Learning Reward Functions for Robotic Manipulation by Observing Humans

Minttu Alakuijala, Gabriel Dulac-Arnold, Julien Mairal, Jean Ponce, Cordelia Schmid

Learning Reward Functions for Robotic Manipulation by Observing Humans

Minttu Alakuijala[1,2], Gabriel Dulac-Arnold[3], Julien Mairal[2], Jean Ponce[1] and Cordelia Schmid[3]

***Abstract*— Observing a human demonstrator manipulate objects provides a rich, scalable and inexpensive source of data for learning robotic policies. However, transferring skills from human videos to a robotic manipulator poses several challenges, not least a difference in action and observation spaces. In this work, we use unlabeled videos of humans solving a wide range of manipulation tasks to learn a task-agnostic reward function for robotic manipulation policies. Thanks to the diversity of this training data, the learned reward function sufficiently generalizes to image observations from a previously unseen robot embodiment and environment to provide a meaningful prior for directed exploration in reinforcement learning. The learned rewards are based on distances to a goal in an embedding space learned using a time-contrastive objective. By conditioning the function on a goal image, we are able to reuse one model across a variety of tasks. Unlike prior work on leveraging human videos to teach robots, our method, Human Offline Learned Distances (HOLD) requires neither a priori data from the robot environment, nor a set of task-specific human demonstrations, nor a predefined notion of correspondence across morphologies, yet it is able to accelerate training of several manipulation tasks on a simulated robot arm compared to using only a sparse reward obtained from task completion.**

I. Introduction

Deep learning has greatly advanced the state of the art in applications ranging from computer vision [1], [2] to natural language processing [3], [4] to speech recognition [5], but its significance in robotics has been blunted by limited access to large-scale data. Although previous efforts have attempted to sufficiently cover a specific embodiment and task [6], [7], collecting a massive dataset for each robot and environment of interest is simply not feasible due to the cost of maintenance and human oversight, hardware wear and tear as well as the bottleneck imposed by real-time execution. For these reasons, creative reuse of data is of central importance for unlocking the benefits of powerful function approximation and data-driven learning in robotics.

One potential source of external data is videos of humans performing arbitrary tasks, widely available on the internet and inexpensive to produce. We focus on manipulation tasks in this work, with the aim of learning from crowd-sourced videos of human arms and hands. However, replicating the demonstrated actions and object interactions with a robot is a challenging open problem. On the perception side, there is a significant visual domain gap between observations of a person and of a robot. Human and robot arms usually have very different morphologies and dynamics, particularly in the end-effector, creating a physical domain gap and making a 1:1 mapping between poses ill-defined in general. Moreover, the actions taken by humans are not observed unless explicitly recorded with specialized equipment, and hence conventional imitation learning [8], [9] or offline reinforcement learning [10], [11] methods are not applicable.

To overcome these challenges, we investigate the use of videos of people solving manipulation tasks to learn a notion of distance between images from the observation space of a task. We leverage this learned distance as a reward signal on tasks with similar structure but very different visual appearance on a set of robotic manipulation domains that the model has never observed. By training on diverse human demonstrations, we employ a strategy analogous to domain randomization [12] used in sim-to-real transfer, which applies variations to visual and physical simulation parameters at training time so that a real-world robotic task with unknown physical properties is more likely to fall in the training distribution. Similarly, when trained with different demonstrators, backgrounds, viewpoints, lightings, objects and tasks, our distance model learns to generalize to a variety of manipulator appearances. Furthermore, several aspects of the task as solved by a human are preserved in the robot workspace. For example, object displacements must respect the laws of physics regardless of the actor.

The learned distance function captures roughly how long it takes for an expert to transition from one state to another, and is therefore closely related to a dense reward function representing task progress that can be optimized with reinforcement learning (RL). Learning dense rewards is especially useful in hard exploration tasks where it is straightforward to define a sparse task-completion reward, but laborious and error-prone to specify a well-shaped dense reward.

Instead of model-free RL, reward functions estimating task progress can also be optimized with model predictive control [13], [14], in which case both a predictive forward model of the environment dynamics and a state-action value function need to be learned, typically from undirected exploration data in the target environment. However, extensive a priori data collection with sufficient coverage on a target robot environment and its action space are required for these methods to be applicable, and learning accurate video prediction models remains a challenging open problem in

[1]Inria, École Normale Supérieure, ENS, CNRS, PSL Research University
{minttu.alakuijala,julien.mairal,jean.ponce}@inria.fr
[2]Inria, Univ. Grenoble Alpes, CNRS, Grenoble INP, LJK
[3]Google Research
{dulacarnold,cordelias}@google.com

itself. We instead propose to learn a state-value function from observation-only data which allows for the reuse of data from different embodiments, and train a policy for the target embodiment with online RL. We empirically show better sample efficiency per task in online training than was required to learn a model in prior work [13].

Our contributions are as follows: i) We train HOLD, a global goal-conditioned distance model, which removes the need for demonstration task labels and exact alignment between robot tasks and demonstrated tasks required by prior work [13], [15]–[17]. ii) We show that time-contrastive embeddings [18] can successfully represent distances for multiple tasks at once despite a high degree of multi-modality in mixed-task training data. iii) We show generalization of reward functions trained from unconstrained human videos to robot arms of various morphologies and environments. iv) We demonstrate up to 18x accelerated training of model-free RL on 5 simulated manipulation tasks by either providing shaped rewards in sparse-reward tasks, or even entirely replacing the reward signal in some tasks. v) We show our method to significantly outperform existing cross-domain imitation [18] and representation learning [19] approaches.

II. Related work

a) Intermediate representations: Several prior works have addressed learning robotic policies from human videos via intermediate representations such as pose estimation or keypoint tracking [20]–[22]. In this work, our aim is to advance the capabilities of learning from raw video data, without depending on hand-crafted intermediate representations of human hands or an object database.

b) Imitation learning: Our work is related to imitation learning from observation, although this line of work has mostly addressed the case of demonstrations from the same observation space [9], [23]–[25]. We instead tackle the more difficult problem of inverse RL from observation under significant observational and dynamical domain shift.

c) Offline RL: Similarly to HOLD, Offline RL [11], [26]–[28] also aims to learn a value function from a dataset of existing trajectories. However, our setting is significantly different from the offline RL problem as we do not have access to either the actions or the rewards of the demonstrator in our dataset, nor do we have a forward model of which states are reachable from a given state, making temporal difference based methods not applicable.

d) Mapping methods: Many methods for learning from videos seek to learn a direct mapping between demonstration videos and robot states and/or actions, such as an inverse model labeling each human transition with an action from the robot action space [15], or an image-to-image translation of a human demonstration to a corresponding robot demonstration [29], [30]. By contrast, our method does not assume a precise 1:1 mapping between the observation and action spaces of the human and the robot and can therefore leverage arbitrarily large amounts of human demonstration videos without any manual supervision cost.

e) Consistency methods: A line of prior work has proposed to learn domain-invariant features capturing task progress regardless of whether the actor is a human or a robot arm [15], [16], [18] with reward defined as distance to a human demonstration [18] or to a goal state [16] in the feature space. One issue with using geometrical distances is that transition times between states are not symmetrical if the environment includes unidirectional transitions, such as dropping an object or knocking something down. To account for this, we also propose an alternative which predicts distances as a function of two ordered states. Sequence-based objectives such as temporal cycle consistency [16] are well suited for single-task learning where all trajectories can be aligned along a global task progression, but it is unclear whether these approaches would work on data from several tasks. Our task-invariant distance function is instead able to fully take advantage of the diversity of hands, objects, backgrounds, lighting variations etc. across tasks, even when a grouping of demonstrations into distinct tasks is not available. Most existing approaches to learning robotic manipulation from human videos also require exact overlap between tasks demonstrated by humans and the robot tasks [13], [15]–[17], and some require robot demonstrations for some of the same tasks [13]. As our model is not specialized for any single task and learns from human data only, no robot demonstrations are needed and the target robot task does not need to be strictly included in the training data as long as a goal image is available to specify the new task.

f) Time-contrastive embeddings: Sermanet *et al.* [18] propose to use distances in an embedding space learned with a time-contrastive objective, but only consider reward learning for a single task, whereas we learn a single multi-task reward model. Moreover, while [18] propose to directly imitate a human demonstration at 1:1 speed, we instead define the task with a goal image from the robot's observation space. As we show experimentally, [18] needs a nearly identical alignment in the initial states, execution speed and cropping between the video and the robot observations, which is a significant limitation. By contrast, our inverse RL approach requires less supervision and allows the robot to potentially outperform the demonstrator, either by executing the task faster or by finding a more optimal trajectory.

g) Functional distance: Our work is also related to estimating functional (also called dynamical) distance between states from online [31] or offline robot data [14]. Most related to our method, Tian *et al.* [14] use an offline dataset and use estimated time-to-goal as a value function in a model predictive control loop. However, both works use only robot data from the same environment, without transfer of the action or observation spaces. Our approach is instead based on estimating the state-value function of the demonstrated behavior drawn from an unknown action space.

III. Human Offline Learned Distances

A. Functional distances from observation-only data

We propose to learn about distances in state space by observing humans and using this prior knowledge of en-

vironment dynamics to accelerate training of RL policies on a robot manipulator. Specifically, our goal is to estimate *functional distance* $d(s, g)$, as defined by Tian *et al.* [14], between an image s of the current state and a goal image g, where $s, g \in \mathcal{S}_r$, the set of camera observations from the robot's observation space. This metric should correlate with $\delta(s, g)$, the number of time steps it takes for an expert policy π^* to reach the goal g from the state s:

$$\delta(s, g) = \mathbb{E}[T | s_T = g, s_0 = s, a_t \sim \pi^*(s_t, g), \\ s_{t+1} \sim p(s_t, a_t)], \tag{1}$$

where p are the transition dynamics of the environment, modeled as a Markov Decision Process (MDP). The negated time difference $-\delta(s, g)$ is equal to the value function V^* for an optimal policy π^* for the reward function

$$r(s, g) = \begin{cases} 0 & s = g \\ -1 & \text{otherwise.} \end{cases} \tag{2}$$

However, this is not the only reward function that can be optimized to recover π^*. In order to serve as a useful reward function for the task defined by g, the value of d does not need to perfectly correlate with δ; instead, pairwise rankings should be preserved for all states:

$$\delta(s, g) > \delta(s', g) \implies d(s, g) > d(s', g) \forall s, s', g \in \mathcal{S}_r. \tag{3}$$

Although defined in terms of an expert policy π^*, $\delta(s, g)$, and consequently the functions $d(s, g)$ that preserve its rankings, can be estimated from observation-only data, without access to actions a, the expert π^*, or even its action space, by obtaining self-supervised time deltas without manual annotation. While Tian *et al.* [14] learn the Q-function corresponding to Eq. (2) from offline trajectories from the robot, our choice of a state-value function, agnostic to a specific action space, allows reuse of data gathered with different but related morphologies, such as other robots or humans. Strictly speaking, the ability to share the function δ between human and robot MDPs relies on them being isomorphic [15], requiring a 1:1 mapping between the action and observation spaces that preserves dynamics p. While this may not fully hold in practice, and the distribution of δ in human data may not necessarily match the robot's dynamics in absolute terms due to embodiment differences, the rankings produced by d can be transferred under fewer assumptions. For example, one embodiment may be twice as fast as the other while still preserving all pairwise rankings.

We assume access to a dataset of N video demonstrations of humans executing a variety of manipulation tasks using approximately shortest paths. In practice, the precise length of time may vary significantly across trials and human demonstrators, and depend on the optimality of the demonstration. Although the absolute length of such time intervals may not be consistent across demonstrators, their relative durations provide a useful learning signal; in order to push an object to the right, one must first approach its current position from the left before starting the pushing maneuver, and not the other way around. We present two methods for learning d on this data.

a) Direct regression (HOLD-R): We assume the demonstrations are optimal and pose the functional distance learning problem as a supervised regression task:

$$\theta^* = \text{argmin} \sum_{i=1}^{N} \sum_{t=1}^{T_i} \sum_{\delta=1}^{T_i - t} ||d_\theta(s_t^i, s_{t+\delta}^i) - \delta||_2^2 \tag{4}$$

where s_t^i is the tth frame of the ith video, T_i is the length of the ith video, and d_θ is a function parameterized by θ trained to predict δ from Eq. (1). The third summation corresponds to data augmentation allowing any future time step in the video to be considered the goal rather than only the last.

b) Time-contrastive embeddings (HOLD-C): Since directly predicting time intervals is difficult and sensitive to noise, we may also consider learning an embedding space where distances can be defined. We propose to use a single-view time-contrastive objective as in TCN [18]. Frames within a small temporal window are encouraged to lie close together in embedding space, whereas embeddings for frames outside some temporal neighborhood are pushed apart. Specifically, if s_p is a positive instance for anchor s, and s_n is a negative instance, for all triplets, we want:

$$||f(s) - f(s_p)||_2^2 + m < ||f(s) - f(s_n)||_2^2 \tag{5}$$

where the margin m is a hyperparameter. However, unlike the single-task setup proposed in [18], we train f on multi-task data and show it to accelerate robot learning across tasks. Moreover, our method improves upon TCN in several ways at the policy training stage: i) HOLD enables the robot to outperform the demonstrations by learning relevant shortcuts through interaction, or by simply moving faster, whereas TCN aims to imitate the human. TCN defines the task using a human video, and minimizes distance to each of its states at 1:1 speed – although the distances are minimized with RL, the best possible reward is defined as matching the human performance. ii) HOLD requires less supervision: TCN needs one human trajectory of the full task whereas we use distance to a goal image only and require no task demonstrations. iii) We use a simpler Euclidean distance to define the metric $d(s, g)$ in the space f, whereas [18] apply a weighted mixture of squared Euclidean and a Huber-style loss $d(s_t, g_t) = \alpha||f(s_t) - f(g_t)||_2^2 + \beta\sqrt{\gamma + ||f(s_t) - f(g_t)||_2^2}$, requiring two additional hyperparameters to be tuned in an already computationally expensive RL training setup.

B. Policy learning

We propose to use the learned functional distance to define a dense reward function for an RL policy. Although our reward function is goal-conditioned and shared across tasks, we train one policy per robot task. As we want to minimize distance to the goal frame, we define reward as follows:

$$r(s_t, a_t, s_{t+1}, g) = -\max(0, d(s_{t+1}, g) - d(g, g))/T \tag{6}$$

where $s_t, s_{t+1}, g \in \mathcal{S}_r$, a_t is an action from the robot's action space, and T is an optional normalizer. We subtract the baseline $d(g, g)$ from the distance estimates to ensure arriving at the goal has reward 0 and no other state has higher

(a) Pushing mouse from left to right

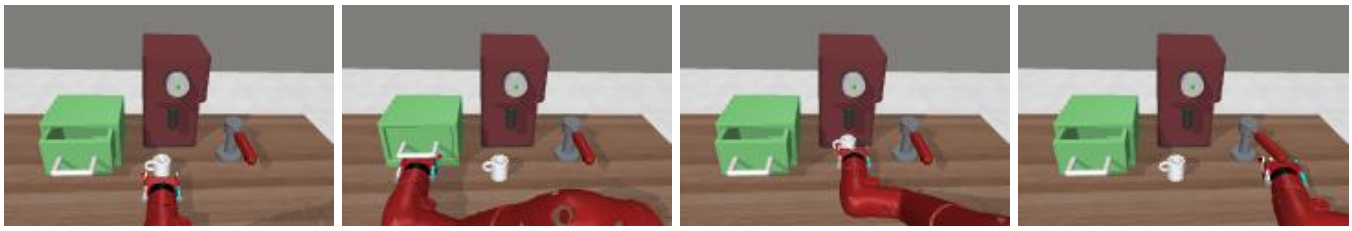
(b) Putting paint brush underneath magazine

(c) Moving book up

Fig. 1: Example human videos from Something-Something v2 used to train the distance models.

(a) Pushing: start (b) Pushing: goal (c) Drawer: start (d) Drawer: subtask (e) Drawer: goal

Fig. 2: The RLV tasks.



(a) Start state for all tasks (b) Close Drawer (c) Push Cup Forward (d) Turn Faucet Right

Fig. 3: The DVD tasks: a Sawyer arm in a tabletop environment adapted from Meta-World [32].

reward; $d(g, g)$ may be positive for the regression models due to untrimmed training videos where the demonstrator idles after solving the task. This definition of reward corresponds to minimizing the sum of distances until the end of the episode, as done by [14], [31]. Alternatively, r could be defined based on the difference $d(s_{t+1}, g) - d(s_t, g)$, such that only the reduction is maximized for each time step. However, we found the cumulative form to perform better empirically, possibly due to being less sensitive to noise.

IV. Experimental Results

A. Distance learning

a) Dataset: We train HOLD on Something-Something v2 (SSv2) [33], a crowd-sourced dataset of 220,847 video clips of 174 action classes (with examples in Fig. 1). Each action is demonstrated with arbitrary objects, matching templates such as *Moving [something] closer to [something]*. The clips last 4 seconds on average and are mostly filmed using handheld devices, with non-negligible camera motion. Although SSv2 videos are grouped into discrete action classes, we do not make use of these labels[1], making our method applicable on any large-scale goal-oriented manipulation data. As we train a single goal image-conditioned distance function, there also does not need to be exact overlap between the demonstrated tasks and the target tasks on the robot, unlike in prior works [13], [15]–[17].

b) Training details: We consider two sizes of network architecture: a ResNet-50 [1] and a Video Vision Transformer (ViViT) [35] pretrained on SSv2 classification. As the single-view time-contrastive objective only supports embedding single images, for HOLD-C we instead use either a ResNet or a Vision Transformer (ViT) [2] pretrained on ImageNet-21K. We train the ResNet models from scratch, and fine-tune the pretrained models on SSv2 without labels after replacing their classification heads. To adapt the pretrained ViViT model for regression, we also reinitialize its temporal position embeddings and shorten the temporal window to 4, including the 3 most recent frames and one goal frame. We also reduce the temporal filter dimension to 1 as there is no longer a computational benefit to shortening the sequence length. For time-contrastive training, we sample batches of 32 subsequent frames per video and use a positive window of 0.2 seconds and a negative window of 0.4s, as done by [18]. For both objectives, we apply the same data augmentation procedure as [35], but leave out MixUp. For further training details, see Appendix A. We observed better policy training performance for the ResNet model

[1]Only HOLD-R with ViViT architecture made use of labels in pretraining, whereas HOLD-R with ResNet backbones as well as all HOLD-C models did not. However, the pretraining could have potentially been done on a different labeled dataset such as Kinetics [34] or skipped altogether, and no labels are needed for the regression task. We are working on evaluating a fully label-free ViViT regression model.

for HOLD-C, and for ViViT for HOLD-R, so we report results using these backbones in Section IV-B. Ablations using the other architectures is included in Appendix C, and strategies for evaluating the distance models on human data before testing them in robot policy training are discussed in Appendix B.

B. Policy Learning

To demonstrate the utility of our method as a reward function for training RL policies, we evaluate it on the Pushing and Drawer Opening tasks defined by RLV [15] (Fig. 2) and on the Close Drawer, Push Cup Forward and Turn Faucet Right tasks defined by DVD [13] (Fig. 3). We follow prior work [15], [16] in using Soft Actor-Critic (SAC) [36] as the underlying RL algorithm and evaluate it on 20 episodes for all tasks. All policies use images as input, and we reuse the policy and critic architectures as well as algorithm hyperparameters from Schmeckpeper *et al.* [15]. Like [15], we augment our learned reward from Eq. (6) with a sparse task reward: 1 for success, and 0 otherwise, defined by each environment based on distance to the target configuration. Since the predicted distances can be significantly larger than 1 but should not override the sparse reward, we scale the predicted rewards by $1/T$, where T is set such that the scale of initial state distances is $\approx 1/3$. Ablations for other values are included in Appendix C.

1) RLV tasks: As shown in Fig. 4, the sum of both reward functions, appropriately balanced, significantly accelerates training compared to using the sparse reward alone. In our experiments, using only sparse reward required 10x more samples for Pushing and >18x more for Drawer to reach the return of HOLD-C. We find that HOLD-C outperforms HOLD-R overall for Pushing, both with and without sparse reward, and in the sparse reward setting for Drawer Opening.

a) Pushing: Without added sparse reward, a single failure case is prominent: while the policy quickly learns to match the end-effector position in the goal frame, it fails to pay attention to the puck position. As observed by Tian *et al.* [14], it is easy for the distance function to excessively focus on fully actuated components in the scene as these are highly predictive of temporal offset. Although HOLD is able to generalize from human arms to a robot arm, for tasks with variable object positions, it may be better suited as an exploration strategy used together with an otherwise rarely-observed sparse reward than a standalone multi-task reward. Note that although Zakka *et al.* [16] also evaluate on Pushing, their results are not comparable as their method is trained on the easier RLV Pushing dataset [15] collected to match the appearance of the robot task, and report on the simpler State Pusher task where the policy directly observes the 2D puck position and the 3D end-effector position.

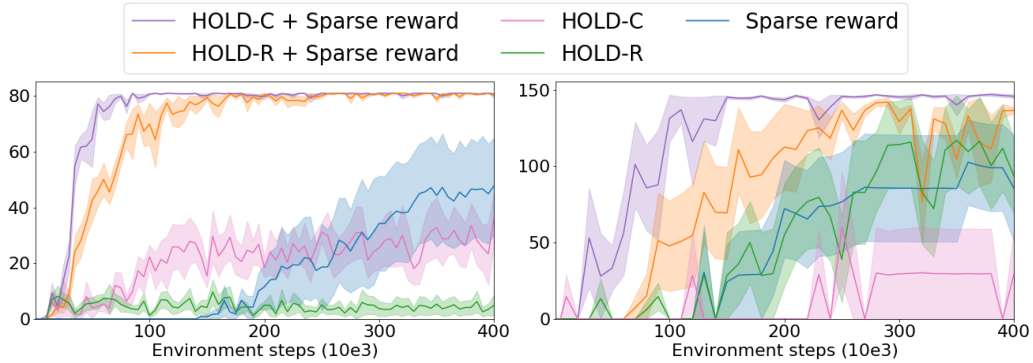
b) Drawer: The Drawer Opening task has double the episode length (200 steps) of Pushing, and consists of two distinct motions: approaching and inserting the gripper into the handle, and pulling the drawer open once there. We find that applying the HOLD models on the full task suffers from the local minimum of only matching the arm position in the goal image. However, if we instead define the task in two parts using an intermediate goal image (in Fig. 2d), our rewards significantly improve sample efficiency compared to the sparse task-completion reward provided by the environment alone, as shown in Fig. 4b. Moreover, HOLD-R alone without any environment reward performs on par with sparse reward in this setting. We train a single policy for both subtasks, which is conditioned on the active goal image by concatenating it to the observation s. For all distance functions, we switch to the next subtask when $d < 1$ for at least 3 consecutive time steps.

2) DVD tasks: We report success rate for the DVD tasks in Fig. 5. These tasks are significantly easier than the RLV tasks and quickly learned even using only sparse reward. To estimate the upper bound in learning speed achievable by improving the reward alone, we define an oracle reward using knowledge of robot and object positions. Since we observe only a narrow performance gap between the oracle and the sparse reward, the learning speed in these tasks is limited mostly by the RL algorithm. Although it is therefore difficult to show much improvement over the sparse reward, both HOLD models outperform it, particularly for Close Drawer and Turn Faucet Right. For Close Drawer, HOLD also solves the task without sparse reward. Unlike [13], we do not first collect a dataset of 10,000 trajectories, or 600,000 steps, of random exploration on the robot to learn a model of the environment, but instead focus on the model-free setting. We show adaptation to a new robot, set of objects and environment in just 12,000–18,000 steps, or 200 to 300 episodes, when sparse environment reward is available, or 22,000 without sparse reward for Close Drawer.

C. Baseline comparisons

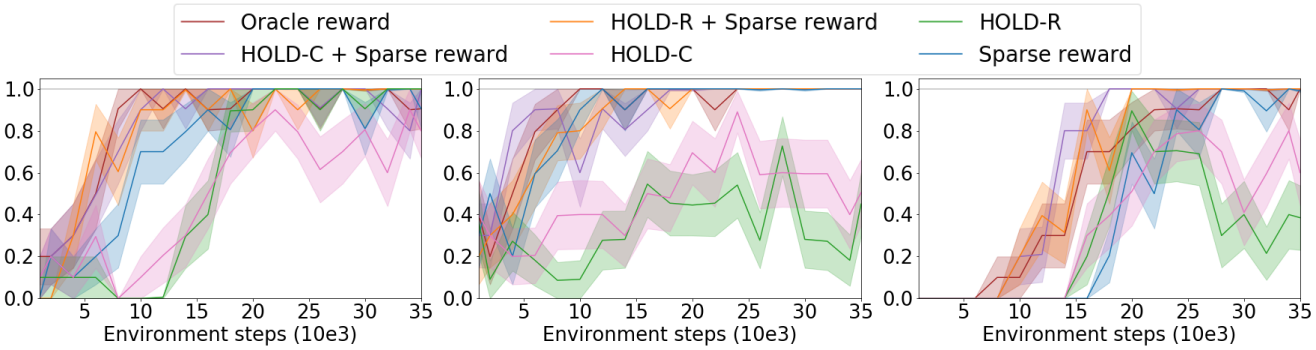
We compare HOLD to rewards defined by two prior methods: TCN [18] and R3M [19]. TCN proposes to transfer a policy given a human video demonstration by minimizing distance to the embeddings of each of the visited states g_t in turn. We empirically set the hyperparameters of $d(s_t, g_t)$ to $\alpha = 0.005, \beta = 0.02$, and $\gamma = 0.2$. As performance may vary based on the exact demonstration video used, we evaluate 3 demonstrations per task from the RLV dataset, which is collected to closely match the RLV robot tasks, and report the average performance across demonstrations (trained with 5 seeds each) in Figure 6. Even the closely aligned demonstrations transfer poorly to policy learning, especially for Pushing, due to slight differences in initial state, cropping or execution speed, highlighting the brittleness of the trajectory-following objective of TCN.

Although R3M is proposed as a general feature representation, we also compare against using Euclidean distance in the representation space for defining dense rewards. We used the ResNet-50 model checkpoint from [19], trained on the much larger Ego4D [37] (3,500 hours) rather than SSv2 (200 hours). As shown in Figure 6, HOLD-C outperforms R3M in both RLV tasks despite having been trained on less data and requiring no language descriptions. Like our



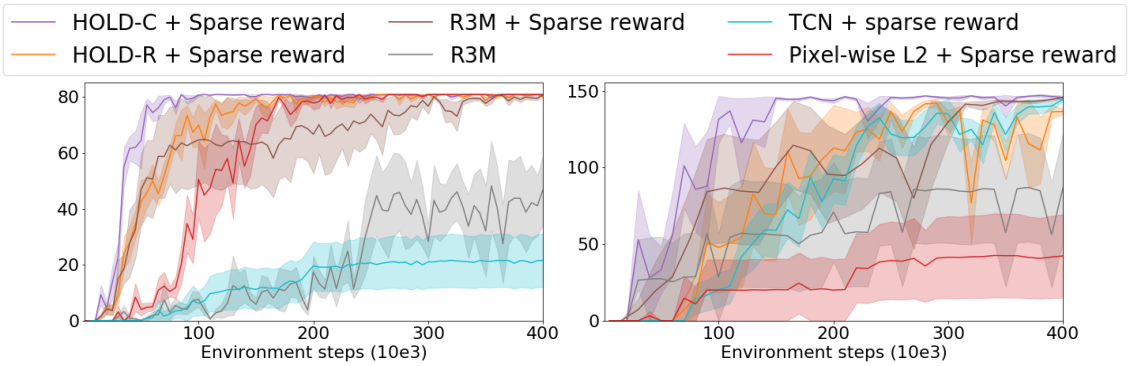
(a) Pushing (b) Drawer Opening (2 subtasks)

Fig. 4: Return on the RLV tasks (5 random seeds, with standard error).



(a) Close Drawer (b) Push Cup Forward (c) Turn Faucet Right

Fig. 5: Success rates on the DVD tasks (10 random seeds, with standard error). Our reward functions improve over sparse reward, and learn the Close Drawer task without using sparse reward.



(a) Pushing (b) Drawer Opening (2 subtasks)

Fig. 6: HOLD-C outperforms TCN and distances in R3M representation space on both RLV tasks.

method, R3M also requires sparse rewards to fully solve the tasks, and an intermediate goal for Drawer opening.

We also include a simple baseline of using the negative pixel-wise distance in image space between the current observation and the goal image as a reward. While using distances in image space with sparse reward also learns the Pushing task faster than sparse reward alone, as shown in Figure 6, this still requires many more training samples than either HOLD-R or HOLD-C, and fails to reliably learn the Drawer task.

V. Conclusion

We have presented a method for learning goal image conditioned reward functions for robotic manipulation from unlabeled human videos, in a challenging setting which no prior work has addressed to our knowledge. Learning a prior for robot behavior from a dataset of human demonstrations without task labels requires generalization both across tasks and across a significant domain shift. While most accurate for short-horizon tasks with single-step movements, the distance functions we train produce useful rewards for visually dif-

ferent robot environments that are able to accelerate training over using sparse reward alone, and can be composed to perform more general multi-step manipulation tasks using subgoals. Finally, we have shown that for some tasks, the predicted rewards alone are sufficient to learn the task without any additional success signals.

ACKNOWLEDGEMENTS

This work was in part suppported by the Inria / NYU collaboration, the Louis Vuitton / ENS chair on artificial intelligence and the French government under management of Agence Nationale de la Recherche as part of the *Investissements d'avenir* program (PRAIRIE 3IA Institute). It was performed using HPC resources from GENCI–IDRIS (Grant 2022-AD011013362). Minttu Alakuijala was supported in part by a Google CIFRE PhD Fellowship. We would like to thank Elliot Chane-Sane for reviewing this manuscript.

REFERENCES

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Computer Vision and Pattern Recognition*, 2016.

[2] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2021.

[3] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems*, 2020.

[4] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann *et al.*, "Palm: Scaling language modeling with pathways," *arXiv preprint arXiv:2204.02311*, 2022.

[5] Y. He, T. N. Sainath, R. Prabhavalkar, I. McGraw, R. Alvarez, D. Zhao, D. Rybach, A. Kannan, Y. Wu, R. Pang *et al.*, "Streaming end-to-end speech recognition for mobile devices," in *International Conference on Acoustics, Speech and Signal Processing*, 2019.

[6] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," in *The International Journal of Robotics Research*, vol. 37, no. 4-5, 2018, pp. 421–436.

[7] H.-S. Fang, C. Wang, M. Gou, and C. Lu, "Graspnet-1billion: A large-scale benchmark for general object grasping," in *Computer Vision and Pattern Recognition*, 2020.

[8] D. A. Pomerleau, "Efficient training of artificial neural networks for autonomous navigation," in *Neural computation*, vol. 3, no. 1, 1991, pp. 88–97.

[9] J. Ho and S. Ermon, "Generative adversarial imitation learning," in *Advances in Neural Information Processing Systems*, 2016.

[10] A. Kumar, J. Fu, M. Soh, G. Tucker, and S. Levine, "Stabilizing off-policy q-learning via bootstrapping error reduction," in *Advances in Neural Information Processing Systems*, 2019.

[11] S. Fujimoto, D. Meger, and D. Precup, "Off-policy deep reinforcement learning without exploration," in *International Conference on Machine Learning*, 2019.

[12] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *International Conference on Intelligent Robots and Systems*, 2017.

[13] A. S. Chen, S. Nair, and C. Finn, "Learning generalizable robotic reward functions from "in-the-wild" human videos," in *Robotics: Science and Systems*, 2021.

[14] S. Tian, S. Nair, F. Ebert, S. Dasari, B. Eysenbach, C. Finn, and S. Levine, "Model-based visual planning with self-supervised functional distances," in *International Conference on Learning Representations*, 2021.

[15] K. Schmeckpeper, O. Rybkin, K. Daniilidis, S. Levine, and C. Finn, "Reinforcement learning with videos: Combining offline observations with interaction," in *Conference on Robot Learning*, 2020.

[16] K. Zakka, A. Zeng, P. Florence, J. Tompson, J. Bohg, and D. Dwibedi, "Xirl: Cross-embodiment inverse reinforcement learning," in *Conference on Robot Learning*, 2022.

[17] L. Shao, T. Migimatsu, Q. Zhang, K. Yang, and J. Bohg, "Concept2robot: Learning manipulation concepts from instructions and human demonstrations," in *The International Journal of Robotics Research*, vol. 40, no. 12-14, 2021, pp. 1419–1434.

[18] P. Sermanet, C. Lynch, Y. Chebotar, J. Hsu, E. Jang, S. Schaal, S. Levine, and G. Brain, "Time-contrastive networks: Self-supervised learning from video," in *International Conference on Robotics and Automation*, 2018.

[19] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta, "R3m: A universal visual representation for robot manipulation," *arXiv preprint arXiv:2203.12601*, 2022.

[20] Y. Qin, Y.-H. Wu, S. Liu, H. Jiang, R. Yang, Y. Fu, and X. Wang, "Dexmv: Imitation learning for dexterous manipulation from human videos," *arXiv preprint arXiv:2108.05877*, 2021.

[21] V. Petrík, M. Tapaswi, I. Laptev, and J. Sivic, "Learning object manipulation skills via approximate state estimation from real videos," in *Conference on Robot Learning*, 2020.

[22] N. Das, S. Bechtle, T. Davchev, D. Jayaraman, A. Rai, and F. Meier, "Model-based inverse reinforcement learning from visual demonstrations," *arXiv preprint arXiv:2010.09034*, 2020.

[23] F. Torabi, G. Warnell, and P. Stone, "Behavioral cloning from observation," in *International Joint Conference on Artificial Intelligence*, 2018.

[24] Y. Aytar, T. Pfaff, D. Budden, T. Paine, Z. Wang, and N. De Freitas, "Playing hard exploration games by watching youtube," in *Advances in Neural Information Processing Systems*, 2018.

[25] I. Kostrikov, K. K. Agrawal, D. Dwibedi, S. Levine, and J. Tompson, "Discriminator-actor-critic: Addressing sample inefficiency and reward bias in adversarial imitation learning," *arXiv preprint arXiv:1809.02925*, 2019.

[26] Y. Wu, G. Tucker, and O. Nachum, "Behavior regularized offline reinforcement learning," *arXiv preprint arXiv:1911.11361*, 2019.

[27] Z. Wang, A. Novikov, K. Zolna, J. S. Merel, J. T. Springenberg, S. E. Reed, B. Shahriari, N. Siegel, C. Gulcehre, N. Heess *et al.*, "Critic regularized regression," in *Advances in Neural Information Processing Systems*, 2020.

[28] X. B. Peng, A. Kumar, G. Zhang, and S. Levine, "Advantage-weighted regression: Simple and scalable off-policy reinforcement learning," *arXiv preprint arXiv:1910.00177*, 2019.

[29] H. Xiong, Q. Li, Y.-C. Chen, H. Bharadhwaj, S. Sinha, and A. Garg, "Learning by watching: Physical imitation of manipulation skills from human videos," in *International Conference on Intelligent Robots and Systems*, 2021.

[30] J. Li, T. Lu, X. Cao, Y. Cai, and S. Wang, "Meta-imitation learning by watching video demonstrations," in *International Conference on Learning Representations*, 2021.

[31] K. Hartikainen, X. Geng, T. Haarnoja, and S. Levine, "Dynamical distance learning for semi-supervised and unsupervised skill discovery," in *International Conference on Learning Representations*, 2020.

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

[33] R. Goyal, S. Ebrahimi Kahou, V. Michalski, J. Materzynska, S. Westphal, H. Kim, V. Haenel, I. Fruend, P. Yianilos, M. Mueller-Freitag *et al.*, "The "something something" video database for learning and evaluating visual common sense," in *International Conference on Computer Vision*, 2017.

[34] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev *et al.*, "The kinetics human action video dataset," *arXiv preprint arXiv:1705.06950*, 2017.

[35] A. Arnab, M. Dehghani, G. Heigold, C. Sun, M. Lučić, and C. Schmid, "Vivit: A video vision transformer," in *International Conference on Computer Vision*, 2021.

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

[37] K. Grauman, A. Westbury, E. Byrne, Z. Chavis, A. Furnari, R. Girdhar, J. Hamburger, H. Jiang, M. Liu, X. Liu *et al.*, "Ego4d: Around the world in 3,000 hours of egocentric video," in *Computer Vision and Pattern Recognition*, 2022.

OVERVIEW OF SUPPLEMENTARY MATERIALS

This appendix is organised as follows: in Section A, we describe implementation details of the distance model and policy training. In Section B, we discuss how HOLD models can be evaluated on human data. Ablations over several design choices are included in Section C. We estimate the increase in sample efficiency achieved with HOLD in Section D, and look into the coverage of the proposed robot tasks in SSv2 in Section E.

A supplementary video visualizing examples of distances predicted by HOLD-C as well as policy roll-outs is included on the project website[2]. We also qualitatively evaluate the predictions on episodes from Bridge Data [40], a diverse dataset of robot manipulation tasks recorded on real robots, and include examples in the video. The results suggest our distance model may well generalize to training manipulation policies on real robots.

APPENDIX A
TRAINING DETAILS

Our distance models are implemented in JAX [38] using the Scenic library [39]. Hyperparameter settings are shown in Table I for the regression models and in Table II for time-contrastive training. For policy training, we reuse the implementation of SAC from [15] based on Softlearning [41]. All RL hyperparameter settings are unchanged (included in Table III for reference).

Parameter	ViViT	ResNet-50
Epochs	20	100
Base learning rate	0.1	3e-4
Optimizer	Momentum	Adam
Batch size	64	32

TABLE I: Training hyperparameters for HOLD-R.

Parameter	ViT	ResNet-50
Epochs	5	100
Sequence length	32	32
Base learning rate	1e-4	1e-4
Optimizer	Adam	Adam
Batch size	8	8

TABLE II: Training hyperparameters for HOLD-C.

Parameter	Value
Initial exploration steps	1000
Learning rate	3e-4
Batch size	256
Optimizer	Adam
Gradient steps per environment step	1

TABLE III: Training hyperparameters for policy training.

APPENDIX B
DISTANCE MODEL EVALUATION ON HUMAN DATA

To avoid evaluating every variation of HOLD in the target robot environment, it would be preferable to be able to rank and pre-select models based on their performance on held-out human data, and test only the most promising ones in robot policy training. However, it is difficult to evaluate generalization without access to robot data, and it is not straightforward to design a suitable test metric that captures both smoothness and correct ranking of states. In this section, we propose several relevant metrics.

For the regression models, in addition to the training objective mean squared error (MSE), we can also evaluate mean absolute error in time steps and in seconds. However, these metrics assume uniform progress at each time step toward task completion, and require high-scoring models to match the scale of ground truth time intervals. Using a hinge loss instead allows non-uniform progress and only penalizes out-of-order predictions:

$$\mathcal{L}_h = \frac{\sum_{i=1}^{N}\sum_{j,k=1}^{T_i-1}\max(0, d(s_k^i, s_{T_i}^i) - d(s_j^i, s_{T_i}^i))\mathbb{I}[j<k]}{\sum_{i=1}^{N}(T_i-1)}. \tag{7}$$

Another option is to not use time-based metrics at all. As explained in Section III-A, it is ultimately more important for the distance models to preserve the ranking of states with respect to a goal frame than to reproduce δ in absolute terms. With the aim of maximally preserving pairwise rankings as defined in Eq. (3), we propose two further metrics, namely misclassification rate:

$$\mathcal{L}_{miscl} = \frac{\sum_{i=1}^{N}\sum_{j,k=1}^{T_i-1}\mathbb{I}[d(s_k^i, s_{T_i}^i) > d(s_j^i, s_{T_i}^i)]\mathbb{I}[j<k]}{\sum_{i=1}^{N}(T_i-1)}, \tag{8}$$

and Spearman correlation, i.e., the correlation between rankings assigned to each frame in the full sequence $s_{1:T_i-1}$, and the ground truth order.

The scores of each of the models we present are shown in Table IV. As we assume no access to robot data at distance training time, we use the SSv2 validation set as a proxy for model performance, and use Spearman correlation as an early stopping criterion. However, we observe that the scores on human data are not predictive of the downstream robot performance these models obtain, highlighting the difficulty of the domain transfer.

APPENDIX C
DISTANCE MODEL ABLATIONS

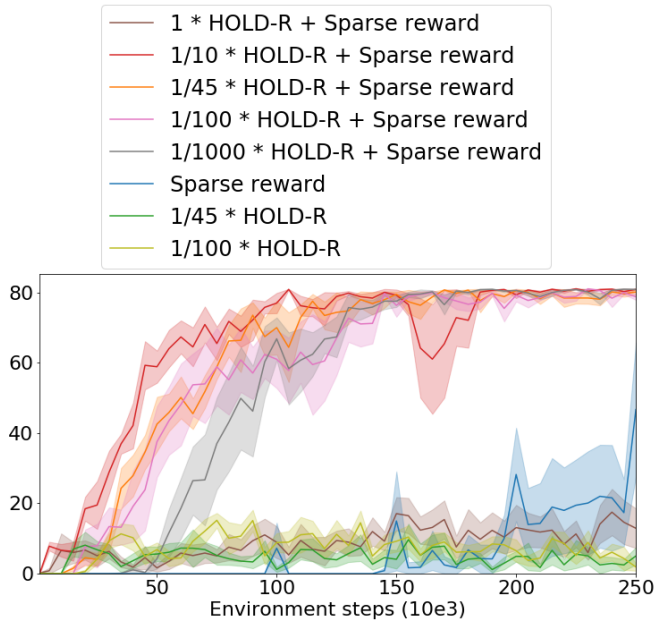
We evaluate a variety of design choices in HOLD models on the RLV Pushing task. Specifically, we compare several values for the reward normalizer T introduced in Eq. (6), as well as variants of the network architecture and the form of the reward (cumulative vs. instantaneous, as described in Section III-B).

The effect of reward scale for HOLD-R is shown in Fig. 7a. While $T = 10$ increases return the fastest, its performance is less stable towards the end of training and it suffers a momentary drop in performance when the policy appears to overfit to the distance reward over the sparse task reward. The normalizer $T = 45$, equal to the average length of a training video in SSv2, provides the best trade-off in sample efficiency and stability and we therefore report results

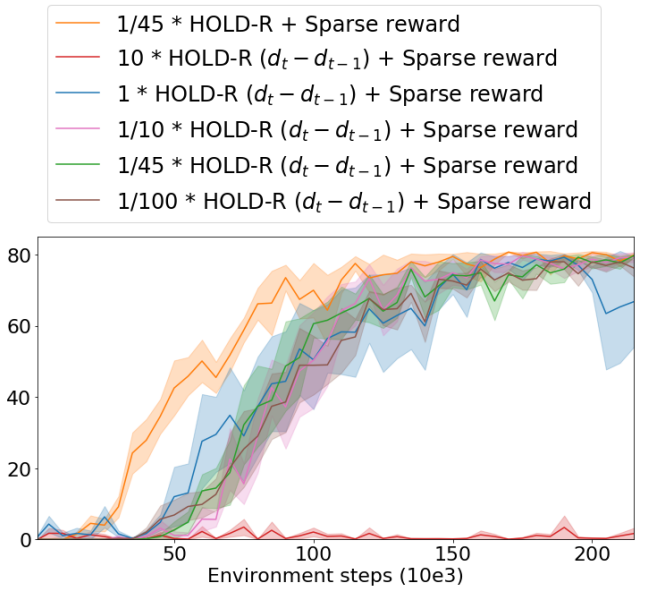
[2]sites.google.com/view/hold-rewards

Model	Network	# frames	Spearman	Misclassification rate	MSE	Mean error	Hinge loss
HOLD-R	ViViT	3	0.6709	0.4539	499.2	18.0 (1.54 s)	0.0663
HOLD-R	ResNet-50	1	0.7136	**0.3976**	**482.1**	**17.3 (1.48 s)**	**0.0233**
HOLD-R	ResNet-50	3	**0.7139**	0.4385	514.1	18.1 (1.55 s)	0.0611
HOLD-C	ResNet-50	1	0.6246	0.4015			
HOLD-C	ViT	1	0.6559	0.4006			

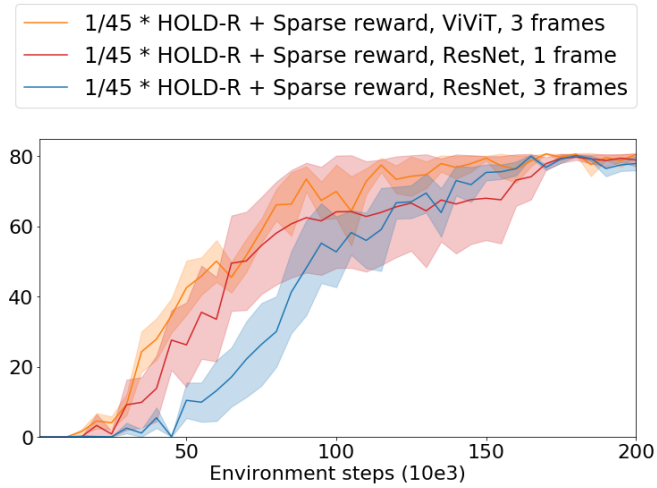
TABLE IV: Evaluation scores on the Something-Something v2 validation set. Time-based metrics are only defined for HOLD-R as HOLD-C models do not predict time.



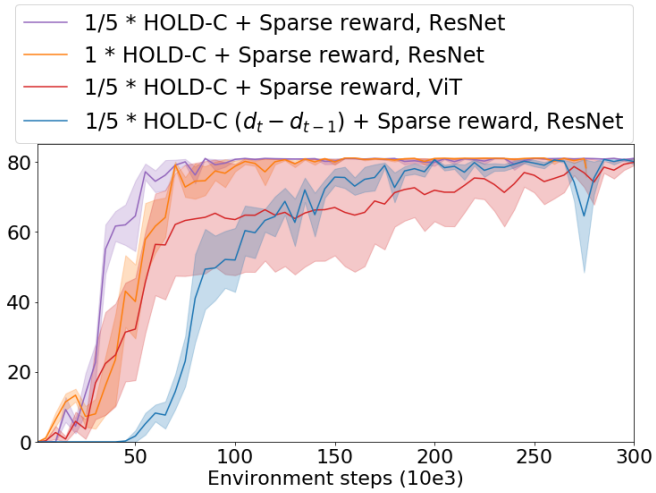
(a) Effect of reward scale for HOLD-R.



(b) Cumulative distance vs. subtracting the previous distance for HOLD-R.



(c) Effect of architecture choice for HOLD-R.



(d) Effect of reward definition, scaling and architecture choice for HOLD-C.

Fig. 7: HOLD model ablations on RLV Pushing.

using this setting in Section IV-B. This value is used for all tasks except RLV Open Drawer, where the task horizon is twice as long and we found $T = 100$ to work significantly better (see Fig. 8). To avoid further extensive tuning of the reward scale for HOLD-C and for the Pixel-wise L2 baseline, we simply set T such that the scale of initial predictions is

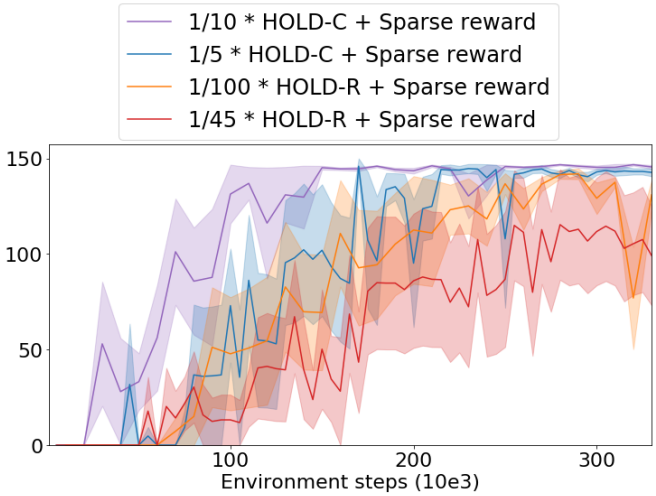


Fig. 8: Effect of reward scale on RLV Open Drawer.

approximately $1/3$, the same scale as HOLD-R with $T = 45$. Using this strategy, we obtain $T = 5$ for HOLD-C and $T = 30$ for the L2 baseline (or $T = 10$ and $T = 100$ for RLV Drawer, respectively), which we indeed found to perform better than respective alternatives $T = 1$ and $T = 45$.

As for reward definition, the cumulative distance reward clearly outperforms instantaneous distance reward (i.e. subtracting the distance at the previous time step) for both HOLD-R (Fig. 7b) and HOLD-C (Fig. 7d). Although the scale of $d_t - d_{t-1}$ is expected to be different from the scale of d_t and hence the normalizer T may need to be set differently, we found the HOLD-R instantaneous distance form to perform consistently worse for a wide range of values of T: $\{0.1, 1, 10, 45, 100\}$. Moreover, the choice of T seemed to have very little effect on the learning performance for $T \leq 1$.

Finally, in Fig. 7c, we compare the HOLD-R ViViT model against ResNet-50 conditioned on either 1 or 3 frames, but found that these smaller models had slightly worse sample efficiency in RL training than ViViT. For HOLD-C models, we found ResNet to outperform ViT, however, ViT may have benefited from longer training. In Section IV-B, we therefore report HOLD-C results using the ResNet architecture.

Appendix D
Longer training for sparse reward

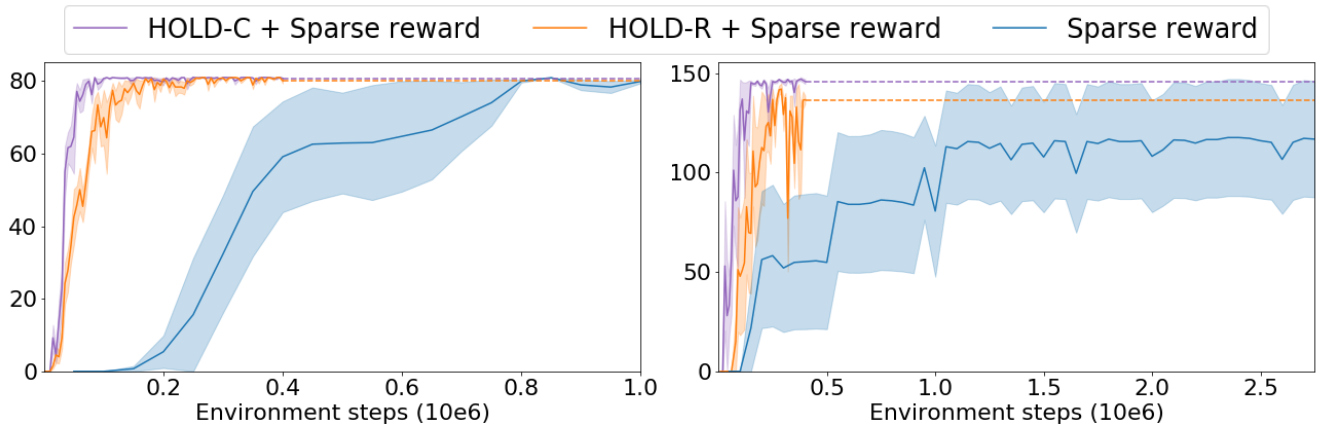
HOLD-C converges to a return of 80 for Pushing after 80,000 environment steps of training and a return of 145 for Drawer after 150,000 steps. In order to estimate how much training time is accelerated compared to using only the sparse reward, we also run experiments with considerably longer training for the sparse reward baseline. The return of HOLD is eventually reached after 800,000 samples for Pushing, whereas for Drawer, it is not reached even after 2.75 million steps. We therefore obtain a speedup of 10x for Pushing (Fig. 9a) and at least 18x for Drawer (Fig. 9b).

Appendix E
Training data coverage of evaluated robot tasks

Our distance models are not specialized for any specific task and can therefore be applied on previously unseen manipulation tasks, or tasks with very few human demonstrations. In this section, we investigate to what extent the robot tasks we evaluate on are covered in SSv2 training data. Note that our models never observe the task labels and are only trained on ungrouped SSv2 videos.

The tasks included in SSv2 are intentionally very diverse. As task templates include generic movements such as "Moving something up", "Moving something down", "Pushing something from left to right", "Pushing something from right to left", it is genuinely difficult to find manipulation tasks unrelated to any of the 174 templates. However, several tasks include drastically different manipulations depending on the objects considered: e.g., opening the screw cap of a bottle and opening a book use the same template "Opening something" but very different motions. As shown in Table V, the action-object pairs we evaluate in the robot tasks have never been demonstrated for Turn Faucet Right, and have been demonstrated <5 times for Push Cup Forward and RLV Pushing (moving a cap toward the camera). The objects "puck" or "disk" do not appear in SSv2, so we replace "puck" by "cap".

Moreover, on closer inspection of the three videos labeled "Moving cap towards the camera", we observe significant variation in the interpretation of the action labels themselves. Instead of pushing an object along a surface like in our robot task, one out of three videos in fact shows pulling a (baseball) cap along a surface, and the remaining two show a person holding a bottle cap and moving it directly towards the camera lens without using a surface at all. The same two motions are demonstrated for related objects such as "lid" (2 videos) and "bottle cap" (1 video). It seems unlikely that the same task, *pushing* a puck towards the camera is demonstrated at all (the pushing templates featuring horizontal directions



(a) Pushing

(b) Drawer Opening

Fig. 9: Eventual return for the sparse reward on the RLV tasks after 1–2.75 million environment steps (2.5–7x increase from Fig. 4).

Robot task	Closest action	Closest object	*a*	*o*	*(a, o)*
Pushing	Moving sth towards the camera	cap	927	611	3
Open Drawer	Opening sth	drawer	1585	619	67
Close Drawer	Closing sth	drawer	1296	619	62
Push Cup Forward	Moving sth away from the camera	mug	937	951	4
Turn Faucet Right	Pushing sth from left to right	faucet	3199	28	0

TABLE V: SSv2 training examples most closely matching the evaluated robot tasks. Column 2 corresponds to the most similar action template *a* to each robot task, whereas column 3 lists the most similar object *o* among the objects manipulated across all tasks templates. In columns 3–5, we give the number of videos in the training and validation sets labeled with either *a*, *o*, or both, respectively. The full dataset consists of 220,847 videos: 168,913 in the train set, 24,777 in the validation set and the remaining 27,157 in the test set.

only). Similarly, only 1/4 videos labeled "Moving mug away from the camera" and 1/4 videos labeled "Moving cup away from the camera" push an object along a surface at all, and the rest perform the maneuver in the air.

We conclude that we have shown generalization to at least one and possibly multiple novel tasks which were not included in the training dataset.

References

[38] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018.

[39] Mostafa Dehghani, Alexey Gritsenko, Anurag Arnab, Matthias Minderer, and Yi Tay. Scenic: A JAX library for computer vision research and beyond. *arXiv preprint arXiv:2110.11403*, 2021.

[40] Frederik Ebert, Yanlai Yang, Karl Schmeckpeper, Bernadette Bucher, Georgios Georgakis, Kostas Daniilidis, Chelsea Finn, and Sergey Levine. Bridge data: Boosting generalization of robotic skills with cross-domain datasets. *arXiv preprint arXiv:2109.13396*, 2021.

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