041.

Bıyık, E., Losey, D. P., Palan, M., Landolfi, N. C., Shevchuk, G., and Sadigh, D. Learning reward functions from diverse sources of human feedback: Optimally integrating demonstrations and preferences. *The International Journal of Robotics Research*, 41(1):45–67, 2022.

Bradley, R. A. and Terry, M. E. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Christiano, P. F., Leike, J., Brown, T., Martic, M., Legg, S., and Amodei, D. Deep reinforcement learning from human preferences. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Chu, K., Zhao, X., Weber, C., Li, M., and Wermter, S. Accelerating reinforcement learning of robotic manipulations via feedback from large language models. *arXiv preprint arXiv:2311.02379*, 2023.

Cui, Y., Niekum, S., Gupta, A., Kumar, V., and Rajeswaran, A. Can foundation models perform zero-shot task specification for robot manipulation? *Learning for Dynamics and Control Conference*, 2022a.

Cui, Y., Niekum, S., Gupta, A., Kumar, V., and Rajeswaran, A. Can foundation models perform zero-shot task specification for robot manipulation? In Firoozi, R., Mehr, N., Yel, E., Antonova, R., Bohg, J., Schwager, M., and Kochenderfer, M. (eds.), *Proceedings of The 4th Annual Learning for Dynamics and Control Conference*, volume 168 of *Proceedings of Machine Learning Research*, pp. 893–905. PMLR, 23–24 Jun 2022b. URL https://proceedings.mlr.press/v168/cui22a.html.

Fan, L., Wang, G., Jiang, Y., Mandlekar, A., Yang, Y., Zhu, H., Tang, A., Huang, D.-A., Zhu, Y., and Anandkumar, A. Minedojo: Building open-ended embodied agents with internet-scale knowledge. *Advances in Neural Information Processing Systems*, 2022.

Fu, J., Singh, A., Ghosh, D., Yang, L., and Levine, S. Variational inverse control with events: A general framework for data-driven reward definition. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Gupta, A., Pacchiano, A., Zhai, Y., Kakade, S. M., and Levine, S. Unpacking reward shaping: Understanding the benefits of reward engineering on sample complexity, 2022.

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*, pp. 1861–1870. PMLR, 2018.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Ho, J. and Ermon, S. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29, 2016.

Ibarz, B., Leike, J., Pohlen, T., Irving, G., Legg, S., and Amodei, D. Reward learning from human preferences and demonstrations in atari. *Advances in neural information processing systems*, 31, 2018.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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 preprint arXiv:2310.00166*, 2023.

Kwon, M., Xie, S. M., Bullard, K., and Sadigh, D. Reward design with language models. In *International Conference on Learning Representations (ICLR)*, 2023.

Laud, A. D. *Theory and application of reward shaping in reinforcement learning*. PhD thesis, USA, 2004. AAI3130966.

Lee, K., Smith, L., and Abbeel, P. Pebble: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training, 2021a.

Lee, K., Smith, L., Dragan, A., and Abbeel, P. B-pref: Benchmarking preference-based reinforcement learning. *arXiv preprint arXiv:2111.03026*, 2021b.

Leike, J., Krueger, D., Everitt, T., Martic, M., Maini, V., and Legg, S. Scalable agent alignment via reward modeling: a research direction. *arXiv preprint arXiv:1811.07871*, 2018.

Li, J., Li, D., Savarese, S., and Hoi, S. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597*, 2023.

Lin, X., Wang, Y., Olkin, J., and Held, D. Softgym: Benchmarking deep reinforcement learning for deformable object manipulation. In *Conference on Robot Learning*, pp. 432–448. PMLR, 2021.

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 preprint 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 preprint arXiv: Arxiv-2310.12931*, 2023b.

Mahmoudieh, P., Pathak, D., and Darrell, T. Zero-shot reward specification via grounded natural language. In *International Conference on Machine Learning*, pp. 14743–14752. PMLR, 2022.

Miech, A., Zhukov, D., Alayrac, J.-B., Tapaswi, M., Laptev, I., and Sivic, J. HowTo100M: Learning a Text-Video Embedding by Watching Hundred Million Narrated Video Clips. In *ICCV*, 2019.

Myers, V., Biyik, E., Anari, N., and Sadigh, D. Learning multimodal rewards from rankings. In *5th Annual Conference on Robot Learning*, 2021.

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

Ng, A. Y. and Russell, S. J. Algorithms for inverse reinforcement learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, pp. 663–670, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc. ISBN 1558607072.

Ni, T., Sikchi, H., Wang, Y., Gupta, T., Lee, L., and Eysenbach, B. f-irl: Inverse reinforcement learning via state marginal matching. In *Conference on Robot Learning*, pp. 529–551. PMLR, 2021.

Nottingham, K., Ammanabrolu, P., Suhr, A., Choi, Y., Hajishirzi, H., Singh, S., and Fox, R. Do embodied agents dream of pixelated sheep?: Embodied decision making using language guided world modelling. *arXiv preprint arXiv:2301.12050*, 2023.

OpenAI. Gpt-4v(ision) system card. 2023.

OpenAI, 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., Petrov, M., d. O. 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, 2019.

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*, pp. 8748–8763. PMLR, 2021.

Rocamonde, J., Montesinos, V., Nava, E., Perez, E., and Lindner, D. Vision-language models are zero-shot reward models for reinforcement learning. In *NeurIPS 2023 Foundation Models for Decision Making Workshop*, 2023. URL https://openreview.net/forum?id=JUwczEJY8I.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

Sontakke, S. A., Zhang, J., Arnold, S., Pertsch, K., Biyik, E., Sadigh, D., Finn, C., and Itti, L. RoboCLIP: One demonstration is enough to learn robot policies. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=DVlawv2rSI.

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

Team, G., Anil, R., Borgeaud, S., Wu, Y., Alayrac, J.-B., Yu, J., Soricut, R., Schalkwyk, J., Dai, A. M., Hauth, A., et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Wang, Y., Xian, Z., Chen, F., Wang, T.-H., Wang, Y., Fragkiadaki, K., Erickson, Z., Held, D., and Gan, C. Robogen: Towards unleashing infinite data for automated robot learning via generative simulation. *arXiv preprint arXiv:2311.01455*, 2023.

Wirth, C., Akrour, R., Neumann, G., and Fürnkranz, J. A survey of preference-based reinforcement learning methods. *Journal of Machine Learning Research*, 18(136):1–46, 2017. URL http://jmlr.org/papers/v18/16-634.html.

Xie, S., Sun, C., Huang, J., Tu, Z., and Murphy, K. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 305–321, 2018.

Xie, T., Zhao, S., Wu, C. H., Liu, Y., Luo, Q., Zhong, V., Yang, Y., and Yu, T. Text2reward: Automated dense reward function generation for reinforcement learning. *arXiv preprint arXiv:2309.11489*, 2023.

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*, pp. 1094–1100. PMLR, 2020.

Yu, W., Gileadi, N., Fu, C., Kirmani, S., Lee, K.-H., Gonzalez Arenas, M., Lewis Chiang, H.-T., 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. *Arxiv preprint arXiv:2306.08647*, 2023.

Ziebart, B. D., Maas, A., Bagnell, J. A., and Dey, A. K. Maximum entropy inverse reinforcement learning. In *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 3*, AAAI'08, pp. 1433–1438. AAAI Press, 2008. ISBN 9781577353683.

Appendix

A. Details on Tasks and Environments

We run our method and baselines on *CartPole* from openAI Gym (Brockman et al., 2016), three rigid and articulated object manipulation tasks from MetaWorld (Yu et al., 2020), and three deformable object manipulation tasks from SoftGym (Lin et al., 2021). For the three MetaWorld tasks, we modified the gripper initial state such that it starts close to the target object to manipulate. Figure 3 in the paper shows the initial state for these 3 tasks. We also adjusted the camera view such that the target object is clearly visible at around the center of the image, to provide good images for VLM to give preferences. We describe the observation space and action space for those tasks as follows:

A.1. Observation Space

For policy learning with SAC, we use state-based observations; for reward learning, we use high dimensional RGB image observations, rendered by the simulator. We now detail the state-based observation space for each task.

MetaWorld Tasks. For MetaWorld tasks, we follow the setting in the original paper (Yu et al., 2020). The state observation always has 39 dimensions. It consists of the position and gripper status of the robot's end-effector, the position and orientation of objects in the scene, and the position of the goal.

CartPole. The state observation has 4 dimensions, including the position and velocity of the cart, as well as the angle and angular velocity of the pole.

Cloth Fold. The state observation is the position of a subset of the particles in the cloth mesh. The cloth is of size 40 x 40, and we uniformly subsample it to be of size 8 x 8. The state is then the position of the picker, and the positions of all those subsampled particles.

Straighten Rope. The state observation is the positions of all particles on the rope and has 36 dimensions.

Pass Water. The state observation includes the size (width, length, height) of the container, the target container position, height of the water in the container, amount of water inside and outside of the container. The state observation has 7 dimensions.

A.2. Action Space

For all environments, we normalize the action space to be within $[-1, 1]$. Below we describe the action space for each environment.

MetaWorld Tasks. For MetaWorld tasks, the action space always has four dimensions. It includes the change in 3D position of the robot's end-effector followed by a normalized torque that the gripper fingers should apply.

CartPole. The original action space is a discrete value in $0, 1$, indicating the direction of the fixed force the cart is pushed with. We modified it to be continuous within range [0, 1] such that SAC can be used as the learning algorithm. The continuous action represents the force applied to the pole.

Cloth Fold. For this task, we use a pick-and-place action primitive. We assume that the corner of the cloth is grasped when the task is initialized. The action is the 2D target place location.

Straighten Rope. For this task, we use two pickers, one at each end of the rope, to control the rope. Therefore, the action space is the 3D delta positions for each picker and has 6 dimensions in total. We assume the two end points of the rope is already grasped at the beginning of the task.

Pass Water. The motion of the glass container is constrained to be in one dimension. Therefore, the action also has a dimension of 1 and is the delta position of the container along the dimension.

B. Hyper-parameters and Network Architectures

B.1. Image-based Reward Learning

For the image-based reward model, we use a 4-layer Convolutional Neural Network for MetaWorld tasks and *CartPole* and a standard ResNet-18 (He et al., 2016) for the three deformable object manipulation tasks. Following PEBBLE (Lee et al., 2021a), we also use an ensemble of three reward models and use tanh as the activation function for outputting reward. For RL-VLM-F, we train the model by optimizing the cross-entropy loss, defined in Equation 2. For VLM Score, we train the mode by optimizing the MSE loss between the predicted score and ground-truth score output by the VLM. For both methods, we use ADAM (Kingma & Ba, 2014) as the optimizer with an initial learning rate of 0.0003.

B.2. Policy Learning

Following PEBBLE (Lee et al., 2021a), we use SAC as the off-policy learning algorithm. We follow the network architectures for the actor and critic and all the hyper-parameter settings in the original paper for policy learning.

B.3. Training details

Our implementation is based on PEBBLE (Lee et al., 2021a). Below we describe the feedback collection schedule for each task. For all tasks, we use a segment size of 1. We

	M	K	N
Open Drawer	40	4000	20000
Soccer	40	4000	20000
Sweep Into	40	4000	20000
CartPole	50	5000	10000
Cloth Fold	50	1000	500
Straighten Rope	100	5000	12000
Pass Water	100	5000	12000

Table 1. Hyper-parameters for feedback learning schedule.

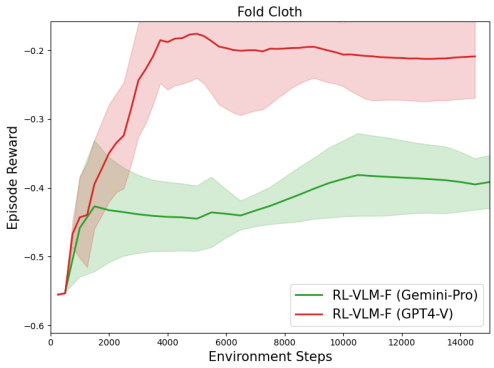

Figure 9. On the Fold Cloth task, we find the performance of GPT-4V to be better than Gemini-Pro, possibly due to the complex visual appearance of the cloth.

summarize the number of queries per feedback session (M in Algorithm 1), the frequency at which we collect feedback in terms of environment steps (K in Algorithm 1), and the maximum budget of queries (N) for each task in Table 1. For Cloth Fold, we have to use a small number of maximum budget of queries due to the quota limitation of GPT-4V.

C. Baselines

C.1. VLM score

For this baseline, we use the same amount of queries (K) at the same frequency (M) as in our method to ask VLM to directly output a score between 0 to 1. The reward model's architecture is the same as our method, except that the model is trained with regression loss to regress to VLM's output score instead of classification loss as done in our method.

C.2. RoboCLIP

In RoboCLIP, the backbone video-language model is S3D (Xie et al., 2018), trained on clips of human activities paired with textual descriptions from the HowTo100M dataset (Miech et al., 2019). Given the assumption that the model generalizes to unseen robotic environments, we applied this baseline solely to the three MetaWorld tasks that contain a robot in the scene. We obtain the implementation directly from the authors. To maintain uniform assumptions across methods, we compare against the RoboCLIP variant that only uses a text description instead of a video demonstration to compute the similarity score with the agent's episode rollout for reward computation. According to the original paper, this text-only variant of RoboCLIP underperforms the video-based method, corroborating the lower performance observed in our tasks.

D. Prompts

D.1. RL-VLM-F and VLM Score

For both RL-VLM-F and VLM Score, we use a unified query template combined with specific task goal descriptions. The templates for RL-VLM-F and VLM Score are shown in Figure 11 and Figure 13:

The only task-specific part in both prompts is the task goal description. We use the same set of descriptions for both methods. We summarize the textual description for each task in Table 2.

D.2. CLIP Score and BLIP-2 Score

The task descriptions for both CLIP Score and BLIP-2 Score baselines are summarized in Table 3. The semantic meaning is almost identical to those used by RL-VLM-F and VLM Score, except that the description is structured differently. For *CartPole*, we used the exact same prompt as in (Rocamonde et al., 2023), since they reported successful learning of this task using that prompt.

D.3. RoboCLIP

For the task descriptions for the RoboCLIP baseline, we followed the format used in the original paper (Sontakke et al., 2023). We summarize the text descriptions in Table 4.

D.4. RL-VLM-F single stage prompt

In Section 6.5 we compared to an ablated version of RL-VLM-F where a single-stage prompting strategy is used. The single-stage prompt used is shown in Figure 12. For a fair comparison, it is kept to be the same as the two-stage prompt with only minor differences.

E. Additional Experiment Results

E.1. GT Task Reward (Oracle) and GT Sparse Reward (Oracle)

To better contextualize the results from different reward models, we test two more baselines, i.e., GT Task Reward (Oracle) and GT Sparse Reward (Oracle). For GT Task Reward (Oracle), we use the original ground-truth human-written reward function with SAC as the RL algorithm to train the policy. For GT Sparse Reward (Oracle), we use

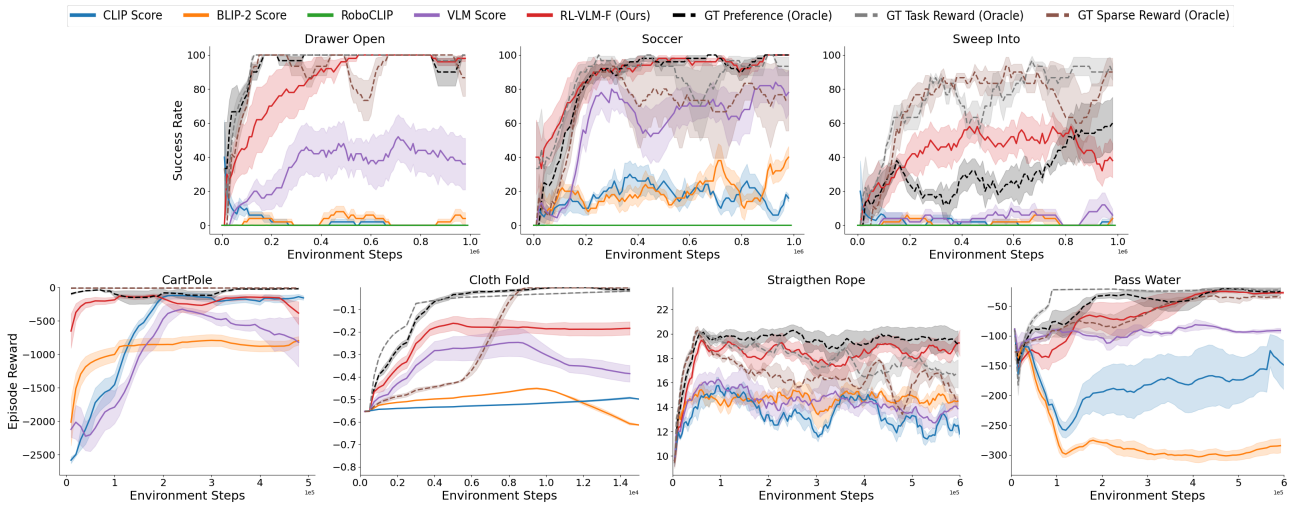

Figure 10. Learning curves of GT Task Reward (Oracle) and GT Sparse Reward (Oracle), along with RL-VLM-F and all baselines.

sparse reward with SAC. The reward is 1 when the goal is achieved and 0 otherwise. The results of GT Task Reward (Oracle) and GT Sparse Reward (Oracle), along with our method and all baselines, are shown in Figure 10. For most tasks, RL-VLM-F 's final performance can match that of using ground-truth reward, highlighting the effectiveness of our method.

E.2. Ablation Study: Influence of Using Different VLMs

For RL-VLM-F and the VLM score baseline, we use Gemini-Pro (Team et al., 2023) as the VLM for all tasks except Fold Cloth. We find Gemini-Pro to perform poorly on Fold Cloth, so we instead use GPT-4V (OpenAI, 2023) as the VLM for this task for both methods. Figure 9 compares the learning performance of Gemini-Pro versus GPT-4V on the task of Fold Cloth. We do observe GPT-4V to achieve much better performance on this task than Gemini-Pro. The poorer performance of Gemini-Pro on this task could be possibly due to the more complex visual reasoning required for deformable cloth.

E.3. More Visualization of the Learned Reward

Here we show the learned reward from RL-VLM-F and the VLM Score baseline, as well as the CLIP and BLIP-2 score along an expert trajectory on three MetaWorld tasks. We compare the learned reward from RL-VLM-F and the VLM Score / CLIP and BLIP-2 score to the ground-truth task progress. The results are shown in Figure 14. For all three tasks, the reward learned by RL-VLM-F aligns the best with the ground-truth task progress.

Prompt Template for RL-VLM-F (ours)

Analysis Template
Consider the following two images:
Image 1:
[Image 1]
Image 2:
[Image 2]

1. What is shown in Image 1?
2. What is shown in Image 2?
3. The goal is to [task description]. Is there any difference between Image 1 and Image 2 in terms of achieving the goal?

Labeling Template
Based on the text below to the questions:
[Repeat the 3 questions in the Analysis Template]
[VLM response]
Is the goal better achieved in Image 1 or Image 2? Reply a single line of 0 if the goal is better achieved in Image 1, or 1 if it is better achieved in Image 2.
Reply -1 if the text is unsure or there is no difference.

Figure 11. Prompt Template for RL-VLM-F.

Task Name	Goal Description
Open Drawer	to open the drawer
Soccer	to move the soccer ball into the goal
Sweep Into	to minimize the distance between the green cube and the hole
CartPole	to balance the brown pole on the black cart to be upright
Cloth Fold	to fold the cloth diagonally from top left corner to bottom right corner
Straighten Rope	to straighten the blue rope
Pass Water	to move the container, which holds water, to be as close to the red circle as possible without causing too many water droplets to spill

Table 2. Goal description used in RL-VLM-F and VLM Score baseline.

Single Stage Prompt Template for RL-VLM-F

Consider the following two images:
Image 1:
[Image 1]
Image 2:
[Image 2]

1. What is shown in Image 1?
2. What is shown in Image 2?
3. The goal is [task description]. Is there any difference between Image 1 and Image 2 in terms of achieving the goal?

Is the goal better achieved in Image 1 or Image 2? Reply a single line of 0 if the goal is better achieved in Image 1, or 1 if it is better achieved in Image 2.
Reply -1 if the text is unsure or there is no difference.

Figure 12. The single stage prompt Template for RL-VLM-F.

Prompt Template for VLM Score

Analysis Template Consider the following image:
[Image]
1. What is shown in the image?
2. The goal is [task description]. On a scale of 0 to 1, the score is 1 if the goal is achieved. What score would you give the image in terms of achieving the goal?
Labeling Template
Based on the text below to the questions:
[Repeat the 3 questions in the Analysis Template]
[VLM response]
Please reply a single line of the score the text has given. Reply -1 if the text is unsure.

Figure 13. Prompt Template for VLM Score.

Task Name	Goal Description
Open Drawer	The drawer is opened.
Soccer	The soccer ball is in the goal.
Sweep Into	The green cube is in the hole.
CartPole	pole vertically upright on top of the cart.
Cloth Fold	The cloth is folded diagonally from top left corner to bottom right corner.
Straighten Rope	The blue rope is straightened.
Pass Water	The container, which holds water, is as close to the red circle as possible without causing too many water droplets to spill.

Table 3. Goal description used in CLIP Score and BLIP-2 Score.

Task Name	Goal Description
Open Drawer	robot opening green drawer
Soccer	robot pushing the soccer ball into the goal
Sweep Into	robot sweeping the green cube into the hole on the table

Table 4. Goal description used in RoboCLIP.

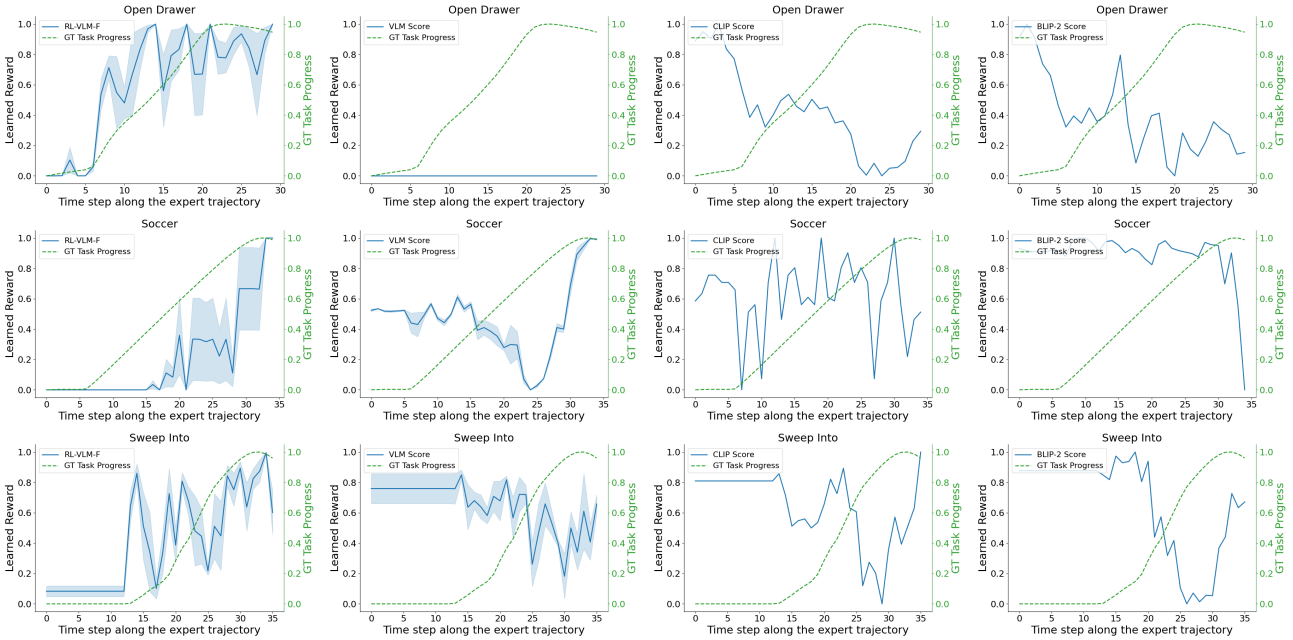

Figure 14. Comparison of learned reward functions from RL-VLM-F and VLM Score, as well as CLIP and BLIP-2 score to the ground-truth task progress along a trajectory rollout on three MetaWorld tasks. From left column to right: reward learned by RL-VLM-F, reward learned by VLM Score, CLIP Score, BLIP-2 Score. From top row to bottom: *Open Drawer*, *Soccer*, and *Sweep Into*. The reward learned by RL-VLM-F aligns the best across all compared methods.