Video-Enhanced Offline Reinforcement Learning: A Model-Based Approach

Minting Pan [1] Yitao Zheng [1] Jiajian Li [1] Yunbo Wang [1] Xiaokang Yang [1]

Abstract

Offline reinforcement learning (RL) enables policy optimization using static datasets, avoiding the risks and costs of extensive real-world exploration. However, it struggles with suboptimal offline behaviors and inaccurate value estimation due to the lack of environmental interaction. We present **Video-Enhanced Offline RL (VeoRL)**, a model-based method that constructs an interactive world model from diverse, unlabeled video data readily available online. Leveraging model-based behavior guidance, our approach transfers commonsense knowledge of control policy and physical dynamics from natural videos to the RL agent within the target domain. VeoRL achieves substantial performance gains (over 100% in some cases) across visual control tasks in robotic manipulation, autonomous driving, and open-world video games. Project page: https://panmt.github.io/VeoRL.github.io.

1. Introduction

When humans tackle long-term control tasks they are unfamiliar with, they typically start by watching instructional videos on platforms like YouTube to develop an abstract understanding of the task, before transitioning to hands-on practice in a real environment to refine their skills. Can machines replicate this learning process? In this work, we introduce video-enhanced offline reinforcement learning (VeoRL) to explore this problem.

Offline reinforcement learning refers to the scenario where an RL system is restricted to learning from a limited, static dataset. This approach offers several benefits, as it allows agents to learn from existing data, reducing the need for costly and potentially unsafe real-world engagement (Fujimoto et al., 2019; Kumar et al., 2019; 2020; Levine et al., 2020; Kostrikov et al., 2022). However, a key challenge in offline RL lies in the inherent incompleteness of fixed datasets, which can result in distributional shifts from the target environment. This issue, compounded by the well-documented overestimation bias in value approximations, creates significant difficulties for current offline RL methods (Yu et al., 2022; Zang et al., 2023; Lu et al., 2023).

The core insight of VeoRL is to leverage unlabeled Internet videos to broaden the agent's understanding of human behaviors and the real physical world, going beyond the limited scope of fixed, predefined datasets. Notably, the vast volume of online video data possesses several key advantages: (i) It can be acquired at a significantly lower cost compared to data collected through real interactions with the target environments. (ii) It implicitly contains demonstration policies that, while potentially suboptimal, can still be beneficial for current visual control tasks.

Nonetheless, the online videos typically lack action labels or reward function annotations and may present remarkable data distributional shifts compared to the target RL domain. These properties introduce unique challenges when using the auxiliary videos for policy optimization. Existing RL pretraining methods generally operate within the raw action space (*e.g.,* through behavior cloning) or perform action-free model warmup (*e.g.,* R3M (Nair et al., 2022) and APV (Seo et al., 2022)), making them ill-suited for transferring underlying behaviors from unlabeled videos.

To address these challenges, we begin by constructing a discrete latent action space to recover the underlying, unobservable control policies from raw video data (see Figure 1a). Specifically, we model the temporal events in the video as a Markov decision process with a continuous state space and a discrete action space. To optimize a finite set of "*behavior abstractions*", we apply vector quantization techniques and use the learned behavior vectors to drive the training of a video prediction model. Empirical results demonstrate that the latent behaviors learned through this approach exhibit strong semantic alignment with the actual actions observed in the target environment.

VeoRL is essentially a model-based RL algorithm, where the agent interacts with a world model that generates state transitions and reward feedback. We train a world model comprising two state transition branches: One branch pre-

[1] MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University. Correspondence to: Yunbo Wang <yunbow@sjtu.edu.cn>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025. Copyright 2025 by the author(s).

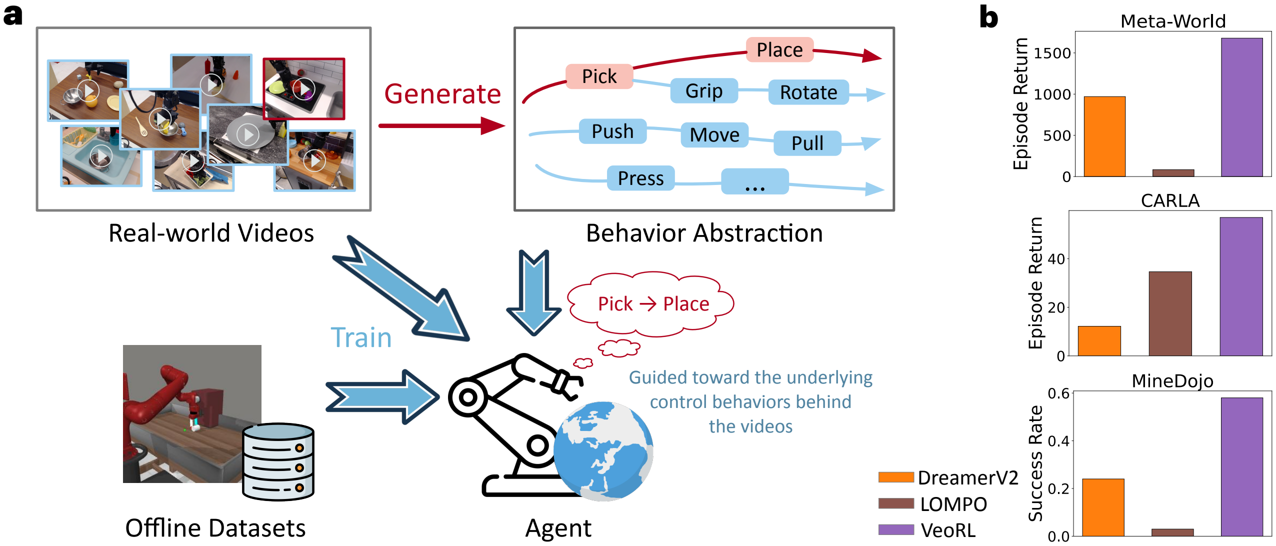

Figure 1. **Overview of the training setup of VeoRL that leads to improved offline RL performance. a**, We extract latent behavior abstractions from task-agnostic, unannotated natural video data to enrich the world model's commonsense understanding of the physical world. By interacting with this world model, the agent performs policy optimization guided explicitly by the latent policies learned from the natural videos. **b**, Overall performance of VeoRL. It demonstrates significant improvements over existing offline RL methods across different visual control benchmarks, including Meta-World robotic manipulation, CARLA autonomous driving, and Minecraft open-world video games. We present the average performance on all tasks on each benchmark.

dicts future state evolution based on the agent's real actions (*trunk net*), while the other branch predicts longer-term environmental feedback derived from latent behaviors (*plan net*). We further introduce an intrinsic reward that encourages the state-action rollouts along the trunk net to progressively align with the long-term future state predicted by the plan net. This approach connects the target policy with the behavior abstractions learned from auxiliary videos, while mitigating the overestimation bias that arises from solely relying on environmental rewards to train the value network.

We evaluate VeoRL on a diverse set of visual control benchmarks. For the Meta-World robotic manipulation tasks, we use the BridgeData-V2 dataset (Walke et al., 2023) as the source of auxiliary real-world videos. For the CARLA autonomous driving tasks, we employ the NuScenes dataset (Caesar et al., 2019), a collection of 1,000 diverse real-world driving scenes, as the auxiliary source domain. For the MineDojo open-world gaming environment, where the agent must navigate a vast state space, we use online Minecraft videos created by human players to provide unlabeled video demonstrations. As summarized in Figure 1b, VeoRL achieves substantial performance improvements over existing RL approaches.

2. Related Work

Model-based visual RL. For visual control tasks, agents learn control policies from high-dimensional visual inputs. By explicitly modeling state transitions and predicting behavioral outcomes, model-based RL approaches (Ha & Schmidhuber, 2018; Hafner et al., 2019; 2020) typically offer better sample efficiency than model-free methods (Yarats et al., 2021b; Kostrikov et al., 2021; Laskin et al., 2020). World Models (Ha & Schmidhuber, 2018) were introduced to first learn compressed latent representations of the environment in a self-supervised manner, and then use these latent states to train the agent. Dreamer (Hafner et al., 2020) and DreamerV2 (Hafner et al., 2021) optimize behavior by predicting and then acting upon expected values over the latent states generated by the Recurrent State-Space Model. Recently, research on goal-conditioned RL using hierarchical world model architectures (Veeriah et al., 2021; Rao et al., 2022; Hafner et al., 2022; Gumbsch et al., 2023; Park et al., 2024) has shown how decomposing tasks into subgoals can facilitate efficient decision-making in complex environments. However, none of these methods address the challenges of offline RL or facilitate unsupervised policy transfer from enriched out-of-domain video data.

Offline RL. Offline RL offers a cost-effective alternative by reducing interactions with online environments. Previous research has primarily focused on addressing the challenge of value overestimation, falling into two broad categories: model-free methods (Cen et al., 2024; Wu et al., 2019; Kumar et al., 2020; Osa & Harada, 2024; Li et al., 2023; Fujimoto & Gu, 2021; Hong et al., 2024) and model-based approaches (Chen et al., 2023; Yu et al., 2021; Hatch et al., 2023; Mazoure et al., 2023; Hatch et al., 2022; Sun et al., 2023; Rigter et al., 2022; Yang et al., 2024b). For example, CQL (Kumar et al., 2020) introduces a model-free offline RL method that mitigates the value overestimation bias by learning a conservative state-action value function. This method can be seamlessly integrated with existing visual RL approaches. In the context of offline visual RL, an additional challenge is effectively handling high-dimensional visual inputs while preventing overfitting during the representation learning of latent states (Shah et al., 2022; Tian et al., 2020; Yadav et al., 2023; Nair et al., 2022; Wang et al., 2024). Specifically, LOMPO (Rafailov et al., 2021) presents a model-based offline visual RL approach that explicitly manages the dynamics uncertainty by incorporating a reward penalty during world model training. In this work, we compare our approach with the model-based LOMPO method and the model-free CQL method integrated into the visual RL backbone of DrQ-V2 (Yarats et al., 2021a).

Video-enhanced RL. In recent years, there has been growing interest in harnessing auxiliary video data to enhance RL, as such data is more accessible and affordable in real-world settings. We can broadly categorize these approaches into two groups. The first group focuses on using videos to improve agent performance through representation learning (Seo et al., 2022; Ma et al., 2023; Wu et al., 2023). For instance, APV (Seo et al., 2022) pre-trains an *action-free* video prediction model using videos and then finetunes the agent with these pretrained representations. VIP (Ma et al., 2023) enhances generalization to downstream offline tasks by leveraging self-supervised visual representations. The second group focuses on learning transferable behavior policies from the source video domain (Torabi et al., 2018; Chang et al., 2022; Baker et al., 2022; Schmeckpeper et al., 2021; Ye et al., 2022). For example, the VPT model (Baker et al., 2022) trains an inverse dynamics model with a small set of *labeled* auxiliary videos and then assigns pseudo-actions to unlabeled video data. This method requires strong consistency across the source to target domains. Another line of work focuses on learning interactive world models from the unlabeled *target-domain* videos to generate playable virtual environments (Valevski et al., 2024; Yang et al., 2024a). While promising, these methods highlight the potential of interactive video generation, without explicitly using them for downstream control tasks. Besides, these approaches have yet to be formally published. In summary, unlike VeoRL, previous video-enhanced approaches typically require the source video set to be either annotated with action/reward labels or derived from the same domain as the target environment for policy deployment, or not directly designed for sequential decision-making tasks.

3. Method

We aim to enhance the performance of offline visual RL trained on an offline target dataset $\mathcal{B}^{\text{tar}}$ by leveraging task-agnostic video data from a source domain $\mathcal{B}^{\text{src}}$. Two key challenges arise in this context: (i) The auxiliary videos generally lack action labels and reward function annotations; (ii) There exist inherent distributional shifts between $\mathcal{B}^{\text{src}}$ and $\mathcal{B}^{\text{tar}}$. To address these challenges, VeoRL employs a three-stage approach (as detailed in Algorithm 1):

1. We introduce a *behavior abstraction network* designed to extract underlying control behaviors directly from the raw observations of unlabeled videos.
2. We design a hierarchical world model to capture state transitions in two ways: through real actions using a *trunk net* and through latent behaviors using a *plan net*.
3. We propose a model-based RL method that optimizes the target policy over the trunk net transitions while being guided by latent behavior rollouts from the plan net.

3.1. Latent Behavior Abstraction

To address the challenge of video data lacking action labels, previous methods (Baker et al., 2022; Zhang et al., 2022) typically focus on using target domain observations with real action labels (*i.e.,* the offline RL dataset) to train an inverse dynamics model, which is then used to assign pseudo-action labels to the source domain auxiliary videos. However, this approach assumes that the source and target domains share the same action space, leading to two potential issues: First, action predictions for source video data may suffer from domain distribution discrepancies. Second, the learned actions are limited in their ability to provide high-level behavior guidance for subsequent policy learning. Instead, we propose a fully unsupervised approach for behavior inference, leveraging a compact categorical latent action space significantly smaller than the raw continuous action space. This approach avoids the challenge of real action prediction, generating higher-level behavior representations that are distinct from the original actions for low-level visual control.

To obtain a discretized latent action space, we design a *behavior abstraction network* (BAN) based on vector quantization (Gray, 1984). As shown in Figure 2a, this module is trained within a one-step video prediction framework, where its output is used to drive a *recurrent state-space model* (RSSM) (Hafner et al., 2019) for forward dynamics modeling. The learning process begins by taking two observations $(o_{t-1}^{\text{tar}}, o_t^{\text{tar}})$ from the target dataset. We extract image encodings from these observations using a CNN model (E_θ) and concatenate them into a latent vector, $[e_{t-1}^{\text{tar}}, e_t^{\text{tar}}] \in \mathbb{R}^D$, which captures the temporal difference between consecutive video frames. We maintain a learnable latent codebook $\mathbb{R}^{K\times D}$ consisting of K continuous vectors $c_{k=1,\dots,K} \in \mathbb{R}^D$, each representing a distinct behavior abstraction. Given $[e_{t-1}^{\text{tar}}, e_t^{\text{tar}}]$, BAN identifies the nearest neighbor c_i from the set $\{c_k\}_{k=1}^K$ and outputs this vector as the chosen latent

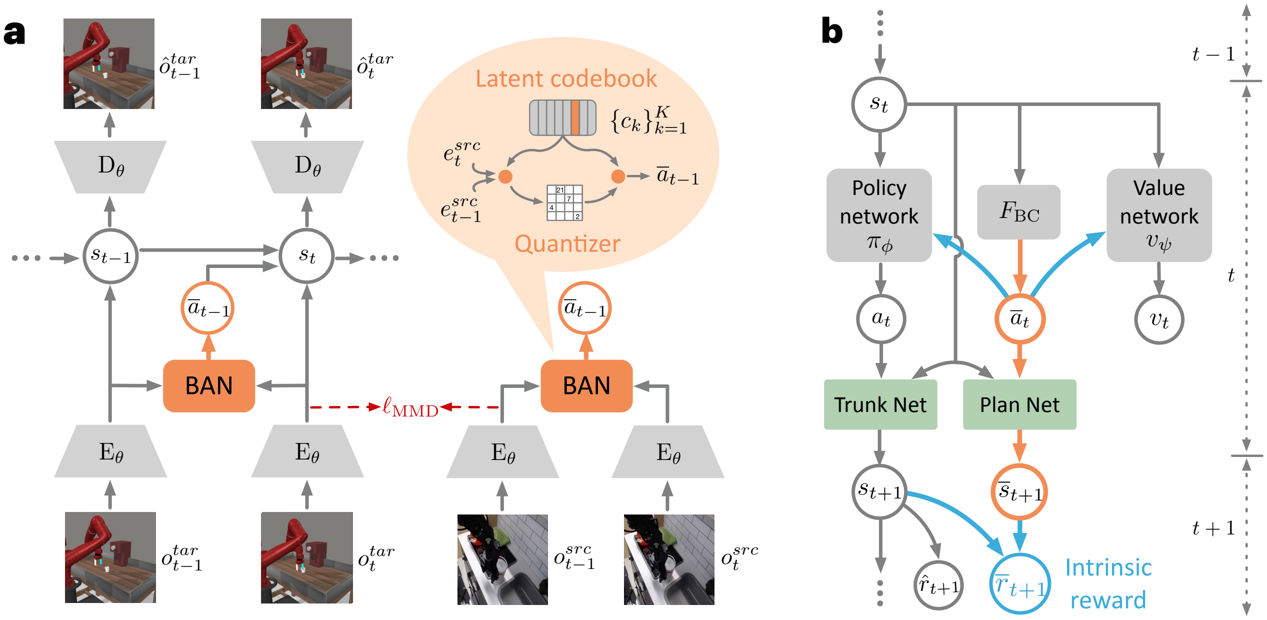

Figure 2. **Model architecture. a,** We construct a discrete, high-level latent action space by training the BAN, enabling forward dynamics modeling independent of real actions. **b,** The visualization of model-based actor-critic learning at a single rollout step. We leverage F_{BC} to replay the video-informed latent behaviors, serving as the inputs of the actor and critic for producing goal-conditioned policies and value estimations, as well as the plan net for generating a long-term state rollout.

behavior $\bar{a}_{t-1}$, corresponding to the transition from o_{t-1}^{tar} to o_t^{tar}. Similar to VQ-VAE (Van Den Oord et al., 2017), the optimization objective of the vector quantization process can be formulated as

$$\ell_{\text{VQ}} = \underbrace{\left\|\text{sg}[e_{t-1}^{\text{tar}}, e_t^{\text{tar}}] - c_i\right\|_2}_{\text{codebook loss}} + \underbrace{\left\|[e_{t-1}^{\text{tar}}, e_t^{\text{tar}}] - \text{sg}(c_i)\right\|_2}_{\text{commitment loss}}, \quad (1)$$

where $\text{sg}(\cdot)$ denotes gradient stopping. The codebook loss encourages the selected code to be closer to the input, while the commitment loss facilitates gradient backpropagation into the encoder. This helps stabilize the training process by preventing the image embedding from switching indiscriminately between different codes in the codebook.

As illustrated in Figure 2a, the selected latent behavior serves as input to a state transition model referred to as the *plan net*. We backpropagate the video prediction error to the BAN to refine the selection of latent behaviors, ensuring that they better predict subsequent frames. The overall objective function of BAN can be formulated as $\ell_{\text{BAN}} = \ell_{\text{VQ}} + \alpha \ell_{\text{plan}}$, where ℓ_{plan} denotes the loss function associated with the plan net, which will be detailed subsequently.

To leverage the BAN trained on $\mathcal{B}^{\text{tar}}$ as an estimator of latent behaviors for OOD videos from $\mathcal{B}^{\text{src}}$, we adopt the *maximum mean discrepancy* (MMD) loss (Borgwardt et al., 2006), a method widely used in domain adaptation (Huang et al., 2006; Li et al., 2013; Gong et al., 2013). The MMD loss is defined as $\ell_{\text{MMD}} = \left\|\mathbb{E}_{o_t^{\text{src}}} e_t^{\text{src}} - \mathbb{E}_{o_t^{\text{tar}}} e_t^{\text{tar}}\right\|_2^2$ and is applied to the outputs of the image encoder to align visual embeddings across different domains.

3.2. Two-Stream World Model

We design a two-stream interactive world model consisting of: (i) a *Trunk net* that captures future state transitions driven by real actions and the associated environmental rewards, $(s_1, a_1, r_1, s_2, a_2, r_2, \ldots) \sim \text{Trunk}(o_1; \pi_\phi)$, and (ii) a *Plan net* that learns to predict future trajectories based on high-level behavior abstractions, $(\bar{s}_1, \bar{a}_1, \bar{s}_2, \bar{a}_2, \ldots) \sim \text{Plan}(o_1; \text{BAN})$. Here, $s_t = [h_t, z_t]$ and $\bar{s}_t = [\bar{h}_t, \bar{z}_t]$, where $(h_t, \bar{h}_t)$ and $(z_t, \bar{z}_t)$ represent the deterministic and stochastic components of the predicted states, respectively. Specifically, the plan net includes a behavior cloning module F_{BC} that predicts $\bar{a}_t$ based solely on $\bar{s}_t$. The learning target for F_{BC} is the latent behavior inferred by BAN from the state transition pair $(\bar{s}_t, \bar{s}_{t+1})$. The world model can be formulated as

$$\begin{array}{ll}
\text{Trunk Net:} & \text{Plan Net:} \\
\left\{\begin{array}{l}
h_t = \text{GRU}(s_{t-1}, a_{t-1}; \theta) \\
z_t \sim p(h_t; \theta) \\
z'_t \sim q(h_t, e_t; \theta) \\
\hat{o}_t \sim p(s_t; \theta) \\
\hat{r}_t \sim p(s_t; \theta)
\end{array}\right. &
\left\{\begin{array}{l}
\bar{h}_t = \text{GRU}(\bar{s}_{t-1}, \bar{a}_{t-1}; \bar{\theta}) \\
\bar{z}_t \sim p(\bar{h}_t; \bar{\theta}) \\
\bar{z}'_t \sim q(\bar{h}_t, e_t; \bar{\theta}) \\
\bar{o}_t \sim p(\bar{s}_t; \bar{\theta}) \\
\bar{a}_t = F_{\text{BC}}(\bar{s}_t; \bar{\theta})
\end{array}\right.
\end{array} \quad (2)$$

where $(z'_t, \bar{z}'_t)$ indicate the posteriors inferred from the current image observation and $(\hat{o}_t, \bar{o}_t)$ are the reconstructed images. θ represents the combined parameters of the trunk net, while $\bar{\theta}$ represents the combined parameters of the plan net. The trunk net and plan net share the same model parameters for the image encoder and decoder, but have separate parameters for the remaining parts of the model. The trunk net is trained exclusively on the offline RL dataset $\mathcal{B}^{\text{tar}}$. The objective function is as follows:

$$\begin{aligned}\ell_{\text{trunk}}(\theta) = \ \mathbb{E}_{\tau\in\mathcal{B}^{\text{tar}}} \sum_{t=1}^{T} \underbrace{-\ln p(\hat{o}_t \mid s_t)}_{\text{image reconstruction}} - \underbrace{\ln p(\hat{r}_t \mid s_t)}_{\text{reward prediction}} \\ + \underbrace{\text{KL}[q(z'_t \mid h_t, e_t) \parallel p(z_t \mid h_t)]}_{\text{KL divergence}}.\end{aligned} \tag{3}$$

The plan net is trained using trajectories from the target dataset $\mathcal{B}^{\text{tar}}$ and unlabeled videos from the source dataset $\mathcal{B}^{\text{src}}$. We adopt a specialized training strategy to ensure that the plan net models meaningful long-term state transitions. Instead of focusing on one-step future predictions, we consider the state transition "*shortcuts*" that occur when BAN predicts a different $\bar{a}_t$ compared to the previous step $\bar{a}_{t-1}$. Specifically, starting with an original trajectory $(o_1, \bar{a}_1, o_2, \bar{a}_2, \ldots, o_{T-1}, \bar{a}_{T-1}, o_T)$ sampled from the dataset and precomputed by BAN, we retain only those state-action pairs $(\bar{a}_t, o_{t+1})$ where a change in latent behavior is detected, *i.e.,* , $\bar{a}_t \neq \bar{a}_{t-1}$. This process generates a refined trajectory that captures significant behavioral changes over time, represented as $(o_{j_1}, \bar{a}_{j_1}, o_{j_2}, \bar{a}_{j_2}, \ldots, o_{j_{n-1}}, \bar{a}_{j_{n-1}}, o_{j_n})$, where $o_{j_1} = o_1$. These refined trajectories are subsequently used to optimize the plan network through the following objective function:

$$\begin{aligned}\ell_{\text{plan}}(\bar{\theta}) = \ \mathbb{E}_{\tau\in\mathcal{B}^{\text{tar}},\mathcal{B}^{\text{src}}} \sum_{t=j_1}^{j_n} \underbrace{-\ln p(\bar{o}_t \mid \bar{s}_t)}_{\text{image reconstruction}} - \underbrace{\ln p(\bar{a}_t \mid s_t)}_{\text{behavior cloning}} \\ + \underbrace{\text{KL}[q(\bar{z}'_t \mid \bar{h}_t, e_t) \parallel p(\bar{z}_t \mid \bar{h}_t)]}_{\text{KL divergence}}.\end{aligned} \tag{4}$$

This approach enables the plan net to capture significant behavior shifts, enhancing its ability to predict impactful long-term transitions. It provides supplementary information that complements short-term, real action-based state transitions during the subsequent policy optimization stage.

3.3. Model-Based Policy Learning

We perform temporal difference learning using a probabilistic policy network $\pi_\phi(\cdot)$ and a deterministic value network $v_\psi(\cdot)$. As shown in Figure 2b, these model components are optimized over two-stream imagined trajectories generated by the trunk net and the plan net, *i.e.,* $(s_1, \bar{s}_1, \hat{a}_1, \bar{a}_1, \hat{r}_1, s_2, \bar{s}_2, \hat{a}_2, \bar{a}_2, \hat{r}_2, \ldots, s_L, \bar{s}_L, \hat{a}_L, \bar{a}_L, \hat{r}_L)$, where s_1 is derived from an observation o_1 randomly sampled from the target set $\mathcal{B}^{\text{tar}}$.

Unlike previous methods, both the policy network and the value network are additionally conditioned on the estimated latent behavior to incorporate high-level control guidance, such that $\pi_\phi(a_t \mid s_t, \bar{a}_t)$ and $v_\psi(s_t, \bar{a}_t)$, where $\bar{a}_t = F_{\text{BC}}(s_t)$. This modification allows for more informed decision-making by integrating abstract behavioral patterns from the auxiliary videos into the learning process.

The learning approach in Eq. (4) enables the plan net to predict long-term states that can serve as decision-making *subgoals*. Accordingly, we propose a goal-conditioned intrinsic reward that quantifies the similarity between short-term state transitions induced by real actions and long-term state transitions driven by high-level latent behaviors:

$$\begin{aligned}&\bar{r}_t = -\left\| s_t, \bar{s}_t \right\|_2, \\ &s_t \sim \text{Trunk}(s_{t-1}, \pi_\phi(s_{t-1}, \bar{a}_{t-1})), \\ &\bar{s}_t \sim \text{Plan}(s_{t-1}, F_{\text{BC}}(s_{t-1})).\end{aligned} \tag{5}$$

The policy network is trained to output actions that lead to states that maximize the value network's output, while the value network seeks to accurately estimate the discounted future returns achieved by the policy network starting from each imagined state. Specifically, the value target V_t^λ for the value network incorporates a weighted average of reward information over an n-step future horizon, which is defined recursively as follows:

$$\begin{aligned}V_t^\lambda \doteq\ & (\hat{r}_t + \omega \bar{r}_t) \\ & + \gamma \begin{cases} (1-\lambda)\, v_\psi(s_{t+1}, \bar{a}_{t+1}) + \lambda V_{t+1}^\lambda, & \text{if} \quad t < L, \\ v_\psi(s_L), & \text{if} \quad t = L. \end{cases}\end{aligned} \tag{6}$$

where λ is set to 0.95 for considering more on long horizon targets. γ is a discount factor set to 0.99 in practice. Given a trajectory of model states and rewards, the value network v_ψ is trained to regress V_t^λ and the policy network π_ϕ is optimized with entropy regularization:

$$\begin{aligned}\ell(\psi) &= \mathbb{E}_{p_\theta, p_\phi} \sum_{t=1}^{L-1} \frac{1}{2} (v_\psi(s_t, \bar{a}_t) - \text{sg}(V_t^\lambda))^2, \\ \ell(\phi) &= \mathbb{E}_{p_\theta, p_\phi} \sum_{t=1}^{L-1} \Bigg(\underbrace{-(1-\rho) V_t^\lambda}_{\text{Dynamics BP}} \underbrace{-\eta H[a_t | s_t, \bar{a}_t]}_{\text{Entropy}} \\ &\qquad \underbrace{-\rho \ln p_\phi(\hat{a}_t \mid s_t, \bar{a}_t) \text{sg}(V_t^\lambda - v_\psi(s_t, \bar{a}_t))}_{\text{REINFORCE}} \Bigg).\end{aligned} \tag{7}$$

For the first term in the actor loss, we employ the straight-through gradients (Bengio et al., 2013) to backpropagate the value targets of the sampled actions and state sequences directly through the learned dynamics. In continuous control tasks like Meta-World and CARLA, we set $\rho = 0$. In discrete control tasks, we set $\rho = 1$ and observe that integrating REINFORCE gradients yields improved results.

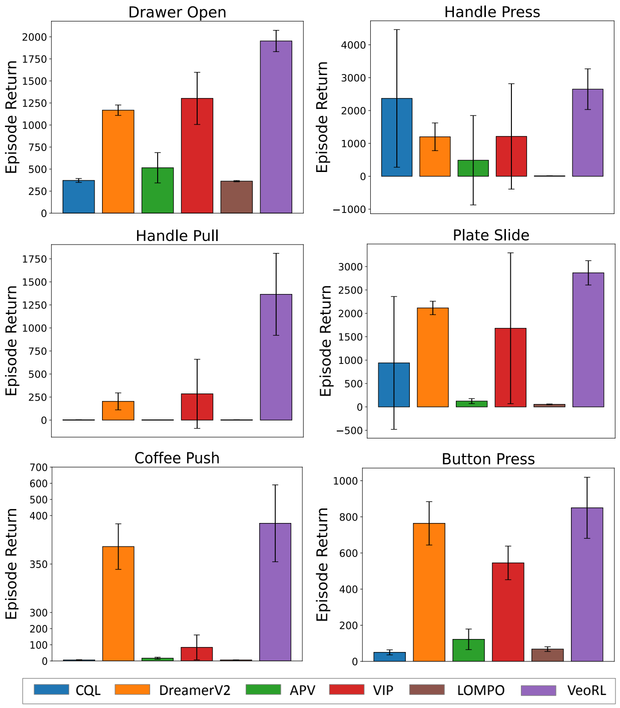

Figure 3. **Performance on Meta-World robotic manipulation tasks in episode return.** Error bars indicate standard deviation across the 50 evaluation episodes, with 3 random training seeds.

4. Experiments

In this section, we present (i) quantitative comparisons with existing RL methods on a diverse set of visual control benchmarks, (ii) offline-to-online transfer learning results on novel interactive control tasks, (iii) ablation studies for each proposed model component in VeoRL, and (iv) hyperparameter analyses along with visualizations of the learned latent behavior abstractions. The hyperparameters for implementing VeoRL are provided in Appendix Table 2.

To demonstrate the capability of VeoRL, we conduct experiments on three visual RL environments, including *Meta-World* (Yu et al., 2019), *CARLA* (Dosovitskiy et al., 2017), and *MineDojo* (Fan et al., 2022) . Like D4RL (Fu et al., 2020), we collect the offline RL datasets of *medium*-quality trajectories using a partially-trained DreamerV2 agent (Hafner et al., 2020). Please refer to Appendix Section C for more details of the benchmarks.

We compare VeoRL with: (1) *DrQ+CQL* (Kumar et al., 2020) and *LOMPO* (Rafailov et al., 2021) are specifically designed for offline RL tasks. (2) *DreamerV2* (Hafner et al., 2020) serves as our primary baseline. It follows a similar model-based RL framework but lacks the video-enhancement method. (3) Like our approach, *APV* (Seo et al., 2022) and *VIP* (Ma et al., 2023) exploit auxiliary videos to improve model performance. (4) Finally, *VPT* (Baker et al., 2022) is particularly designed for Minecraft tasks, so we only use it as the baseline model on the MineDojo environment. Further details of the compared models can be found in Appendix Section D.

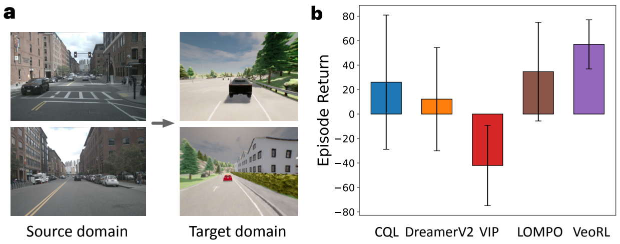

Figure 4. **Experiments of autonomous driving. a**, Showcases of the source NuScenes and target CARLA datasets. **b**, Performance comparison on CARLA, measured by averaged episode returns.

4.1. Main Results

Meta-World robotic manipulation. For the Meta-World offline RL tasks, we use the BridgeData-V2 dataset (Walke et al., 2023) with real-world videos as the unlabeled source domain. We conduct all tasks with 3 random seeds and report the mean results and standard deviations over 50 episodes. As shown in Figure 3, our approach achieves state-of-the-art performance in episodic returns over all six tasks in this domain. Specifically, VeoRL outperforms DreamerV2 by a large margin in *Drawer Open* (1168 → 1953), *Handle Press* (1201 → 2650) and *Handle Pull* (203 → 1365). Compared to APV and VIP, which also employ video-enhanced pretraining, VeoRL demonstrates a significant advantage by effectively exploiting and transferring the underlying control behaviors behind these videos. Furthermore, we provide additional model comparisons measured by success rate in Appendix Table 3, where VeoRL consistently achieves the best performance.

CARLA autonomous driving. We employ the large-scale NuScenes dataset (Caesar et al., 2019) with 1,000 diverse real-world driving scenes to improve the autonomous driving performance on the CARLA benchmark. Figure 4 shows the performance of our model compared to different baselines across three random training seeds. As we can see, our VeoRL achieves notably the best mean results with the smallest standard deviations, outperforming DreamerV2 by as much as **350%**. Notably, although the VIP method also involves pretraining on the same NuScenes dataset, its policy transfer approach (based on a self-supervised value-function objective) struggles to achieve satisfactory results in this driving task. We exclude the results from APV in Figure 4 to provide a clearer comparison of the other approaches, as it yields an average episode return of approximately −500.

MineDojo open-world games. MineDojo is an open-world 3D environment with a vast state space for the agent

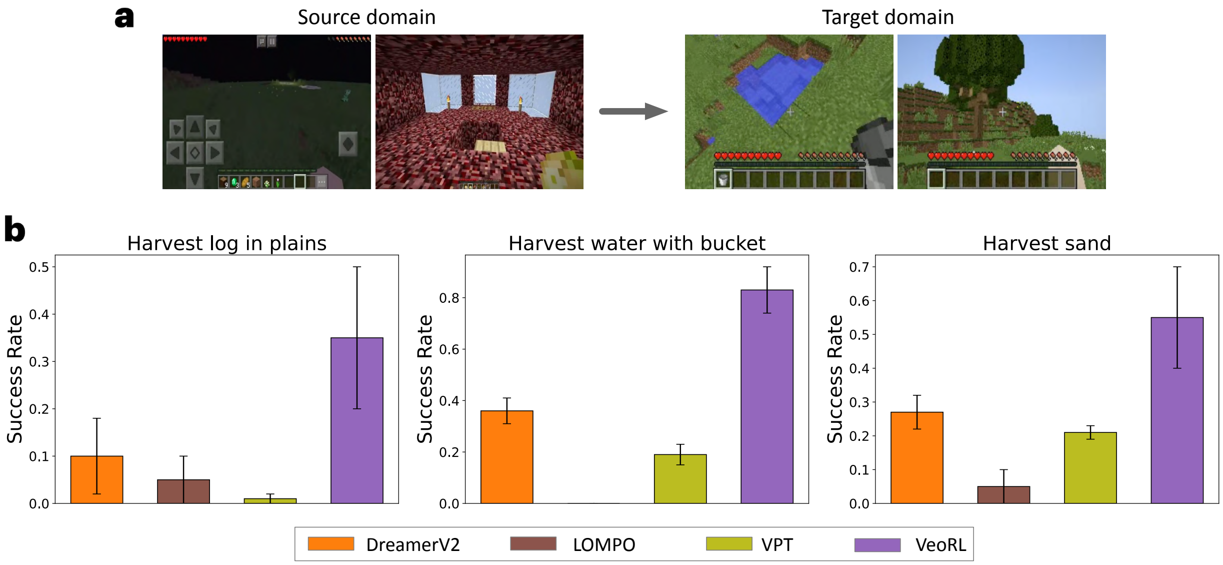

Figure 5. **Experiments of the MineDojo 3D navigation and control tasks. a**, Showcases of source online videos and target offline datasets. **b**, Performance comparison in success rate.

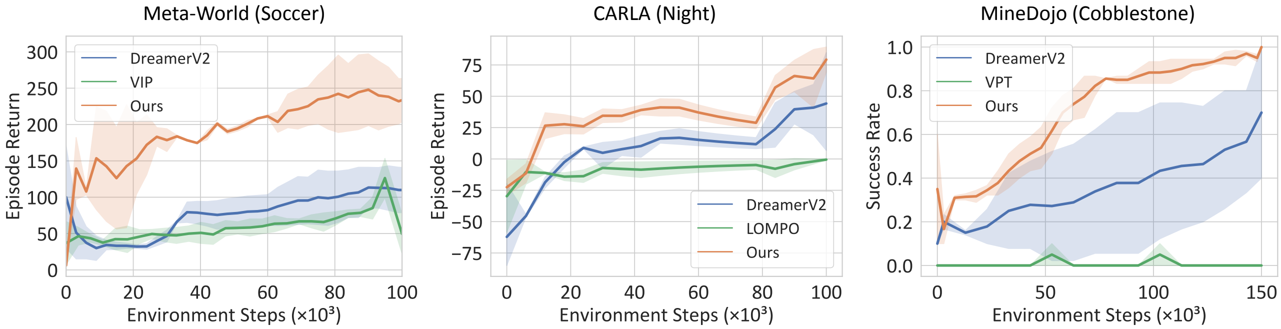

Figure 6. **Performance of offline-to-online finetuning on novel tasks.** By conducting offline pretraining, VeoRL demonstrates improved training efficiency and final performance when adapting to unseen, more challenging visual control tasks. Notably, we train the DreamerV2 model under the same offline-to-online setup. The results are averaged over 3 random training seeds.

to explore, making effective video demonstrations particularly essential for the target offline learning scenarios. As shown in Figure 5a, we use online videos of human players as the auxiliary source video set. These videos contain random Minecraft tasks that may differ significantly in gaming tactics from those for the target tasks. The results in Figure 5b show that VeoRL outperforms the other baseline models. Specifically, it improves upon DreamerV2 by **250%** (0.10 → 0.35) in *Harvest log in plains*, by approximately **130%** (0.36 → 0.83) in *Harvest water with bucket*, and by around **100%** (0.27 → 0.55) in *Harvest sand*.

Offline-to-online transfer learning. To evaluate the generalizability of our model, we finetune the pretrained offline RL agents in *unseen* visual control tasks through online interactions with the environment. We select a more challenging *Soccer* task as our offline-to-online transfer target on the Meta-World benchmark. This task typically involves a two-stage decision-making process where the agent must first fetch a ball and then push it to a goal position. For CARLA, the offline RL agents are pretrained using the *Day-mode* dataset and then finetuned in the *Night mode* driving scenarios. For MineDojo, the agents are pretrained on a fixed dataset for the *Harvest sand* task and finetuned on the *Harvest cobblestone with wooden pickaxe* task. As shown in Figure 6, VeoRL demonstrates superior training efficiency compared to DreamerV2, which follows the same offline-to-online domain adaptation setup, highlighting the effectiveness of our offline training approach.

4.2. Ablation Studies

As we will discuss in Section 3, VeoRL introduces three key innovations to the model-based actor-critic learning framework of DreamerV2 (Hafner et al., 2021): (i) It uses latent behaviors as inputs to the policy network (*i.e.,* actor); (ii) It

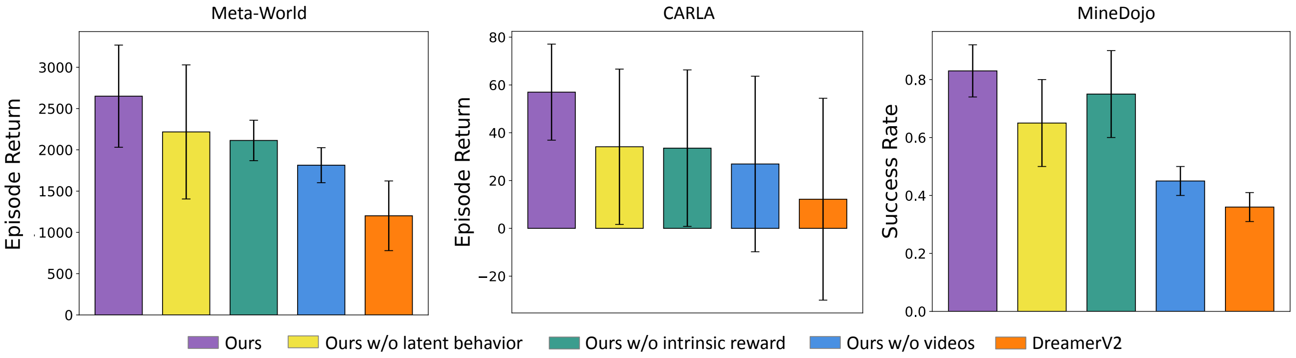

Figure 7. **Ablation studies of VeoRL.** We show the impact of latent behavior guidance (*yellow*) and the intrinsic reward (*green*). We also compare VeoRL with a baseline model that does not use any source video data throughout the training process (*blue*).

Table 1. **Performance comparison with DreamerV3 on Meta-World robotic manipulation tasks in success rate and episode return.** We compute the mean and standard deviation of 50 episodes over 3 random training seeds.

Methods	DreamerV2	DreamerV3	VeoRL(DV2)	VeoRL(DV3)
		Success Rate		
Drawer Open	0.18 ± 0.04	0.00 ± 0.00	0.70 ± 0.07	0.55 ± 0.15
Handle Press	0.33 ± 0.11	0.05 ± 0.05	0.60 ± 0.12	0.35 ± 0.15
		Episode Return		
Drawer Open	1168.35 ± 59.55	674.55 ± 79.04	1953.60 ± 121.48	1393.50 ± 122.50
Handle Press	1201.75 ± 422.10	257.85 ± 247.05	2650.90 ± 619.60	1360.15 ± 547.85

employs latent behaviors to compute an intrinsic reward for training the value network (*i.e.,* critic), and (iii) It trains the *plan net* in the world model using the auxiliary video set and the offline RL dataset jointly. Accordingly, we conduct ablation studies to evaluate the effectiveness of individual techniques. Figure 7 provides corresponding results on Meta-World (*Handle Press*), CARLA and MineDojo (*Harvest water with bucket*). As illustrated by the yellow and green bars, excluding the latent behavior input in the policy network or removing its influence from the value network's training objective both result in significant performance drops in VeoRL. These results highlight the crucial role of latent behaviors as effective high-level guidance in policy learning. The blue bar illustrates that training all network components solely on the target offline dataset (without access to auxiliary videos) leads to a significant performance decline. This highlights that our video-enhanced method effectively extracts valuable prior knowledge of underlying expert behaviors from the unlabeled videos.

4.3. Comparison with DreamerV3

The decision to use DreamerV2 as the backbone is driven by two key considerations: (i) *Offline setting performance*. While DreamerV3 (Hafner et al., 2023) achieves strong performance across diverse tasks with fixed hyperparameters, our experiment results in Table 1 demonstrate that it underperforms in offline RL settings without hyperparameter tuning compared to DreamerV2. (ii) *Backbone consistency*. LAMPO, a key baseline in our work, uses DreamerV2 as its backbone. To ensure direct and fair comparisons between our method and LAMPO, we maintain consistency by adopting the same backbone. Notably, we also conduct additional experiments on Meta-World by integrating VeoRL with DreamerV3. The results demonstrate that our approach outperforms vanilla DreamerV3, showcasing its ability to seamlessly integrate with different architectures.

4.4. Analyses of Latent Actions

To illustrate the correspondence between latent behaviors and real actions, we evaluate the distribution of the learned latent behavior abstractions in Meta-World *Coffee Push* in Figure 8. In VeoRL, we roll out the trunk net and plan net simultaneously for model-based policy learning. Starting from the same input state s_t, we obtain two future trajectories of real actions $a_{t:t+H}$ and corresponding latent behaviors $\bar{a}_{t:t+H}$. To show the correspondence between them, we specifically select trajectories of four consecutive real actions $a_{t:t+4}$ with time-invariant latent behaviors $\bar{a}_t = \bar{a}_{t:t+4}$, and then use the t-SNE algorithm (Van der Maaten & Hinton, 2008) to project $a_{t:t+4}$ into a 2D plane.

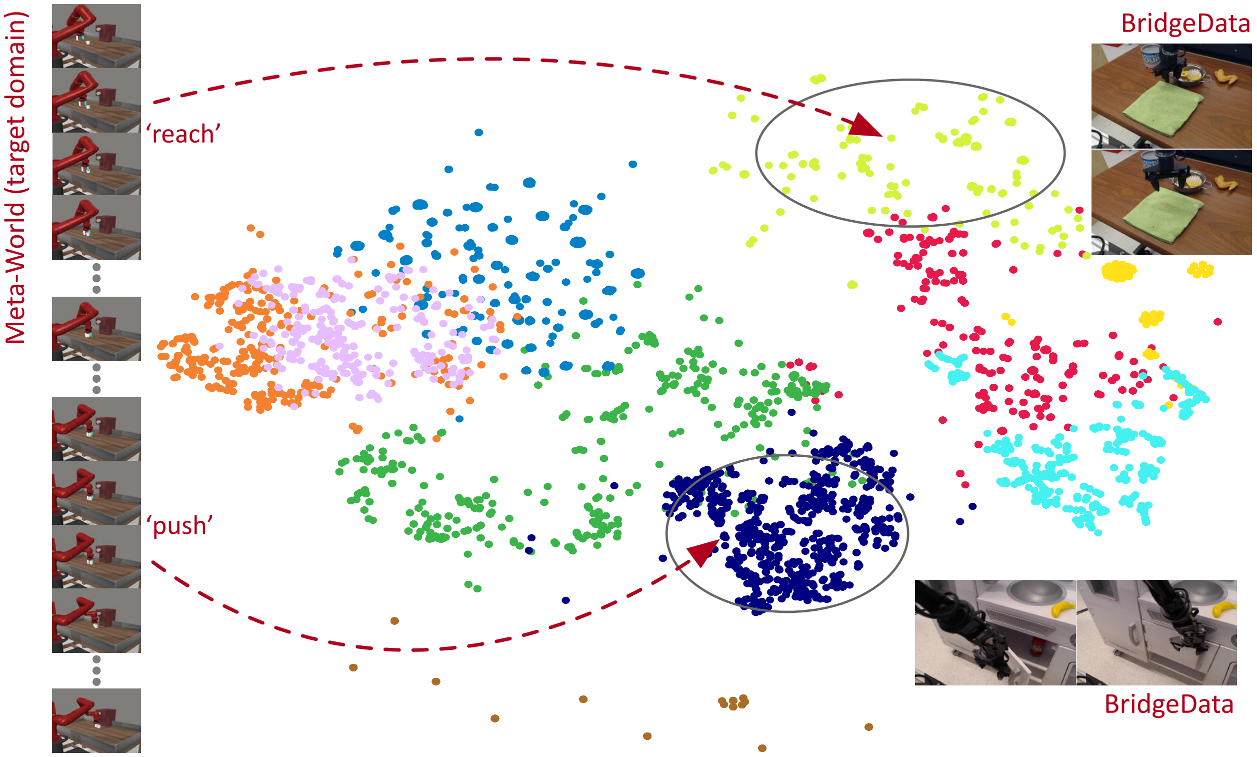

Figure 8. **Correspondence between latent behaviors and real actions in Meta-World.** During model-based state rollouts, starting from the same state s_t of the target domain, VeoRL simultaneously generates real action trajectories $a_{t:t+H}$ and latent behaviors $\bar{a}_{t:t+H}$. We consider trajectories of four consecutive real actions $a_{t:t+4}$, where the latent behaviors are time-invariant, *i.e.,* $\bar{a}_t = \bar{a}_{t:t+4}$. We then use t-SNE to project $a_{t:t+4}$ into a 2D plane. To distinguish between different latent behaviors, we assign distinct colors to the points, visualizing 10 latent behaviors for clarity. We present BridgeData examples where VeoRL generates the corresponding latent behaviors.

Different colors assigned to the points indicate distinct latent behaviors. From Figure 8, we observe that the latent behaviors align well with distinct clusters of real action sequences, demonstrating the high representational fidelity of the learned latent action space. These latent behaviors effectively capture semantic information from online videos and can be interpreted as fundamental operational skills, which can then be used as policy guidance for the RL agent. Furthermore, we provide the ablation results of varying the number of discrete latent behavior in Appendix Figure 9.

4.5. Further Results and Discussion

Please refer to Appendices F-I for additional experimental results, including the quality analysis of source videos and target domain trajectories, the hyperparameter analysis of latent action space, the generalizability across tasks, and the impact of MMD in domain adaptation.

5. Conclusions and Limitations

In this paper, we introduced VeoRL, a novel approach that leverages freely available, unlabeled video data to enhance offline visual RL. VeoRL constructs a world model with two state transition branches: one conditioned on latent behaviors extracted from video data and the other on the agent's real actions. This dual-branch structure allows the agent to effectively learn control policies by aligning state rollouts from two branches, thereby transferring knowledge from the diverse, out-of-domain video data to the RL agents. Our results demonstrate that VeoRL significantly outperforms existing offline RL methods across a wide range of challenging visual control tasks, including robotic manipulation, autonomous driving, and open-world video games. The integration of video data not only boosts the agent's ability to generalize across different domains but also accelerates the learning process by incorporating a broader understanding of the physical world and control policies.

The limitation of this work is the computational overhead. The training process of VeoRL involves extracting latent behaviors and optimizing a world model with two state transition branches. Despite the performance gain, the dual-branch setup could increase the computational complexity and memory requirements, making the approach more resource-intensive than simpler RL algorithms.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (62250062), the Smart Grid National Science and Technology Major Project (2024ZD0801200), the Shanghai Municipal Science and Technology Major Project (2021SHZDZX0102), and the Fundamental Research Funds for the Central Universities.

Impact Statement

VeoRL represents a significant advancement in offline RL by leveraging unlabeled video data to enhance agent policy learning and generalization. It enables agents to be trained on vast amounts of real-world data without the need for labor-intensive annotations, making it more accessible for researchers, developers, and practitioners. VeoRL brings RL closer to how humans learn by observing and interacting with the world. Additionally, by aligning RL with human learning processes, VeoRL enables the development of more intuitive AI systems.

References

Baker, B., Akkaya, I., Zhokov, P., Huizinga, J., Tang, J., Ecoffet, A., Houghton, B., Sampedro, R., and Clune, J. Video pretraining (VPT): Learning to act by watching unlabeled online videos. In *NeurIPS*, volume 35, pp. 24639–24654, 2022.

Bengio, Y., Léonard, N., and Courville, A. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Borgwardt, K. M., Gretton, A., Rasch, M. J., Kriegel, H.-P., Schölkopf, B., and Smola, A. J. Integrating structured biological data by kernel maximum mean discrepancy. *Bioinformatics*, 22(14):e49–e57, 2006.

Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., and Beijbom, O. nuscenes: A multimodal dataset for autonomous driving. *arXiv preprint arXiv:1903.11027*, 2019.

Cen, Z., Liu, Z., Wang, Z., Yao, Y., Lam, H., and Zhao, D. Learning from sparse offline datasets via conservative density estimation. *arXiv preprint arXiv:2401.08819*, 2024.

Chang, M., Gupta, A., and Gupta, S. Learning value functions from undirected state-only experience. In *ICLR*, 2022.

Chen, C., Zhang, Y., Liu, X., and Coates, M. Bidirectional learning for offline model-based biological sequence design. In *ICML*, pp. 5351–5366, 2023.

Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., and Koltun, V. CARLA: An open urban driving simulator. In *CoRL*, pp. 1–16, 2017.

Fan, L., Wang, G., Jiang, Y., Mandlekar, A., Yang, Y., Zhu, H., Tang, A., Huang, D.-A., Zhu, Y., and Anandkumar, A. Minedojo: Building open-ended embodied agents with internet-scale knowledge. In *NeurIPS*, volume 35, pp. 18343–18362, 2022.

Fu, J., Kumar, A., Nachum, O., Tucker, G., and Levine, S. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Fujimoto, S. and Gu, S. S. A minimalist approach to offline reinforcement learning. *NeurIPS*, 34:20132–20145, 2021.

Fujimoto, S., Meger, D., and Precup, D. Off-policy deep reinforcement learning without exploration. In *ICML*, pp. 2052–2062, 2019.

Gong, B., Grauman, K., and Sha, F. Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In *ICML*, pp. 222–230, 2013.

Gray, R. Vector quantization. *IEEE Assp Magazine*, 1(2): 4–29, 1984.

Gumbsch, C., Sajid, N., Martius, G., and Butz, M. V. Learning hierarchical world models with adaptive temporal abstractions from discrete latent dynamics. In *ICLR*, 2023.

Ha, D. and Schmidhuber, J. Recurrent world models facilitate policy evolution. In *NeurIPS*, 2018.

Hafner, D., Lillicrap, T., Fischer, I., Villegas, R., Ha, D., Lee, H., and Davidson, J. Learning latent dynamics for planning from pixels. In *ICML*, pp. 2555–2565, 2019.

Hafner, D., Lillicrap, T., Ba, J., and Norouzi, M. Dream to control: Learning behaviors by latent imagination. In *ICLR*, 2020.

Hafner, D., Lillicrap, T., Norouzi, M., and Ba, J. Mastering atari with discrete world models. In *ICLR*, 2021.

Hafner, D., Lee, K.-H., Fischer, I., and Abbeel, P. Deep hierarchical planning from pixels. In *NeurIPS*, volume 35, pp. 26091–26104, 2022.

Hafner, D., Pasukonis, J., Ba, J., and Lillicrap, T. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

Hatch, K., Yu, T., Rafailov, R., and Finn, C. Example-based offline reinforcement learning without rewards. *Proceedings of Machine Learning Research vol*, 144:1–17, 2022.

Hatch, K. B., Eysenbach, B., Rafailov, R., Yu, T., Salakhutdinov, R., Levine, S., and Finn, C. Contrastive example-based control. In *L4DC*, pp. 155–169, 2023.

Hong, J., Dragan, A., and Levine, S. Offline rl with observation histories: Analyzing and improving sample complexity. In *ICLR*, 2024.

Huang, J., Gretton, A., Borgwardt, K., Schölkopf, B., and Smola, A. Correcting sample selection bias by unlabeled data. In *NeurIPS*, volume 19, 2006.

Kostrikov, I., Yarats, D., and Fergus, R. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *ICLR*, 2021.

Kostrikov, I., Nair, A., and Levine, S. Offline reinforcement learning with implicit Q-learning. In *ICLR*, 2022.

Kumar, A., Fu, J., Soh, M., Tucker, G., and Levine, S. Stabilizing off-policy q-learning via bootstrapping error reduction. In *NeurIPS*, volume 32, 2019.

Kumar, A., Zhou, A., Tucker, G., and Levine, S. Conservative q-learning for offline reinforcement learning. In *NeurIPS*, volume 33, pp. 1179–1191, 2020.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Laskin, M., Lee, K., Stooke, A., Pinto, L., Abbeel, P., and Srinivas, A. Reinforcement learning with augmented data. In *NeurIPS*, 2020.

Levine, S., Kumar, A., Tucker, G., and Fu, J. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Li, A., Boots, B., and Cheng, C.-A. Mahalo: Unifying offline reinforcement learning and imitation learning from observations. In *ICML*, pp. 19360–19384, 2023.

Li, W., Duan, L., Xu, D., and Tsang, I. W. Learning with augmented features for supervised and semi-supervised heterogeneous domain adaptation. *IEEE Transactions on Pattern analysis and machine intelligence*, 36(6):1134–1148, 2013.

Lu, C., Ball, P. J., Rudner, T. G., Parker-Holder, J., Osborne, M. A., and Teh, Y. W. Challenges and opportunities in offline reinforcement learning from visual observations. *Transactions on Machine Learning Research*, 2023.

Ma, Y. J., Sodhani, S., Jayaraman, D., Bastani, O., Kumar, V., and Zhang, A. Vip: Towards universal visual reward and representation via value-implicit pre-training. In *ICLR*, 2023.

Mazoure, B., Eysenbach, B., Nachum, O., and Tompson, J. Contrastive value learning: Implicit models for simple offline RL. In *CoRL*, pp. 1257–1267, 2023.

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

Nottingham, K., Ammanabrolu, P., Suhr, A., Choi, Y., Hajishirzi, H., Singh, S., and Fox, R. Do embodied agents dream of pixelated sheep: Embodied decision making using language guided world modelling. In *ICML*, pp. 26311–26325, 2023.

Osa, T. and Harada, T. Discovering multiple solutions from a single task in offline reinforcement learning. In *ICML*, 2024.

Park, S., Ghosh, D., Eysenbach, B., and Levine, S. HIQL: Offline goal-conditioned RL with latent states as actions. In *NeurIPS*, volume 36, 2024.

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

Rafailov, R., Yu, T., Rajeswaran, A., and Finn, C. Offline reinforcement learning from images with latent space models. In *L4DC*, pp. 1154–1168, 2021.

Rao, D., Sadeghi, F., Hasenclever, L., Wulfmeier, M., Zambelli, M., Vezzani, G., Tirumala, D., Aytar, Y., Merel, J., Heess, N., et al. Learning transferable motor skills with hierarchical latent mixture policies. In *ICLR*, 2022.

Rigter, M., Lacerda, B., and Hawes, N. Rambo-RL: Robust adversarial model-based offline reinforcement learning. In *NeurIPS*, volume 35, pp. 16082–16097, 2022.

Schmeckpeper, K., Rybkin, O., Daniilidis, K., Levine, S., and Finn, C. Reinforcement learning with videos: Combining offline observations with interaction. In *CoRL*, volume 155, pp. 339–354, 2021.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Seo, Y., Lee, K., James, S. L., and Abbeel, P. Reinforcement learning with action-free pre-training from videos. In *ICML*, pp. 19561–19579. PMLR, 2022.

Shah, D., Bhorkar, A., Leen, H., Kostrikov, I., Rhinehart, N., and Levine, S. Offline reinforcement learning for visual navigation. *arXiv preprint arXiv:2212.08244*, 2022.

Sun, Y., Zhang, J., Jia, C., Lin, H., Ye, J., and Yu, Y. Model-bellman inconsistency for model-based offline reinforcement learning. In *ICML*, pp. 33177–33194, 2023.

Tian, S., Nair, S., Ebert, F., Dasari, S., Eysenbach, B., Finn, C., and Levine, S. Model-based visual planning with self-supervised functional distances. *arXiv preprint arXiv:2012.15373*, 2020.

Todorov, E., Erez, T., and Tassa, Y. Mujoco: A physics engine for model-based control. In *IROS*, pp. 5026–5033, 2012.

Torabi, F., Warnell, G., and Stone, P. Behavioral cloning from observation. In *IJCAI*, pp. 4950–4957, 2018.

Valevski, D., Leviathan, Y., Arar, M., and Fruchter, S. Diffusion models are real-time game engines. *arXiv preprint arXiv:2408.14837*, 2024.

Van Den Oord, A., Vinyals, O., et al. Neural discrete representation learning. In *NeurIPS*, volume 30, 2017.

Van der Maaten, L. and Hinton, G. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

Veeriah, V., Zahavy, T., Hessel, M., Xu, Z., Oh, J., Kemaev, I., van Hasselt, H. P., Silver, D., and Singh, S. Discovery of options via meta-learned subgoals. In *NeurIPS*, volume 34, pp. 29861–29873, 2021.

Walke, H., Black, K., Lee, A., Kim, M. J., Du, M., Zheng, C., Zhao, T., Hansen-Estruch, P., Vuong, Q., He, A., Myers, V., Fang, K., Finn, C., and Levine, S. Bridgedata v2: A dataset for robot learning at scale. In *CoRL*, 2023.

Wang, Q., Yang, J., Wang, Y., Jin, X., Zeng, W., and Yang, X. Making offline rl online: Collaborative world models for offline visual reinforcement learning. In *NeurIPS*, 2024.

Wu, J., Ma, H., Deng, C., and Long, M. Pre-training contextualized world models with in-the-wild videos for reinforcement learning. In *NeurIPS*, 2023.

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

Yadav, K., Ramrakhya, R., Majumdar, A., Berges, V.-P., Kuhar, S., Batra, D., Baevski, A., and Maksymets, O. Offline visual representation learning for embodied navigation. In *Workshop on Reincarnating Reinforcement Learning at ICLR 2023*, 2023.

Yang, M., Li, J., Fang, Z., Chen, S., Yu, Y., Fu, Q., Yang, W., and Ye, D. Playable game generation. *arXiv preprint arXiv:2412.00887*, 2024a.

Yang, R., Zhong, H., Xu, J., Zhang, A., Zhang, C., Han, L., and Zhang, T. Towards robust offline reinforcement learning under diverse data corruption. In *ICLR*, 2024b.

Yarats, D., Fergus, R., Lazaric, A., and Pinto, L. Mastering visual continuous control: Improved data-augmented reinforcement learning. *arXiv preprint arXiv:2107.09645*, 2021a.

Yarats, D., Zhang, A., Kostrikov, I., Amos, B., Pineau, J., and Fergus, R. Improving sample efficiency in model-free reinforcement learning from images. In *AAAI*, pp. 10674–10681, 2021b.

Ye, W., Zhang, Y., Abbeel, P., and Gao, Y. Become a proficient player with limited data through watching pure videos. In *ICLR*, 2022.

Yu, T., Quillen, D., He, Z., Julian, R., Hausman, K., Finn, C., and Levine, S. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *CoRL*, 2019.

Yu, T., Kumar, A., Rafailov, R., Rajeswaran, A., Levine, S., and Finn, C. Combo: Conservative offline model-based policy optimization. In *NeurIPS*, volume 34, pp. 28954–28967, 2021.

Yu, T., Kumar, A., Chebotar, Y., Hausman, K., Finn, C., and Levine, S. How to leverage unlabeled data in offline reinforcement learning. In *ICML*, pp. 25611–25635, 2022.

Zang, H., Li, X., Yu, J., Liu, C., Islam, R., Combes, R. T. D., and Laroche, R. Behavior prior representation learning for offline reinforcement learning. In *ICLR*, 2023.

Zhang, Q., Peng, Z., and Zhou, B. Learning to drive by watching youtube videos: Action-conditioned contrastive policy pretraining. In *ECCV*, pp. 111–128. Springer, 2022.

Appendix

A. Algorithm of VeoRL

We provide the overall training scheme of VeoRL in Algorithm 1.

Algorithm 1 The training scheme of VeoRL

```
Require: Target offline RL dataset $\mathcal{B}^{\text{tar}}$ and source video set $\mathcal{B}^{\text{src}}$.
Initialize: Random policy network $\pi_\phi$ and value network $v_\psi$.
while not converged do
  // Behavior abstraction learning
  for update $m_1$ steps do
    Draw observation sequences $o^{\text{tar}}_{1:T} \sim \mathcal{B}^{\text{tar}}$ and $o^{\text{src}}_{1:T} \sim \mathcal{B}^{\text{src}}$.
    Train BAN using $\ell_{\text{VQ}} + \alpha \ell_{\text{plan}}$ on $o^{\text{tar}}_{1:T}$.
    Train the image encoder using $\ell_{\text{MMD}}$ on $o^{\text{tar}}_{1:T}$ and $o^{\text{src}}_{1:T}$.
  end for
  // World model learning
  for update $m_2$ steps do
    Draw trajectories $\tau^{\text{tar}} = \{(o^{\text{tar}}_t, a^{\text{tar}}_t, r^{\text{tar}}_t)\}_{t=1}^{T} \sim \mathcal{B}^{\text{tar}}$ and $o^{\text{src}}_{1:T} \sim \mathcal{B}^{\text{src}}$.
    Train the low-level trunk net using Objective (3) on $\tau^{\text{tar}}$.
    Predict latent behaviors $\bar{a}^{\text{tar}}_{1:T-1} \sim \text{BAN}(o^{\text{tar}}_{1:T})$ and $\bar{a}^{\text{src}}_{1:T-1} \sim \text{BAN}(o^{\text{src}}_{1:T})$.
    Train the high-level plan net using Objective (4) on $(o^{\text{tar}}_{1:T}, \bar{a}^{\text{tar}}_{1:T-1})$ and $(o^{\text{src}}_{1:T}, \bar{a}^{\text{src}}_{1:T-1})$.
    Train the behavior cloning network $F_{\text{BC}}$ using $\bar{a}^{\text{tar}}_{1:T-1}$ and $\bar{a}^{\text{src}}_{1:T-1}$.
  end for
end while
// Model-based policy learning
while not converged do
  Draw $\{(o^{\text{tar}}_t, a^{\text{tar}}_t, r^{\text{tar}}_t)\}_{t=1}^{T} \sim \mathcal{B}^{\text{tar}}$.
  for each $t$ do
    Generate $\{(s_t, a_t, r_t)\}_t^{t+L}$ using $\pi_\phi$ and the low-level trunk net.
    Generate $\{(\bar{s}_t, \bar{a}_t)\}_t^{t+L}$ using $F_{\text{BC}}$ and the high-level plan net.
    Compute the intrinsic rewards $\{\bar{r}_t\}_t^{t+L}$ using Eq. (5).
    Update $v_\psi$ and $\pi_\phi$ using Objective (7).
  end for
end while
```

B. Implementation Details

The model configurations and hyperparameter settings are detailed in Table 2.

C. Benchmark Details

We evaluate the performance of VeoRL in the following three visual RL environments:

- **Meta-World** (Yu et al., 2019): The Meta-World benchmark simulates 50 manipulation tasks with complex visual dynamics, all executed by the same robotic arm. These tasks involve a variety of actions such as reaching, pushing, and grasping, implemented with the MuJoCo physics engine (Todorov et al., 2012). The action space is represented as a 2-tuple: the 3D movement of the end-effector and a normalized torque for the gripper fingers, with values ranging from -1 to 1. We collect six offline datasets from the tasks *button press*, *drawer open*, *handle pull*, *handle press*, *plate slide*, and *coffee push*. Each dataset consists of 200 trajectories, each with 500 time steps.

- **CARLA** (Dosovitskiy et al., 2017): CARLA provides realistic visual observations for autonomous driving research, including varied weather conditions, road layouts, and traffic signs. The agent controls the vehicle using steering, acceleration, and braking commands. The goal is to maximize driving distance over 1,000 time steps while avoiding

Table 2. An overview of layers and hyperparameters used in VeoRL in the three environments.

Item	Meta-World	CARLA	MineDojo
World Model			
Image encoder	Conv3-32	Conv3-32	Conv3-96
GRU hidden size	200	200	4096
RSSM number of units	200	200	1024
Stochastic latent dimension	50	50	32
Discrete latent classes	0	0	32
Weighting factor α	1	1	1
World model learning rate	$3 \cdot 10^{-4}$	$3 \cdot 10^{-4}$	$1 \cdot 10^{-4}$
BAN's update iterations	40K	40K	30K
Behavior Learning			
Imagination horizon L	15	15	15
λ-target	0.95	0.95	0.95
Discount γ in Eq. (6)	0.99	0.99	0.99
ω in Eq. (6)	0.05	0.1	$1 \cdot 10^{-5}$
ρ in Eq. (7)	0	0	1
η in Eq. (7)	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$3 \cdot 10^{-4}$
MLP number of Policy network	4	4	5
MLP number of Value network	3	3	5
Policy network learning rate	$8 \cdot 10^{-5}$	$8 \cdot 10^{-5}$	$3 \cdot 10^{-5}$
Value network learning rate	$8 \cdot 10^{-5}$	$8 \cdot 10^{-5}$	$3 \cdot 10^{-5}$
Environment Setting			
Time limit	500	1000	1000
Action repeat	1	4	1
Image size	64×64	64×64	64×64

collisions with 30 vehicles or barriers. The reward function incorporates velocity, collision impact, and steering effort, encouraging safe and efficient driving. The offline dataset includes approximately 1,000 episodes.

- **MineDojo** (Fan et al., 2022): This platform provides convenient APIs built on Minecraft, standardizing task specifications, world settings, and observation and action spaces for agents. In our study, we focus on three challenging tasks: *harvest log in plains*, *harvest water with bucket*, and *harvest sand*. We set the maximum time steps of each episode as 1,000 and adopt a binary reward indicating task completion, supplemented by the MineCLIP reward (Fan et al., 2022).

For the auxiliary video datasets, we select two datasets, BridgeData-V2 and NuScenes, as source domains due to their relevance to Meta-World and CARLA, as well as publicly available Minecraft videos for MineDojo:

- **BridgeData-V2** (Walke et al., 2023): This is a large and diverse robot manipulation dataset featuring 24 environments and 60,096 trajectories. Most trajectories are video recordings from the Toy Kitchen and Toy Tabletops environments. For our experiments, we use trajectories longer than 20 steps as the source domain data for Meta-World, resulting in approximately 50,000 trajectories.

- **NuScenes** (Caesar et al., 2019): This large-scale autonomous driving dataset contains 1,000 diverse scenes with detailed 3D object annotations. For our experiments, we use only the video recordings, excluding annotations, as the source domain data for tasks in the CARLA environment. In total, we collect 850 episodes with varying time step lengths.

- **Minecraft videos:** For the source domain of MineDojo, we use publicly available videos of humans playing Minecraft. We collect a dataset of 5,000 episodes with varying lengths.

Table 3. **Performance on Meta-World robotic manipulation tasks in success rate.** We compute the mean and standard deviation of 50 episodes over 3 random training seeds.

Methods	DrQ + CQL	DreamerV2	APV	VIP	LOMPO	Ours
Drawer Open	0 ± 0	0.18 ± 0.04	0.02 ± 0.05	0.02 ± 0.04	0 ± 0	**0.70 ± 0.07**
Handle Press	0.60 ± 0.48	0.33 ± 0.11	0.10 ± 0.09	0.56 ± 0.17	0 ± 0	**0.60 ± 0.12**
Handle Pull	0 ± 0	0.04 ± 0.01	0 ± 0	**0.36 ± 0.15**	0 ± 0	0.30 ± 0.10
Plate Slide	0.23 ± 0.41	0.48 ± 0.02	0 ± 0	0.34 ± 0.12	0 ± 0	**0.67 ± 0.03**
Coffee Push	0 ± 0	0.29 ± 0.01	0.12 ± 0.11	0.16 ± 0.15	0 ± 0	**0.31 ± 0.05**
Button Press	0 ± 0	0.58 ± 0.08	0.01 ± 0.01	0.18 ± 0.08	0 ± 0	**0.62 ± 0.02**
Average	0.14	0.32	0.04	0.27	0	**0.53**

Table 4. **The ablation study of the quality of source videos on Meta-World (Handle Press).** As the number of source domain videos increases, the model's performance improves accordingly.

Handle Press	All videos	1/2 videos	1/4 videos	DreamerV2
Episode return	2651 ± 620	2477 ± 441	1859 ± 423	1202 ± 422

D. Compared Methods

We compare VeoRL against existing model-based and model-free offline RL methods:

- **CQL** (Kumar et al., 2020): A model-free method that conservatively learns value functions to address value overestimation. Following (Fu et al., 2020), we integrate CQL regularizers into the DrQ-V2 algorithm (Yarats et al., 2021a) to adapt it for pixel-based inputs.

- **DreamerV2** (Hafner et al., 2019): A model-based RL approach that learns policies directly from latent states within the world model. We adapt it to the offline setting and refer to this version as Offline DV2.

- **APV** (Seo et al., 2022): This method builds an action-conditional RSSM model on top of an action-free RSSM model pretrained on video datasets. We adapt APV to the offline scenario for a fair comparison.

- **VIP** (Ma et al., 2023): A pretraining approach that focuses on learning universal visual representations from video datasets, enabling generalization to downstream offline tasks. For this comparison, we pair VIP with the AWR algorithm (Peng et al., 2019) for offline policy learning.

- **LOMPO** (Rafailov et al., 2021): A model-based offline visual RL method that addresses model uncertainty in the latent space while incorporating explicit reward estimation.

- **VPT** (Baker et al., 2022): A foundational model for Minecraft trained using standard behavior cloning. VPT uses an inverse dynamics model trained on limited labeled data to annotate a vast set of unlabeled online videos. For task-specific performance, we finetune VPT on the offline dataset using PPO (Schulman et al., 2017), following the methodology outlined in DECKARD (Nottingham et al., 2023).

E. Further Results of Performance Comparison

Table 3 provides a detailed comparison of VeoRL and existing methods on the Meta-World benchmark, evaluated by success rate. VeoRL outperforms DreamerV2 with a remarkable advantage in *Drawer Open* (0.18 → 0.70), *Handle Pull* (0.04 → 0.30) and *Handle Press* (0.33 → 0.60) in terms of success rate.

F. Analyses of the Quality of Source Videos and Target Domain Trajectories

To partially investigate the impact of using fewer high-quality videos, we have conducted an ablation study analyzing how the quantity of unlabeled source videos affects policy performance. The results are summarized in Table 4. As the number of source domain videos increases, the model's performance improves accordingly, demonstrating that our method effectively leverages information from source domain videos and exhibits strong scalability. Even when using only a quarter of the videos, VeoRL's results still significantly outperform DreamerV2 (which does not utilize any source domain videos for training), indicating that our method can effectively extract useful skills from the videos.

Our framework trains the Plan Net and the corresponding BC model using data from both the source domain (BridgeData) and the target domain (Meta-World). This design leverages the diversity of BridgeData to enable the BC model to infer high-level behavioral abstractions (*e.g.*, "reach" or "grasp") that are applicable to the target task (*e.g.*, "Button press"), even when the offline data is of low quality and lacks successful trajectories. To validate this, we conduct a new experiment in which the offline Meta-World data consists of random trajectories, potentially containing extremely poor demonstrations. As shown in Table 5, the results indicate that our method could still benefit from the latent skills (estimated by BC) from the auxiliary videos, demonstrating the robustness of our approach in challenging offline RL settings.

Table 5. **The ablation study of the quality of target domain trajectories on Meta-World (Button Press) in episode return.** It demonstrates the robustness of our approach in challenging offline RL settings.

Button Press	Medium offline data	Random offline data
Ours	850 ± 169	638 ± 150
DreamerV2	765 ± 120	505 ± 150
VIP	545 ± 94	276 ± 99

G. Hyperparameter Analysis of Latent Action Space

Figure 9 provides the results of varying the number of discrete latent behaviors in the Meta-World (*Handle Press*) and CARLA benchmarks. We observe that setting $num = 50$ consistently yields the best performance across all tested environments, striking a good balance between the complexity of the learned latent action space and its stability of policy learning. Furthermore, VeoRL demonstrates robust performance across varying numbers of latent behaviors, compared with DreamerV2 without training with natural videos from the BridgeData-V2 and NuScenes datasets (dashed lines).

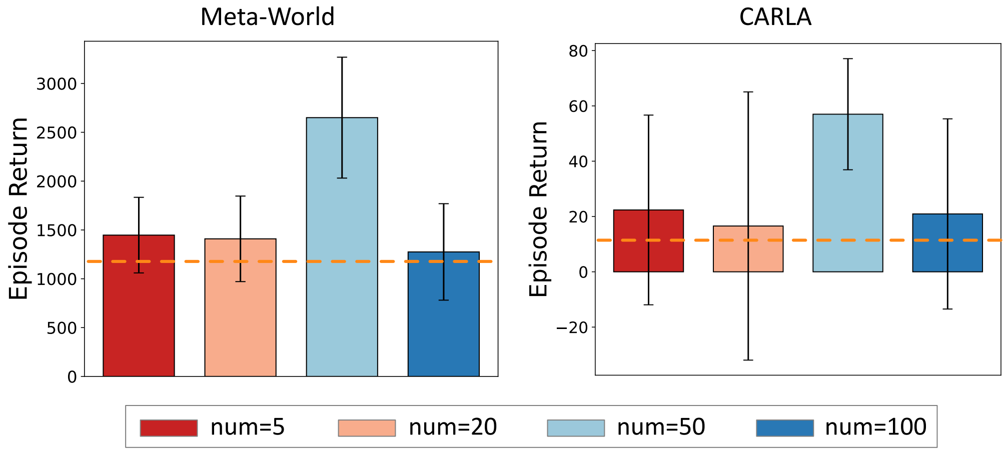

Figure 9. **Hyperparameter analyses of the dimension of the latent action space.** We observe that (i) VeoRL demonstrates robust performance across varying numbers of latent actions, compared with DreamerV2 without training with natural videos from the BridgeData-V2 and NuScenes datasets; (ii) A setting of 50 latent actions consistently yields the best performance across different tasks.

H. Generalizability across Tasks

We conduct the experiments on Meta-World and MineDojo to validate the transferability of latent behaviors across tasks. In the training phase, the behavior abstraction network is frozen after training on Task A, with a fixed latent codebook. For Task B, this pre-trained network is directly deployed without undergoing task-specific fine-tuning on source video data. As presented in Table 6, the minimal performance degradation confirms that our method's latent behaviors trained on Task A remain effective for Task B, consistently outperforming other baselines trained specifically on Task B. This transferability significantly reduces training costs for downstream tasks, as it spares the time for re-training on the auxiliary videos.

Table 6. **The study of the generalization ability of the latent behaviors across tasks on Meta-World and MineDojo.** We compute the mean and standard deviation over 3 random training seeds.

Methods	Codebook construction	Downstream task	Success rate
	Meta-World		
VeoRL	Button Press	Button Press	0.62 ± 0.02
VeoRL	Handle Press	Button Press	0.63 ± 0.12
DreamerV2	N/A	Button Press	0.58 ± 0.08
VIP	N/A	Button Press	0.18 ± 0.08
	MineDojo		
VeoRL	Harvest sand	Harvest sand	0.55 ± 0.05
VeoRL	Harvest water with bucket	Harvest sand	0.50 ± 0.10
DreamerV2	N/A	Harvest sand	0.25 ± 0.06
VPT	N/A	Harvest sand	0.20 ± 0.04

I. Analyses of MMD in Domain Adaptation

We performed an ablation study in the Meta-World environment to evaluate the role of MMD loss. To quantify its impact, we compared performance under conditions with and without MMD loss, accompanied by feature distribution visualizations derived from principal component analysis (PCA). The results in Table 7 validate that MMD loss critically bridges domain discrepancies. By aligning feature distributions, it enables the target domain to effectively transfer and adapt knowledge from the source domain, leading to a 33% improvement in success rates (0.60 vs. 0.45). Furthermore, as shown in the Figure 10, incorporating MMD significantly aligns the feature distributions between the source and target domains. This alignment correlates with policy generalization.

Table 7. **The ablation study on the MMD loss in the Meta-World environment.**

Handle Press	w/ MMD loss	w/o MMD loss	DreamerV2
Episode return	2651 ± 620	1961 ± 585	1202 ± 422
Success rate	0.60 ± 0.12	0.45 ± 0.15	0.33 ± 0.11

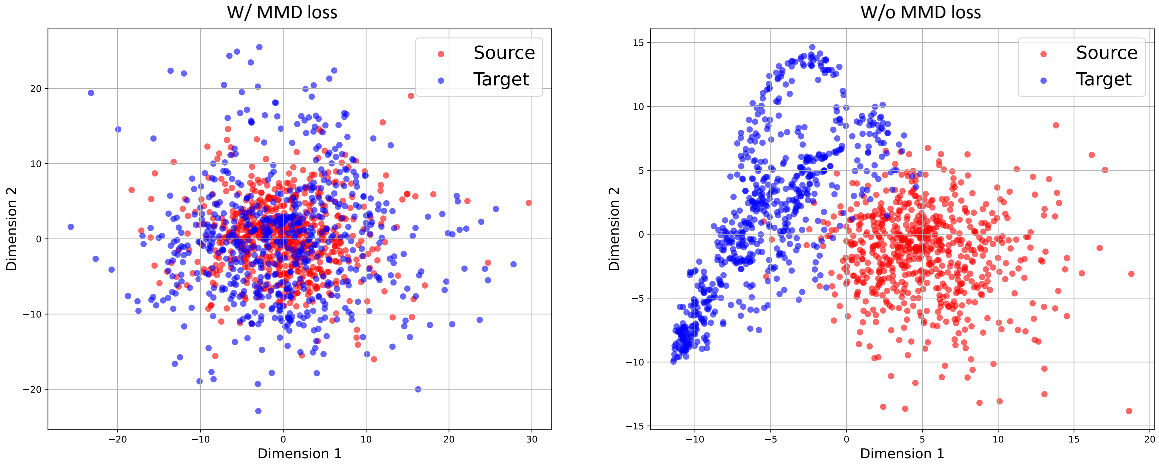

Figure 10. **Distribution visualization using PCA.** We observe that incorporating MMD significantly aligns the feature distributions between the source and target domains.