2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Robotic Offline RL from Internet Videos via Value-Function Learning

Chethan Bhateja[1], Derek Guo[1], Dibya Ghosh[1], Anikait Singh[1,2], Manan Tomar[1],
Quan Vuong[2], Yevgen Chebotar[2], Sergey Levine[1,2], Aviral Kumar[1,2]
[1]UC Berkeley; [2]Google DeepMind

***Abstract*— Pre-training on Internet data has proven to be a key ingredient for broad generalization in many modern ML systems. What would it take to enable such capabilities in robotic reinforcement learning (RL)? Offline RL methods, which learn from datasets of robot experience, offer one way to leverage prior data into the robotic learning pipeline. However, these methods have a "type mismatch" with video data (such as Ego4D), which are the largest prior datasets available for robotics, since video offers observation-only experience without the action or reward annotations needed for RL methods. In this paper, we develop a system for leveraging large-scale human video datasets in robotic offline RL, based entirely on learning value functions via temporal-difference learning. We show that value learning on video datasets learns representations that are more conducive to downstream robotic offline RL than other approaches for learning from video data. Our system, called V-PTR, combines the benefits of pre-training on video data with robotic offline RL approaches that train on diverse robot data, resulting in value functions and policies for manipulation tasks that perform better, act robustly, and generalize broadly. On several manipulation tasks on a real WidowX robot and in simulated settings, our framework produces policies that greatly improve over other prior methods. Our video and additional details can be found at https://dibyaghosh.com/vptr/.**

I. Introduction

Developing methods capable of acquiring robotic skills that generalize widely to new scenarios is an important problem in robotic learning. In other areas of machine learning, broad generalization has been fueled primarily by pre-training on large datasets with a diversity of behavior. It seems compelling that the same formula may be applied to robotic learning, but in practice, even our largest robotic datasets contain data for relatively few tasks, and from a limited number of scenarios. In principle, robotic reinforcement learning (RL) should be able to learn from more general sources of data like human video, which are far more abundant and capture a broader set of skills, situations, and interactions. However, these datasets are difficult to incorporate into RL methods that exist today, since Internet-scale video data does not come with action or reward annotations present in typical robot data.

Existing works [23, 24, 38] include video data in the robot learning pipeline by performing self-supervised visual representation learning on video data [12], followed by downstream policy learning via behavioral cloning using the learned representations. While such an approach can extract visual features from video, it is limited in its ability to extract a deeper "functional" understanding of the world from video data: in principle, despite differences in embodiment, human videos can still be used to understand intents and affordances that can be executed in the real world, the dynamics, and the eventual outcomes that can be attained by acting.

Motivated by the above desiderata for video pre-training, in this work, we aim to develop an approach that pre-trains on Internet-scale human video to produce representations for downstream offline RL. Our main contribution is a system, which we call **V**ideo **P**re-**T**raining for **R**obots (V-PTR), that fits value functions to model long-term outcomes achieved when solving tasks on action-free video data.

Concretely, V-PTR pre-trains on human videos by learning an intent-conditioned value function [10] via temporal-difference learning (TD-learning). This approach eschews self-supervised representation learning objectives utilized in prior works [19, 23, 24] in favor of a TD value learning objective, just as how downstream offline RL agents will fine-tune task-specific value functions. Next, we fine-tune on a multi-task robotic dataset, which is annotated with actions, tasks, and rewards, using value-based offline RL [18]. Downstream, when a target task is specified, V-PTR fine-tunes the multi-task policy on this task. Each phase of our system gradually incorporates the knowledge of "what future outcomes can be achieved" (video pre-training), "what robot actions lead to these outcomes" (robot pre-training), and "how can the desired task be solved" (fine-tuning).

Our experiments on several robotic manipulation tasks on a real WidowX robot show that by pre-training on human video data (Ego4D [11]) and multi-task robot data (Bridge data [9]), V-PTR endows downstream offline RL methods with significantly improved zero-shot generalization and robustness to different target objects, distractors, and other variations in the workspace compared to prior methods that learn from videos, significantly outperforming prior methods including VIP [19]. To our knowledge, our work presents the first large-scale demonstration showing that TD-learning is an effective approach to pre-train from video for robotic RL.

II. Related Work

A number of prior approaches learn representations from video by applying image-level representation objectives on individual frames in the video or by modeling temporal dependencies along the frames in a given trajectory. The former includes objectives like reconstruction [14, 21, 27, 37] or contrastive learning on images [30]. While these objectives are widely used in computer vision, resulting representations do not capture any information about environment dynamics. The latter approaches model long-term dynamics from video by predicting the next frame [28], learning value functions [10, 19], or running time-contrastive learning [22, 29].

979-8-3503-8457-4/24/$31.00 ©2024 IEEE

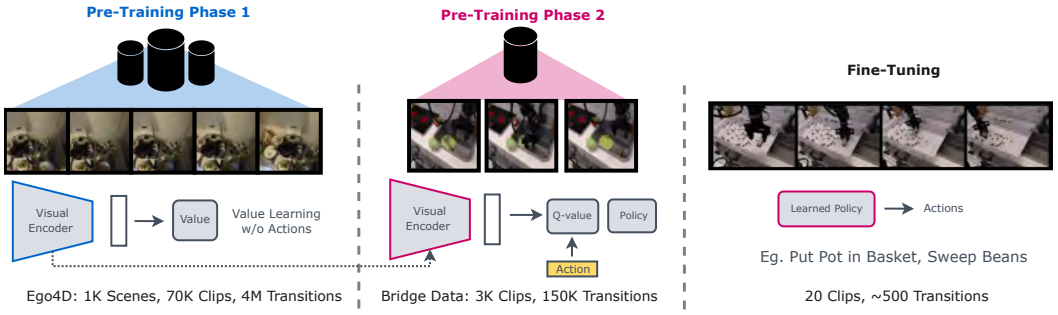

Fig. 1: **Video pre-training for robots (V-PTR)** involves pre-training representations on large-scale video datasets such as Ego4D, then pre-training the representation on multi-task robot datasets such as Bridge data, and finally fine-tuning on the downstream robot task.

In the context of robot learning, recent works learn representations from internet video datasets like Ego4D [11], using masked auto-encoders [14, 23], language-video alignment [14], or time-contrastive learning [19, 23], and train downstream policies on frozen features using behavioral cloning. In our experiments, we compare to several of these methods, and find that V-PTR attains a higher performance on real-world tasks, especially when evaluated with high initial state variability and in the presence of distractors. Specifically for egocentric human video, Bahl et al. [3] predicts wrist trajectories, and uses them to guide intermediate robot waypoints. This approach is orthogonal to our goal of learning state representations; in principle, our approach may be combined with this waypoint guidance.

The most closely related work is value-implicit pre-training (VIP) [19], which pre-trains a value function using time-contrastive prediction for downstream reward shaping. Both learn value functions during pre-training, albeit with entirely different algorithms (contrastive learning vs. TD learning), for different policies (dataset policy vs. intent-conditioned policy), exhibiting different generalization properties [4, 8]. Furthermore, the system desiderata differ for VIP and V-PTR: VIP focuses on learning good visual reward functions for weighted behavioral cloning, while we seek good value function initializations for downstream offline RL. In our experiments, we find that when both pre-training approaches are evaluated on the quality of downstream offline RL, those initialized with V-PTR improve over VIP in terms of generalization and robustness. While obtaining reward functions is not the focus of this paper, our experiments show that V-PTR outperforms VIP with reward shaping.

Finally, a different class of methods attempt to modify the downstream RL algorithm to train on video and robot data together in lieu of a video pre-training stage. For instance, [5, 7, 25, 33] train an inverse dynamics model to label video transitions with action pseudo-labels to use alongside the robotic experience; [31, 32, 34] use inverse RL to imitate the state-distribution present in the video; [2] translates human video frames to equivalent robot frames. These methods succeed only when a small domain gap exists between video data and robot data, so that observations from video data can plausibly be interpreted as robot data. This condition fails in our setup, as we utilize Ego4D [11] and the Bridge [9] datasets, where observations in these datasets differ significantly from each other, including a major difference in viewpoint (e.g., egocentric view in video data [11] & shoulder view in robot data [9]). To our knowledge, no method of this type has been performant in our setting.

III. Problem Statement and Background

We aim to use Internet-scale video data to boost the robustness and generalization of robotic offline RL agents.

Formal problem statement. We assume access to two pre-training datasets: an Internet-scale video dataset $\mathcal{D}_{\text{video}}$ (e.g., the Ego4D dataset [11]) and a target dataset, $\mathcal{D}_{\text{target}}$ of a limited number of demonstrations for a given target task on the robot. Additionally we are also provided a dataset of multi-task robot behaviors, $\mathcal{D}_{\text{robot}}$, which may not contain any data relevant to the target task. The video dataset $\mathcal{D}_{\text{video}}$ consists of sequences of frames (i.e., observations in the MDP), with no action or rewards. Denoting a frame as $\mathbf{s}_{i,j}$, we define $\mathcal{D}_{\text{video}} := \{(\mathbf{s}_{i,0}, \mathbf{s}_{i,1}, \cdots)\}_{i=1}^{n_{\text{video}}}$. The target dataset, $\mathcal{D}_{\text{target}}$, comprises of a few demonstrations of the target task on the robot $\mathcal{D}_{\text{target}} := \{(\mathbf{s}_{i,0}, \mathbf{a}_{i,0}, r_{i,0}, \mathbf{s}_{i,1}, \cdots)\}_{i=1}^{n_{\text{target}}}$, where the reward, $r_{i,j}$ is annotated to be +1 only on the final three timesteps of the demonstration (following [18]). The multi-task robot dataset $\mathcal{D}_{\text{robot}}$ is organized identically to the target robot dataset, but with an additional task annotation on each trajectory t_i, which is specified either as a one-hot identifier or by natural language. Our goal is train policy π which maximizes the γ-discounted cumulative reward, $\mathbb{E}_{\mathbf{s}_0 \sim \rho_0, \mathbf{a}_{0:\infty}, \mathbf{s}_{1:\infty} \sim \pi}[\sum_{t=0}^{\infty} \gamma^t r(\mathbf{s}_t, \mathbf{a}_t)]$, starting from a more diverse set of initial states indicated by the distribution ρ_0 than what was observed in the target dataset (e.g., more variation in distractor objects).

Background. Our system utilizes a generalized formulation of goal-conditioned RL and temporal-difference learning for pre-training value functions. In a nutshell, the goal-conditioned RL problem trains the agent to achieve arbitrary goal frames $\mathbf{g}$, where rewards are specified by the sparse signal of $\mathbb{I}(\mathbf{s} = \mathbf{g})$ when the frame is identical to the goal frame. Although the reward signal is sparse, goals and rewards can be defined by hindsight relabeling [1]. To learn a policy for the downstream task, we use value-based offline RL methods, which optimize π against a learned Q-function $Q^\pi(\mathbf{s}, \mathbf{a})$. The Q-value function measures the expected long-term reward attained when executing action $\mathbf{a}$ at state $\mathbf{s}$, then following policy π thereafter, and satisfies the Bellman equation $Q^\pi(\mathbf{s}, \mathbf{a}) = r(\mathbf{s}, \mathbf{a}) + \gamma \mathbb{E}_{\mathbf{s}', \mathbf{a}'}[Q^\pi(\mathbf{s}', \mathbf{a}')]$.

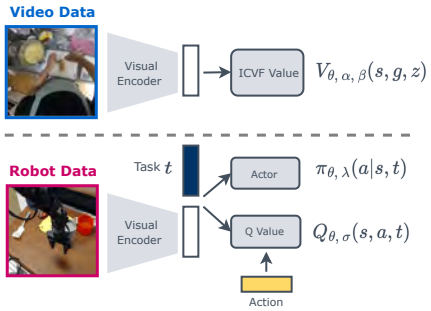

Fig. 2: **Network architecture**. V-PTR first pre-trains image representations by training a general value function from video and then refines this representation via multi-task pre-training on robot data.

IV. Video Pre-Training for Robotic Offline RL

Even though video data contains rich functional and dynamic information useful for downstream skill learning, this data cannot directly be integrated into robotic offline RL pipelines. In this section, we develop V-PTR, our system that pre-trains general value functions on Internet-scale video data, and fine-tunes value functions for the desired downstream robotic task using value-based offline RL.

System overview. Our system, V-PTR, pre-trains in two phases: first on video data, and then on multi-task robot data. In the first phase, we train an intent-conditioned value function [10] on action-less video data using a value-learning objective to model the outcomes associated with solving the parametric family of goal-achieving tasks. Next, we refine this representation with multi-task robot data with actions and rewards, by training a state-action Q-function on this representation using offline RL. We expect the video and multi-task robot data to capture dynamic features in the environment, with the robot data bridging the domain gap between human video and the robot, and aligning the learned representation with the agent's action space. Downstream, to adapt the system to a new target task, our system fine-tunes the Q-function and the policy on the target dataset.

A. Phase 1: Video Pre-Training via TD-Learning

Since the goal of video pre-training is to improve the performance of downstream value-based offline RL, we turn to learning value functions on the video data as a natural pre-training procedure. We choose to pre-train by learning an intent-conditioned value function (ICVF), a recently-proposed general value function that can be efficiently trained on passive data without action labels [10]. An ICVF, annotated $V(\mathbf{s}_{\text{video}}, \mathbf{g}_{\text{video}}, \mathbf{z})$ computes the value obtained towards reaching a goal $\mathbf{g}_{\text{video}}$, assuming the policy *intended* to reach a different intended goal $\mathbf{z}$, and is formally defined as

$$V(\mathbf{s}_{\text{video}}, \mathbf{g}_{\text{video}}, \mathbf{z}) = \mathbb{E}_{a_t \sim \pi^*_{\mathbf{z}}(\cdot|\mathbf{s}_t)}\Big[\sum_t \gamma^t \mathbb{I}\left(\mathbf{s}_{\text{video}} = \mathbf{g}_{\text{video}}\right)\Big].$$

As with a standard value function, the ICVF can be learned by temporal-difference (TD) learning on its corresponding Bellman equation, using a target network to bootstrap the predictions of the learned value function. We follow the goal-sampling strategy from ICVF: after sampling a start frame $\mathbf{s}$ from the video dataset, we choose the goal $\mathbf{g}$ to be either the next observation, a future observation in the video, or a random observation from a different video. The intent $\mathbf{z}$ is also an observation appearing in the video, and is chosen in a similar fashion as the goal state. Additionally, following Ghosh et al. [10], with some probability we set $\mathbf{z} = \mathbf{g}$.

We follow Ghosh et al. [10] and parameterize our estimated value function as

$$V(\mathbf{s}_{\text{video}}, \mathbf{g}_{\text{video}}, \mathbf{z}) := \phi(\mathbf{s}_{\text{video}})^\top T(\mathbf{z}) \psi(\mathbf{g}_{\text{video}}),$$

where ϕ_θ and ψ_α denote models that transform the observation and the goal observation respectively into low-dimensional representations, and T_β, a learned mapping aligning the two representations. At convergence, the ICVF provides a measure of temporal spatiality, and the learned representation $\phi_\theta(\mathbf{s})$ offers a useful feature basis for downstream value functions.

B. Phase 2: Multi-Task Robot Pre-Training via Offline RL

In the next phase, we refine the learned representation on a multi-task robot dataset, $\mathcal{D}_{\text{robot}}$, to narrow the domain gap between robot image observations and human video, and to provide information about the target robot embodiment (i.e., the actions affordable by the robot). To avoid evaluation contamination, the tasks and workspaces in this robot dataset are explicitly disjoint from the target tasks used downstream.

V-PTR uses multi-task robot data to pre-train a Q-function and a policy using multi-task conservative Q-learning (CQL) [18], initializing the parameters of *both* the Q-function and the policy using the backbone learned during video pre-training in Phase 1. Concretely, the Q-function and the policy are conditioned on the pre-trained representation of the robot observation $\phi_\theta(\mathbf{s}_{\text{robot}})$ alongside a task vector t (either a one-hot task identifier or a language embedding from a sentence transformer). At the onset of this phase, we initialize the representation encoder ϕ_θ to the encoder $\phi_{\theta^*_{\text{video}}}$ obtained at the end of phase 1. The value function is trained to satisfy the Bellman equation,

$$\min_\theta \ \alpha \cdot \mathcal{L}_{\text{CQL}}(\theta) + \mathbb{E}\left[\left(Q_\theta(\mathbf{s}, \mathbf{a}; t) - r - \gamma \bar{Q}(\mathbf{s}', \mathbf{a}', t)\right)^2\right],$$

with target Q-network $\bar{Q}$, and CQL regularizer $\mathcal{L}_{\text{CQL}}(\theta)$ [16], and the policy trained to maximize value,

$$\max_\lambda \quad \mathbb{E}_{\mathcal{D}_{\text{robot}}}\left[\mathbb{E}_{\mathbf{a}\sim\pi_\lambda(\cdot|\mathbf{s};t)}[Q(\mathbf{s}, \mathbf{a}; t)]\right] + \beta \mathcal{H}(\pi_\lambda).$$

After pre-training, we have a multi-task Q-function and policy that can be fine-tuned to the desired downstream task using a small target dataset.

C. Phase 3: Fine-Tuning to a Target Task

Finally, we fine-tune the value function and policy from the pre-training stage to the target task by running CQL [16] on the target dataset $\mathcal{D}_{\text{target}}$. We follow Kumar et al. [18] and treat the target data simply as a new task; fine-tuning involves assigning a new task identifier to the target data (either a new one-hot vector or a new language command), and continuing to run multi-task offline CQL on the robot pre-training and target tasks jointly. To address any overfitting induced by the

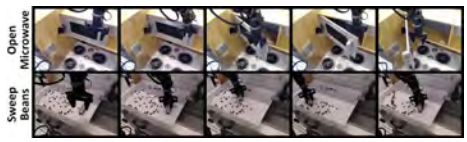

Fig. 3: **Examples of setup and successful rollouts for complex tasks**. We utilize the robot setup from the Bridge dataset [9] for our tasks. **Top:** Two-phase open microwave; **Bottom:** Sweep beans into pile with tool.

small size of $\mathcal{D}_{\text{target}}$, we perform stratified sampling between the multi-task robot data $\mathcal{D}_{\text{robot}}$ and the target data $\mathcal{D}_{\text{target}}$: $1-\tau$ proportion of the training batch comes from $\mathcal{D}_{\text{robot}}$ and τ from $\mathcal{D}_{\text{target}}$, where τ is small (we use $\tau = 0.1$).

D. Implementation Details

Video pre-training: We use video frames at a 224×224 resolution. The three components parameterizing the ICVF are implemented as separate 3-layer MLP heads on a shared visual backbone encoder. We use a Resnetv2-50 [13] as our backbone since smaller convolutional networks led to worse pre-training performance. We replace all batch normalization layers with group normalization [36], since prior works [6, 18] often found batch normalization to be unstable. To avoid overfitting to spurious correlations between consecutive frames in a video clip, we use image augmentation (random crops and color jitter) [15], weight decay, and dropout. We train the model for 2×10^6 gradient steps with a batch size of 64, using Adam, and learning rate 10^{-4} cosine annealed through training. All remaining designs we take from the open-source code [10].

Multi-task robot pre-training: We fine-tune on the multi-task robot dataset primarily following design decisions from Kumar et al. [18]. The CQL policy takes the RGB image, proprioceptive state, and task identifier as input. The RGB image is passed through a visual encoder, then concatenated with the remaining information, and passed through a 2-layer MLP. We additionally concatenate task and action information into each hidden layer of the MLP. The CQL value function is parameterized similarly, but it also takes the action vector as input. Encoder parameters for both the value and policy are initialized from the video pre-trained ICVF, but there is no further weight tying between the networks. We train CQL for 2×10^5 gradient steps with batch size of 64, and Adam with a constant learning rate of 10^{-4}.

V. Experimental Results

The goal of our experiments is to validate the effectiveness of V-PTR in boosting the generalization and robustness of robotic offline RL. We evaluate V-PTR in several scenarios requiring generalization to new scenes, compare to other approaches for incorporating video data, and perform additional diagnostic experiments to understand how value pre-training can provide useful representations for downstream robotic RL. Our settings include several challenging robotic manipulation tasks that require zero-shot generalization to new objects and distractors. A video of our evaluations and diagnostic experiments can be found on our **project website**.

Real-world setup. We conduct our experiments on a WidowX robot platform. We perform video pre-training on Ego4D [11], an egocentric video dataset consisting of 4M transitions of humans attempting diverse tasks in the real world, using the same pre-processed subset from prior works [19, 23]. Then, for multi-task robot pre-training, we utilize the subset of the Bridge dataset [9, 35] used by prior work [18], a dataset with 150K transitions of various manipulation task demonstrations on a WidowX robot in toy kitchens. Downstream, we fine-tune on several tasks on a WidowX robot, in a *previously unseen* toy-kitchen workspace. For each target task, we collect 10 demonstrations using teleoperation, with a range of distractor objects and object variability. Solving these tasks requires skills such as picking and placing a variety of objects, using tools to accomplish tasks (e.g., sweeping), and two-phase door opening (Figures 3 and 4).

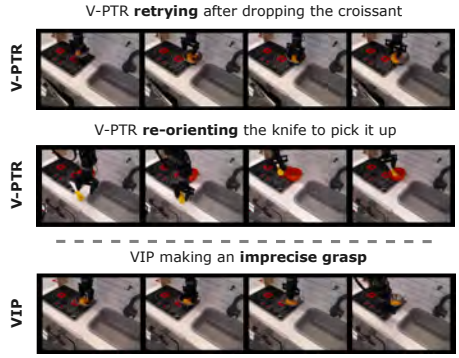

Fig. 4: **Visualizing qualitative performance of V-PTR and VIP**. Here we show rollouts for V-PTR (top) and VIP (bottom) on the real robot manipulation tasks. V-PTR carefully executes the task by orienting the gripper to match the object and retrying on failure whereas VIP grasp objects without this re-orientation, leading to failure.

Comparisons to prior methods. We compare V-PTR to approaches that do not utilize video data (PTR [18], BC [9]), as well as other methods for video pre-training (R3M [23], MVP [24, 38], and VIP [19]). Following the protocols in these prior works, we fine-tune the R3M and MVP representations with imitation learning on multi-task and target robot data (phases 2 and 3), and evaluate the VIP representation both with reward-weighted imitation learning and with downstream offline RL via CQL, to provide an apples-to-apples comparison with V-PTR. We evaluate three versions of VIP: **(i)** "$\text{VIP}_{\text{frozen}}$", which freezes the pre-trained representation learned from video data during fine-tuning, **(ii)** "VIP", which continues to fine-tune on robot data, and **(iii)** "$\text{VIP}_{\text{reward}}$", which not only utilizes the pre-trained representation for initializing the policy but also uses distances between representations of current and future frames as a reward shaping for downstream offline RL via reward-weighted imitation, following [19]. When using language task specification, we compare to language conditioned variants of BC and PTR.

A. Real-World Results

We evaluate V-PTR and comparisons in three testing scenarios. In **Scenario 1**, we evaluate the performance of the policy, varying the robot's initial pose and the position of objects in scene. In **Scenario 2**, we evaluate the policy in the

				Video pre-training				No videos	No robot data
		Task	V-PTR (Ours)	R3M+BC	MVP+BC	VIP+CQL	VIP_{frozen}+CQL	PTR	V-PTR w/o phase 2
Scenario 1		Croissant from bowl	7 / 12	0 / 12	4 / 12	2 / 12	0 / 12	3 / 12	5 / 12
		Sweet potato on plate	6 / 12	0 / 12	1 / 12	0 / 12	0 / 12	1 / 12	1 / 12
		Knife in pot	6 / 12	0 / 12	0 / 12	0 / 12	0 / 12	0 / 12	0 / 12
		Cucumber in pot	5 / 12	0 / 12	1 / 12	0 / 12	0 / 12	1 / 12	1 / 12
		Total	24 / 48	0 / 48	6 / 48	2 / 48	0 / 48	5 / 48	7 / 48
Scenario 2	with distractor objects	Croissant from bowl	8 / 12	0 / 12	3 / 12	2 / 12	0 / 12	0 / 12	3 / 12
		Sweet potato on plate	4 / 12	0 / 12	2 / 12	0 / 12	0 / 12	1 / 12	2 / 12
		Knife in pot	4 / 12	0 / 12	0 / 12	1 / 12	0 / 12	0 / 12	0 / 12
		Cucumber in pot	4 / 12	0 / 12	0 / 12	1 / 12	0 / 12	0 / 12	1 / 12
		Total	20 / 48	0 / 48	5 / 48	4 / 48	0 / 48	1 / 48	6 / 48
Scenario 3	novel target objects	Carrot	2 / 3	0 / 3	0 / 3	1 / 3	0 / 3	0 / 3	2 / 3
		Cucumber	1 / 3	0 / 3	0 / 3	1 / 3	0 / 3	0 / 3	1 / 3
		Ice-cream	0 / 3	0 / 3	0 / 3	1 / 3	1 / 3	1 / 3	0 / 3
		Total	3 / 9	0 / 9	0 / 9	3 / 9	1 / 9	1 / 9	3 / 9

TABLE I: **Task success rates of V-PTR and prior methods** on several manipulation tasks over 12 trials (best-performing method indicated in red). Note that V-PTR outperforms all prior methods, including those approaches that do not fine-tune the learned representation, use imitation learning for downstream control, or do not incorporate video data.

				No CQL
Task	V-PTR	VIP [19]+CQL	PTR [18]	VIP_{reward} [19]
Open Microwave	5 / 12	2 / 12	0 / 12	0 / 12
Sweep Beans	6 / 12	5 / 12	2 / 12	2 / 12

TABLE II: **Performance of V-PTR, VIP, and PTR on more complex tasks**. V-PTR outperforms PTR as well as VIP variants that use downstream CQL or BC weighted by the reward shaping from Ma et al. [19].

Fig. 5: Examples of areas swept by VIP [19] (**top**) and V-PTR (**bottom**) methods. V-PTR sweeps a much larger area (blue), and consistently begins a second sweep, whereas VIP [19] is too slow to sweep a second time.

presence of novel distractor objects in the scene. We note that some of the target data does contain distractor objects, but we use a different set of distractor objects during evaluation than those seen in training. Finally, in **Scenario 3**, we test the ability of the learned policy to manipulate novel target objects that were never seen during training.

We evaluate different methods on four tasks in Table I. Observe that V-PTR outperforms all other prior approaches, and in some tasks (e.g., "place knife in pot") is the only method that produces any successful trials. We observed that all of these methods do learn behavior that *attempt* to solve the task, for example by moving toward relevant objects in the scene, but do not eventually succeed due to either imprecise localization of target objects or a prematurely executed grasp attempt. On the other hand, we found that in several scenarios V-PTR is able to re-orient the gripper before executing a grasp (e.g., Fig. 4).

Next, we evaluate V-PTR with distractor objects in Table I (Scenario 2). Adding distractors reduces performance for every method (as one may expect), but V-PTR still exhibits the smallest degradation compared to the next best approach. To study generalization to novel target objects (Scenario 3), we also consider a "take [object] from bowl" task, and replace the target object with unseen target objects at evaluation. Performance is low for all comparisons in Table I, but V-PTR succeeds 50% of the time with new objects.

Comparisons to VIP on more complex tasks. We compare V-PTR in more detail to VIP (which also uses similar video data) on manipulation tasks that require learning more complex skills: "open microwave" and "sweep beans" in Table II. Specifically, we compare to different variants of VIP discussed above (VIP+CQL; VIP_{reward}) and find that V-PTR outperforms these variants. Qualitatively in Figure 5, we observe that on the "sweep beans" task, V-PTR sweeps a larger area than the VIP policy, which is too slow to execute the sweep motion a second time. This corroborates our analysis (Figure 6 and 7) that value functions trained on the V-PTR representation tend to have lower error than those utilizing VIP representations.

Language-based task specification. We next study how V-PTR works when the robot pre-training data is labeled with natural language descriptions (a more general format) instead of task identifiers. To handle language, we first encode the task description into an embedding vector using the pre-trained language encoder from GRIF [20], which has been shown to be effective at learning BC policies. The Q-function and the policy then utilize these embedding vectors in lieu of the one-hot task identifiers, processing them identically as before. In Table III, we compare V-PTR to imitation learning and PTR with language embeddings, and find that V-PTR improves over both of these methods by around 50%, indicating that V-PTR can leverage downstream language.

B. Visualizations and Diagnostic Experiments

We now analyze V-PTR more carefully by visualizing the learned features from video-pretraining, probing the generalization of the system, and assessing the quality of value estimation in downstream offline RL.

Task	V-PTR (language)	BC	PTR	No language V-PTR (one-hot)
Croissant	7 / 12	3 / 12	5 / 12	7 / 12
Sweet potato	6 / 12	2 / 12	6 / 12	6 / 12
Knife in Pan	5 / 12	0 / 12	3 / 12	6 / 12
Cucumber in Pot	9 / 12	0 / 12	3 / 12	5 / 12
Open Microwave	9 / 12	1 / 12	0 / 12	5 / 12
Sweep Beans	8 / 12	2 / 12	4 / 12	6 / 12
Total	**44 / 72**	8 / 72	21 / 72	35 / 72

TABLE III: **Performance of language-conditioned V-PTR compared to language-conditioned BC and PTR** for the six manipulation tasks that we study. Note that V-PTR outperforms both of these prior methods, indicating that V-PTR can favorably learn in settings with alternate task specification.

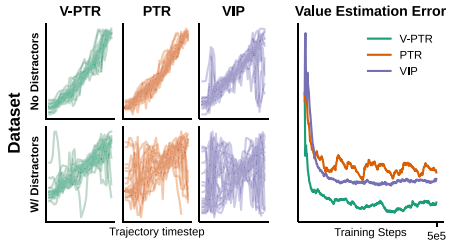

Fig. 6: **Visualizing the learned values $V(\mathbf{s}_t)$ w.r.t. time-step t** on rollouts from training data (**top**), held-out data (**middle**), and rollouts with distractor objects (**bottom**) obtained after multi-task pre-training in phase 2 ($V(\mathbf{s}_t)$ is computed using the average of the multi-task Q-value under actions sampled from the learned policy). Note that values trained by PTR and VIP tend to be highly non-smooth, especially on held-out rollouts with novel distractors, whereas V-PTR produces smooth value functions.

Video pre-training via V-PTR improves target value estimation. We visualize the learned value function on frames from different rollouts in Figure 6 (left), where the true value function should monotonically increase from initial states ($t = 0$) to the final states ($t = 30$) of a successful rollout. We visualize the downstream value $V(\mathbf{s}_t)$ for two kinds of rollouts: **(i)** rollouts from training data and **(ii)** rollouts with distractor objects. All three visualized methods – PTR, VIP, and V-PTR– are able to learn smooth value functions on the data without distractor objects. However, in the presence of distractors, the value functions trained by PTR and VIP are non-smooth and non-monotonic (bottom), while V-PTR retains higher-quality value functions.

Next, we more precisely measure the ability of V-PTR to fit downstream value functions. We train a SARSA value function (for which we may compute a closed-form optimal solution) on top of frozen pre-trained representations from PTR (no video data), VIP [19], and V-PTR, and report the error between the predicted value estimate and the ground-truth value on a held-out dataset in Figure 6 (right). V-PTR attains smaller fitting error than PTR and VIP.

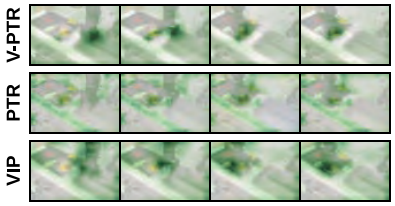

Fig. 7: **Grad-CAM visuals superimposed on frames from robot data**. Regions highlighted in green denote patches of the observation with the most significant influence on the learned policy. Without PTR and VIP, background areas in the image highly influence the output of the learned policy. In contrast, initializing the policy with the representation obtained from V-PTR focuses more on gripper and object positions.

What kind of visual cues do representations trained by V-PTR capture? We probe which parts of the image influence the output of the learned policy for V-PTR and other baselines, by utilizing Grad-CAM [26] to mark patches of the frame that influence the output of the learned policy in green in Figure 7. We observe that V-PTR policies discard the scene background and focus on cues relevant for robot control (e.g., object, gripper positions), while PTR and VIP place higher focuses on the scene background. This evidence provides an explanation for why V-PTR robustly attains better performance in our experiments.

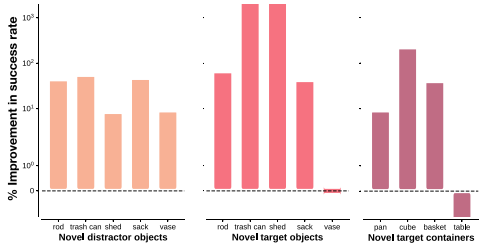

Fig. 8: **Simulation results**. Percentage improvement in success rates of V-PTR over PTR [18], that does not utilize any video data for pre-training. y-axis is in log scale. In all but one scenario, V-PTR improves performance.

Simulation results. We also evaluate V-PTR in a simulated pick-and-place task [17, 18] where generalization to new objects can be measured more precisely. The task is to place a target object into a container, in the presence of a distractor object. Like our real-world experiments, we pre-train on video data from Ego4D, and use simulated robot demonstrations for multi-task pre-training and task fine-tuning. We evaluate V-PTR under multiple conditions that require policies to behave robustly in the presence of distractor objects and generalize to novel target objects and workspaces. In Figure 8, we present our results in terms of the percentage improvement in success rates obtained by V-PTR over those of PTR [18] (with no video data). Consistently, across all but one scenario with a novel target container, V-PTR improves over PTR, demonstrating that value-based video pre-training on Ego4D boosts performance and robustness of robotic RL.

VI. Discussion and Conclusion

In this paper, we designed a robotic system, V-PTR, that uses value function pre-training on the large-scale Ego4D video dataset [11] and the robot Bridge dataset [9] to improve the performance of policies learned downstream. While V-PTR outperforms prior methods for learning from video data, we found that all the current methods remain sensitive to deviations in workspace height, camera angle, and robot configurations. There also exist many opportunities to scale, whether incorporating multi-robot datasets, larger human video datasets with language, or larger models. Nevertheless, our evaluations and diagnostic experiments indicate the promise of using RL-like value pre-training on video data for improving the general ability of robot learning algorithms.

References

[1] Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. *Advances in neural information processing systems*, 30, 2017. 2

[2] Shikhar Bahl, Abhi Gupta, and Deepak Pathak. Human-to-robot imitation in the wild. *ArXiv*, abs/2207.09450, 2022. URL https://api.semanticscholar.org/CorpusID:248941578. 2

[3] Shikhar Bahl, Russell Mendonca, Lili Chen, Unnat Jain, and Deepak Pathak. Affordances from human videos as a versatile representation for robotics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13778–13790, 2023. 2

[4] Leemon Baird. Residual Algorithms : Reinforcement Learning with Function Approximation. In *International Conference on Machine Learning (ICML)*, 1995. 2

[5] Bowen Baker, Ilge Akkaya, Peter Zhokhov, Joost Huizinga, Jie Tang, Adrien Ecoffet, Brandon Houghton, Raul Sampedro, and Jeff Clune. Video pretraining (vpt): Learning to act by watching unlabeled online videos. *ArXiv*, abs/2206.11795, 2022. 2

[6] Aditya Bhatt, Max Argus, Artemij Amiranashvili, and Thomas Brox. Crossnorm: Normalization for off-policy td reinforcement learning. *arXiv preprint arXiv:1902.05605*, 2019. 4

[7] Matthew Chang, Arjun Gupta, and Saurabh Gupta. Learning value functions from undirected state-only experience. *ArXiv*, abs/2204.12458, 2022. 2

[8] Christoph Dann, Gerhard Neumann, Jan Peters, et al. Policy evaluation with temporal differences: A survey and comparison. *Journal of Machine Learning Research*, 15:809–883, 2014. 2

[9] 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. 1, 2, 4, 6

[10] Dibya Ghosh, Chethan Bhateja, and Sergey Levine. Reinforcement learning from passive data via latent intentions. *arXiv preprint arXiv:2304.04782*, 2023. 1, 3, 4

[11] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18995–19012, 2022. 1, 2, 4, 6

[12] K He, X Chen, S Xie, Y Li, P Dollár, and RB Girshick. Masked autoencoders are scalable vision learners. arxiv. 2021 doi: 10.48550. *arXiv preprint arXiv.2111.06377*, 2021. 1

[13] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision*, 2016. 4

[14] Siddharth Karamcheti, Suraj Nair, Annie S Chen, Thomas Kollar, Chelsea Finn, Dorsa Sadigh, and Percy Liang. Language-driven representation learning for robotics. *arXiv preprint arXiv:2302.12766*, 2023. 1, 2

[15] Ilya Kostrikov, Denis Yarats, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. *arXiv preprint arXiv:2004.13649*, 2020. 4

[16] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *arXiv preprint arXiv:2006.04779*, 2020. 3

[17] Aviral Kumar, Anikait Singh, Stephen Tian, Chelsea Finn, and Sergey Levine. A workflow for offline model-free robotic reinforcement learning. In *5th Annual Conference on Robot Learning*, 2021. URL https://openreview.net/forum?id=fy4ZBWxYbIo. 6

[18] Aviral Kumar, Anikait Singh, Frederik Ebert, Yanlai Yang, Chelsea Finn, and Sergey Levine. Pre-training for robots: Offline rl enables learning new tasks from a handful of trials. *RSS 2023; arXiv:2210.05178*, 2023. 1, 2, 3, 4, 5, 6

[19] 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. *arXiv preprint arXiv:2210.00030*, 2022. 1, 2, 4, 5, 6

[20] Vivek Myers, Andre He, Kuan Fang, Homer Walke, Philippe Hansen-Estruch, Ching-An Cheng, Mihai Jalobeanu, Andrey Kolobov, Anca Dragan, and Sergey Levine. Goal representations for instruction following: A semi-supervised language interface to control. *arXiv preprint arXiv:2307.00117*, 2023. 5

[21] A. Nair, S. Bahl, A. Khazatsky, V. Pong, G. Berseth, and S. Levine. Contextual imagined goals for self-supervised robotic learning. In *Conference on Robot Learning (CoRL)*, 2019. 1

[22] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhi Gupta. R3m: A universal visual representation for robot manipulation. *ArXiv*, abs/2203.12601, 2022. 1

[23] 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. 1, 2, 4

[24] Ilija Radosavovic, Tete Xiao, Stephen James, Pieter Abbeel, Jitendra Malik, and Trevor Darrell. Real-world robot learning with masked visual pre-training. *arXiv preprint arXiv:2210.03109*, 2022. 1, 4

[25] Karl Schmeckpeper, Oleh Rybkin, Kostas Daniilidis, Sergey Levine, and Chelsea Finn. Reinforcement learning with videos: Combining offline observations with interaction. In *Conference on Robot Learning*, 2020. 2

[26] RR Selvaraju, M Cogswell, A Das, R Vedantam, D Parikh, and D Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. arxiv 2016. *arXiv preprint arXiv:1610.02391*. 6

[27] Younggyo Seo, Danijar Hafner, Hao Liu, Fangchen Liu, Stephen James, Kimin Lee, and P. Abbeel. Masked world models for visual control. *ArXiv*, abs/2206.14244, 2022. 1

[28] Younggyo Seo, Kimin Lee, Stephen James, and P. Abbeel. Reinforcement learning with action-free pre-training from videos. *ArXiv*, abs/2203.13880, 2022. 1

[29] Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, and Sergey Levine. Time-contrastive networks: Self-supervised learning from video. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1134–1141, 2017. 1

[30] A. Srinivas, Michael Laskin, and P. Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. In *International Conference on Machine Learning*, 2020. 1

[31] Bradly C. Stadie, P. Abbeel, and Ilya Sutskever. Third-person imitation learning. *ArXiv*, abs/1703.01703, 2017. 2

[32] Gerald Tesauro, Nicholas K Jong, Rajarshi Das, and Mohamed N Bennani. A hybrid reinforcement learning approach to autonomic resource allocation. In *2006 IEEE International Conference on Autonomic Computing*, pages 65–73. IEEE, 2006. 2

[33] Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation. *ArXiv*, abs/1805.01954, 2018. 2

[34] Faraz Torabi, Garrett Warnell, and Peter Stone. Generative adversarial imitation from observation. *ArXiv*, abs/1807.06158, 2018. 2

[35] Homer Walke, Kevin Black, Abraham Lee, Moo Jin Kim, Max Du, Chongyi Zheng, Tony Zhao, Philippe Hansen-Estruch, Quan Vuong, Andre He, et al. Bridgedata v2: A dataset for robot learning at scale. *arXiv preprint arXiv:2308.12952*, 2023. 4

[36] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018. 4

[37] Tete Xiao, Ilija Radosavovic, Trevor Darrell, and Jitendra Malik. Masked visual pre-training for motor control. *ArXiv*, abs/2203.06173, 2022. 1

[38] Tete Xiao, Ilija Radosavovic, Trevor Darrell, and Jitendra Malik. Masked visual pre-training for motor control. *arXiv preprint arXiv:2203.06173*, 2022. 1, 4