

Vid2Robot: End-to-end Video-conditioned Policy Learning with Cross-Attention Transformers

Vidhi Jain^{1,2}, Maria Attarian^{1,3}, Nikhil J Joshi¹, Ayzaan Wahid¹, Danny Driess¹, Quan Vuong¹, Pannag R Sanketi¹, Pierre Sermanet¹, Stefan Welker¹, Christine Chan¹, Igor Gilitschenski³, Yonatan Bisk², Debidatta Dwibedi¹

¹Google DeepMind, ²Carnegie Mellon University, ³University of Toronto

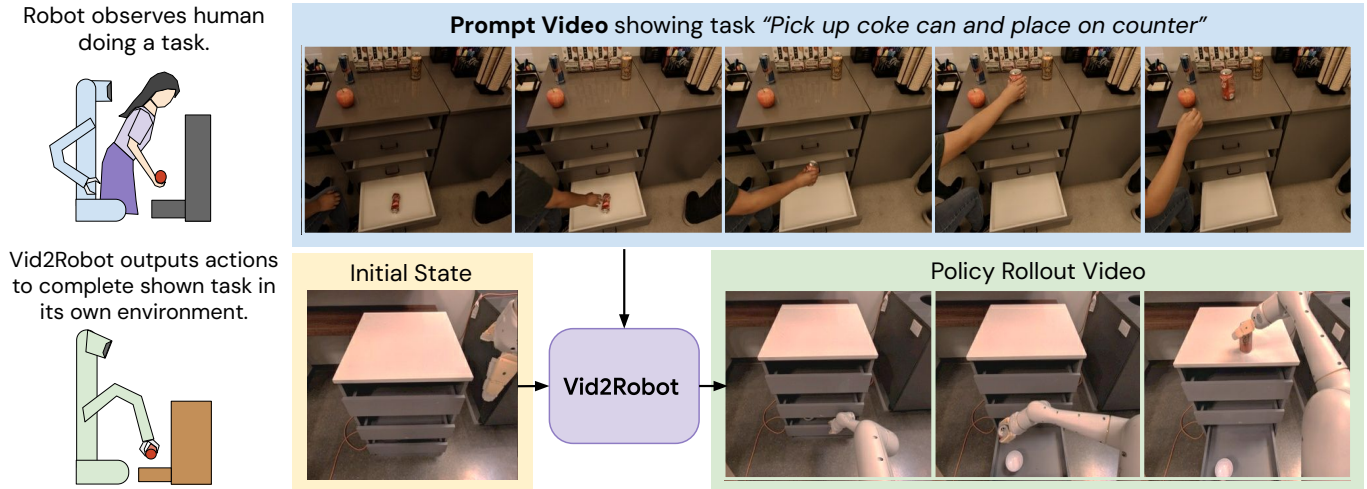


Fig. 1: Overview. Vid2Robot is a video-conditioned robot policy. Given a human demonstration (top), Vid2Robot recognizes the task semantics and performs the same task based on the robot’s current visual observation (bottom left). A successful trajectory is presented on the bottom right.

Abstract—While large-scale robotic systems typically rely on textual instructions for tasks, this work explores a different approach: can robots infer the task directly from observing humans? This shift necessitates the robot’s ability to decode human intent and translate it into executable actions within its own physical constraints and environment.

We introduce Vid2Robot, a novel end-to-end video-based learning framework for robots. Given a video demonstration of a manipulation task and current visual observations, Vid2Robot directly produces robot actions. This is achieved through a unified representation model trained on a large dataset of human video and robot trajectory. The model leverages cross-attention mechanisms to fuse prompt video features to the robot’s current state and generate appropriate actions that mimic the observed task. To further improve policy performance, we propose auxiliary contrastive losses that enhance the alignment between human and robot video representations.

We evaluate Vid2Robot on real-world robots, demonstrating a 23% improvement in performance compared to other video-conditioned policies when using human demonstration videos. Additionally, our model exhibits emergent capabilities, such as successfully transferring observed motions from one object to another, and long-horizon composition, thus showcasing its potential for real-world applications.

I. INTRODUCTION

The path to creating versatile robots that provide assistance in people’s daily routines requires them to learn new skills on-the-go. These range from small preferences like which brand of dishwasher a specific household uses to entirely different

ways to clean the house. For known skills, humans simply communicate in natural language, but when nuance is required or a skill is novel, we revert to demonstrations. For example, we might show how a particular microwave works or how we prefer our cabinets to be organized. To enable seamless robot deployment, robots need the same ability for generalization from demonstration for learning new policies that comes so naturally to humans.

Humans can infer the intentions of other humans based on third-person visual observations. Oftentimes, we use social reasoning and common sense to understand others’ goals implicitly. This ability is leveraged both as children and adults (e.g. via How-To videos [31]) for learning anything where the mechanical nuance of the task is hard to capture in still images or text [6] (e.g. how to knead dough or knit). If robots can also be taught to understand the intentions of other agents, it might allow them to better interact with humans and perform tasks more efficiently.

This work focuses on visual imitation learning, where robots learn to perform tasks by watching video demonstrations. This setup offers several advantages. First, it allows robots to learn from agents with different embodiment, enabling new skill acquisition without tele-operation. Second, it enables robots to learn from experts, even if they are not situated with the robot. Finally, visual imitation learning is ideal for teaching tasks that are difficult or impossible to describe in words.

Existing multi-task robot manipulation models (e.g. RT-1 [8], RT-2 [9], and RT-X [34]) use language conditioning to output a robot trajectory. This reliance of text alone for task specification makes it difficult for robots to handle polysemy and tasks whose executions vary dramatically based on context. For example, ‘open drawer’, ‘open cabinet’, ‘open container with lid’ and ‘open jar with screw cap’ might share the same verb but require very different motor control for each interaction. Here the agent should not generalize its policy, whereas it should generalize from one *drawer* to others that vary in type, color and shape. For this reason, there are a broad range of tasks for which it is hard to design primitives for high-level planning approaches [26, 2].

Another common approach has been to use a final goal image in goal-conditioned behavior cloning tasks [32, 25]. While several task specifications can be defined in terms of the resulting state of the environment, there are others for which the manner in which the action is performed is part of the specification. For example, ‘hold the flag’ and ‘wave the flag’ can have the same final goal image. This ambiguity can be resolved through the use of several sub-goal frames, i.e. video conditioning.

While language conditioned policies achieve somewhat high success rates, video conditioned policies have lagged behind in performance, as shown in prior work [22]. Cases of good performance [41] with video conditioning require the provided video to be from the same workspace with limited variability. Based on observations, we identify three main challenges for video conditioned policies: (1) *High dimensional data*: Raw videos are high dimensional data that require more compute and memory to process. This makes video conditioned multi-task policies difficult to train at scale. (2) *Variability in task specification*: There can be significant variance in how people perform the same task. Demonstrations for a task like ‘unstuck the cups’ can have both visually distinctive and physically diverse cups, in addition to changes in the background distractors and lighting conditions. This leads to high variability in task specification for a policy that needs to perform the same task in a new setting. (3) *Limited Availability of Training Data*: While there is an abundance of unlabeled video data on the internet, obtaining labeled video datasets for specific tasks that our robots are capable of doing is challenging.

Despite these challenges, as noted, video conditioned policy learning is a core challenge robots need to master. Therefore, to reduce the reliance on detailed and potentially ambiguous language prompts, we aim to enable physical visual demonstrations as a another way for task specification. To this end, we study how end-to-end models with video-conditioning can be used to specify tasks to robot.

We aim to develop an end-to-end system that enables rapid adaptation to tasks specified in the form of video demonstration. Unlike prior work that either learned representations from videos for only object and verb recognition [22] or learned motor control in simulation [45], our work demonstrates the applicability of end-to-end learned video representations for real-world multi-task robotic control. We present the key con-

tributions of our work as follows: (1) We present a transformer-based policy to encode video task specification, demonstrated by either robot or human agent embodiments (§II). (2) We encourage alignment between the prompt and robot video representations using three contrastive auxiliary losses during training (§II-E) (3) Through real robot experiments, we find our video conditioned policy is better than baselines on human prompt videos. Furthermore, our policy is better at cross-object motion transfer (§III).

II. APPROACH

A. Preliminaries

Our objective is to design a robotic system that takes in a *prompt video* of a manipulation task and outputs actions that accomplish the task demonstrated in the video. This system needs to infer the underlying task from the prompt video (which might have a different setup or embodiment than the robot) and then manipulate the objects in its own environment to achieve the inferred task. Specifically, we are given a prompt video V and the robot state $S_t = \{x_i\}_{i=t-k-1}^t$ where x_i is the frame from the robot’s camera stream at time i , k is the maximum number of historical frames, and t is the current time-step. We train a policy $\pi(a_t|S_t, V)$ that infers the underlying task from V and predicts task relevant action a_t . To train this model, we need a dataset of paired prompt videos and robot trajectories. We will discuss in detail how to create paired datasets below.

B. Datasets

To train a video-conditioned robot policy we need a dataset of pairs: prompt videos and robot trajectories performing the same task. In this work, we explore reference videos where the task is performed by both humans and robots. To create this dataset, we rely on three classes of data:

- 1) **Robot-Robot**: We pair existing robot-robot videos of the same task. For this pairing we consider two videos to match if they are performing the same task in different settings. We define ‘*task*’ based on natural language instructions used when recording robot trajectories. These instructions typically consist of one or two verbs surrounded by nouns, such as ‘*place* water bottle upright’, ‘*move* the coke can to the green chip bag’ or ‘*open* top drawer’. The objective of this pairing is two-fold: first, to be able to take advantage of an already labeled and collected dataset of robot trajectories and second to ensure robots are able to imitate when the same task is demonstrated in a different environment.
- 2) **Hindsight Human-Robot**: Here we use the task instructions from the robot trajectories dataset and ask one to five human participants to perform the task and record a demonstration video from the robot’s perspective/view. The set of instructions are the same as before, but there is a significant embodiment and speed variability due to different humans performing the task with left or right hands and at a randomized robot camera angle. This requires some manual effort but provides us with a lot of paired data for training the policy for the available set of


































Dataset Name	Prompt Video Embodiment	Prompt v/s Robot Scene	Prompt Video					Robot Video					
Robot-Robot	Robot	Different											
Hindsight Human-Robot	Human	Different											
Co-located Human-Robot	Human	Same											

Fig. 2: Dataset creation. (top row) Here we show a Robot-Robot video pair for *placing the rxbar into top drawer*. We similarly pair existing robot-robot videos performing the same task. (middle row) Here we show Hindsight Human-Robot paired videos for *picking a coke can from the bottom drawer and placing it on the counter* task. We use the task instructions from robot trajectories and ask human participants to perform the task and record a demonstration video from robot’s perspective/view. (bottom row) Here we show a Co-located Human-Robot pair of videos for *placing the pipe wrench in the toolkit*. We record both a human demonstration and by a robot teleoperation in a same workspace. Different workspaces can be used to perform the same task instruction, thus, eventually resulting in pairs with visually diverse prompts and robot state observations. More details in §II-B.

instructions in the robot dataset without having to collect new robot trajectories.

3) **Co-located Human-Robot** In this case, a human and a robot perform the same task in the same workspace. We used this approach to collect human demonstrations and robot trajectories in diverse spaces such as living space with sofas, meeting room with whiteboards, hardware workstations with toy tools, kitchen with countertop, refrigerator and sink, storage supplies area, and more.

We show examples of paired prompt and robot videos from each of the three datasets in Figure 2. As can be seen, there is a considerable difference in the backgrounds and distractor objects in the Hindsight Human-Robot and Co-located Human-Robot datasets. A different complexity arises when comparing the first approach (Robot-Robot) where the actor is a robot with same morphology to the other two cases where the human is the actor in the prompt videos.

After combining all the datasets, we have $\sim 100k$ robot videos and $\sim 10k$ human videos covering the tasks introduced in RT-1 [8] and RT-2 [9]. We include videos from all three data sources as they represent varying levels of difficulty and expense to collect. Pairing existing robot datasets requires less additional effort but lacks diversity in how the task is done. The second source of data is created by asking humans to mimic existing robot trajectories. While this adds some diversity in prompt videos, it does not cover any new tasks on the robot side. Finally, the presumed gold-standard is to collect data where both humans and robots are co-located in the same environment and perform diverse tasks. This takes the most amount of time as labor is required both of the humans and robot trajectories collected through tele-operation.

C. Model Architecture

Our policy takes as input the prompt video and the current robot state and outputs robot actions. It consists of four modules: (1) prompt video encoder (2) robot state encoder, (3) state-prompt encoder, and (4) robot action decoder. The full

architecture is illustrated in Figure 3 and each of the modules are detailed below:

(1) **Prompt Video Encoder** encodes the video demonstration provided as a reference to convey the desired task semantics. The prompt video encoder implicitly learns to infer what task should be performed and how it needs to be done. The prompt encoder consists of a per-frame Image encoder ϕ_p (ViT [15]) followed by a Perceiver Resampler [1, 20] ψ_p . The output of the prompt encoder $\psi_p(\phi_p(V)) = z_{prompt}$ is a set of N tokens of d -dimension to condition the policy with the task relevant attributes from the video.

(2) **Robot State Encoder** encodes the current state of the robot given the current frame and last k frames as input. Note that this module also encodes information about the objects and environment of the robot. The architecture is similar to the prompt encoder, that is, a per-frame Image encoder ϕ_s followed by a Perceiver Resampler ψ_s . Similar to the prompt encoder’s outputs, the output of the state encoder is $\psi_s(\phi_s(S_t)) = z_{state}$ that encodes the latent environment and robot state information from the history of recent observations.

We use the same image encoder weights for both (1) and (2), that is, $\phi_p = \phi_s = \phi$. The role of the image encoder ϕ is to capture spatial visual information in each frame. The Perceiver Resampler is used to enable temporal learning across frames as well as reduce the number of video tokens that must be passed into the action decoder.

(3) **State-Prompt Encoder** The state-prompt encoder takes the prompt video encoding z_{prompt} and robot state encoding z_{state} and outputs a task encoding relevant for action prediction $z_{state|prompt}$. The module is trained to output robot actions by cross-attending between the state encoding as queries and the prompt video encoding as keys and values. Intuitively, the state-prompt encoder enables fusion of the state and prompt information. For example, if the prompt video demonstrates picking up of an apple in the basket and the current state contains apple, banana and orange, then the cross attention between the state and prompt encoding enables learning for

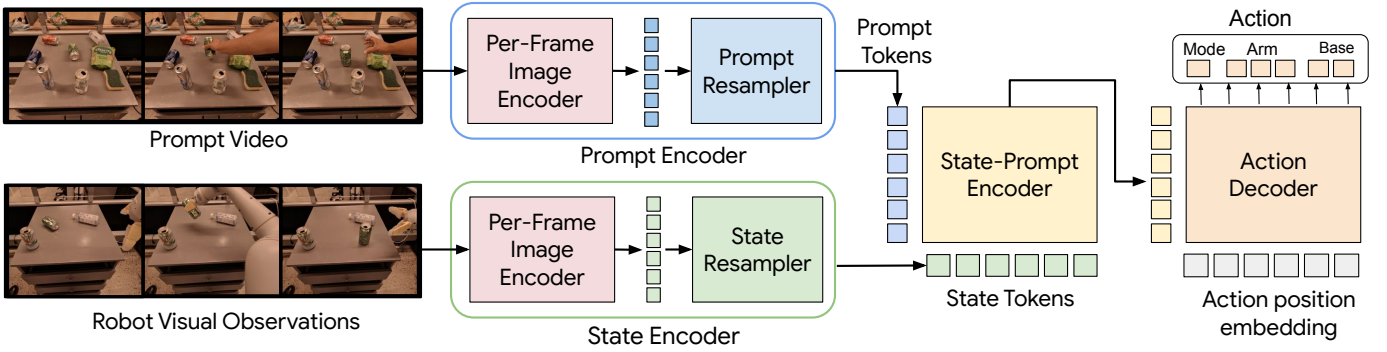


Fig. 3: Architecture. Our model takes as input frames of the prompt video and the robot’s current observations, encodes those into prompt video and robot state token embeddings, which are then processed through into state-prompt encoder and decoded into a robot action for the current timestep. More details in §II-C.

which object to attend to in the state, which is crucial for the next step of action decoding. We refer to the output of the state-prompt encoder as prompt-aware state tokens.

(4) Robot Action Decoder The goal of the action decoder is to predict the action vector a_t for the current state S_t such that it completes the task shown in the prompt video V_p . The action decoder is a transformer decoder architecture that takes in the fixed action position tokens [50] as input queries and the prompt-aware state tokens $z_{state|prompt}$ for keys and values. The size of the action position embedding is $N \times d$ where N is the number of action dimensions and d is the transformer embedding dimension. More details on the action vector in §II-D.

The action position embeddings cross-attend to the prompt-aware state tokens to predict the target binned action values as output. Each output token of the action decoder corresponds to an action dimension for the mode, arm and base. Specifically, each token embedding is projected to 256 dimensions and a softmax layer is applied on the top to obtain the bin corresponding to the target action vector. Unlike prior work [8, 9] that use autoregressive action decoding that requires multiple forward passes during inference, we use action position embeddings for one forward pass prediction like in ACT [50]. Instead of predicting one action for the next timestep, we follow the approach outlined in [22, 50] and train the policy with a prediction horizon of four steps. We always use the action bin that has the highest probability, i.e. argmax over predicted probabilities, to choose the action value for execution.

Cross-Attention Layers. In the Vid2Robot architecture, we use Cross-Attention Transformer layers extensively. They are used in the following modules: Prompt Resampler, State Resampler, State-Prompt Encoder and Action Decoder. We found Cross-Attention layers are helpful in managing the high number of tokens and the resulting large attention matrices when processing both prompt videos and robot state videos. This is because the standard self-attention layers would require orders of magnitude more memory to process the same video. For example, when using ViT-B/16 the total number

of video tokens for a 16 frame reference video and a 8 frame robot state video at 224×224 resolution would be $8 \times 196 + 16 \times 196 = 4704$. A full self-attention operation on this would lead to an attention matrix with $4704^2 \sim 22\text{M}$ entries. However, by using two Perceiver Resamplers with 64 latents we were able to train with attention matrices of the size $8 \times 196 \times 64 + 16 \times 196 \times 64 \sim .3\text{M}$. Thus, cross attention layers in Vid2Robot play an important role in reducing attention computation and enabling training with paired videos.

D. Preprocessing

To handle the varying lengths of videos for efficient training, we randomly sample $N = 16$ frames always including first and last frames and sort them in increasing order of time. During training, we sample a robot state S_t by sampling a random timestep first. We then select the preceding $k - 1$ frames to create a robot state video comprising of a total of $k = 8$ frames before. In case there are less than $k - 1$ frames before the current time-step, we repeat the first frame to create a fixed size robot state video. The pixel values in each frame are normalized between 0 to 1. Each frame is resized to $(224, 224)$. Photometric distortions like cropping, brightness, contrast, hue and saturation are applied during training.

The action at that timestep consists of the 3 components: *Mode*: (m) whether to terminate episode, move only arm, move only base or both. *Arm*: gripper position (x, y, z), orientation (rotation along xy, yz, zx), and the degree of closedness (c). *Base*: displacement (x, y) and rotation. Overall, the action $a_t = [m, g_x, g_y, g_z, \theta_{xy}, \theta_{yz}, \theta_{zx}, c, b_x, b_y, b_\theta]$ is an 11-dimensional vector. Each of the values have different ranges, which we first use to scale the values in between 0 and 1. We then discretize the values into 256 bins each. In total, we construct 11-dim action vector as target, each of which has value between $[0, 255]$.

E. Training

Action Prediction Loss We train Vid2Robot end-to-end with behavior cloning. The idea is to learn video representations from raw pixels to recognize task verb and objects, as well as

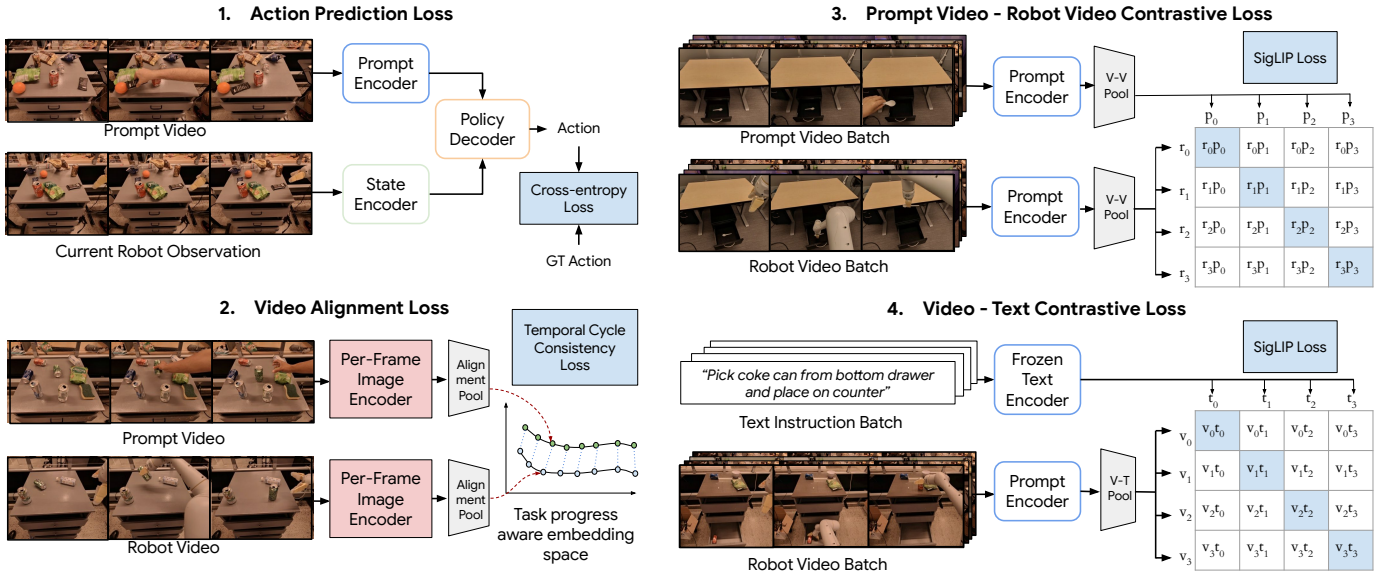


Fig. 4: Training Setup. We show all the losses Vid2Robot is trained with and how each loss is connected to its different modules. Along with (1) the main action prediction loss, we apply three auxiliary losses: (2) temporal video alignment loss, (3) a contrastive loss between the prompt and robot video performing the same task, and (4) a contrastive loss between a prompt/robot video with the language embedding. More details in §II-E.

learn motor control to accomplish it. We use a classification loss on actions that have been tokenized into $N=256$ bins. Given the robot trajectory for performing a task with current visual observations x_t , we have the corresponding expert action a_t . The action prediction loss is Cross Entropy between the predicted action and the expert action as:

$$L_{CE}(a_t, \hat{a}_t) = \sum_{\tau} a_t \log \hat{a}_t \quad (1)$$

This trains all the model parameters, as shown in Fig 3.

Auxiliary Losses. Although our dataset size is substantial, it is insufficient for training large Transformer based models. In order to prevent over-fitting by just predicting actions correctly on the training set, we add three auxiliary losses that encourage learning features that are helpful in understanding semantics in prompt videos.

Video Alignment Loss: We want to encourage temporal alignment between prompt videos and robot videos performing that show the same task. By aligning prompt videos and robot videos, we want the image encoder to learn to be invariant to different embodiments, lighting, backgrounds, view-angles and distractor objects while still encoding features relevant to predicting task progress. Our choice of loss is the temporal-cycle consistency loss introduced in [18]. This loss has been shown to encode task progress when trained on videos of different agents performing the same task [48]. This loss is applied on per-frame image embeddings of the prompt V_p and robot V_r videos during training. To apply the loss, we average pool the per-frame embeddings output in spatial dimensions from image encoder ϕ and apply a projector head of 2-layer MLP [11]. We call this as *alignment pooling layer* Φ on the per-frame image embeddings, as shown in Fig 4.

For each video V_i , this results in a sequence of embeddings $E_i = \{\Phi(v_i^1), \Phi(v_i^2), \dots, \Phi(v_i^{L_i})\}$, where L_i is the length of the i^{th} video.

We apply TCC loss on encoding E_p , and E_r for prompt and robot video respectively. The intuitive idea of TCC loss is that we want to ensure the representation of every frame of E_p should have a correspondence in E_r and vice versa. This involves two steps: First, we compute soft neighbor of t^{th} frame of E_p (E_p^t in short) in E_r and call it \widehat{E}_{pr}^t .

$$\widehat{E}_{pr}^t = \sum_k^{L_r} \alpha_k E_r^k, \quad \text{where } \alpha_k = \frac{e^{-\|E_i^t - E_j^k\|^2}}{\sum_k^{L_j} e^{-\|E_i^t - E_j^k\|^2}} \quad (2)$$

Second, we find the corresponding frame for this newly computed soft-neighbour in E_p . This is called *cycle-back* in [18] and it involves similar soft-neighbour computation as in Equation 2 to obtain say \widehat{E}_{pr}^t , which ideally should be same as t , that is, $(\widehat{E}_{pr}^t - t)^2$ should be minimized. TCC loss minimizes such mean squared error between all frames for prompt and robot video encodings and vice-versa, that is,

$$L_{TCC}(E_p, E_r) = \sum_{t \in V_p} (\widehat{E}_{pr}^t - t)^2 \quad (3)$$

$$L_{TCC} = \frac{L_{TCC}(E_p, E_r) + L_{TCC}(E_r, E_p)}{2}$$

Prompt-Robot Video Contrastive Loss (VVCL): We apply contrastive loss between prompt tokens produced by robot or prompt video performing the same task. This loss encourages the prompt encodings to learn task semantics from video tokens only in a self-supervised manner. A thing to note here is that the initial pairing of prompt and robot video has been done using natural language. However, by using a contrastive

loss only on video embeddings with a self-supervised loss, we hope to encode features not covered by the short natural language embedding itself. Examples of these features include similar motions like reaching for objects, and rotating the robot arm. We use a Attention Pooling layer to merge features from the N prompt tokens to produce a single embedding for each video. We apply the SigLIP [49] loss between video-video pairs to encourage videos showing same task, involving similar motions and interacting objects, to be close to each other while being away from other videos in the batch. A batch contains the same number of robot videos and prompt videos, say B . We use the prompt encoder $\psi_p(\phi(\cdot))$ to obtain a batch of full robot video embeddings Z_{robot} and prompt video embeddings Z_{prompt} , each of size $B \times d$. We multiply them, $Z_{robot} \cdot Z_{prompt}^T$ to obtain a $B \times B$ matrix. Adding a learnable temperature τ and bias b , we have our logit matrix as $\hat{Y} = (Z_{robot} \cdot Z_{prompt}^T) * \tau + b$. We consider the videos of robot and prompt performing the same task as positives and assign them a label of 1 along the diagonal and -1 for off-diagonal pairs, that is, the label matrix $Y = 2I_B - 1$. SigLIP loss is the negative loglikelihood, specifically, $\sigma'(Z_1, Z_2) = -\sum \log \sigma(Y \cdot (Z_1 \cdot Z_2^T) * t + b)$, where $Y = 2I_B - 1$. The video-video contrastive loss is then defined as:

$$L_{VVCL} = \sigma'(Z_{prompt}, Z_{robot}) \quad (4)$$

Video-text Contrastive Loss (VTCL): We add a contrastive loss between prompt tokens Z_{prompt} and those produced by robot video Z_{robot} and the embedding of the text instructions of the task Z_{text} . This encourages a part of the embedding space to be aware of object names and verbs present in the prompt and the robot videos. A version of this loss has been applied before by BC-Z [22] as auxiliary language regression loss. We use an Attention Pooling layer [47] with one latent query to merge features from the N prompt tokens to produce a single embedding for each video. Within a batch, we retrieve B pairs of video and text embeddings. Similar to Equation 4, we apply SigLIP [49] loss to get

$$L_{VTCL} = \frac{\sigma'(Z_{prompt}, Z_{text}) + \sigma'(Z_{robot}, Z_{text})}{2} \quad (5)$$

This encourages every video to have similar embeddings to their textual description embeddings, while being different from the text embeddings corresponding to other videos in the batch.

Overall, we apply the mean of all four losses for training that is $L = \frac{1}{4}(L_{CE} + L_{TCC} + L_{VVCL} + L_{VTCL})$.

F. Implementation

We trained the model (implemented in Jax) for 200K iterations. We use AdamW optimizer with an initial learning rate of $8e-5$ using a cosine learning rate schedule with warmup steps 2,000 and final learning rate of $1e-6$. For both the Prompt and State Resamplers, we use 2 Perceiver Resampler layers with 64 latents. Both state-prompt encoder and action decoder are 4 layer deep cross-attention transformers.

III. EXPERIMENTS

We present results with real robot evaluations for our multi-task video-conditioned policy. One of the key questions that we tackle in this work is how well robots can imitate humans performing manipulation tasks. Because of differences in embodiments, humans perform manipulation tasks at a different speed and style. We study the effect of using robot as well as human videos as prompts.

Metrics. We refer to a *rollout* as a sequence of actions inferred from the policy and executed on the robot from an initial state observation and prompt video, until the policy terminates or a maximum number of steps are taken, whichever is lower. We define *success* for a rollout when the policy executes the task instruction shown in the prompt video. A successful rollout involves correct actions to be taken successively in the environment, without any assistance for resets or recovery. For each task instruction, we record many rollouts per policy. We take the average of success recorded across all the rollouts and call it the *Success Rate* for that task. Aggregated success rate across tasks is referred as *Overall Success Rate*.

A mistake made early on in a rollout can result in poor success rate, even if the model’s offline overall prediction accuracy is high. For example, if a policy makes an error while grasping a water bottle early on in the task and it slips to an unreachable location, the rollout will be marked as a failure, even if the policy had good action predictions for the later steps. To record partial progress for a rollout, we annotate whether the robot *reached* the correct location, *grasped* the correct object, *released* the object at the correct location, and *terminated* the task correctly. More details on partial success analysis in §III-A.

Evaluation Setup. We ask human raters to evaluate success for a policy’s rollout on a robot. We evaluate the policies by varying the robots, lighting conditions, chest of drawers, and objects. We ensure the policies being evaluated are shown similar initial object configurations during rollouts. The initial state is randomized after all policies have been evaluated for a given initial state. For all rollouts, we sample prompt videos that are not seen by the models during training. This ensures that the policies are evaluated for whether they can recognize the task from new prompt videos or not.

Baselines. We compare our model with BC-Z [22], a video conditioned policy using a ResNet-18 encoder. BC-Z [23] involves demonstration-observation pairs processed via a FiLM [37] conditioned ResNet encoder and fed into a ResNet based policy network to predict robot actions. For a fair comparison, we train the BC-Z model with the same training data used to train the Vid2Robot model. BC-Z doesn’t have a terminate action, so we run its rollouts for a fixed maximum number of steps.

Key Questions and Results We address the following questions in this work:

- 1) How well do video-conditioned policies perform when they are shown a task in an unseen video? (Fig 5, § III-A)

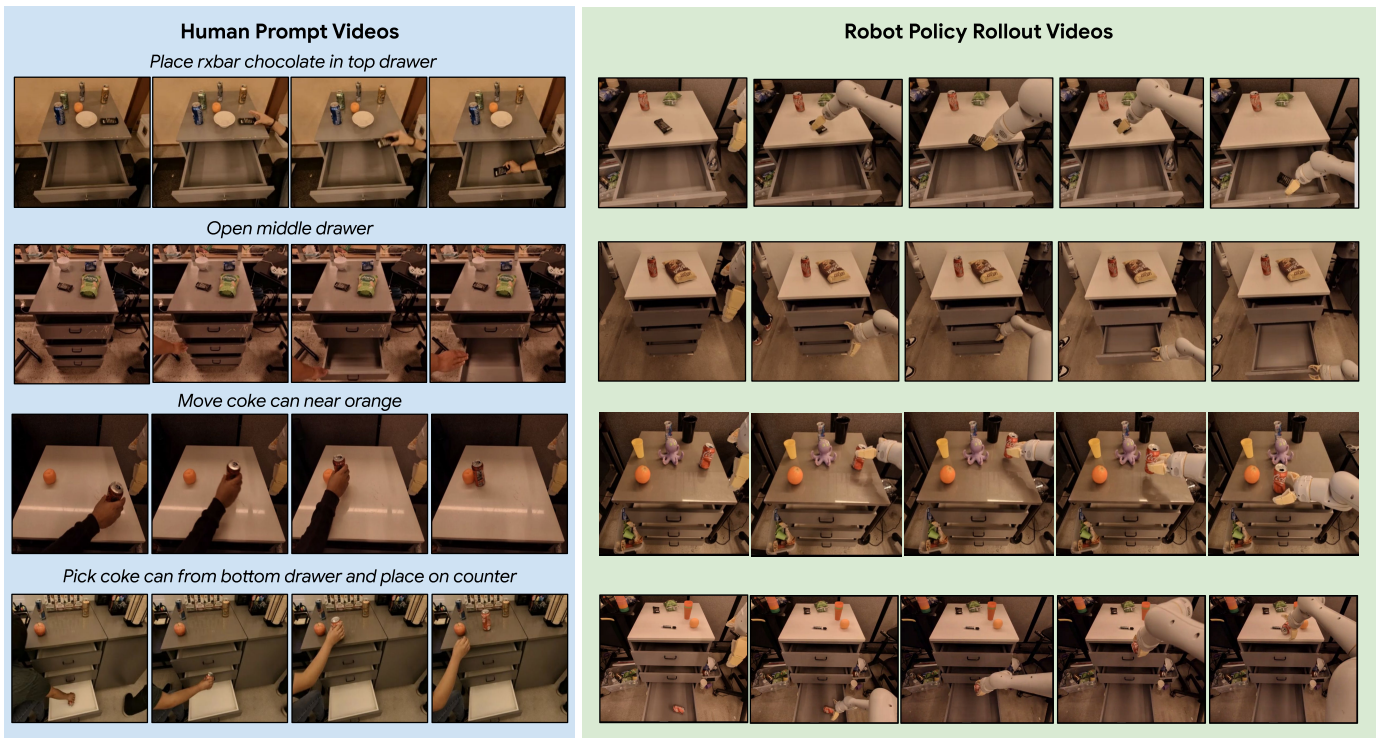


Fig. 5: Policy Rollouts. Each row shows a prompt video of a human doing a task on the left, and on the right we show the corresponding successful robot rollouts using Vid2Robot. Note how visually different the prompts are, while the policy rollouts are recorded with different lighting, background, as well as number and placement of the distractor objects.

TABLE I: Task Success Rate for Robot and Human prompts.

Prompter	Model	<i>pick</i>	<i>pick-place on</i>	<i>place into</i>	<i>open</i>	<i>close</i>	<i>move near</i>	<i>knock over</i>	<i>place upright</i>	Overall
Robot	BC-Z	75.0%	50.0%	61.5%	16.7%	66.7%	44.0%	58.3%	50.0%	52.6%
	Vid2Robot	75.0%	58.8%	50.0%	91.7%	100.0%	33.3%	41.7%	16.7%	54.9%
Human	BC-Z	50.0%	12.5%	12.5%	0.0%	50.0%	43.8%	12.5%	50.0%	30.6%
	Vid2Robot	100.0%	50.0%	50.0%	62.5%	87.5%	43.8%	25.0%	12.5%	52.8%

- 2) What is the gap in success rate due to prompt embodiment difference (robot v/s human)? (§ III-A)
- 3) Can we leverage the learned motion representations for out-of-distribution object interactions? (§ III-B)

A. Task-based success

We compare our Vid2Robot model and baseline BC-Z with robot and human prompt videos in Table I. Both Vid2Robot and BC-Z were trained on a same data mixture containing robot-robot and human-robot paired data. Prompt videos cover a subset of the training tasks but the videos themselves are new for the models. In this evaluation, we investigate what each model’s ability is to infer the task specification from prompt video as well as the current observed state of the robot.

In order to test the capabilities of the model in different settings on real robot, we evaluate it across eight categories of manipulation tasks as shown in Table I. Specifically, we evaluate for nine tasks: ‘knock water bottle over’, ‘move rxbar chocolate near coke can’, ‘move green jalapeno chip bag near

coke can’, ‘pick green rice chip bag’, ‘place coke can upright’, ‘pick coke can from bottom drawer and place on counter’, ‘open middle drawer’, ‘close middle drawer’, and ‘place apple into top drawer’.

We ask four evaluators to carry out two rollouts per task for a prompt video dataset and policy setting (a row in Table I), that implies, we have eight trials per task to evaluate a policy’s task success rate. We report overall success rate per row over nine tasks with eight trials per task, that is, $9 \times 8 = 72$ trials. In total, our evaluations in Table I required $72 \times 4 = 288$ real robot rollouts.

1) *What is the gap in success rate due to embodiment difference in prompt videos?*: We compare our model with BC-Z when prompted with robot and human videos. BC-Z serves as a strong baseline for our comparisons. The overall success rate of our model Vid2Robot outperforms BC-Z for Human prompt videos by 20%, and is comparable for Robot prompt videos. Note that there is an order of magnitude more training samples for robot trajectories than human videos in our training mixture. Hence, there isn’t a significant gap

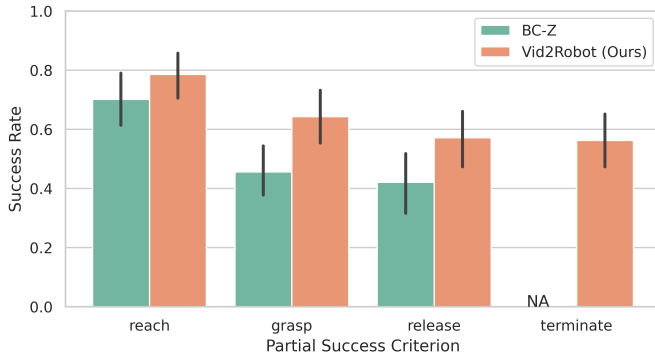


Fig. 6: Partial Success Rate for BC-Z and Vid2Robot. Our policy Vid2Robot outperforms BC-Z in terms of *reaching* the correct object, *grasping* it, *releasing* it at the correct location and then *terminating* the episode correctly. Note that BC-Z does not have terminate control.

TABLE II: Cross-object motion transfer success.

Model	<i>pick</i>	<i>pick-place on</i>	<i>place into</i>	<i>place upright</i>	<i>knock over</i>	Overall
BC-Z	45.8%	0.0%	29.2%	12.5%	0.0%	17.5%
Vid2Robot	45.8%	25.0%	54.2%	16.7%	29.2%	34.2%

in performance for robot prompt videos. For human prompt videos, our model outperforms BC-Z in most tasks, showing that Vid2Robot captures the task semantics from prompt videos better than the baseline. Our model outperforms in tasks like picking from drawer and placing on the counter, and opening/closing drawer tasks by a large margin. The most challenging task is *placing upright* and *knocking over*. We analyze the failure reasons in §V Fig 9.

2) *How well do video-conditioned policies perform when they are shown a task in an unseen video?:* In addition to marking a rollout as a success, we recorded partial success annotations per rollout. In Fig 6, we observe that our model *reaches* to the correct object 78% of the time, about 8% more as compared to baseline. The policies sometimes fail to reach the correct object and go towards a distractor instead. Next, *grasping* errors happen, particularly with small and deformable objects and in collision prone areas like drawer handle or counter’s edge. Here our model (65%) outperforms BC-Z (45%) by a large margin of 20%. A successful grasp often the most difficult part in a rollout, and the most crucial for success. After grasping, most tasks require *releasing* at a correct location. There is a slight drop in success rate in both models due to incorrect release during the rollouts. While BC-Z runs for a fixed number of steps, our policy Vid2Robot predicts when to terminate. We observe that the rate of *release* and *terminate* is almost identical, about 57% for our model, that implies, that after releasing at correct location, Vid2Robot mostly terminates successfully.

B. Cross-object motion transfer

Our policy and baseline were trained with paired videos as discussed in §II-B. This implies that the training data included

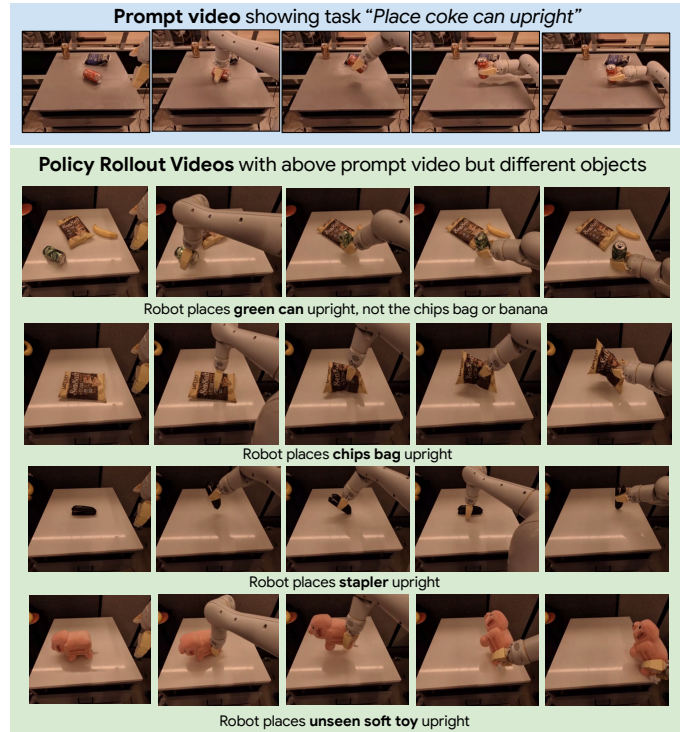


Fig. 7: Qualitative results for cross-object motion transfer. Given a prompt video of *placing coke can upright*, we rollout the policy with a *green can*, *chips bag*, *stapler* and a *soft toy* in front of the robot. We observe that our model can infer the motion of *place upright* in the prompt video and apply it on other objects. There is an implicit notion of pragmatics in the policy as shown by the selection of green can over other objects.

only those scenarios where the interaction object shown in prompt is present in the current robot observations. But *what if we provided a prompt video of one object and tested on other objects. Does it do the same motion as shown in the prompt video?* Interestingly, we found our model to perform learned manipulation actions on objects that it has not seen in train set. We call this emergent behavior as *cross-object motion transfer*.

We compare Vid2Robot with BC-Z for cross object motion transfer ability with five prompt videos, namely, ‘knock water bottle over’, ‘pick green rice chip bag’, ‘place coke can upright’, ‘pick coke can from bottom drawer and place on counter’, and ‘place apple into top drawer’. Each prompt video is evaluated with unrelated objects in robot’s initial observation. The objects used for evaluation are ‘orange’, ‘green can’, ‘chips bag’, ‘banana’, ‘pink piggy soft toy’, ‘wrist watch’. We selected objects to have diverse shape, size, and deformability to evaluate situations that require different grasps for success.

The evaluation setup is similar to §III-A. Here the evaluator sets up one of the object for a task and records rollouts for each model. We compare 2 models on 5 tasks with 6 objects, so every evaluator runs $2 \times 5 \times 6 = 60$ rollouts. We repeat the evaluation with four raters, thus reporting results in Table II

on a total of $4 \times 60 = 240$ rollouts.

1) *Can we provide a prompt video of one object and test it on other objects? Does the policy do the same motion as shown in the prompt video?:* In Fig 7, we show the above experimental setup qualitatively. We use a prompt video to ‘place coke can upright’. We observe that the policy is able transfer the action of ‘placing upright’ to several objects, like a green can, a chips bag, a stapler, and a soft toy. Note that the policy adheres to the prompt video and chooses green can over chips bag or banana for placing upright.

Quantitatively, we observe that BC-Z is often unable to successfully complete the tasks when testing cross-object motion transfer, as shown in each task in Table II. In contrast, our model (34%) performs better than BC-Z (17%) in this setting and performs the motion indicated in the prompt video. Our model is comparable to BC-Z with 45% success rate on *picking* out-of-distribution objects. More importantly, tasks involving placing into drawers demonstrates significant improvement (29% \rightarrow 54%). For certain tasks like picking from drawers and placing on counters and knocking over, BC-Z is unable to perform at all whereas Vid2Robot is able to complete the task 25% – 29% of the time.

C. Ablations

In §II-E, we presented action prediction loss and three auxiliary losses. Here we analyze the role of these additional loss functions to the overall success rate. We investigate the impact of (1) not using any auxiliary loss, and (2) adding auxiliary language loss.

We consider the tasks similar to that described in §III-A, that is, 9 tasks for evaluating each policy.

We have 3 model variants, namely, the original Vid2Robot, the one without video-text contrastive loss (CL) and the one with only action prediction loss. We ask 3 human evaluators to run the each model variant with 2 rollouts each. In total, we report results with $3 \times 3 \times 9 \times 2 = 162$ rollouts in Fig 8. The error bars indicate the standard deviation for success reported on rollouts with each model variant.

1) *What is the impact of not using any auxiliary loss?:* We observe that the performance of our model (61%) is significantly improved by enforcing representation constraints through auxiliary losses, in comparison to using only action prediction loss (45%). It highlights the importance of the proposed auxiliary losses in §II-E.

2) *What is the impact of the auxiliary language loss?:* BC-Z proposed to use language representations to improve video representations for conditioning the policy. We compare our policy with another variant trained with all losses but the Video-Text CL. We observe only marginal improvement of 1-2% in success rate when using the language loss. This implies that video alignment and video contrastive loss contribute significantly towards performance improvement. Our results hope to serve as a promising evidence that effective video representations can be learned without auxiliary losses that use pre-trained language embeddings.

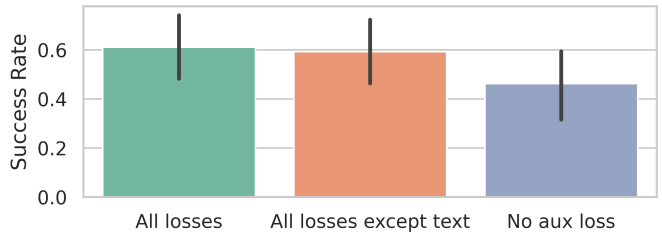


Fig. 8: Ablation for auxiliary losses used in Vid2Robot. We compare our proposed approach that has all auxiliary losses (green, left) with a variant without language contrastive loss that was originally proposed in BC-Z (orange, middle) and a version with no auxiliary losses (blue, right). More details in (§III-C)

IV. RELATED WORK

Task Specifications for Robots The development of general-purpose robots hinges on effectively grounding task specifications. Videos are a dense source of information that not only provide what to do but also how to do it in physical world. Recent works have used videos for task specification [4, 24, 41]. Another line of work uses videos to learn world models to predict future visual observations [30, 27, 10, 32, 16]. While language [46, 8, 34, 35], final goal images [25, 7], and others like hand-drawn inputs [44] have been proposed as means for task specification, learning from prompt videos is complementary to these approaches and inevitable for rapid adaptation of trained policies to perform new manipulation skills at deployment.

Learning from Human Demonstrations As videos of humans performing various tasks proliferate the internet, several works aim to address how to best leverage this information for robot learning. The difference in robot vs human embodiment poses a significant challenge, for which existing approaches range from translating image of a human into the robot [43] to inpainting for agent-agnostic representations [3]. Many prior works propose to leverage off-the-shelf models for hand pose estimation and contact tracking [5, 13, 38], object-centric representations [39, 21], as well as reward functions for reinforcement learning [3, 28, 43]. Other methods [33, 45, 5] cast this problem into visual representation learning to accelerate learning of downstream motor control tasks. While these modular learning solutions work well in limited datasets, these are prone to compounding error of each of its component, and thus, not efficiently scalable. End-to-end training approaches for goal-conditioned imitation learning [12, 42, 19, 14] and reinforcement learning [40, 36] are promising alternatives to these techniques, but these results have been largely limited in simulation and hindered by sim-to-real gap. In contrast, we choose to tackle this as an end-to-end large multi-task learning from human videos with real robot evaluations.

Imitation via Paired Demonstrations Our setup of paired prompt videos and robot trajectory is most similar to One-Shot Visual Imitation literature. Many prior works assume access to pairs, where the first is used as the demonstration of the task to be performed, and the second as the observation of the agent.

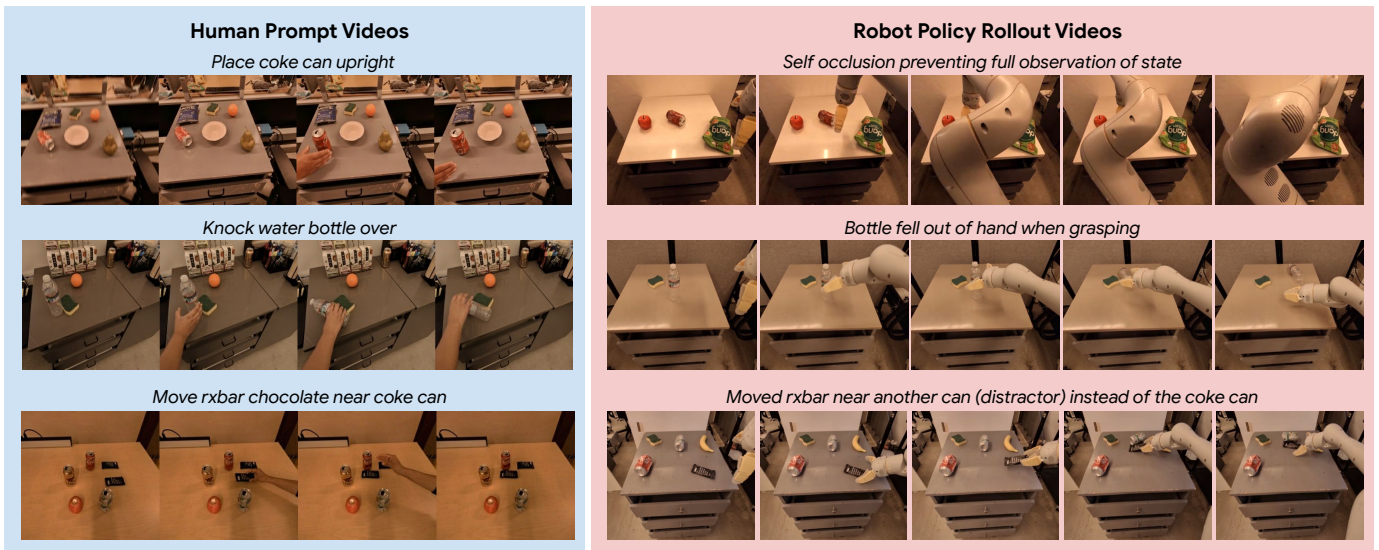


Fig. 9: Failure analysis with policy rollouts. (Top) Policy predicts gripper pose and depends on the IK solver to move the arm. Sometimes, the IK solution can block the robot’s camera view. (Middle) Grasping failures happen, especially with transparent and deformable objects. (Bottom) Distractor objects as well as difference in lighting and background may cause recognition errors, where policy might perform the correct motion but with incorrect object(s).

Some of the early works [17] proposed training a demonstration network via temporal convolution and neighborhood attention to condition a manipulation policy network. In more recent approaches like [12, 29, 21], paired demonstrations and observations are used to train a transformer policy, often with additional constraints like inverse dynamics prediction[12] or contrastive representation learning [29]. However, these approaches are largely evaluated in specific set of simulated tasks, and not compared on real robots. Most similar to our work is BC-Z [23] which reports evaluations with real robot tasks. While our setup is similar to some of this prior art, our model Vid2Robot couples large image encoders, cross-attention layers, and contrastive auxiliary losses to learn a manipulation policy that imitates a human showing a task.

V. LIMITATIONS AND FUTURE DIRECTIONS

In §III, we show that our approach has improved over previous work but there is a gap in performance for video-conditioned policies. Language conditioned policies like [9] shows a higher success for known set of tasks with several hundreds of teleoperation trajectories for training. We, on the other hand, accomplish the first milestone of evaluating the video-conditioned policies in the similar setup. We discuss three limitations of our work and provide insights for future directions here.

First, we qualitatively investigate some reasons for failure of a policy rollout. In Fig 9, we illustrate and explain 3 examples showing how self occlusion, grasping errors and presence of distractors can lead to failure during any rollout. Second, we observe a significant drop in the grasping success in Fig 6. While we use robot camera observation to estimate the state and implicitly learn depth estimation, it is often incomplete when there is occlusion or when the robot gripper

is out of camera view. By enhancing the state information with multimodal sensor fusion, we may improve the grasp success rate. Third, we consider carefully collected short task instruction demonstrations from three different sources as shown in §II-B, all of which are 5 to 20 seconds videos. To test our models on long horizon demonstrations or ‘in-the-wild’ videos online, we need effective pairing strategies for videos and a few corresponding robot trajectories to train the policy.

VI. CONCLUSION

We demonstrate novel methods for both data collection and modeling for video conditioned skill learning. These skills generalize to novel object configurations and more abstracted verb meanings when no immediately obvious object is visible. The skills and generality provided by our model complement other approaches to widen the set of skills that robots have access to, and to include skills not otherwise easily acquired. Future work can leverage these learned primitives to execute novel task plans. We hope our cross-object motion transfer experiments will encourage further research in transferring motion to new objects and settings for bootstrapping data collection, and enabling human-robot interaction with rapid adaptation to new skills.

ACKNOWLEDGMENTS

We would like to thank Yansong Pang, Grecia Salazar, Utsav Malla, Deeksha Manjunath, Jornell Quiambao, Sarah Nguyen, Sangeetha Ramesh, Tran Pham, Samuel Wan, Tomas Jackson, Jodilyn Peralta, Celeste Barajas, Elio Prado, Rochelle Dela Cruz, Alex Luong and Krista Reymann for supporting data collection via teleoperation. Special thanks to Jornell Quiambao, Grecia Salazar, Utsav Malla, Deeksha Manjunath,

TABLE III: Author Contributions

Name	Data Collection	Model Training	Evaluations	Infrastructure	Leadership	Paper Writing
Ayzaan Wahid		✓				
Christine W.Y. Chan	✓					
Danny Driess				✓		✓
Debidatta Dwibedi	✓	✓	✓	✓	✓	✓
Igor Gilitschenski					✓	✓
Maria Attarian	✓	✓		✓		✓
Nikhil J Joshi	✓		✓	✓	✓	✓
Pannag R Sanketi	✓			✓	✓	
Pierre Sermanet	✓					
Quan Vuong		✓		✓		
Stefan Welker	✓			✓		
Vidhi Jain	✓	✓	✓	✓	✓	✓
Yonatan Bisk					✓	✓

Sarah Nguyen, Sangeetha Ramesh, and Jaspiar Singh for evaluations on robot; Michael Ahn, Anthony Brohan and Keerthana Gopalakrishnan for policy evaluation infrastructure; Suneel Belkhale, Dorsa Sadigh, Chelsea Finn, and Sergey Levine for helpful discussions; Jonathan Tompson, and Vincent Vanhouke for thorough feedback on the writing.

REFERENCES

[1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andy Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. Flamingo: a visual language model for Few-Shot learning. *ArXiv*, abs/2204.14198, 2022.

[2] Montserrat Gonzalez Arenas, Ted Xiao, Sumeet Singh, Vidhi Jain, Allen Z. Ren, Quan Vuong, Jake Varley, Alexander Herzog, Isabel Leal, Sean Kirmani, Dorsa Sadigh, Vikas Sindhwani, Kanishka Rao, Jacky Liang, and Andy Zeng. How to prompt your robot: A prompt-book for manipulation skills with code as policies. In *2nd Workshop on Language and Robot Learning: Language as Grounding*, 2023. URL <https://openreview.net/forum?id=T8AiZj1QdN>.

[3] Shikhar Bahl, Abhinav Gupta, and Deepak Pathak. Human-to-robot imitation in the wild. *arXiv preprint arXiv:2207.09450*, 2022.

[4] Shikhar Bahl, Russell Mendonca, Lili Chen, Unnat Jain, and Deepak Pathak. Affordances from human videos as a versatile representation for robotics. In *Computer Vision and Pattern Recognition*, April 2023.

[5] Shikhar Bahl, Russell Mendonca, Lili Chen, Unnat Jain, and Deepak Pathak. Affordances from human videos as a versatile representation for robotics. 2023.

[6] Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about Physical Commonsense in Natural Language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2020. URL <https://yonatanbisk.com/piqa>.

[7] Konstantinos Bousmalis, Giulia Vezzani, Dushyant Rao, Coline Devin, Alex X. Lee, Maria Bauzá, Todor Davchev, Yuxiang Zhou, Agrim Gupta, Akhil S. Raju, Antoine Laurens, Claudio Fantacci, Valentin Dalibard, Martina Zambelli, Murilo Fernandes Martins, Rugile Pevcevičiute, Michiel Blokzijl, Misha Denil, Nathan Batchelor, Thomas Lampe, Emilio Parisotto, Konrad Zolna, Scott E. Reed, Sergio Gomez Colmenarejo, Jonathan Scholz, Abbas Abdolmaleki, Oliver Groth, Jean-Baptiste Regli, Oleg O. Sushkov, Tom Rothorl, José Enrique Chen, Yusuf Aytar, David Barker, Joy Ortiz, Martin A. Riedmiller, Jost Tobias Springenberg, Raia Hadsell, Francesco Nori, and Nicolas Manfred Otto Heess. Robocat: A self-improving generalist agent for robotic manipulation. September 2023.

[8] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil J Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jor-nell Quiambao, Kanishka Rao, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-1: Robotics transformer for Real-World control at scale. In *arXiv preprint arXiv:2212.06817*, December 2022.

[9] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alex Herzog, Jasmine Hsu, Brian Ichter, Alex Irpan, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Lisa Lee, Tsang-Wei Edward Lee, Sergey Levine, Yao Lu, Henryk Michalewski, Igor Mordatch,

- Karl Pertsch, Kanishka Rao, Krista Reymann, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Pierre Sermanet, Jaspiar Singh, Anikait Singh, Radu Soricut, Huong Tran, Vincent Vanhoucke, Quan Vuong, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Jialin Wu, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control. In *arXiv preprint arXiv:2307.15818*, 2023.
- [10] Elliot Chane-Sane, Cordelia Schmid, and Ivan Laptev. Goal-Conditioned reinforcement learning with imagined subgoals. July 2021.
- [11] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E Hinton. Big self-supervised models are strong semi-supervised learners. *Advances in neural information processing systems*, 33: 22243–22255, 2020.
- [12] Sudeep Dasari and Abhinav Gupta. Transformers for one-shot visual imitation. In *Conference on Robot Learning*, pages 2071–2084. PMLR, 2021.
- [13] Eadom Dessalene, Chinmaya Devaraj, Michael Maynard, Cornelia Fermuller, and Yiannis Aloimonos. Forecasting action through contact representations from first person video. *IEEE Trans. Pattern Anal. Mach. Intell.*, 45(6): 6703–6714, June 2023.
- [14] Yiming Ding, Carlos Florensa, Mariano Phielipp, and Pieter Abbeel. Goal-conditioned imitation learning. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 15324–15335. Curran Associates Inc., Red Hook, NY, USA, December 2019.
- [15] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=YicbFdNTTy>.
- [16] Yilun Du, Sherry Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Joshua B. Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL <https://openreview.net/forum?id=bo8q5MRcwY>.
- [17] Yan Duan, Marcin Andrychowicz, Bradly Stadie, OpenAI Jonathan Ho, Jonas Schneider, Ilya Sutskever, Pieter Abbeel, and Wojciech Zaremba. One-shot imitation learning. *Advances in neural information processing systems*, 30, 2017.
- [18] Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman. Temporal Cycle-Consistency learning. In *Computer Vision and Pattern Recognition*, 2019.
- [19] Oliver Groth, Chia-Man Hung, Andrea Vedaldi, and Ingmar Posner. Goal-Conditioned End-to-End visuomotor control for versatile skill primitives. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1319–1325. IEEE, May 2021.
- [20] Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, Olivier Hénaff, Matthew M Botvinick, Andrew Zisserman, Oriol Vinyals, and João Carreira. Perceiver IO: A general architecture for structured inputs & outputs. July 2021.
- [21] Vidhi Jain, Yixin Lin, Eric Undersander, Yonatan Bisk, and Akshara Rai. Transformers are adaptable task planners. In Karen Liu, Dana Kulic, and Jeff Ichnowski, editors, *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pages 1011–1037. PMLR, 14–18 Dec 2023. URL <https://proceedings.mlr.press/v205/jain23a.html>.
- [22] Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. BC-Z: Zero-Shot Task Generalization with Robotic Imitation Learning. In *Proceedings of the 5th Conference on Robot Learning*, pages 991–1002, 2022.
- [23] Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, pages 991–1002. PMLR, 2022.
- [24] Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. VIMA: General robot manipulation with multimodal prompts. In *Fortieth International Conference on Machine Learning*, 2023.
- [25] Jacob Krantz, Theophile Gervet, Karmesh Yadav, Austin Wang, Chris Paxton, Roozbeh Mottaghi, Dhruv Batra, Jitendra Malik, Stefan Lee, and Devendra Singh Chaplot. Navigating to objects specified by images. April 2023.
- [26] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023.
- [27] Yuxuan Liu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Imitation from observation: Learning to imitate behaviors from raw video via context translation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1118–1125. IEEE, May 2018.
- [28] Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. VIP: Towards universal visual reward and representation via Value-Implicit Pre-Training. September 2022.
- [29] Zhao Mandi, Fangchen Liu, Kimin Lee, and Pieter Abbeel. Towards more generalizable one-shot visual imitation learning. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2434–2444. IEEE, 2022.
- [30] Russell Mendonca, Shikhar Bahl, and Deepak Pathak.

- Structured world models from human videos. 2023.
- [31] Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. HowTo100M: Learning a Text-Video Embedding by Watching Hundred Million Narrated Video Clips. In *ICCV*, 2019.
- [32] Ashvin Nair, Vitchyr Pong, Murtaza Dalal, Shikhar Bahl, Steven Lin, and Sergey Levine. Visual reinforcement learning with imagined goals. July 2018.
- [33] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022.
- [34] Open X-Embodiment Collaboration, Abhishek Padalkar, Acorn Pooley, Ajay Mandlekar, Ajinkya Jain, Albert Tung, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anikait Singh, Animesh Garg, Anthony Brohan, Antonin Raffin, Ayzaan Wahid, Ben Burgess-Limerick, Beomjoon Kim, Bernhard Schölkopf, Brian Ichter, Cewu Lu, Charles Xu, Chelsea Finn, Chenfeng Xu, Cheng Chi, Chenguang Huang, Christine Chan, Chuer Pan, Chuyuan Fu, Coline Devin, Danny Driess, Deepak Pathak, Dhruv Shah, Dieter Büchler, Dmitry Kalashnikov, Dorsa Sadigh, Edward Johns, Federico Ceola, Fei Xia, Freek Stulp, Gaoyue Zhou, Gaurav S Sukhatme, Gautam Salhotra, Ge Yan, Giulio Schiavi, Gregory Kahn, Hao Su, Hao-Shu Fang, Haochen Shi, Heni Ben Amor, Henrik I Christensen, Hiroki Furuta, Homer Walke, Hongjie Fang, Igor Mordatch, Ilija Radosavovic, Isabel Leal, Jacky Liang, Jad Abou-Chakra, Jaehyung Kim, Jan Peters, Jan Schneider, Jasmine Hsu, Jeannette Bohg, Jeffrey Bingham, Jiajun Wu, Jialin Wu, Jianlan Luo, Jiayuan Gu, Jie Tan, Jihoon Oh, Jitendra Malik, Jonathan Booher, Jonathan Tompson, Jonathan Yang, Joseph J Lim, João Silvério, Junhyek Han, Kanishka Rao, Karl Pertsch, Karol Hausman, Keegan Go, Keerthana Gopalakrishnan, Ken Goldberg, Kendra Byrne, Kenneth Oslund, Kento Kawaharazuka, Kevin Zhang, Krishan Rana, Krishnan Srinivasan, Lawrence Yunliang Chen, Lerrel Pinto, Li Fei-Fei, Liam Tan, Lionel Ott, Lisa Lee, Masayoshi Tomizuka, Max Spero, Maximilian Du, Michael Ahn, Mingtong Zhang, Mingyu Ding, Mohan Kumar Srirama, Mohit Sharma, Moo Jin Kim, Naoaki Kanazawa, Nicklas Hansen, Nicolas Heess, Nikhil J Joshi, Niko Suenderhauf, Norman Di Palo, Nur Muhammad Mahi Shafiqullah, Oier Mees, Oliver Kroemer, Pannag R Sanketi, Paul Wohlhart, Peng Xu, Pierre Sermanet, Priya Sundaresan, Quan Vuong, Rafael Rafailov, Ran Tian, Ria Doshi, Roberto Martín-Martín, Russell Mendonca, Rutav Shah, Ryan Hoque, Ryan Julian, Samuel Bustamante, Sean Kirmani, Sergey Levine, Sherry Moore, Shikhar Bahl, Shivin Dass, Shubham Sonawani, Shuran Song, Sichun Xu, Siddhant Halder, Simeon Adebola, Simon Guist, Soroush Nasiriany, Stefan Schaal, Stefan Welker, Stephen Tian, Sudeep Dasari, Suneel Belkhale, Takayuki Osa, Tatsuya Harada, Tatsuya Matsushima, Ted Xiao, Tianhe Yu, Tianli Ding, Todor Davchev, Tony Z Zhao, Travis Armstrong, Trevor Darrell, Vidhi Jain, Vincent Vanhoucke, Wei Zhan, Wenxuan Zhou, Wolfram Burgard, Xi Chen, Xiaolong Wang, Xinghao Zhu, Xuanlin Li, Yao Lu, Yevgen Chebotar, Yifan Zhou, Yifeng Zhu, Ying Xu, Yixuan Wang, Yonatan Bisk, Yoonyoung Cho, Youngwoon Lee, Yuchen Cui, Yueh-Hua Wu, Yujin Tang, Yuke Zhu, Yunzhu Li, Yusuke Iwasawa, Yutaka Matsuo, Zhuo Xu, and Zichen Jeff Cui. Open X-Embodiment: Robotic learning datasets and RT-X models. *arXiv 2310.08864*, 2023.
- [35] Priyam Parashar, Vidhi Jain, Xiaohan Zhang, Jay Vakil, Sam Powers, Yonatan Bisk, and Chris Paxton. Spatial-Language Attention Policies for Efficient Robot Learning. In *Conference on Robot Learning*, 2023. URL <https://arxiv.org/abs/2304.11235>.
- [36] Xue Bin Peng, Angjoo Kanazawa, Jitendra Malik, Pieter Abbeel, and Sergey Levine. SFV: reinforcement learning of physical skills from videos. *ACM Trans. Graph.*, 37(6):1–14, December 2018.
- [37] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI conference on artificial intelligence*, 2018.
- [38] Vladimír Petrík, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. Learning object manipulation skills via approximate state estimation from real videos. In Jens Kober, Fabio Ramos, and Claire Tomlin, editors, *Proceedings of the 2020 Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, pages 296–312. PMLR, 2021.
- [39] Sören Pirk, Mohi Khansari, Yunfei Bai, Corey Lynch, and Pierre Sermanet. Online object representations with contrastive learning. 2019.
- [40] Karl Schmeckpeper, Oleh Rybkin, Kostas Daniilidis, Sergey Levine, and Chelsea Finn. Reinforcement learning with videos: Combining offline observations with interaction. November 2020.
- [41] Rutav Shah, Roberto Martín-Martín, and Yuke Zhu. MUTEX: Learning unified policies from multimodal task specifications. In *7th Annual Conference on Robot Learning*, 2023.
- [42] Pratyusha Sharma, Deepak Pathak, and Abhinav Gupta. Third-Person visual imitation learning via decoupled hierarchical controller. *Adv. Neural Inf. Process. Syst.*, 32, 2019.
- [43] Laura Smith, Nikita Dhawan, Marvin Zhang, Pieter Abbeel, and Sergey Levine. AVID: Learning Multi-Stage tasks via Pixel-Level translation of human videos. December 2019.
- [44] Austin Stone, Ted Xiao, Yao Lu, Keerthana Gopalakrishnan, Kuang-Huei Lee, Quan Vuong, Paul Wohlhart, Sean Kirmani, Brianna Zitkovich, Fei Xia, Chelsea Finn, and Karol Hausman. Open-World object manipulation using pre-trained Vision-Language models. March 2023.

- [45] Tete Xiao, Ilija Radosavovic, Trevor Darrell, and Jitendra Malik. Masked visual pre-training for motor control. *arXiv preprint arXiv:2203.06173*, 2023.
- [46] Sriram Yenamandra, Arun Ramachandran, Karmesh Yadav, Austin Wang, Mukul Khanna, Theophile Gervet, Tsung-Yen Yang, Vidhi Jain, Alexander William Clegg, John Turner, Zsolt Kira, Manolis Savva, Angel Chang, Devendra Singh Chaplot, Dhruv Batra, Roozbeh Mottaghi, Yonatan Bisk, and Chris Paxton. HomeRobot: Open-Vocabulary Mobile Manipulation. In *Conference on Robot Learning*, 2023. URL <https://arxiv.org/abs/2306.11565>.
- [47] Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. Coca: Contrastive captioners are image-text foundation models. *arXiv preprint arXiv:2205.01917*, 2022.
- [48] Kevin Zakka, Andy Zeng, Pete Florence, Jonathan Tompson, Jeannette Bohg, and Debidatta Dwibedi. Xirl: Cross-embodiment inverse reinforcement learning. In *Conference on Robot Learning*, pages 537–546. PMLR, 2022.
- [49] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 11941–11952, Los Alamitos, CA, USA, oct 2023. IEEE Computer Society. doi: 10.1109/ICCV51070.2023.01100.
- [50] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning Fine-Grained bimanual manipulation with Low-Cost hardware. In *Robotics: Science and Systems*, 2023.