Primitive Skill-based Robot Learning from Human Evaluative Feedback

Ayano Hiranaka*[1], Minjune Hwang*[2], Sharon Lee[2], Chen Wang[2], Li Fei-Fei[2], Jiajun Wu[2], Ruohan Zhang[2]

Abstract— **Reinforcement learning (RL) algorithms face significant challenges when dealing with long-horizon robot manipulation tasks in real-world environments due to sample inefficiency and safety issues. To overcome these challenges, we propose a novel framework, SEED, which leverages two approaches: reinforcement learning from human feedback (RLHF) and primitive skill-based reinforcement learning. Both approaches are particularly effective in addressing sparse reward issues and the complexities involved in long-horizon tasks. By combining them, SEED reduces the human effort required in RLHF and increases safety in training robot manipulation with RL in real-world settings. Additionally, parameterized skills provide a clear view of the agent's high-level intentions, allowing humans to evaluate skill choices before they are executed. This feature makes the training process even safer and more efficient. To evaluate the performance of SEED, we conducted extensive experiments on five manipulation tasks with varying levels of complexity. Our results show that SEED significantly outperforms state-of-the-art RL algorithms in sample efficiency and safety. In addition, SEED also exhibits a substantial reduction of human effort compared to other RLHF methods. Further details and video results can be found at `https://seediros23.github.io/`.**

I. Introduction

Long-horizon manipulation tasks pose a significant challenge for robot learning due to the limitations of reinforcement learning (RL) [1] in physical, real-world environments. While RL has shown remarkable success in simulation environments, its application to real-world robotics is hampered by sample inefficiency and safety concerns, as it is impractical to allow robots to engage in unbridled trial-and-error interactions with the physical environment for extended periods. Sparse reward signals in long-horizon tasks exacerbate these difficulties. In response, recent research has proposed two promising approaches to enhance RL in real-world robot applications: leveraging human evaluation and augmenting robots with primitive skills. Here, we present a novel framework that complements these approaches to tackle long-horizon manipulation tasks in the physical world.

First, different types of human guidance [2] are often introduced to speed up learning and reduce risks. This is known as reinforcement learning from human feedback (RLHF). For instance, humans can provide real-time evaluative feedback ("good", "neutral", or "bad") [3], [4] to indicate how desirable the observed behavior is. Evaluation is an attractive approach for robot learning because it is relatively easy to collect. For physical robot learning tasks, it may be infeasible for a human trainer to define a reward function (for RL) or provide a demonstration (as in imitation learning, IL) due to safety concerns or limited human expertise. Nevertheless, similarly to how sports coaches provide valuable feedback for professional athletes, it is still often possible for humans to guide the learning agent through useful evaluations. This underscores the potential utility of non-expert feedback in skill acquisition and mastery. Even in cases where RL or IL approaches are viable, evaluation can be used to increase the speed of task learning.

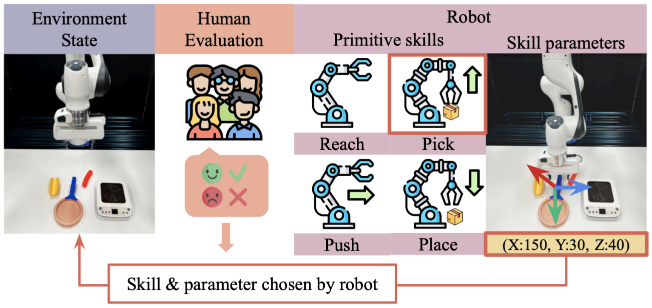

Fig. 1. An overview of skill-based evaluative feedback (SEED). Human trainers provide evaluative feedback on a robotic learning agent's choice of primitive skills and skill parameters.

Another such approach is the augmentation of robots with a pre-defined library [5] of parameterized primitive skills, such as `Pick(obj-A)` or `MoveTo(x,y)`. Although deep RL has the potential to learn a policy with low-level, high-dimensional action space like joint commands, augmenting robots with skills has emerged as a promising approach to improve the efficiency and scalability of robot learning in physical environments.

To overcome the challenges faced by robot learning in real-world manipulation tasks, we propose a novel framework, SEED (**S**kill-based **E**valuative f**EeD**back), as shown in Fig. 1, which synergistically integrates two approaches: learning from human evaluative feedback and primitive skill-based motion control. The combination of primitive skills and evaluative feedback is highly advantageous for RL agents for several reasons. Firstly, by breaking down complex, long-horizon tasks into a sequence of primitive skills, evaluative feedback can provide dense training signals, which makes long-horizon tasks with sparse rewards more tractable. Secondly, evaluating low-level robotic actions can be a time and resource-intensive task for humans [6], but evaluating primitive skills require significantly less effort. Thirdly, primitive skills are intuitive and can reveal the high-level intentions of the robot, allowing humans to evaluate skill choices before they are executed. This "evaluation

*Equal contribution, alphabetically ordered

[1]Department of Mechanical Engineering, [2]Department of Computer Science, Stanford University, CA, USA, `zharu@stanford.edu`

without execution" design is difficult for robots that only have low-level actions. The use of negative feedback from humans to prevent the robot from executing the action can ensure safety during RL training in real-world settings.

We conducted extensive experiments on five manipulation tasks of varying complexities in both the Robosuite [7] simulator and in the real world. Our empirical results demonstrate that SEED significantly outperforms alternative approaches in terms of sample efficiency, safety, and human effort, particularly in long-horizon tasks with sparse rewards.

Our experiments also highlight emerging capabilities of SEED, including zero-shot generalization ability in unseen scene configurations through reward composition. Additionally, SEED outperforms imitation learning-based methods when suboptimal demonstrations or multimodal demonstrations are present.

II. Related Work

Learning from human evaluative feedback for tasks with sparse rewards. In this framework, human trainers monitor the learning process of an agent and provide a learning signal to indicate whether the observed behavior is desirable, in the form of continuous scalar signals [8], binary values [9], [10], or trajectory-level critiques [11] through different means of providing feedback [12], [13], [14], [10], [15]. The agent then learns a policy to maximize positive feedback from humans. Human evaluation is often interpreted as value function [8], [16], or advantage function [17], [18]. Human evaluation can be naturally combined with environment rewards so the agent learns simultaneously from both sources [19], [20], [21]. Applying evaluative feedback-based RL to physical robots is challenging: [6], [22], [23], [24] shows that this is feasible, but without primitive skills, we are limited to shorter-horizon tasks such as reaching and placing, or tasks with low-dimensional state and action space.

Leveraging primitive skills for long-horizon robot learning tasks. A plethora of recent research has explored leveraging parameterized skills to solve long-horizon robot manipulation tasks [25], [5], [26], [27], [28], [29], [30], [31], [32], [33]. Traditional search-based algorithms, such as task and motion planning (TAMP) [34], [35], [36], [37], have been widely utilized for effective multi-step parameterized skill optimization. However, these methods heavily depend on analytically-defined components, such as preimage functions and environment kinematics models. Recent learning-based approaches have been developed, leveraging deep neural networks to learn to solve long-horizon tasks from either human demonstrations [27], [28] or task rewards [29], [31], [32]. Although learning-based methods provide greater flexibility in solving complex tasks, they often require a significant number of demonstrations and well-defined reward functions for learning primitive skills which can be both costly and challenging to scale up.

Our work is closely related to the approach introduced in MAPLE [5], which aims to enhance the sample efficiency of learning manipulation policies by augmenting deep RL with parameterized skills. However, MAPLE still faces limitations that hinder its ability to generalize to novel scenes and imposes safety risks in deployment on real-world robots. In contrast, SEED, addresses these issues by leveraging simple yet effective evaluative feedback as reward signals. Our approach not only significantly improves the sample efficiency of the model but also ensures the training process is safe and user-friendly in novel environments.

III. Method

Our method is designed to overcome challenges in learning long-horizon robot manipulation with deep RL. By leveraging RLHF and parameterized skills, we propose a novel framework — SEED, to improve sample efficiency, reduce human effort, and ensure safety in RL tasks with physical robots.

A. Parameterized Skills

We represent the robot decision-making problem as a Markov decision process denoted by the tuple $\langle \mathcal{S}, \mathcal{A}, P, R, \gamma \rangle$, representing the state space, the action space, the transition function, the reward function, and the discount factor. A policy π is a mapping from observation state space $\mathcal{S}$ to a probability distribution over the robot action space $\mathcal{A}$.

However, naively learning RL robot agents with low-level joint control or operational space control is impractical in real world due to the sample inefficiency and safety concerns. Since skill-augmented RL has shown promising results in solving long-horizon tasks with better sample efficiency, we follow recent works in learning RL agents with parameterized skills [5] and augment our action space $\mathcal{A}$ of the manipulation agent with the following primitive skills (a) and their parameters ($\mathbf{x}$):

- *Reaching:* Moves end-effector to location (x, y, z).
- *Picking:* Picks up an object at location (x, y, z).
- *Placing:* Places an object at location (x, y, z).
- *Pushing:* Reaches to starting location (x, y, z) and pushes end-effector in x or y direction by δ.
- *Gripper Release:* Opens gripper (has no parameters).

By leveraging the parameterized skills, the control policy π is able to focus on learning skill and parameter selection, which bypasses the burden of learning low-level motor control and improves learning efficiency.

Since our decision-making algorithm does not require knowledge of the primitive skills' underlying control mechanism, the skills can be implemented in any method as long as they are robust and adaptive to various situations encountered during the task. In our implementation, each of the skills is predefined by closed-loop controllers that move the end effector in straight-line paths between a series of waypoints. Robosuite's built-in controller and Deoxys's API for Franka Emika Panda arm controller [38] are used for the simulation and real-world experiments respectively. Operational space control (OSC) [39] is used for both scenarios.

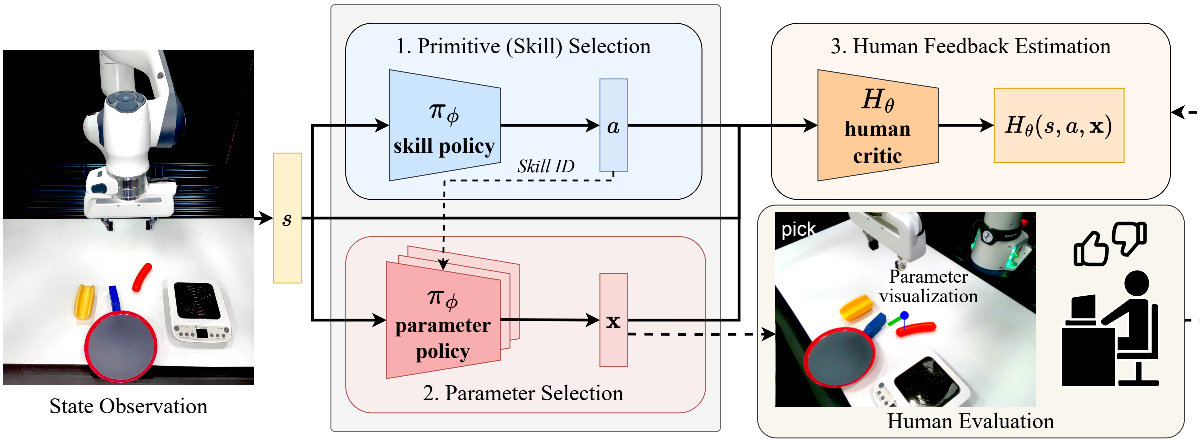

Fig. 2. Neural network architecture for SEED. The network consists of a critic network that predicts human evaluative feedback, a skill actor network that selects primitive skill, and a parameter actor network that selects parameters for the selected skill. Each skill has a unique parameter policy. The skill policy outputs the ID of the selected skill, which is used to invoke the parameter policy corresponding to the selected skill. Outputs of the skill and parameter networks are used by human trainers to provide evaluative feedback. This evaluation signal, combined with the skill and parameter selection, is used to train the human critic.

B. Skill-based evaluative feedback.

Leveraging human evaluative feedback can further improve sample efficiency and safety in long-horizon robot manipulation tasks. TAMER [8], [16] is a widely used framework for RL from evaluative feedback. Instead of using the environment reward, human trainers provide a scalar signal to indicate whether the observed decision is desirable or not. We denote this signal as $H(s, a, \mathbf{x}) \in \{-1, 0, +1\}$, where s is the state vector, a is the one-hot skill selection vector, and $\mathbf{x}$ is the skill parameter vector. Since part of the action space (skill parameters) is continuous, we use Soft Actor-Critic (SAC) [40] as the RL backbone. We use MAPLE [5] as the framework for simultaneously learning a skill-policy that selects a primitive skill, and unique parameter policies for each skill to select the skill parameters. The key difference is that MAPLE is purely based on RL, but SEED's critic is trained using supervised learning where the objective is to predict human evaluative feedback.

The model architecture is shown in Fig. 2. We can estimate human evaluative feedback $\hat{H}(s, a, \mathbf{x})$ using a critic head in SAC. We assume θ, ϕ, ψ, parameterize the critic, the skill selection actor network, and the parameter selection actor network, respectively. The learning objective for the critic is an L2 loss:

$$\mathcal{L}(\theta) = \mathbb{E}_{(s,a,\mathbf{x},H)\sim\mathcal{D}} \|\hat{H}_\theta(s, a, \mathbf{x}) - H(s, a, \mathbf{x})\|_2^2 \quad (1)$$

Similar to MAPLE, we have separate loss functions for the skill actor and the parameter actor:

$$\mathcal{L}(\phi) = \mathbb{E}_{a\sim\pi_\phi} \Big[\alpha_\phi \log(\pi_\phi(a|s)) - \mathbb{E}_{\mathbf{x}\sim\pi_\psi} \hat{H}_\theta(s, a, \mathbf{x})\Big] \quad (2)$$

$$\mathcal{L}(\psi) = \mathbb{E}_{a\sim\pi_\phi} \mathbb{E}_{\mathbf{x}\sim\pi_\psi} \Big[\alpha_\psi \log(\pi_\psi(\mathbf{x}|s, a)) - \hat{H}_\theta(s, a, \mathbf{x})\Big], \quad (3)$$

α_ϕ, α_ψ are the temperature parameter for the maximum entropy objective in SAC [40], [41]. The actors update the policy distribution in the direction suggested by the critic. The agent learns a policy to maximize expected feedback from humans.

C. Balanced replay buffer.

During the initial stages of training with human feedback, the policy network mostly proposes suboptimal actions. As a result, the model's replay buffer will predominantly be filled with actions labeled with negative human feedback. Following prior works on resampling methods for imbalanced learning [42], [43], we sample an equal number of "good" and "bad" samples in each batch during off-policy learning stages, promoting faster convergence of the critic and the actor networks. In the absence of positive samples in the early training stage, we resort to the standard batch sampling approach using negative samples. This method can be considered to be a special case of prioritized replay buffer [44], in which we prioritize sampling transitions with positive human feedback.

D. Facilitating learning with affordances.

MAPLE has shown that adding an affordance score as a small auxiliary reward can facilitate exploration and learning [5], e.g., a pushing skill is only appropriate in the vicinity of pushable objects, and the agent should be penalized with a negative reward for using the skill inappropriately. MAPLE utilizes well-crafted, skill-specific affordance scores that scale with distance from keypoints to encourage the agent to specify position parameters near important sites for each primitive. To accelerate learning in real robot experiments, we adopt a simplified version of MAPLE's affordance score where we add a small penalty of -0.1 when the skill parameter is not near any task-relevant objects. This affordance reward design is more general and involves less human engineering effort.

E. Evaluation without execution.

Primitive skills and their parameters like those defined in MAPLE [5] have clear semantics and are intuitive to humans.

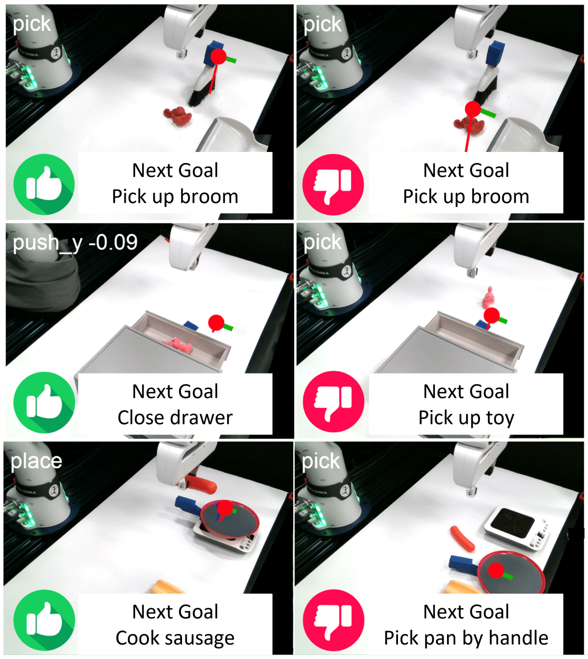

Fig. 3. SEED user interface. The red dot and line show (x, y, z) positions of the chosen skill parameter projected on the 2D camera view. Green bar shows the gripper orientation. For example, in the top right image, the next subgoal is to pick up the broom. Human gives negative feedback because the proposed parameters are too far away from the broom handle.

Algorithm 1 Skill-based Evaluative Feedback (SEED)

```
Initialize network weights for skill policy $\pi_\phi(a|s)$, parameter policy $\pi_\psi(\mathbf{x}|s, a)$, and human feedback critic network $H_\theta(s, a, \mathbf{x})$; initialize replay buffer $\mathcal{D}$
for episode = 1, ..., N do
    Initialize $t \leftarrow 0$
    while episode not terminated do
        Sample skill $a_t = \pi_\phi(\cdot|s_t)$
        Sample skill parameter $\mathbf{x}_t = \pi_\psi(\cdot|s_t, a_t)$
        Query human for evaluative feedback $H_t(s_t, a_t, \mathbf{x}_t)$
        Store transition $(s_t, a_t, \mathbf{x}_t, H_t)$ in $\mathcal{D}$
        if $H_t(s_t, a_t, \mathbf{x}_t) = +1$ then
            Execute $(a_t, \mathbf{x}_t)$
        end if
        Sample a minibatch of $(s_t, a_t, \mathbf{x}_t, H_t)$ from $\mathcal{D}$ to perform gradient updates on $\theta, \phi, \psi$ based on Eqs 1, 2, and 3.
        $t \leftarrow t + 1$
    end while
end for
```

Therefore humans can evaluate robot's selection of skills and parameters even before the robots execute the action.

The interface for evaluation without execution is shown in Fig. 3. During the training process, a depiction of the robot's workspace, marked with annotations of the agent's skill and parameter selections, is presented to a human evaluator. During the initial stage of training, the human trainer evaluates each action as "good" or "bad" based solely on the visual representation of the robots' skills and parameters. Only when the human is confident of the robot's ability to make good decisions, will the robot be allowed to execute the action, making training more safe and efficient. The full pipeline of SEED is shown in Algorithm 1.

IV. Experiments Setups

A. Baselines

To understand the effect of human evaluation and to compare skill-based learning with low-level action-based learning, we compare SEED with the following baselines:

- **SAC** [40] is the standard actor-critic algorithm that optimizes the stochastic policy with entropy regularization.
- **TAMER** [8] is an existing framework for RLHF. To adapt TAMER to continuous actions space, we used TAMER+SAC, which replaces the standard critic with a human feedback critic which estimates a scalar signal from human trainers. Human trainer evaluates every low-level step, or a single command to the OSC controller. The agent is trained on dense human rewards and sparse environment rewards.
- **MAPLE** [5] is an existing framework for RL with behavior primitives. Compared to SAC, MAPLE replaces the standard actor with a hierarchical policy that has a high-level policy that determines the skill selection, and a low-level parameter policy that determines the parameter selection given the primitive skill. This algorithm is trained on sparse environment rewards and does not involve human evaluations.
- **MAPLE-aff** is a variant of MAPLE that leverage affordance score as a dense reward signal [5]. We intend to show that human feedback is a more powerful learning signal than this hand-designed affordance reward. This algorithm has no human evaluations.

TAMER and MAPLE can be viewed as ablated versions of SEED. The former does not have primitive skills and the latter does not have human feedback. Training hyperparameters are shown in Table I. *Train frequency* refers to the number of environment steps between each gradient descent, in which we take *gradient steps* times of gradient updates.

Synthetic human feedback in simulation. We utilized synthetic feedback in simulated environments instead of real human feedback as in real-world environments. Synthetic feedback assumes idealized human feedback behaviors, which allows us to focus on the learning algorithms themselves and perform more extensive and controlled experiments. For SEED, we utilize predefined heuristics (i.e., skill-specific affordance reward) to generate binary human feedback for each high-level step.

In contrast, TAMER was trained with an oracle which is a fully trained SAC agent, since heuristics for low-level actions are difficult to specify. Specifically, the TAMER agent chooses an action a in state s, and the oracle chooses an action a^*. The oracle SAC computes the Q values for these actions: $Q(s, a)$ and $Q(s, a^*)$. If the learning agent chooses an action that has a Q-value close enough to $Q(s, a^*)$, it is a good action and the agent should receive positive feedback. Otherwise, it should receive negative feedback: $H(s, a) = +1, \text{if } \; Q(s, a) \geq \alpha Q(s, a^*) \text{ and} -1 \text{ otherwise}$. The α value is initially set as 0.999, and it increases over time to encourage the agent to learn to choose better actions during training.

Human feedback in real world. For each of the three tasks, a single human trains the agent twice by providing evaluation signals via a keyboard key press.

B. Long-horizon manipulation tasks

The robot we use is a Franka Emika robot arm. The simulation tasks are implemented in the Robosuite [7]. We evaluate SEED and baseline algorithms in simulation and in the real world, in the following long-horizon tasks (as shown in Fig. 4). Because our skill implementation is independent of robot proprioception, this information is omitted for SEED and MAPLE, enabling reduced state space.

`Reaching` is a simulation task in which the robot has to move its gripper to the fixed area from a random starting location. The state, $s \in \mathbb{R}^4$ represents the gripper position in 3D and a binary state indicating the gripper status for all algorithms. In this environment, we provide *Reaching* and *Gripper Release* as available skills to skill-based algorithms; however, the desired outcome is for the model to learn to solely rely on the reaching primitive. This task is relatively short-horizon, and is mainly a sanity check for our implementation of baseline algorithms since many of them have zero performance in more challenging tasks described below.

`Stacking` is a simulation task in which the robot has to stack a small block on top of a larger block. The initial locations of both blocks and the robot are randomized. For low-level baselines, the state space $s \in \mathbb{R}^{10}$ comprises 3D positions of the gripper and blocks, as well as a binary state indicating whether the gripper is closed. On the other hand, MAPLE and SEED utilize a reduced state space $s \in \mathbb{R}^6$, which includes 3D positions for both blocks. In this task, *Picking* and *Placing* are available.

`Sweeping` is in the real world. The robot is required to pick up a broom and sweep a toy into a dustpan. For SAC and TAMER, the state space ($s \in \mathbb{R}^{10}$) includes the 3D positions of the gripper and broom, 2D position of the toy, the gripper state, and a flag indicating whether the broom is being grasped. MAPLE and SEED's state space omits the gripper position and state. Available primitive skills for this task include *Picking* and *Pushing*.

`Collecting-Toy` is in the real world. The robot is tasked with picking up a toy, placing it in a drawer, and pushing the drawer closed. For SAC and TAMER, the state space ($s \in \mathbb{R}^{10}$) includes the 3D positions of the gripper and toy, the gripper state, the delta value of the drawer's current position from the closed position, and flags indicating whether the toy is being grasped and whether it is in the drawer. MAPLE and SEED omit the gripper position and state from their state space. Available primitive skills for this task include *Picking*, *Pushing*, and *Placing*.

TABLE I

TRAINING HYPERPARAMETERS (SIM / REAL)

	SAC	TAMER	MAPLE	SEED
learning rate	3e-5 / -	3e-5 / 3e-4	3e-3 / 3e-3	3e-3 / 3e-3
batch size	256 / -	256 / 1024	256 / 1024	256 / 1024
γ (discount rate)	0.99 / -	0.99 / 0.99	0.99 / 0.4	0.99 / 0.4
gradient steps	5 / -	5 / 30	5 / 3	5 / 30
train frequency	1 / -	1 / 25	1 / 2	1 / 25

`Cooking-Hotdog` is in the real world. The task requires the robot to perform a series of actions, including picking up a skillet and placing it on a stove, placing a sausage on the skillet, picking up the sausage again, and placing it in a bun. In both SAC and TAMER, the state space ($s \in \mathbb{R}^{15}$) contains 3D positions of the gripper, sausage, and skillet, as well as flags indicating the gripper state and the status of various steps in the task. However, the state space for MAPLE and SEED does not include the gripper position or state. Available skills for this task are *Picking* and *Placing*.

C. Hardware setup

The robot we use is a Franka Emika robot arm. The experiment is run on a PC operating on Ubuntu 20.04 with Intel® Core i7-7700K CPU and Nvidia® GTX 1080 Ti graphics card. For object position estimation, two calibrated Intel® Realsense™Depth Camera D415 are used.

V. Results

Evaluation metrics. The primary performance metric utilized in our simulations is the task success rate, as the algorithms can be trained until convergence. To measure the task success rate, we conducted 100 evaluations throughout the training process, each consisting of 10 rollouts.

In real-world scenarios, extensive evaluations prove to be costly. As a result, we have adopted an approach wherein we measure the number of successes over the course of the training steps. This allows us to monitor the progress of the algorithms in real-time without incurring significant expenses. Furthermore, it is crucial to consider safety concerns while evaluating the performance of robots. We have identified two critical safety scenarios that need to be monitored: a safety violation resulting in damage to the robot or objects, and a safety violation leading to task failure. In the former scenario, the emergency stop button is pressed; in the latter case, a manual reset is required to restore normalcy. The count of safety violations for each of the trials is documented.

SEED is sample efficient. Simulation experiment results are shown in Fig. 5 comparing the performance of several RL algorithms on two different robotic manipulation tasks: `Reaching` and `Stacking`. In the simple, short-horizon task, `Reaching`, both TAMER and SEED algorithms exhibit rapid learning which highlights the advantage of using evaluative feedback. As expected, MAPLE and MAPLE-aff algorithms demonstrate faster learning rates than the SAC algorithm in this task. However, in the more complex and challenging `Stacking` task, SEED outperforms all other

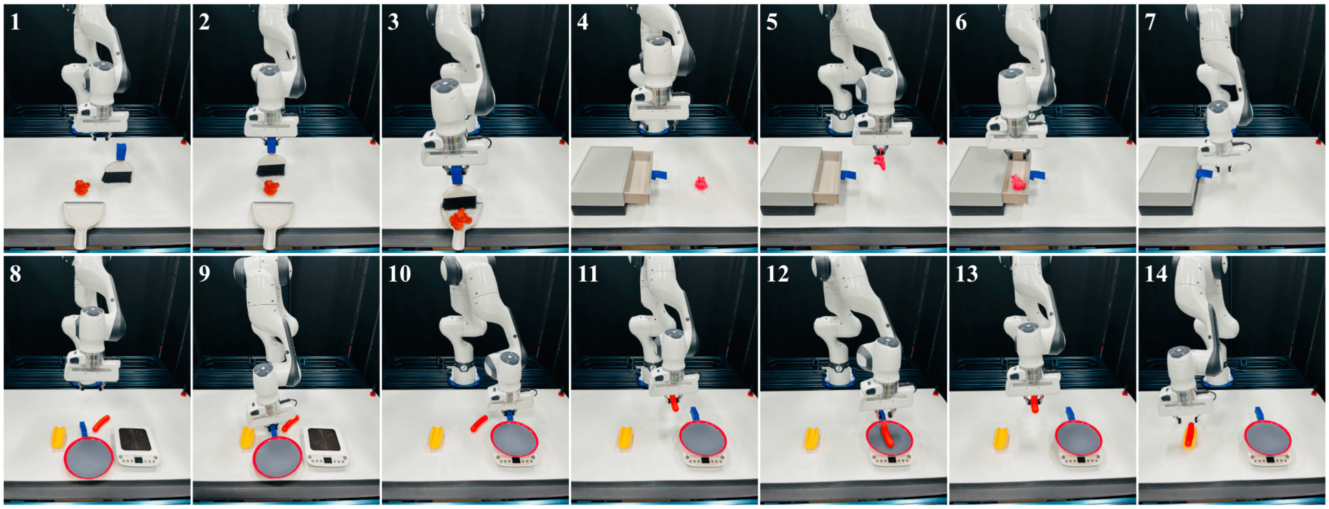

Fig. 4. Visualizations of real-world, long-horizon manipulation tasks with intermediate steps. Top row: `Sweeping` (1-3) and `Collecting-Toy` (4-7) tasks; bottom row (8-14): `Cooking-Hotdog`, the task with the longest horizon.

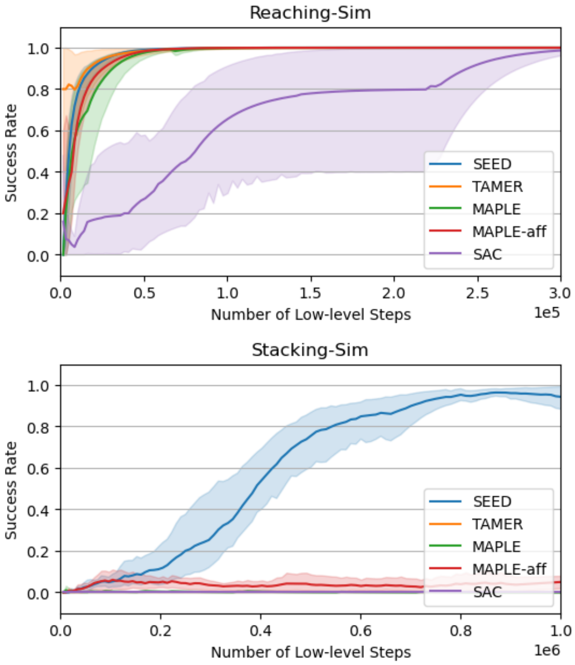

Fig. 5. Average success rate over training steps (low-level steps, one low-level action per step) for `Reaching` and `Stacking` in simulation. SEED learns to solve the tasks more efficiently compared to the baselines. Error bars indicate the standard error of the means ($n = 5$).

algorithms by a substantial margin. It is worth noting that although MAPLE-aff may eventually learn the `Stacking` task after four million steps, as reported in the original research [5], SEED learns to solve the task in only 800,000 steps.

While the original work on MAPLE-aff demonstrated the advantage of skill-based actions in sim2real transfer [5], this approach is only applicable when a digital twin setting can be prepared. However, this is not always possible, calling for methods that allow the robot to train from scratch in the real world.

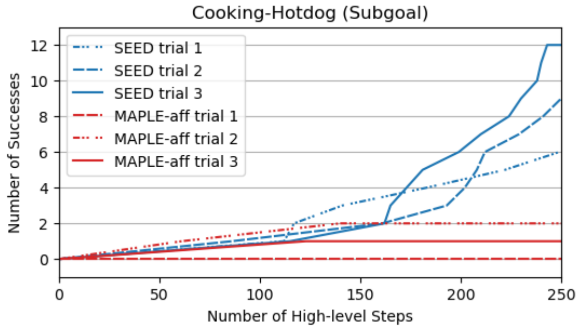

Fig. 6. Number of successes over training steps (high-level steps, one skill per step) for the first goal in `Cooking-Hotdog` (picking up the skillet). SEED learns to solve this subgoal more efficiently than MAPLE-aff.

The first challenge in training MAPLE-aff from scratch in real-world settings is the time-consuming process. To overcome this challenge, the SEED algorithm employs evaluation without execution and relies on human feedback to optimize training time, which is around ten times faster than MAPLE-aff. The second challenge of training MAPLE-aff is that physical robots pose a safety risk. Nonetheless, to compare the performance of SEED and MAPLE-aff, we conducted experiments using a simplified version of the `Cooking-Hotdog` task, which involves only the first subgoal of picking up the skillet. The results shown in Fig. 6, indicate that on average, SEED can successfully complete the subgoal nine times within 250 high-level steps, while MAPLE-aff only succeeds once. Given the low success rate of MAPLE-aff and safety concerns associated with continuing the experiments, we did not run MAPLE-aff on the entire task.

SEED ensures better safety. Table II presents the safety violation ratio, which is given by the number of safety violations divided by the total number of decision steps. Our

TABLE II

SAFETY VIOLATION RATIO.

	TAMER	MAPLE-aff	SEED
`Sweeping`	0.25%	8.50%	1.11%
`Collecting-Toys`	0.26%	1.19%	0.40%
`Cooking-Hotdog`	0.49%	3.54%	0.51%

analysis reveals that SEED exhibits significantly lower safety risks when compared to MAPLE-aff. The chance of safety violation in MAPLE-aff is about 3 to 7 times compared to SEED. The main reason is that SEED enables evaluation without execution, which can prevent dangerous actions from being executed. The findings of our study indicate that SEED holds great promise in enhancing safety in robot learning, a crucial consideration for real-world applications.

It is worth noting that in the case of TAMER, one decision step is equivalent to one low-level step, whereas in MAPLE-aff and SEED, one decision step corresponds to one primitive skill step, which typically involves around 100 low-level steps. Therefore the risk of TAMER is underestimated here (and its performance is zero as shown in Fig. 7).

SEED significantly reduces human effort. Due to concerns over the MAPLE-aff algorithm's sample efficiency and potential safety risks in a physical robot setting, we opted to exclude it from the rest of real-world experiments. Instead, we focused solely on training the TAMER and SEED algorithms until task completion. We compare SEED and TAMER based on the amount of human effort required in real-world experiments. Figure 7 displays the results obtained from providing both TAMER and SEED with the same quantity of human feedback. Notably, SEED was able to learn effectively within the given amount of feedback, while TAMER failed to achieve any successful task completion. Remarkably, for all three long-horizon tasks, SEED has successfully learned to solve them. Additionally, human trainers adapt quickly and learn how to provide better feedback for the robots, as evidenced by the much better results observed in the second trial compared to the first. Please refer to the supplemental video for a detailed analysis of the learning results.

VI. CONCLUSION

Real-world manipulation tasks that involve long horizons present numerous challenges to robotic learning agents, including safety guarantee and sample efficiency. In addition, if human data are required, as in the case of RLHF, it is essential to minimize the associated human effort. This work presents SEED, an innovative approach that synergistically integrates human evaluative feedback and primitive skills to enhance the efficiency and safety of real-world reinforcement learning. The proposed method overcomes the challenges associated with real-world long-horizon manipulation tasks, thereby paving the way for future research to scale up robot learning with improved safety guarantees and affordable human costs.

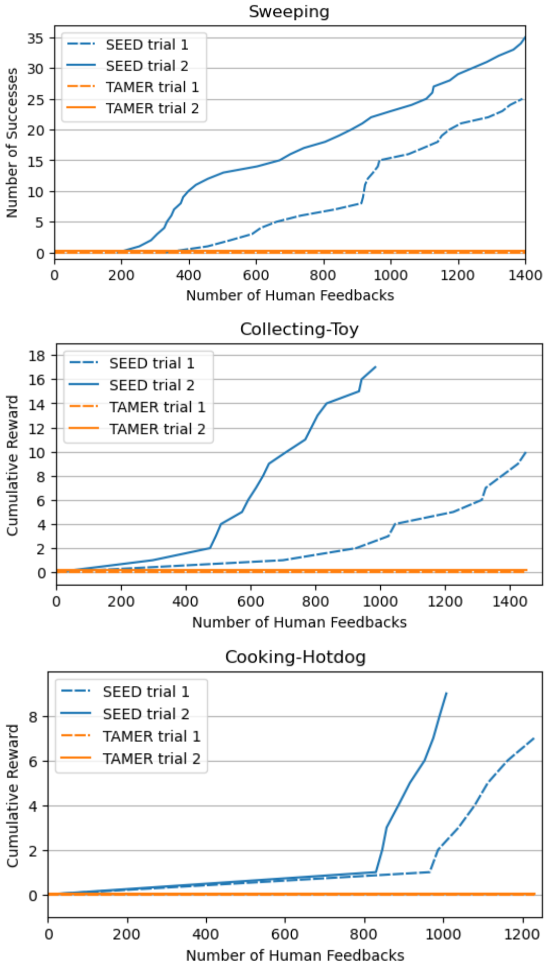

Fig. 7. Number of successes over the number of human feedback for three real-world tasks. SEED learns to solve all tasks efficiently while TAMER cannot. The experiments are terminated when the maximum number of steps is reached or the agent has learned to solve the task consistently.

ACKNOWLEDGMENTS

This work was in part supported by ONR MURI N00014-22-1-2740, ONR MURI N00014-21-1-2801, the Stanford Institute for Human-Centered AI (HAI), Amazon, Analog Devices, JPMC, and Salesforce. Ruohan Zhang is partially supported by Wu Tsai Human Performance Alliance Fellowship.

REFERENCES

[1] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[2] Ruohan Zhang, Faraz Torabi, Lin Guan, Dana H Ballard, and Peter Stone. Leveraging human guidance for deep reinforcement learning tasks. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 6339–6346. AAAI Press, 2019.

[3] Michael L Littman. Reinforcement learning improves behaviour from evaluative feedback. *Nature*, 521(7553):445–451, 2015.

[4] Jinying Lin, Zhen Ma, Randy Gomez, Keisuke Nakamura, Bo He, and Guangliang Li. A review on interactive reinforcement learning from human social feedback. *IEEE Access*, 8:120757–120765, 2020.

[5] Soroush Nasiriany, Huihan Liu, and Yuke Zhu. Augmenting reinforcement learning with behavior primitives for diverse manipulation tasks. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 7477–7484. IEEE, 2022.

[6] Ruohan Zhang, Dhruva Bansal, Yilun Hao, Ayano Hiranaka, Jialu Gao, Chen Wang, Roberto Martín-Martín, Li Fei-Fei, and Jiajun Wu. A dual representation framework for robot learning with human guidance. In *Conference on Robot Learning*, pages 738–750. PMLR, 2023.

[7] Yuke Zhu, Josiah Wong, Ajay Mandlekar, and Roberto Martín-Martín. robosuite: A modular simulation framework and benchmark for robot learning. *arXiv preprint arXiv:2009.12293*, 2020.

[8] W Bradley Knox and Peter Stone. Interactively shaping agents via human reinforcement: The tamer framework. In *Proceedings of the fifth international conference on Knowledge capture*, pages 9–16. ACM, 2009.

[9] Andrea L. Thomaz and Cynthia Breazeal. Reinforcement learning with human teachers: Evidence of feedback and guidance with implications for learning performance. In *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 1*, pages 1000–1005. AAAI Press, 2006.

[10] Anis Najar, Olivier Sigaud, and Mohamed Chetouani. Interactively shaping robot behaviour with unlabeled human instructions. *Autonomous Agents and Multi-Agent Systems*, 34:1–35, 2020.

[11] Yuchen Cui and Scott Niekum. Active reward learning from critiques. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6907–6914. IEEE, 2018.

[12] Charles Isbell, Christian R Shelton, Michael Kearns, Satinder Singh, and Peter Stone. A social reinforcement learning agent. In *Proceedings of the fifth international conference on Autonomous agents*, pages 377–384. ACM, 2001.

[13] Ana C Tenorio-Gonzalez, Eduardo F Morales, and Luis Villaseñor-Pineda. Dynamic reward shaping: training a robot by voice. In *Ibero-American conference on artificial intelligence*, pages 483–492. Springer, 2010.

[14] Shane Griffith, Kaushik Subramanian, Jonathan Scholz, Charles L Isbell, and Andrea L Thomaz. Policy shaping: Integrating human feedback with reinforcement learning. In *Advances in neural information processing systems*, pages 2625–2633, 2013.

[15] Iretiayo Akinola, Zizhao Wang, Junyao Shi, Xiaomin He, Pawan Lapborisuth, Jingxi Xu, David Watkins-Valls, Paul Sajda, and Peter Allen. Accelerated robot learning via human brain signals. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3799–3805. IEEE, 2020.

[16] Garrett Warnell, Nicholas Waytowich, Vernon Lawhern, and Peter Stone. Deep tamer: Interactive agent shaping in high-dimensional state spaces. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*. AAAI Press, 2018.

[17] James MacGlashan, Mark K Ho, Robert Loftin, Bei Peng, Guan Wang, David L Roberts, Matthew E Taylor, and Michael L Littman. Interactive learning from policy-dependent human feedback. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2285–2294. JMLR. org, 2017.

[18] Dilip Arumugam, Jun Ki Lee, Sophie Saskin, and Michael L Littman. Deep reinforcement learning from policy-dependent human feedback. *arXiv preprint arXiv:1902.04257*, 2019.

[19] W Bradley Knox and Peter Stone. Combining manual feedback with subsequent mdp reward signals for reinforcement learning. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*, pages 5–12. International Foundation for Autonomous Agents and Multiagent Systems, 2010.

[20] W Bradley Knox and Peter Stone. Reinforcement learning from simultaneous human and mdp reward. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 475–482. International Foundation for Autonomous Agents and Multiagent Systems, 2012.

[21] Riku Arakawa, Sosuke Kobayashi, Yuya Unno, Yuta Tsuboi, and Shin-ichi Maeda. Dqn-tamer: Human-in-the-loop reinforcement learning with intractable feedback. *arXiv preprint arXiv:1810.11748*, 2018.

[22] W Bradley Knox, Peter Stone, and Cynthia Breazeal. Training a robot via human feedback: A case study. In *Social Robotics: 5th International Conference, ICSR 2013, Bristol, UK, October 27-29, 2013, Proceedings 5*, pages 460–470. Springer, 2013.

[23] Zizhao Wang, Xuesu Xiao, Garrett Warnell, and Peter Stone. Apple: Adaptive planner parameter learning from evaluative feedback. *IEEE Robotics and Automation Letters*, 6(4):7744–7749, 2021.

[24] Anis Najar, Olivier Sigaud, and Mohamed Chetouani. Training a robot with evaluative feedback and unlabeled guidance signals. In *2016 25th IEEE international symposium on robot and human interactive communication (RO-MAN)*, pages 261–266. IEEE, 2016.

[25] Rohan Chitnis, Tom Silver, Joshua B Tenenbaum, Tomas Lozano-Perez, and Leslie Pack Kaelbling. Learning neuro-symbolic relational transition models for bilevel planning. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4166–4173. IEEE, 2022.

[26] Yifeng Zhu, Jonathan Tremblay, Stan Birchfield, and Yuke Zhu. Hierarchical planning for long-horizon manipulation with geometric and symbolic scene graphs. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6541–6548. IEEE, 2021.

[27] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Cliport: What and where pathways for robotic manipulation. In *Conference on Robot Learning*, pages 894–906. PMLR, 2022.

[28] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Conference on Robot Learning*, pages 785–799. PMLR, 2023.

[29] Danfei Xu, Ajay Mandlekar, Roberto Martín-Martín, Yuke Zhu, Silvio Savarese, and Li Fei-Fei. Deep affordance foresight: Planning through what can be done in the future. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6206–6213. IEEE, 2021.

[30] Chen Wang, Danfei Xu, and Li Fei-Fei. Generalizable task planning through representation pretraining. *IEEE Robotics and Automation Letters*, 7(3):8299–8306, 2022.

[31] Shuo Cheng and Danfei Xu. Guided skill learning and abstraction for long-horizon manipulation. In *CoRL 2022 Workshop on Learning, Perception, and Abstraction for Long-Horizon Planning*, 2022.

[32] Christopher Agia, Toki Migimatsu, Jiajun Wu, and Jeannette Bohg. Stap: Sequencing task-agnostic policies. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7951–7958. IEEE, 2023.

[33] Chengshu Li, Ruohan Zhang, Josiah Wong, Cem Gokmen, Sanjana Srivastava, Roberto Martín-Martín, Chen Wang, Gabrael Levine, Michael Lingelbach, Jiankai Sun, et al. Behavior-1k: A benchmark for embodied ai with 1,000 everyday activities and realistic simulation. In *Conference on Robot Learning*, pages 80–93. PMLR, 2023.

[34] Tomás Lozano-Pérez and Leslie Pack Kaelbling. A constraint-based method for solving sequential manipulation planning problems. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3684–3691. IEEE, 2014.

[35] Marc Toussaint. Logic-geometric programming: An optimization-based approach to combined task and motion planning. In *IJCAI*, pages 1930–1936, 2015.

[36] Caelan Reed Garrett, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. Pddlstream: Integrating symbolic planners and blackbox samplers via optimistic adaptive planning. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 30, pages 440–448, 2020.

[37] Caelan Reed Garrett, Rohan Chitnis, Rachel Holladay, Beomjoon Kim, Tom Silver, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Integrated task and motion planning. *Annual review of control, robotics, and autonomous systems*, 4:265–293, 2021.

[38] Yifeng Zhu, Abhishek Joshi, Peter Stone, and Yuke Zhu. Viola: Imitation learning for vision-based manipulation with object proposal priors. In *Conference on Robot Learning*, pages 1199–1210. PMLR, 2023.

[39] Oussama Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal on Robotics and Automation*, 3(1):43–53, 1987.

[40] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018.

[41] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

[42] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research (JAIR)*, 16(1):321–357, jun 2002.

[43] Peter C.R. Lane, Daoud Clarke, and Paul Hender. On developing robust models for favourability analysis: Model choice, feature sets and imbalanced data. *Decision Support Systems*, 53(4):712–718, 2012.

[44] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. In *4th International Conference on Learning Representations, ICLR*, 2016.