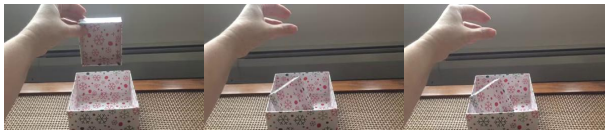
A Training Details

Dataset Details. In prior work [6], the action recognition dataset Something-Something v2 (SSv2) [3] was transformed into a video and language dataset. In our work, we employ 174 templates from SSv2 (e.g., "Opening [Something]") as action descriptions for the videos, and utilize annotated labels from SSv2 (e.g., "Opening drawer") as task descriptors for the videos. In the experiments of the first three chapters, we select videos from the following 12 different human tasks which represent different actions, where each task has 853-3170 training videos: 1) Closing [Something], (2) Moving [Something] away from the camera, (3) Moving [Something] towards the camera, (4) Opening [Something], (5) Pushing [Something] from left to right, (6) Pushing [Something] from right to left, (7) Poking [Something] so lightly that it doesn't or almost doesn't move, (8) Moving [Something] down, (9) Moving [Something] up, (10) Pulling [Something] from left to right, (11) Pulling [Something] from right to left, (12) Pushing [Something] with [Something]. As delineated in DVD [2], the tasks chosen are apt for single-arm setups in desktop scenarios, covering diverse actions. The selection of the initial seven tasks aligns with their pertinence to simulated environments, while the inclusion of the remaining tasks, not guided by specific criteria, allows for their potential replacement with an alternate set of eight suitable tasks. Similarly, in the experiment of Section 4.5, we select videos from the 68 different human tasks related to the Concept2Robot simulation environment. In our experimentation, for each specific task, Adapt2Reward is trained using all corresponding human videos from the Something-Something V2 dataset. To enhance the model's capacity for detailed object recognition, we incorporate task descriptions with annotated labels, coupled with the videos, as model inputs.

template: Opening [Something]
label: Opening drawer



template: Dropping [Something] into [Something]
label: Dropping box into box

Fig. 1: Two samples in Something-Something v2.

In the Meta-World simulation environment, our Adapt2Reward is trained with 280 successful and 280 failed robotic videos per task. The successful robot data consists of two parts: the first part is collected by generating random action

sequences and filtering them using ground truth rewards, and the second part is gathered by a hard-coded script with uniform noise from $[-0.03, 0.03]$. It performs flawlessly successfully by manipulating a robot arm. We posited that the failure data should encompass both failed attempts during random exploration and those nearly successful to reflect realistic learning scenarios. Therefore, the failure robotic data are partly gathered by random shooting and a ground truth reward to each task and partly collected by Soft Actor-Critic (SAC) [4] trained on these ground truth rewards, capturing a comprehensive spectrum of failure modes. For the Concept2Robot simulation environment, in addition to varying amounts of human videos, Adapt2Reward is trained on 500 robot success videos and 500 robot failure videos of 12 tasks. These videos are collected by training single-task policies with a hand-designed, task-specific, binary reward signal. In particular, we also collect failure data during random exploration.

Training Settings. We use the video-language model Singularity [6], pre-trained on 17M human vision-language datasets, and fine-tune it on the Something-Something-V2 (SS-V2) dataset [3]. To recognize fine-grained temporal relationships between video frames, we utilize its multi-frame variant, Singularity-Temporal. This variant introduces a two-layer temporal transformer encoder after the vision encoder, feeding its outputs into the multi-modal encoder. In our experiments, we freeze the text encoder and fine-tune the other modules on mixed human-robot data. We use a learning rate of $5e-6$ with AdamW, momentum values of 0.9 and 0.999, and a weight decay of 0.02. Notably, we employ a higher learning rate of 0.001 for prompts, as they are randomly initialized. Our Adapt2Reward model is trained for 10 epochs using a cosine decay schedule. We set our batch sizes $B_h = B_r = B_f = 8$ and $K = 3$ as default parameters. We maintain a uniform trade-off of 1 across all losses, including $\mathcal{L}_{CDC}$, $\mathcal{L}_{VLC}$, and $\mathcal{L}_{fVLC}$. Additionally, we use 4 frames for both training and inference, and apply random resize, crop, and color jitter as augmentation techniques across all videos.

Training FitVid. To evaluate Adapt2Reward's performance on potentially unseen tasks with a given language instruction, we employ visual model predictive control with FitVid [1] visual prediction models trained on datasets autonomously collected in each environment. FitVid introduces a convolutional non-hierarchical variational model with a fixed prior that can significantly overfit the video prediction datasets. We use the same architecture and default hyperparameters as the original paper.

However, long-distance predictions remain a major cause of low-quality predictions by models. In our experiments, we do not need images of every frame to calculate rewards. Therefore, unlike FitVid, which predicts the next frame of the video frame by frame, we splice 4 frames of actions and feed them to the model together with the current state image, and the model will output the state image after executing the 4 actions. Therefore, a task of predicting 60 frames frame by frame based on an initial state of one frame will be transformed into a task of predicting 16 frames from the initial state. For each of the four simulation environments in which we evaluate Adapt2Reward, we collect 10,000 random episodes, each with 60 total frames, of the agent interacting in that environment

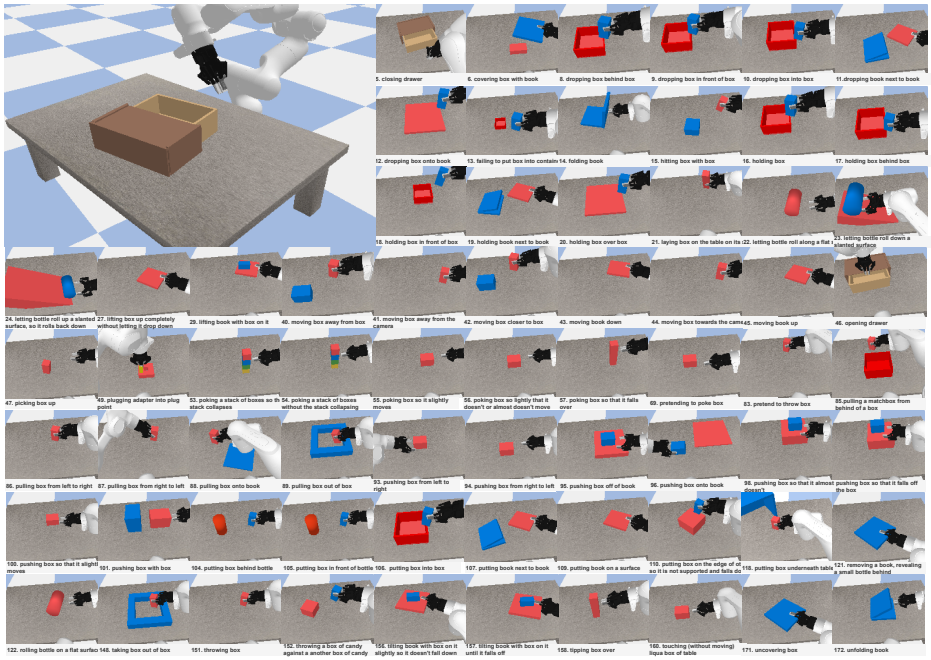


Fig. 2: The modified Concept2Robot environment in our experiments

and train FitVid for 200,000 epochs on all of the data. The models are trained to predict the next 12 frames given an input of init frames. In the inference, the model predicts the next 16 frames given an input of init frames.

B Experimental Details

Meta-World simulation Environments. We utilize the same MuJoCo simulation based on the Meta-World [10] environments as used in DVD [2]. In this simulation, the state space comprises RGB image observations with dimensions $[180, 120, 3]$. Our action space includes continuous actions for the robot's gripper's linear and angular velocities, and discrete actions for the gripper's open/close movement, resulting in a total of five dimensions.

Concept2Robot simulation Environments. Following the environments in Concept2Robot [9], we selected 68 out of the 174 tasks that fit our environment setting. We excluded 10 problematic environments that either couldn't be executed or had irrelevant actions and commands. Additionally, we modify the environment and manually craft reward functions for some tasks to ensure their actions and verbal commands are relevant and unaffected by extreme object states. The remaining 96 tasks were unsuitable as they either required dual-arm manipulation or were challenging to simulate in PyBullet, such as "tearing something". For the selected tasks, we used the same task IDs as in SSv2, detailed in

Figure 2. Unlike Concept2Robot [9], we used labels as textual descriptions, for example, "dropping box into box", instead of templates. We simulated each of the 68 tasks in PyBullet. Our robot model is a 7-DoF Franka Panda robot arm equipped with a two-fingered Robotiq 2F-85 gripper. A static camera captures the environment state as RGB images, downscaled to 120×160. At each reset after an RL episode, the manipulation objects in each environment are initialized with a random pose within manually set bounds for each task.

Details of Evaluation. In the previous study, DVD, trajectories of sixty frames were divided into three twenty-frame segments for action planning. Each twenty-frame segment had 100 action sequences uniformly and randomly sampled. The trajectory with the highest similarity to human demonstrations, as computed by the DVD algorithm, was chosen for environmental execution. The initial state for each subsequent segment was the final state from the previous trajectory. We contend that this segmented approach might lead to longer inference times, particularly with video prediction models that have a high number of parameters. We propose directly comparing the full sixty-frame video with human demonstrations, arguing that it is more intuitive than comparing segmented twenty-frame parts. Segment planning leads to the robot's task performance video being divided into three parts, with each part's similarity to the human demo calculated separately. This could yield inaccurate similarity scores, as DVD trains on unsegmented robot and human videos.

In our experiment, we use continuous sixty-frame trajectories for uninterrupted planning. We randomly generate 300 action sequences and employ the FitVid model to predict 300 six-second videos from initial images. Based on the task's language command, we calculate the rewards for these videos and select the highest-scoring trajectory for execution. For DVD, we execute the trajectory most similar to human demonstrations. For Concept2Robot, we execute the trajectory with the highest classification score in the environment.

We evaluate the following four target tasks: 1) Closing drawer, which is defined as the last frame in the 60-frame trajectory having the drawer pushed in to be less than 0.05, where it starts open at 0.07; 2) Moving cup away from the camera, move the cup away from the camera at least 0.1 distance compared to the initial distance; 3) Moving the handle of the faucet, moving the handle more than 0.01 distance, where it starts at 0; 4) Pushing cup from left to right, pushing the cup at least 0.05 distance from left to right. We run 100 trials for 3 different seeds for each task for every method in all experiments.

In the Concept2Robot environments, following the Concept2Robot [9], we combine Deep Deterministic Policy Gradients (DDPG) [7] with the Cross Entropy Method (CEM) [8] to improve action selection. The DDPG critic network approximates the Q-value function, and the actor network's output initiates the CEM. CEM then searches for a better action near the initial one, aiming to optimize decision-making [5]. Similar to Concept2Robot [9], we use task-specific success metrics to evaluate the tasks. We report the success rate for each task in 100 trials for every method across all experiments.

Failure Prompt Length	Environment Generalization	Task Generalization
1	64.58	60.75
2	**67.76**	**69.33**
4	66.65	63.25

Table 1: Ablation study on Failure Prompt Length. We report the average success rate of Adapt2Reward in environment generalization and task generalization experiments using different failure prompt lengths.

Task	Concept2Robot	Adapt2Reward
pushing cup from left to right	88	**99**
pushing cup onto book	18	**70**
moving box up	98	**100**
moving box down	84	**100**

Table 2: Change Objects. We report the success rates trained Concept2Robot and Adapt2Reward. Adapt2Reward outperformed Concept2Robot, achieving a success rate higher by 2-52%

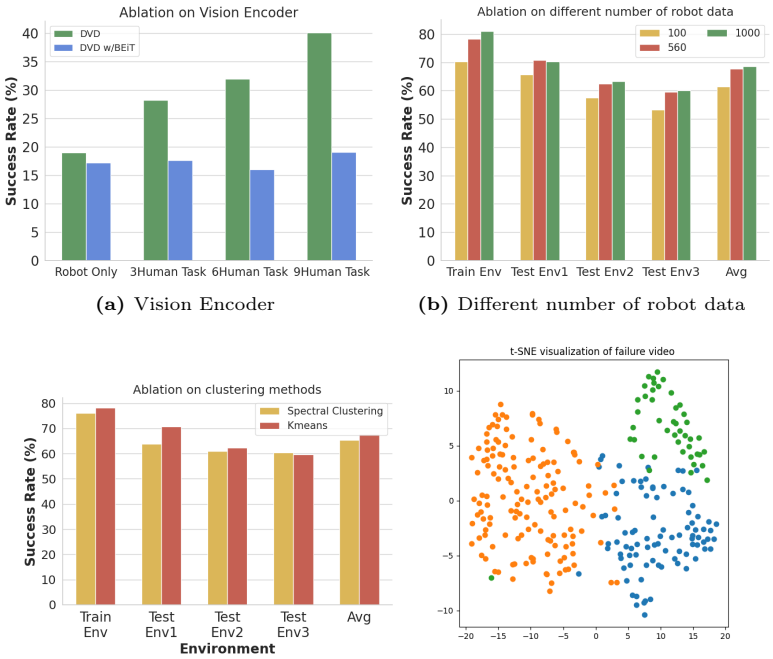
C Additional Results

Prompt Length, Novel Objects and Vision Backbone. In Table 1, our ablation study on the failure prompt length. We revealed that failure prompts of length 2 were most effective, with longer prompts potentially diverting attention from task semantics to prompt content.

In Table 2, we evaluate the robot's adaptability to novel objects by introducing items it has not previously encountered, thereby challenging it to identify these objects and successfully accomplish the task. In this context, Adapt2Reward demonstrates superior performance compared to Concept2Robot, achieving a success rate that surpasses its counterpart by a significant margin of 2-52%.

We undertook an ablation study targeting the visual encoder of the DVD model to ascertain whether the observed performance enhancement was attributable to the utilization of a distinct visual encoder in Adapt2Reward. In this process, we substituted the visual encoder in DVD with BEiT, the same encoder employed in Adapt2Reward, which had undergone pre-training on annotated labels and videos within the SSv2 dataset. The results, depicted in Figure 3(a), revealed a notable decline in DVD's performance following the encoder replacement. This outcome strongly suggests that the enhanced performance of Adapt2Reward cannot be attributed to the use of a different visual encoder.

The number of robot data. Futhermore, we explored how varying amounts of robotic data affect our model's environmental generalization ability. Figure 3(b) shows the success rates across environments when Adapt2Reward is trained with 100, 560, and 1000 robotic data instances. Training with 560 in-



(a) Vision Encoder **(b)** Different number of robot data

(c) Clustering method **(d)** t-SNE visualization

Fig. 3: (a) We present the average success rate of DVD in task generalization experiments across different vision encoders. (b) We show the average success rate of Adapt2Reward in environment generalization experiments with different amounts of training robot data, where 560 is the number adopted in the main text. (c) We display the average success rate of Adapt2Reward in four environments with different clustering methods. (d) We use t-SNE to visualize the failed videos from the "moving the handle of a faucet" task, with different colors representing different failure prompts.

stances significantly enhances performance compared to just 100. However, increasing the dataset to 1000 instances from 560 yields minor improvements, hinting at a possible saturation point or limited data diversity.

Clustering methods and t-SNE visualization. In addition, we explore how different clustering methods impact our experiment. Apart from K-means discussed in the main text, Figure 3(c) presents the results of Adapt2Reward trained using spectral clustering across four environments. To further analyze the learned failure prompts, Figure 3(d) shows the t-SNE visualization of the unsuccessful videos from the "moving the handle of a faucet task" in the dataset. Different colors represent various failure prompts.

Case Study of Failure Clusters. In our study, we used spherical K-means clustering to iteratively update cluster centers for failure videos related to each task at the end of every training epoch. This method assigns new pseudo-labels to the failure videos, highlighting unique failure patterns for each task. As shown in

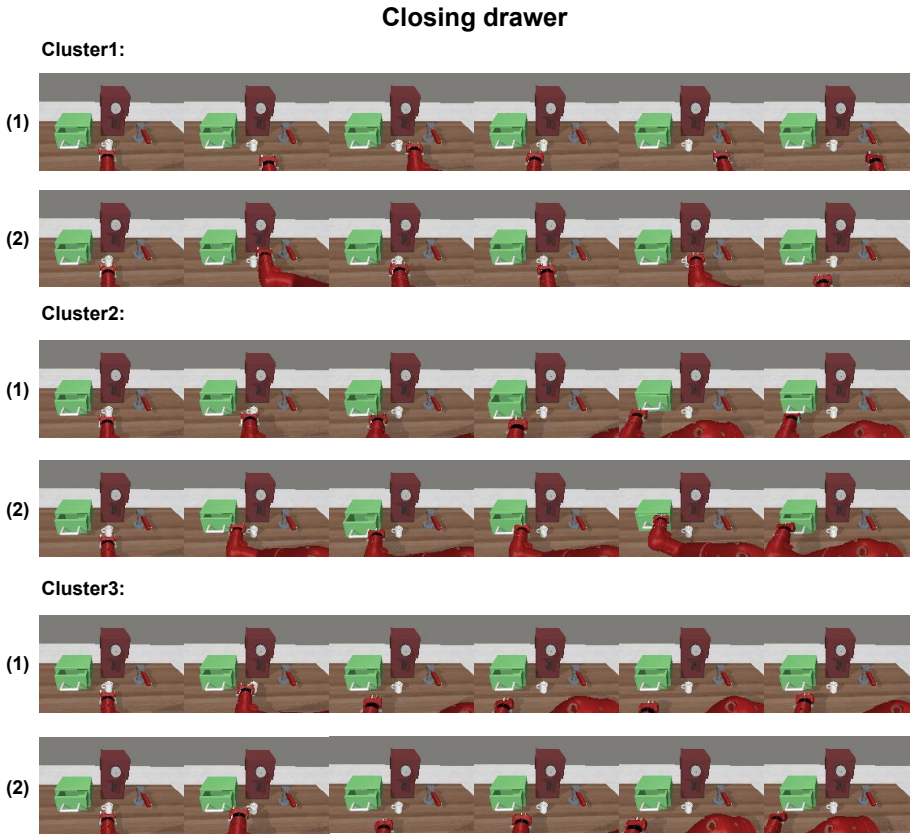


Fig. 4: Examples of Clustering for the Task of Closing Drawer. We display failure videos from the same cluster for the task of closing drawer. With K set to 3, we provide two examples per cluster.

Figure 4, we set $K = 3$ and display two failure videos per cluster for the "closing drawer" task. *Cluster 1* typically includes robots moving randomly across the desktop without interacting with the drawer. *Cluster 2* features robots that close the drawer and then reopen it. *Cluster 3* shows robots that align with the drawer but fail to complete the closing action. These clusters shed light on the various reasons for failure in the drawer-closing task. Figure 5 shows two failure videos per cluster for the "moving faucet handles" task. In *Cluster 1*, robots lock onto the faucet but fail to recognize the handle, leading to movements below the handle and task failure. *Cluster 2* shows robots moving randomly, unable to locate the faucet. In *Cluster 3*, robots correctly identify both the faucet and its handle but are unable to move the handle. Clustering failure videos reveals the different causes of failure for each task, improving the reward model's ability to distinguish between success and failure executions.

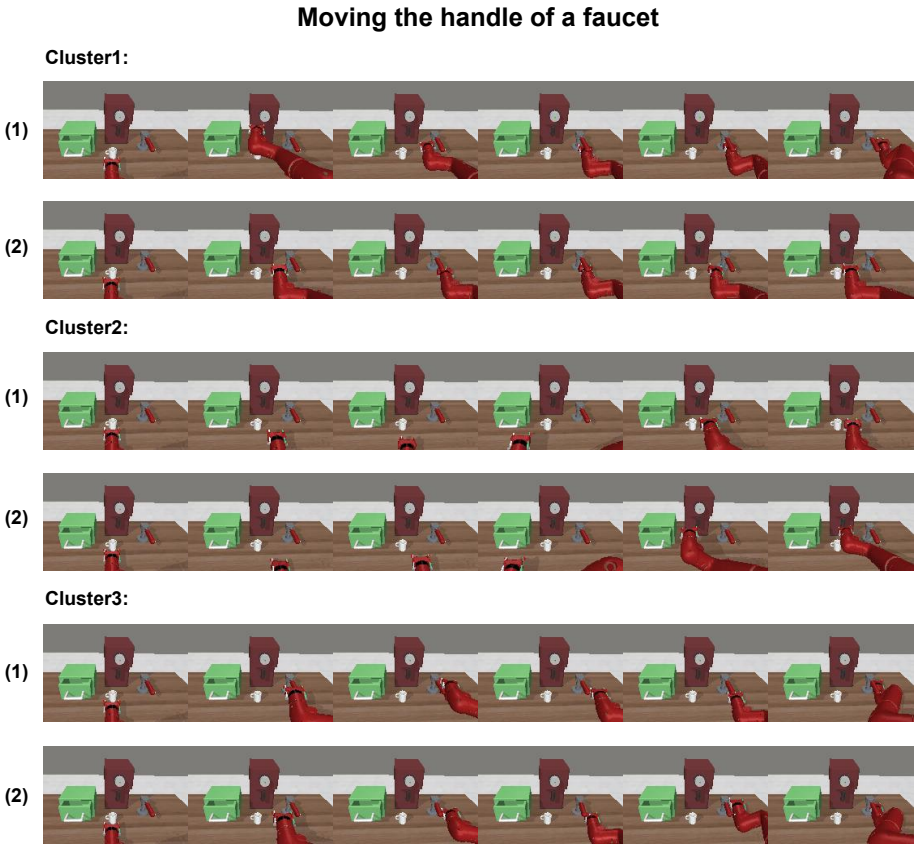


Fig. 5: Examples of Clustering for the Task of Moving the handle of a faucet. We display failure videos from the same cluster for the task of moving the handle of a faucet. With K set to 3, we provide two examples per cluster.

Success Rate per Task. In our simulation environment generalization experiments, we evaluated four tasks: 1) closing drawer, 2) moving cup away from the camera, and 3) moving the handle of a faucet, and 4) pushing cup from left to right. Figure 6 displays the individual task success rate for Adapt2Reward, trained with seven human tasks, and compares these to previous studies.In our task generalization experiment, we explored the effect of incorporating human data on the reward function's ability to adapt to new tasks. Following the methodology in DVD [2], we trained the Adapt2Reward model on robot videos from three distinct tasks within the training environment, deliberately excluding any robot data from the target tasks. The three training tasks are (1) opening drawer, (2) pushing cup from right to left, (3) poking cup so lightly that it doesn't or almost doesn't move. Figure 7 shows the results for each of the target tasks using Adapt2Reward trained with various number of human video tasks,

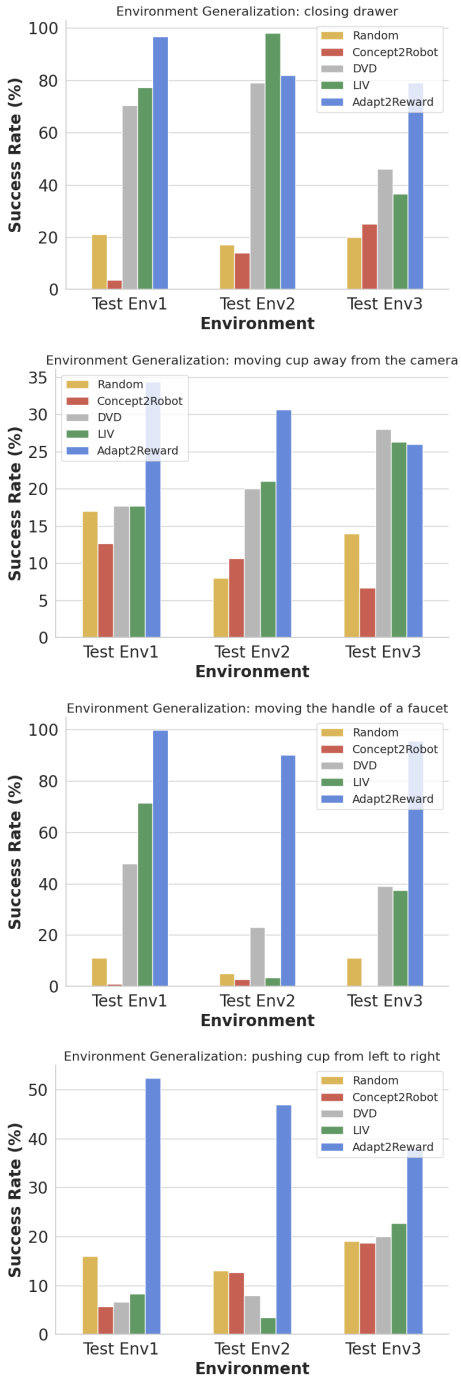


Fig. 6: Environment Generalization. We compare Adapt2Reward's performance to DVD, Concept2Robot, and a random policy. Each bar shows the success rate for the task, computed over 3 seeds of 100 trials.

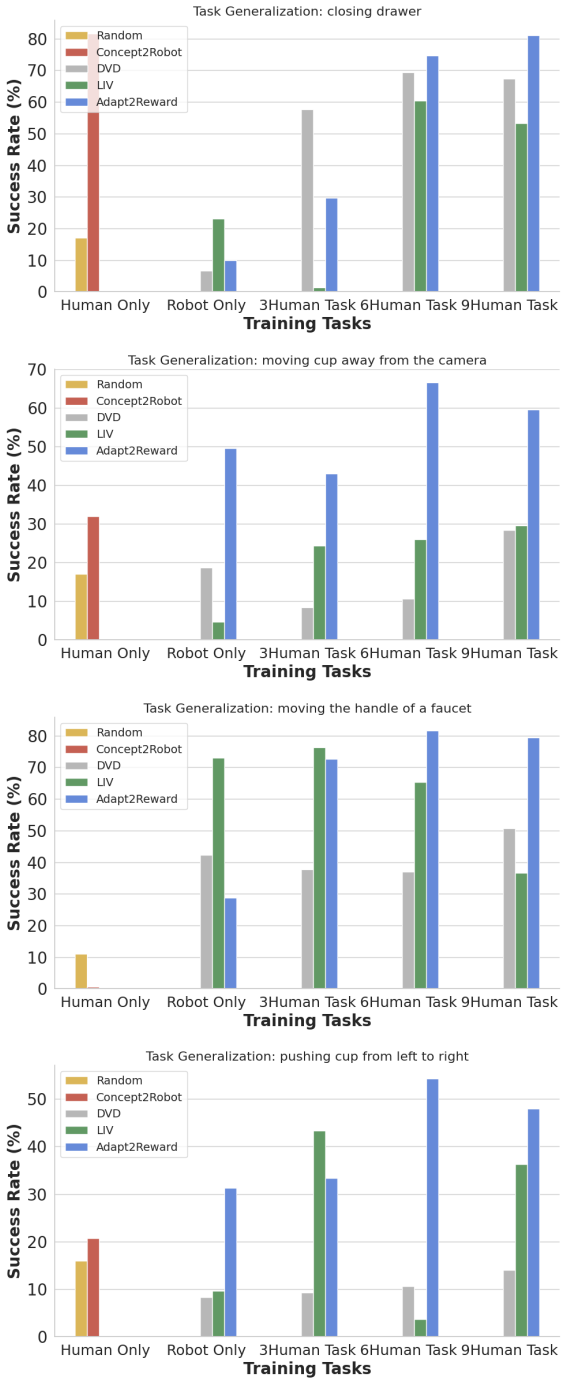


Fig. 7: Task Generalization. We compare Adapt2Reward's performance to DVD, Concept2Robot, and a random policy. Each bar shows the success rate for the task, computed over 3 seeds of 100 trials.

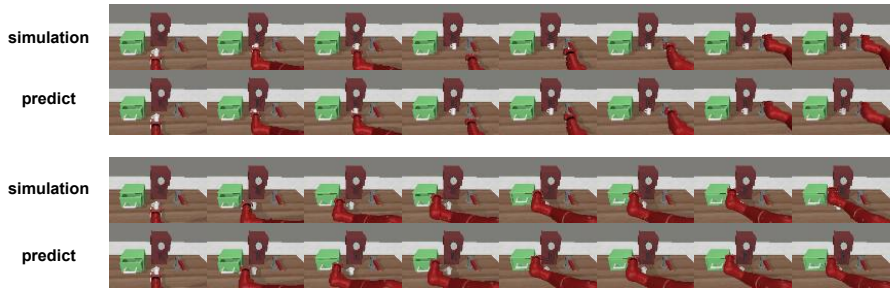
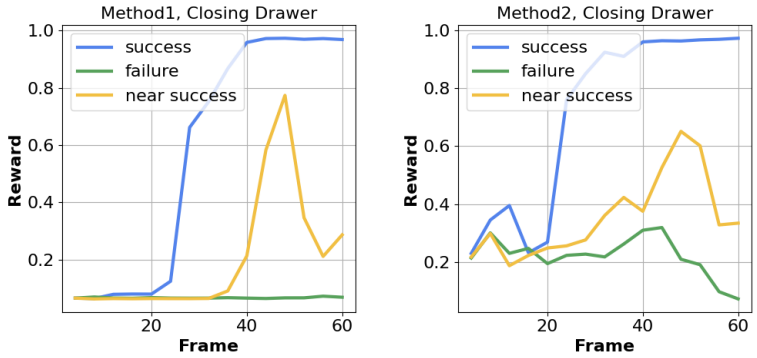


Fig. 8: FitVid Performance. We present the videos predicted by FitVid alongside actual videos from the simulation environment.



(a) Method 1 **(b)** Method 2

Fig. 9: Labeling dense rewards. We employ two distinct evaluation methods to evaluate the reliability of Adapt2Reward in assigning dense rewards and the capability of predicting future success.

comparing them to previous works. The conclusion of these experiments aligns with Section 4 in the main paper.

Performance of FitVid. In Figure 8, we showcase a comparison between the videos predicted by FitVid and the corresponding actual footage from the virtual environment. Our analysis indicates that the variance between these predicted and authentic simulation videos remains within tolerable bounds. This finding effectively diminishes the concern that disparities between the predicted and real videos could substantially impact the precision of reward estimations based on these videos.

Labeling dense rewards. To evaluate the reliability of Adapt2Reward in assigning dense rewards, we analyze the "closing drawer" task with successful trajectories, failed trajectories, and trajectories that were nearly successful (closed the drawer but then opened it again). In Figure 9, we show the results of two distinct evaluation methods. The first method directly calculates the current reward

using Adapt2Reward based on the frame sequence up to the current frame. The second first predicts 100 random complete trajectories from the current state using FitVid and then averages the Adapt2Reward rewards of these trajectories as the reward of the current state.

D Limitations and Future Works

While our methodology has shown proficiency in generalizing to new tasks and environments through learning from a curated set of successful and failed task-specific videos, it is not without its limitations. First, it demands the preliminary collection of a subset of both successful and unsuccessful robotic videos. Second, we evaluated our method in structured simulation environments but did not test its usage in complex real-world applications where more unexpected failure scenarios occur. Third, while our approach is well-suited for straightforward, desktop-centric tasks, its efficacy is somewhat reduced for more intricate tasks that necessitate multi-stage execution processes. In contrast, some studies leveraging Large Language Models (LLMs) like GPT-4 to craft reward functions primarily concentrate on code-based reward structures, using language as a medium for human-machine interaction.

A promising future avenue is to exploit multimodal inputs, combining a concise set of task videos with natural language descriptions to facilitate more nuanced user interactions. This approach holds potential for more elaborate tasks, such as “graceful walking." Though our core strategy is straightforward and effective, there is scope for refinement. Developing methodologies to craft more sophisticated reward functions could significantly elevate success rates and bolster the capacity to tackle more intricate comparative challenges.

References

1. Babaeizadeh, M., Saffar, M.T., Nair, S., Levine, S., Finn, C., Erhan, D.: Fitvid: Overfitting in pixel-level video prediction. ArXiv **abs/2106.13195** (2021), `https://api.semanticscholar.org/CorpusID:235623762`
2. Chen, A.S., Nair, S., Finn, C.: Learning generalizable robotic reward functions from "in-the-wild" human videos. ArXiv **abs/2103.16817** (2021)
3. Goyal, R., Kahou, S.E., Michalski, V., Materzynska, J., Westphal, S., Kim, H., Haenel, V., Fründ, I., Yianilos, P.N., Mueller-Freitag, M., Hoppe, F., Thurau, C., Bax, I., Memisevic, R.: The “something something” video database for learning and evaluating visual common sense. 2017 IEEE International Conference on Computer Vision (ICCV) pp. 5843–5851 (2017)
4. Haarnoja, T., Zhou, A., Abbeel, P., Levine, S.: Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. ArXiv **abs/1801.01290** (2018)
5. Kalashnikov, D., Irpan, A., Pastor, P., Ibarz, J., Herzog, A., Jang, E., Quillen, D., Holly, E., Kalakrishnan, M., Vanhoucke, V., Levine, S.: Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. ArXiv **abs/1806.10293** (2018)

6. Lei, J., Berg, T.L., Bansal, M.: Revealing single frame bias for video-and-language learning. ArXiv **abs/2206.03428** (2022)
7. Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N.M.O., Erez, T., Tassa, Y., Silver, D., Wierstra, D.: Continuous control with deep reinforcement learning. CoRR **abs/1509.02971** (2015)
8. Rubinstein, R.Y., Kroese, D.P.: The cross-entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation, and machine learning, vol. 133. Springer (2004)
9. Shao, L., Migimatsu, T., Zhang, Q., Yang, K., Bohg, J.: Concept2robot: Learning manipulation concepts from instructions and human demonstrations. The International Journal of Robotics Research **40**, 1419 – 1434 (2020)
10. Yu, T., Quillen, D., He, Z., Julian, R.C., Hausman, K., Finn, C., Levine, S.: Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. ArXiv **abs/1910.10897** (2019)