FUTURE REPRESENTATION DISTILLATION FOR WORLD ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

World Action Models (WAM) use the predictive knowledge of pretrained video diffusion models for robot control through future video prediction. However, iterative video denoising is costly, and inaccurate visual predictions can compromise action prediction. Our key idea is to make this predictive knowledge available through visual representations computed from the current observation, enabling action prediction without generating future videos. We introduce DREAMFREE, a WAM that repurposes a pretrained video diffusion model as a predictive visual encoder through representation distillation. The visual encoder learns to predict a frozen video diffusion model's representation using only the current observation. An action module is then trained to predict actions by extracting action-relevant information from the visual encoder. Across real-world and simulated tasks, DREAMFREE achieves higher success rates than state-of-the-art VLAs and WAMs, while performing real-time closed-loop control at 11Hz with a 14B video diffusion backbone. Experiments show that DREAMFREE can translate stronger video priors and additional video data into better robot control and enable learning new robot tasks from video demonstrations, including videos from a different embodiment, without action annotation for the new tasks.

1 INTRODUCTION

Pretrained video diffusion models offer a promising foundation for robot control (Gao et al., 2026; Kim et al., 2026; Zhu et al., 2025). Through large-scale video pretraining, these models learn to predict how physical environments evolve over time, providing *predictive* knowledge that can guide robot actions. World Action Models (WAMs; Ye et al. 2026b) use this knowledge by adapting video models to predict visual futures and actions, demonstrating strong manipulation performance and generalization to new tasks and environments (Kim et al., 2026; Ye et al., 2026b).

A common approach in WAMs is to use this predictive knowledge of pretrained video diffusion models through future video prediction, either by jointly predicting visual futures and actions (Kim et al., 2026; Ye et al., 2026b) or by decoding actions from predicted visual futures (Hu et al., 2025; Pai et al., 2025). These methods leverage the pretrained models' knowledge through their native video prediction process. However, iterative video denoising incurs substantial inference cost, and errors in the predicted visual futures can lead to inaccurate actions. Moreover, it remains unclear whether action prediction requires the same level of visual detail as video generation. These considerations raise the question of how to effectively use the predictive knowledge of pretrained video diffusion models for action prediction.

We hypothesize that this predictive knowledge can be effectively captured in visual representations to support action prediction, without iterative video denoising. Our key idea is to distill the video model's predictive knowledge into the visual representations computed from the current observation. We train a visual encoder to anticipate visual futures from the observation by using the representations extracted from a pretrained video diffusion model. Our analysis shows that distillation enables the encoder to effectively capture future visual information from the current observation and make its features more useful for action prediction. The resulting predictive representations enable action prediction without generating future videos.

Based on this finding, we introduce DREAMFREE, a WAM that repurposes a pretrained video diffusion backbone as a predictive visual encoder through representation distillation. We train the

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

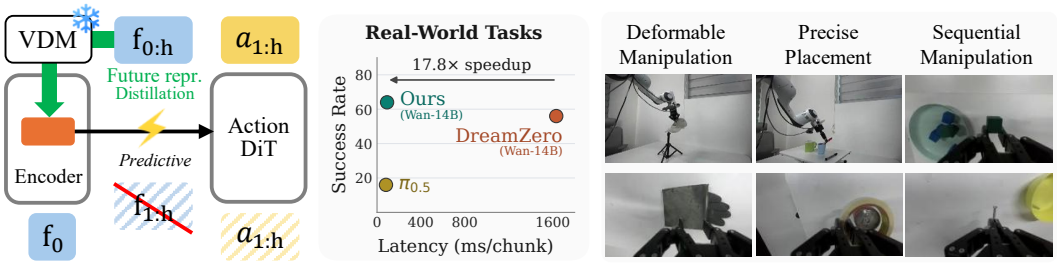


Figure 1: **Overview**. DREAMFREE distills predictive knowledge from a frozen video diffusion model into a predictive visual encoder, enabling action prediction from the current observation without generating future videos. In real-world evaluations, DREAMFREE achieves higher success than DreamZero and $\pi_{0.5}$ with latency comparable to that of $\pi_{0.5}$ and a 17.8× inference speedup over DreamZero. DREAMFREE generalizes across diverse real-world manipulation tasks.

model in two stages, separating predictive representation learning from action learning. In the first stage, a frozen video diffusion model extracts representations from both current and future frames of a video demonstration, and the visual encoder learns to predict these representations from the current observation. The encoder thus learns to anticipate scene evolution in feature space, making the video model's predictive knowledge available in a single forward pass without future video generation. This stage requires no action labels, allowing the visual encoder to learn from any in-the-wild video demonstrations. In the second stage, we train an action module on robot demonstrations to map these visual features to executable actions, with the visual encoder either frozen or jointly fine-tuned.

We evaluate DREAMFREE on diverse manipulation tasks in both simulation and real-world environments (Khazatsky et al., 2024), demonstrating stronger performance compared to state-of-the-art VLA and WAM baselines (Physical Intelligence, 2025; Ye et al., 2026b). In real-world evaluations, DREAMFREE succeeds on tasks involving object deformation or successive rearrangements where video-generating WAM fails (Figure 1). These successes suggest a benefit of using predictive visual representations for control without relying on potentially inaccurate video predictions. Experiments show that using a larger video backbone and more pretraining video data improve success, demonstrating DREAMFREE can translate stronger video priors and additional video data into better robot control without additional action annotation. We additionally show task transfer from video demonstrations, including videos from a different embodiment, showing that DREAMFREE can learn new tasks from videos without collecting action annotations for those tasks.

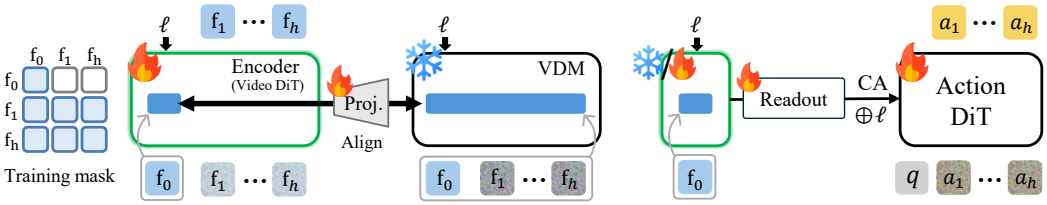
In addition to strong performance, DREAMFREE offers a practical advantage for real-time control. During deployment, the policy obtains predictive visual representations in a single encoder pass and uses them to generate an action chunk without iterative video denoising. DREAMFREE using a 14B video backbone achieves real-time closed-loop control at 11Hz, achieving a 17.8× inference speedup over a video-generating WAM with the same video backbone size and latency comparable to that of a VLA (Figure 1, middle).

We highlight our main contributions below:

- We introduce DREAMFREE, a WAM that repurposes a pretrained video diffusion backbone as a predictive visual encoder by distilling future information from a frozen video diffusion model.
- We outperform state-of-the-art VLA and WAM baselines in real-world manipulation without task-specific adaptation, while achieving real-time closed-loop control at 11Hz.
- We show that DREAMFREE translates stronger video priors and additional video data into better robot control.
- We demonstrate that DREAMFREE can learn new tasks from video demonstrations, including videos from a different embodiment, without action annotations for the new tasks.

2 RELATED WORK

Vision-language-action policies Vision-Language-Action (VLA) policies adapt pretrained vision-language models to map visual observations and language instructions directly to robot actions (Bjorck



(a) Future Representation Distillation (b) Action DiT Training

Figure 2: **Model architecture of DREAMFREE**. Training is separated into two stages: (a) In the first stage, a pretrained video diffusion backbone is repurposed as a visual encoder through video generative training with a structured attention mask and distillation of predictive representations from the frozen video diffusion model. (b) In the second stage, the action readout module and action decoder are trained to predict actions by extracting action-relevant information from the encoder.

et al., 2025; Black et al., 2025; Brohan et al., 2023; Kim et al., 2024; Physical Intelligence, 2025). While their backbones provide semantic and linguistic priors, pretraining on static image-text pairs gives them limited knowledge of the spatiotemporal dynamics important for robot control.

Video model-based robot policies Video model-based robot policies build on pretrained video diffusion models by coupling future visual prediction with action prediction. Some decode actions from video representations through inverse dynamics (Hu et al., 2025; Pai et al., 2025; Rhoda AI, 2026), while others jointly denoise future video and action sequences (Bi et al., 2026; Kim et al., 2026; Li et al., 2026; Ma et al., 2026; Ye et al., 2026b). These World Action Models (WAMs) transfer spatiotemporal video priors to control, achieving strong action-data efficiency and generalization. However, generating future video at deployment introduces substantial inference cost, and errors in the predicted video can lead to inaccurate action prediction. Recent works retain video prediction during training while enabling action prediction without future video generation at deployment (Ye et al., 2026a; Yuan et al., 2026), yet how faithfully their representations preserve the video model's predictive knowledge remains unclear.

Predictive visual representation learning Predictive visual learning methods train encoders to capture scene structure and dynamics by predicting representations of unobserved visual content (Assran et al., 2025). Related predictive supervision is used in VLA policies, where future images or visual features serve as auxiliary training targets alongside action supervision (Wu et al., 2024; Zhang et al., 2026; Zhao et al., 2025). For example, FLARE (Zheng et al., 2025) aligns future tokens in the action expert with vision-language embeddings of future observations. Our approach learns predictive visual representations from the current observation by distilling features from a frozen video diffusion model into a visual encoder.

3 DREAMFREE

As shown in Figure 2, DREAMFREE consists of a predictive visual encoder followed by an action module. The training stage of DREAMFREE is separated into two stages. First, the predictive visual encoder is trained to predict the representations from the frozen video diffusion model (Section 3.1). The frozen video diffusion model extracts representations from both current and future frames of a video demonstration, while the visual encoder conditions on the current observation only. Second, the action module is trained to predict actions by extracting action-relevant information from the predictive visual encoder through a readout module (Section 3.2). During inference, only a single forward pass on the predictive encoder is required to extract visual features relevant for action prediction, enabling real-time closed-loop control up to 11Hz.

3.1 LEARNING A PREDICTIVE VISUAL ENCODER

Predictive visual encoder We initialize the encoder from a pretrained video diffusion backbone and use the hidden states of its current-frame latent tokens as the visual representation. We apply a structured mask to the backbone's bidirectional video self-attention (Figure 2(a) left), restricting current-frame tokens to attend only to each other within self-attention. This ensures that the current-

frame hidden states depend only on the current observation and instruction. During training, the backbone receives a clean current frame and noisy future frames, as shown in Figure 2(a).

Future representation distillation The main idea is to distill the representation of a frozen video diffusion model into the visual encoder that observes only the current observation and instruction. Motivated by REPA (Yu et al., 2025), we introduce *future representation distillation*, which aligns the projection of the encoder's current frame representation with a video diffusion model representation as shown in Figure 2(a).

Formally, let $\boldsymbol{u}_\theta$ be the predictive visual encoder, $\boldsymbol{g}$ be the pretrained video diffusion model, and $\boldsymbol{p}_\psi$ be the trainable projector. Consider a clean video $\boldsymbol{x}_*$ with the first frame $\boldsymbol{x}_*^0$ and language instruction $\boldsymbol{\ell}$, and let $\boldsymbol{h}_i$ be the i-th layer hidden representation of $\boldsymbol{u}_\theta(\boldsymbol{x}_*^0, \boldsymbol{\ell})$. We use $\boldsymbol{g}$ as the teacher and take its hidden representation at layer j, computed from a perturbed latent of $\boldsymbol{x}_*$, as the distillation target. We use a lightweight Transformer decoder for the projector $\boldsymbol{p}_\psi$ that maps the current-frame representation $\boldsymbol{h}_i$ to the spatiotemporal token layout and feature dimension of the teacher's full-video representation. Note that $\boldsymbol{g}$ and $\boldsymbol{p}_\psi$ are used only during training and discarded at deployment.

Future representation distillation is achieved by maximizing the similarity between the current-only encoder representation and the teacher representation as follows:

$$\mathcal{L}_{\text{distill}}(\theta, \psi) \coloneqq \mathbb{E}_{\boldsymbol{x}_*, \boldsymbol{\epsilon}_d}\big[-\operatorname{sim}\big(\boldsymbol{p}_\psi(\boldsymbol{h}_i), \boldsymbol{g}_j(\boldsymbol{x}_{t_d}, t_d, \boldsymbol{\ell})\big)\big], \tag{1}$$

where $\boldsymbol{\epsilon}_d$ is Gaussian noise, $t_d \in [0, 1]$ is a fixed timestep controlling the teacher input's noise level, $\boldsymbol{x}_{t_d} = (1 - t_d)\boldsymbol{x}_* + t_d\boldsymbol{\epsilon}_d$, and $\operatorname{sim}(\cdot, \cdot)$ is a pre-defined similarity function.

To preserve the predictive capability inherited from the pretrained video backbone, we also use the original generative flow matching objective as a regularization:

$$\mathcal{L}_{\text{gen}}(\theta) \coloneqq \mathbb{E}_{\boldsymbol{x}_*, \boldsymbol{\epsilon}, t}\Big[\big\| \boldsymbol{u}_\theta(\boldsymbol{x}_t, t, \boldsymbol{\ell}) - (\boldsymbol{\epsilon} - \boldsymbol{x}_*) \big\|_2^2 \Big], \tag{2}$$

where $\boldsymbol{u}_\theta$ is conditioned on clean current-frame latent tokens with timestep 0, while the future-frame latent tokens in x_t are perturbed with Gaussian noise $\boldsymbol{\epsilon}$ at timestep t.

The encoder is trained with both the native generative loss and the distillation loss as follows:

$$\mathcal{L}(\theta, \psi) \coloneqq \mathcal{L}_{\text{gen}}(\theta) + \mathcal{L}_{\text{distill}}(\theta, \psi). \tag{3}$$

3.2 Learning actions from visual representations

After training the predictive visual encoder, we use action-labeled robot demonstrations to train the action module, which consists of a readout module, state and action encoders, and an action decoder. Given the observation and instruction, the visual encoder produces the predictive representation that serves as a visual prior for the action module. Then the readout module extracts action-relevant features from the representation, and the action module conditions on the action-relevant features together with the instruction and proprioception to predict the action chunk, as illustrated in Figure 2(b).

Action readout module Given observation $\boldsymbol{o}$ and instruction $\boldsymbol{\ell}$, we extract the hidden representation $\boldsymbol{h}_k$ from the visual encoder, where k denotes the layer used for action readout. We transform these visual tokens into a fixed number of action-facing tokens using a latent-query readout inspired by Perceiver (Jaegle et al., 2021). The trainable readout $\boldsymbol{r}_\omega$ updates learned queries through self-attention and cross-attention, with $\boldsymbol{h}_k$ serving as keys and values. The resulting tokens $\boldsymbol{c} = \boldsymbol{r}_\omega(\boldsymbol{h}_k)$ provide the action decoder with action-relevant predictive visual representations.

Action decoder Conditioned on the action-facing tokens $\boldsymbol{c}$, instruction $\boldsymbol{\ell}$, proprioceptive state $\boldsymbol{q}$, and optional context $\boldsymbol{s}$, the action decoder predicts the action chunk. We instantiate the action decoder $\boldsymbol{u}_\phi$ with a cross-attention-conditioned video Diffusion Transformer (DiT) (Peebles & Xie, 2022).

Given an action chunk $\boldsymbol{a}_*$, the action decoder is trained with a flow matching objective:

$$\mathcal{L}_{\text{act}} \coloneqq \mathbb{E}_{\boldsymbol{a}_*, \boldsymbol{\epsilon}, t}\Big[\big\| \boldsymbol{u}_\phi(\boldsymbol{a}_t, t \,|\boldsymbol{c}, \boldsymbol{q}, \boldsymbol{s}, \boldsymbol{\ell}) - (\boldsymbol{\epsilon} - \boldsymbol{a}_*) \big\|_2^2 \Big], \tag{4}$$

where $\boldsymbol{\epsilon}$ is a Gaussian noise, $t \in [0, 1]$, and $\boldsymbol{a}_t = (1 - t)\boldsymbol{a}_* + t\boldsymbol{\epsilon}$.

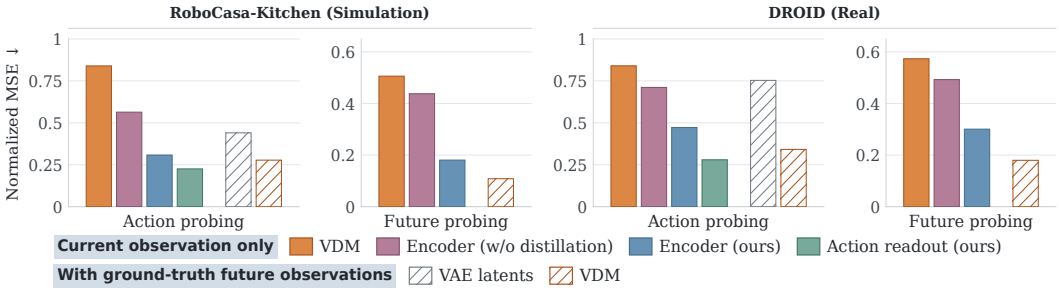


Figure 3: **Representation analysis**. We evaluate visual representations by measuring how accurately action sequences (Action probing) and future-frame VAE latents (Future probing) can be decoded from them. Future representation distillation makes the encoder's features more useful for action prediction and improves their ability to capture future visual information from the current observation.

Inference Without Iterative Video Denoising During inference, the predictive visual encoder processes the observation and instruction in a single forward pass. The resulting predictive visual representation is transformed by the readout into action-facing tokens, and the action decoder conditions on these to generate an action chunk using N denoising steps. This enables real-time closed-loop control up to 11Hz even for a 14B video backbone.

4 Experiments

We validate the performance of DreamFree in both simulation and real-world environments. In particular, we investigate the following questions:

- **(Q1)** Does distillation make visual features more useful for action prediction?
- **(Q2)** Do representations learned by DreamFree translate into strong manipulation performance?
- **(Q3)** Can DreamFree translate stronger video priors and additional video data into better control?
- **(Q4)** How effectively and efficiently does DreamFree perform real-world manipulation?
- **(Q5)** Can DreamFree learn new tasks from video demonstrations without action annotations?

4.1 Experimental setup

Implementation details We use pretrained Wan2.2-5B and Wan2.2-A14B high-noise expert (Wan et al., 2025) as the video backbone, including the video DiT, text encoder, and video VAE. The action decoder is a 1B DiT. For all experiments, we set the action horizon to 32. Following Fast-WAM (Yuan et al., 2026), we temporally downsample the videos by 4×, producing 9 video frames per chunk. DreamFree is trained on each dataset without cross-embodiment pretraining. We provide more details in the Appendix A.

Benchmarks We evaluate our method on two simulation benchmarks and one real-world evaluation setup. For **LIBERO** (Liu et al., 2023), we follow the standard benchmarking protocol, using four task suites (Spatial, Object, Goal, Long). For **RoboCasa-Kitchen** (Nasiriany et al., 2024), we follow the 24-task benchmark protocol of Kim et al. (2026) and train on their released dataset, which contains 50 demos per task. The tasks span object placement and manipulation of kitchen fixtures and appliances. For **real-world evaluation**, we use the DROID (Khazatsky et al., 2024) setup, evaluating on 10 tasks selected from the pretraining distribution spanning diverse capabilities, including push, cover, press, fold, pour, and composite skills (see Appendix C.2). We measure the task progress and success rate using 5 rollouts per task. We provide task descriptions and evaluation details in Appendix C.2.

Baselines On simulation benchmarks, we compare DreamFree against state-of-the-art VLAs (Bjorck et al., 2025; Black et al., 2025; Kim et al., 2025; Physical Intelligence, 2025; Zheng et al., 2025) and WAMs (Bi et al., 2026; Kim et al., 2026; Li et al., 2026; Yuan et al., 2026). For real-world DROID evaluation, we compare with state-of-the-art VLA $\pi_{0.5}$ and WAM DreamZero (Ye et al., 2026b) with their publicly released checkpoints, namely $\pi_{0.5}$-DROID and DreamZero-DROID, following their inference setup.

Table 1: **Average success rate (%) across the four LIBERO task suites.**

Method	Avg. SR
VLAs	
π_0 (Black et al., 2025)	94.1
$\pi_{0.5}$ (Physical Intelligence, 2025)	96.9
OpenVLA-OFT (Kim et al., 2025)	97.1
WAMs	
Motus (Bi et al., 2026)	97.7
LingBot-VA (Li et al., 2026)	98.5
Fast-WAM (Yuan et al., 2026)	97.6
DREAMFREE (Ours)	**98.6**

Table 2: **Average success rate (%) across 24 RoboCasa-Kitchen tasks.**

Method	Avg. SR
π_0 (Black et al., 2025)	62.5
GR00T-N1.5 (Bjorck et al., 2025)	64.1
FLARE (Zheng et al., 2025)	66.4
Cosmos Policy (Kim et al., 2026)	67.1
Fast-WAM (Yuan et al., 2026)	44.4
Fast-WAM 14B (Yuan et al., 2026)	44.7
DREAMFREE (Ours)	
5B video backbone	58.5
14B video backbone	**71.7**
5B video backbone + extra videos	68.8
14B video backbone + extra videos	**74.4**

4.2 Q1: REPRESENTATION ANALYSIS

We examine whether future representation distillation makes the visual encoder's representation more useful for action prediction. We compare features from three models given only the current observation: the pretrained video diffusion model (VDM), a variant of the DREAMFREE visual encoder trained without distillation, and the visual encoder of DREAMFREE. As a reference with direct access to future observations, we also evaluate VDM representations computed from both the current frame and ground-truth future frames. The VDM and encoders use the 14B video backbone and are conditioned on the language instructions. Each encoder is trained separately on Robocasa-Kitchen and DROID datasets. For each representation and input setting, we train a separate probe while keeping the feature extractor frozen. We report normalized mean squared error in Figure 3.

Action probing To assess the usefulness of the features for control, we train a lightweight attention-based probe to predict action sequences from each representation, using paired robot demonstration videos and actions. As an additional reference, we also probe VAE latents of both the current and ground-truth future frames to compare the encoder representations with direct encodings of future observations. Figure 3 shows that DREAMFREE encoder achieves lower action prediction error than the VDM using only the current observation and the encoder variant trained without distillation. These results show that distillation makes the encoder's features more useful for action prediction.

Despite using only the current observation, DREAMFREE encoder achieves lower action prediction error than VAE latents of current and future frames, and approaches the accuracy of the VDM using the future frames. This indicates that DREAMFREE encoder learns features that support action prediction nearly as accurately as directly observing the future. We additionally probe the features produced by the action readout module of DREAMFREE. These features yield even lower prediction error, demonstrating that the readout effectively extracts action-relevant information from the visual representations.

Future probing To better understand the improvement in action decodability from distillation, we examine whether distillation improves the encoder's ability to capture future visual information from the current observation. We train a spatially shared linear probe to reconstruct the VAE latent patches of ground-truth future frames. Figure 3 shows that DREAMFREE encoder achieves lower reconstruction error than the VDM using only the current observation and the encoder variant trained without distillation. Furthermore, its reconstruction error approaches that of the VDM given future frames on RoboCasa-Kitchen, while a gap remains on DROID. These results show that our encoder effectively captures future visual information from the current observation.

4.3 Q2 & Q3: PERFORMANCE AND SCALING ON SIMULATION BENCHMARKS

We first assess in-distribution task execution of DREAMFREE in the LIBERO benchmark compared to VLAs and WAMs. In Table 1, DREAMFREE achieves performance comparable to state-of-the-art baselines even without cross-embodiment pretraining, supporting the effectiveness of DREAMFREE's predictive visual representations for in-distribution manipulation.

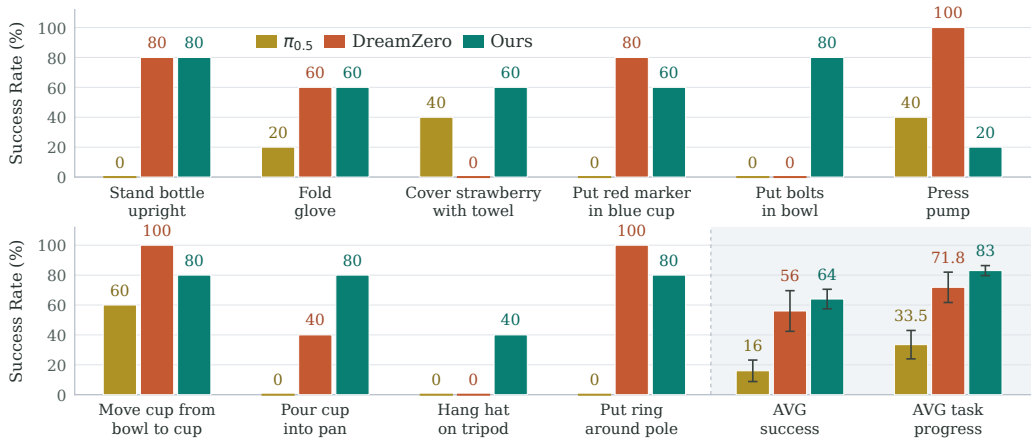
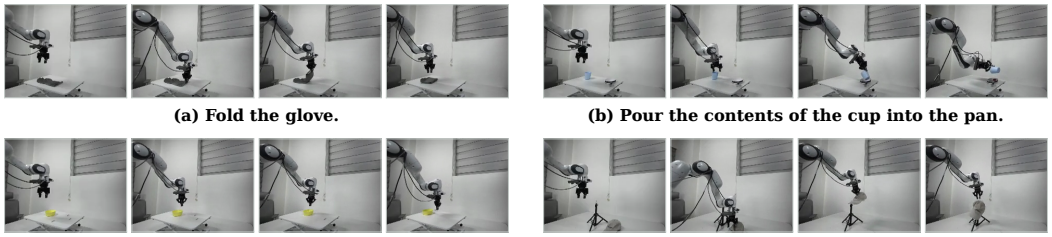


Figure 4: **Real-world evaluation on DROID**. We evaluate policies out of the box on 10 tasks with five trials per task. Task descriptions and the progress scoring criteria are provided in Appendix C.2.



(a) Fold the glove. **(b) Pour the contents of the cup into the pan.**

(c) Put the bolts in the bowl. **(d) Hang the hat on the tripod.**

Figure 5: **Examples of real-world task executions**. We show selected frames from successful rollouts of DREAMFREE on four DROID evaluation tasks.

We next evaluate DREAMFREE on the RoboCasa-Kitchen benchmark to assess multi-task manipulation under object and scene variation, with limited demonstration data. Table 2 shows that DREAMFREE with a 14B video backbone achieves the highest average success rate of 71.7%, demonstrating that our predictive visual representations can translate pretrained video knowledge into strong manipulation performance without generating future videos. Fast-WAM also predicts actions without generating future videos at inference, but underperforms the other baselines with only 44.4% success, suggesting that using video priors without future video generation remains challenging in this setting.

Moreover, we investigate whether DREAMFREE benefits from a larger video backbone and more pretraining video data. Fast-WAM shows little improvement when its backbone is scaled from 5B to 14B, suggesting limited benefit from stronger video priors. In contrast, scaling our video backbone from 5B to 14B improves success rate by 13.2 pp, with particularly large gains on pick-and-place tasks. These gains suggest that stronger video priors provide more useful representations for coordinating grasping, transport, and placement. Also, we additionally use videos from the atomic-task subset of RoboCasa365 (Nasiriany et al., 2026) showing robot demonstrations in simulated kitchen environments. Training the visual encoder with these videos, without their action labels, further improves success. Together, these results show that DREAMFREE can translate stronger video priors and additional video data into better robot control, without additional action annotations.

To assess how each visual encoder training objective contributes to performance, we ablate the generative objective $\mathcal{L}_{\text{gen}}$ and distillation objective $\mathcal{L}_{\text{distill}}$. As shown in Table 3, removing the distillation $\mathcal{L}_{\text{distill}}$ leads to a significant drop in the success rate for both datasets. Removing the generative objective $\mathcal{L}_{\text{gen}}$ also degrades the performance, although the drops are smaller compared to removing the distillation. These results indicate that both objectives contribute to learning predictive visual representations, with distillation playing a particularly important role in translating pretrained video knowledge into strong manipulation performance. This is consistent with the representation analysis in Section 4.2, where removing distillation reduces both action and future decodability.

Table 3: **Ablation of visual encoder training objectives**.

Method	LIBERO	RoboCasa
Ours	98.6	71.7
w/o $\mathcal{L}_{\text{gen}}$	98.0	55.6
w/o $\mathcal{L}_{\text{distill}}$	94.6	47.2

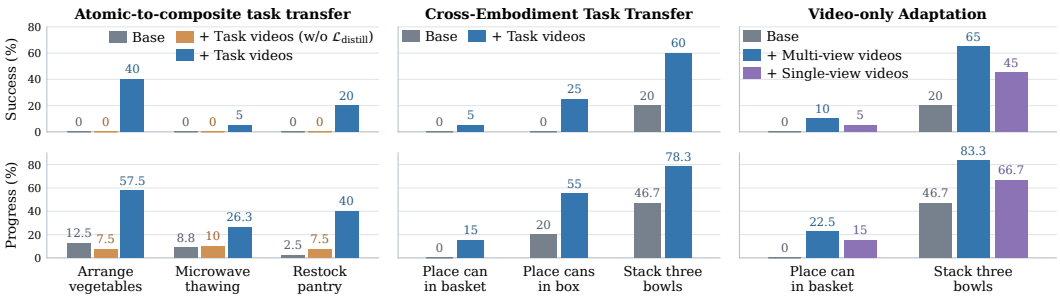


Figure 6: **Task transfer from video demonstrations**. We report success rates and progress over 20 episodes per task. Task descriptions are provided in Appendix C.3.



(a) Microwave thawing **(b) Place can in basket**

Figure 7: **Examples of task execution after transfer from videos**. We show selected frames from successful rollouts of DREAMFREE.

4.4 Q4: REAL-WORLD PERFORMANCE AND EFFICIENCY

To assess real-world manipulation and generalization, we evaluate the models out of the box on 10 task types represented in the pretraining data, using new objects or scene arrangements. Figure 4 reports the task progress and success rates. DREAMFREE achieves the highest average success rate at 64%, compared with 56% for DreamZero and 16% for $\pi_{0.5}$, with successful executions spanning a broader range of task types than either baseline. DREAMFREE also achieves the highest average task progress at 83%, compared with 71.8% for DreamZero and 33.5% for $\pi_{0.5}$.

The gains over DreamZero are notable in tasks involving deformable object manipulation or successive rearrangements, such as cover strawberry with towel, hang hat on tripod, and put bolts in bowl. DreamZero fails in all trials for each of these tasks, with inaccurate video predictions observed in most failures. Qualitatively, DreamZero often persists with unsuccessful actions after an initial error, whereas DREAMFREE often adjusts its actions and recovers. Together with the analysis in Section 4.2, these observations suggest that DREAMFREE benefits from capturing predictive information directly in its visual representations, enabling successful control without relying on potentially inaccurate future video predictions. These results demonstrate that DREAMFREE achieves effective real-world control and generalizes zero-shot to new objects or scene arrangements within seen tasks. We describe the failure cases of DREAMFREE in Appendix D.3.

Latency In Figure 8 (left), we measure the mean inference latency per action chunk on NVIDIA H200 GPUs. Each policy is measured in its deployed configuration. DREAMFREE performs real-time closed-loop control at 11Hz, yielding a 17.8× speedup over DreamZero while achieving a higher average success rate on the real-world tasks. Compared with $\pi_{0.5}$, DREAMFREE achieves 4× the average success rate with 1.1× the latency.

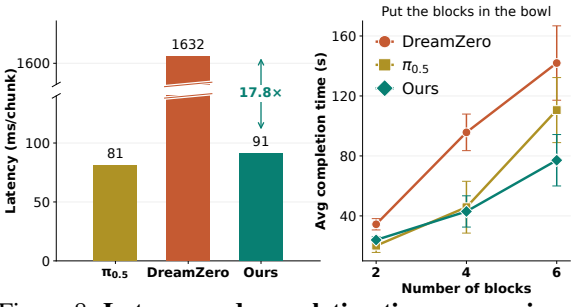


Figure 8: **Latency and completion time comparison**.

Figure 8 (right) further compares the task completion times for placing different numbers of blocks into a bowl. DREAMFREE achieves completion times comparable to those of $\pi_{0.5}$ on simpler settings (2 or 4 blocks), while completing the task much faster in the harder setting (6 blocks). These results highlight the efficiency of using predictive visual representations for action generation without explicit video generation.

4.5 Q5: Task transfer from video demonstrations

We explore task transfer from video demonstrations without action annotations for the new tasks.

Atomic-to-composite task transfer We first investigate whether DreamFree can learn new composite tasks from videos, using action labels only from atomic tasks. Here, atomic tasks involve individual skills while composite tasks combine several of these skills sequentially. We compare a base DreamFree trained on atomic tasks in RoboCasa-Kitchen with two variants that additionally use videos of composite tasks for visual encoder training. One variant uses only $\mathcal{L}_{\text{gen}}$ without distillation, while the other uses both $\mathcal{L}_{\text{gen}}$ and $\mathcal{L}_{\text{distill}}$. All models use the same action labels from atomic tasks, and the composite tasks are absent from the action-labeled training data.

As shown in Figure 6 (left), the base model fails on all tasks, and adding the task videos with distillation raises the average success from 0% to 21.7%, with at least one successful trial on each task. Using the same videos without distillation yields 0% average success. These results demonstrate that DreamFree can acquire new task knowledge from videos and transfer it to action execution without additional action annotations, with distillation playing an important role in enabling this transfer. Figure 7 illustrates a successful microwave thawing task rollout in which DreamFree completes four subgoals in sequence.

Cross-embodiment task transfer We investigate cross-embodiment transfer of DreamFree. We use RoboTwin (Chen et al., 2026) with ALOHA-AgileX as the target robot, and compare a base DreamFree trained on its atomic tasks with a variant whose encoder also learns from composite task videos recorded with a dual-arm Franka Panda setup. Both models use the same ALOHA-AgileX action labels, and the composite tasks are absent from the action-labeled training.

Figure 6 (middle) shows that the base model succeeds only in bowl stacking, with 6.7% average success across the new tasks, and adding task videos from a different embodiment improves the average success to 30%. This demonstrates that DreamFree can transfer task knowledge from different embodiments' videos to action execution.

Video-only adaptation of a trained policy To investigate whether an already trained DreamFree can learn new tasks from videos alone, we fine-tune its visual encoder while keeping the action module fixed. Starting from DreamFree trained on atomic tasks in RoboTwin, we adapt the encoder separately for each task using 10 video demonstrations. Fine-tuning takes approximately 10 minutes per task on a single GPU. We compare fine-tuning with multiple camera views to fine-tuning with only the head camera view.

Figure 6 (right) shows that both single-view and multi-view video fine-tuning improve success rates on both tasks, with larger gains from multi-view videos. This shows that task knowledge acquired through video-only fine-tuning of the encoder can transfer to action execution without updating the action module, even with single-camera demonstrations. Figure 7 illustrates a successful task rollout with the fine-tuned policy.

We find that not all tasks can be learned through video demonstrations. For example, a policy unable to open cabinets also failed to learn to place an object inside a cabinet through video-only fine-tuning. This suggests that learning new tasks from videos may be limited by the policy's ability to execute the required atomic skills. We discuss these failure cases in Appendix D.3.

5 Conclusion

We presented DreamFree, a WAM that repurposes pretrained video diffusion models as predictive visual encoders through future representation distillation. By distilling video priors from the intermediate representations of a frozen video diffusion model with access to future frames, DreamFree learns to predict future visual representations that guide action prediction without generating future videos. Experiments demonstrate strong manipulation performance in both simulation and the real world, with a significant inference speedup over video-generating WAM. DreamFree benefits from a larger video backbone and more video data, and enables task transfer from video demonstrations. These findings highlight predictive representation learning as an effective way to translate pretrained video knowledge into robot control and expand robot capabilities through video-based learning.

6 AI USE STATEMENT

In this work, we did not use generative AI tools for research ideation, problem formulation, or the development of the core methodology. We used generative AI tools to assist with code implementation and grammar and language refinement. All AI-assisted code was reviewed and tested by the authors, and all AI-assisted text was carefully reviewed and revised. We take full responsibility for the final content of this work, including all text, claims, code, and experimental results.

7 ETHICS STATEMENT

This work focuses on robotic manipulation in controlled research environments and does not involve human subjects, sensitive data, or other significant ethical concerns.

8 REPRODUCIBILITY STATEMENT

To ensure reliable and reproducible results, we provide the source code in the supplementary materials and describe detailed experiment settings in Appendix B.

REFERENCES

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, et al. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.

Jose A. Barreiros, Andrew Beaulieu, Aditya Bhat, Rick Cory, Eric Cousineau, Hongkai Dai, Ching-Hsin Fang, Kunimatsu Hashimoto, Muhammad Zubair Irshad, Masha Itkina, Naveen Kuppuswamy, Kuan-Hui Lee, Katherine Liu, Dale McConachie, Ian McMahon, Haruki Nishimura, Calder Phillips-Grafflin, Charles Richter, Paarth Shah, Krishnan Srinivasan, Blake Wulfe, Chen Xu, Mengchao Zhang, Alex Alspach, Maya Angeles, Kushal Arora, Vitor Campagnolo Guizilini, Alejandro M. Castro, Dian Chen, Ting-Sheng Chu, Sam Creasey, Sean Curtis, Richard Denitto, Emma Dixon, Eric Dusel, Matthew Ferreira, Aimee Goncalves, Grant Gould, Damrong Guoy, Swati Gupta, Xuchen Han, Kyle Hatch, Brendan Hathaway, Allison Henry, Hillel Hochsztein, Phoebe Horgan, Shun Iwase, Donovon Jackson, Siddharth Karamcheti, Sedrick Keh, Joseph Masterjohn, Masayuki Masuda, Jean Mercat, Patrick Tree Miller, Paul Mitiguy, Tony Nguyen, Jeremy Nimmer, Yuki Noguchi, Reko Ong, Aykut Özgün Önol, Owen Pfannenstiehl, Richard Poyner, Leticia Priebe Rocha, Gordon Richardson, Christopher Rodriguez, Derick Seale, Michael A. Sherman, Mariah Smith-Jones, David Tago, Pavel Tokmakov, Matthew Tran, Basile Van Hoorick, Igor Vasiljevic, Sergey Zakharov, Mark Zolotas, Rares Ambrus, Kerri Fetzer-Borelli, Benjamin Burchfiel, Hadas Kress-Gazit, Siyuan Feng, Stacie Ford, and Russ Tedrake. A careful examination of large behavior models for multitask dexterous manipulation. *Sci. Robotics*, 11(113), 2026.

Hongzhe Bi, Hengkai Tan, Shenghao Xie, Zeyuan Wang, Shuhe Huang, Haitian Liu, Ruowen Zhao, Yao Feng, Chendong Xiang, Yinze Rong, et al. Motus: A unified latent action world model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv:2503.14734*, 2025.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. In *RSS 2025*, 2025.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. In *7th Annual Conference on Robot Learning*, 2023.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, Weiliang Deng, Yubin Guo, Tian Nian, Xuanbing Xie, Qiangyu Chen, Kailun Su, Tianling Xu, Guodong Liu, Mengkang Hu, Huan ang Gao, Kaixuan Wang, Zhixuan Liang, Yusen Qin, Xiaokang Yang, Ping Luo, and Yao Mu. Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. In *International Conference on Machine Learning*, 2026.

Danny Driess, Jost Tobias Springenberg, brian ichter, LILI YU, Adrian Li-Bell, Karl Pertsch, Allen Z. Ren, Homer Walke, Quan Vuong, Lucy Xiaoyang Shi, and Sergey Levine. Knowledge insulating vision-language-action models: Train fast, run fast, generalize better. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Shenyuan Gao, William Liang, Kaiyuan Zheng, Ayaan Malik, Seonghyeon Ye, Sihyun Yu, Wei-Cheng Tseng, Yuzhu Dong, Kaichun Mo, Chen-Hsuan Lin, et al. Dreamdojo: A generalist robot world model from large-scale human videos. *arXiv preprint arXiv:2602.06949*, 2026.

Yucheng Hu, Yanjiang Guo, Pengchao Wang, Xiaoyu Chen, Yen-Jen Wang, Jianke Zhang, Koushil Sreenath, Chaochao Lu, and Jianyu Chen. Video prediction policy: A generalist robot policy with predictive visual representations. In *International Conference on Machine Learning*, 2025.

Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. Perceiver: General perception with iterative attention. In *International Conference on Machine Learning*, 2021.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv:2403.12945*, 2024.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. OpenVLA: An open-source vision-language-action model. In *8th Annual Conference on Robot Learning*, 2024.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv:2502.19645*, 2025.

Moo Jin Kim, Yihuai Gao, Tsung-Yi Lin, Yen-Chen Lin, Yunhao Ge, Grace Lam, Percy Liang, Shuran Song, Ming-Yu Liu, Chelsea Finn, and Jinwei Gu. Cosmos policy: Fine-tuning video models for visuomotor control and planning. In *International Conference on Learning Representations*, 2026.

Lin Li, Qihang Zhang, Yiming Luo, Shuai Yang, Ruilin Wang, Fei Han, Mingrui Yu, Zelin Gao, Nan Xue, Xing Zhu, et al. Causal world modeling for robot control. *arXiv:2601.21998*, 2026.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, qiang liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. In *Neural Information Processing Systems Datasets and Benchmarks Track*, 2023.

Teli Ma, Jia Zheng, Zifan Wang, Chunli Jiang, Andy Cui, Junwei Liang, and Shuo Yang. Dit4dit: Jointly modeling video dynamics and actions for generalizable robot control. *arXiv:2603.10448*, 2026.

Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. Robocasa: Large-scale simulation of everyday tasks for generalist robots. *arXiv:2406.02523*, 2024.

Soroush Nasiriany, Sepehr Nasiriany, Abhiram Maddukuri, and Yuke Zhu. Robocasa365: A large-scale simulation framework for training and benchmarking generalist robots. In *International Conference on Learning Representations*, 2026.

Jonas Pai, Liam Achenbach, Victoriano Montesinos, Benedek Forrai, Oier Mees, and Elvis Nava. mimic-video: Video-action models for generalizable robot control beyond vlas. *arXiv:2512.15692*, 2025.

William S. Peebles and Saining Xie. Scalable diffusion models with transformers. *International Conference on Computer Vision*, pp. 4172–4182, 2022.

Physical Intelligence. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

Rhoda AI. Causal video models are data-efficient robot policy learners. *Rhoda AI Blog*, 2026. URL https://www.rhoda.ai/research/direct-video-action.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv:2503.20314*, 2025.

Hongtao Wu, Ya Jing, Chilam Cheang, Guangzeng Chen, Jiafeng Xu, Xinghang Li, Minghuan Liu, Hang Li, and Tao Kong. Unleashing large-scale video generative pre-training for visual robot manipulation. In *The Twelfth International Conference on Learning Representations*, 2024.

Angen Ye, Boyuan Wang, Chaojun Ni, Guan Huang, Guosheng Zhao, Hao Li, Hengtao Li, Jie Li, Jindi Lv, Jingyu Liu, et al. Gigaworld-policy: An efficient action-centered world–action model. *arXiv:2603.17240*, 2026a.

Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, et al. World action models are zero-shot policies. *arXiv:2602.15922*, 2026b.

Sihyun Yu, Sangkyung Kwak, Huiwon Jang, Jongheon Jeong, Jonathan Huang, Jinwoo Shin, and Saining Xie. Representation alignment for generation: Training diffusion transformers is easier than you think. In *International Conference on Learning Representations*, 2025.

Tianyuan Yuan, Zibin Dong, Yicheng Liu, and Hang Zhao. Fast-wam: Do world action models need test-time future imagination? *arXiv:2603.16666*, 2026.

Lihan Zha, Asher J Hancock, Mingtong Zhang, Tenny Yin, Yixuan Huang, Dhruv Shah, Allen Z Ren, and Anirudha Majumdar. Lap: Language-action pre-training enables zero-shot cross-embodiment transfer. *arXiv preprint arXiv:2602.10556*, 2026.

Wenyao Zhang, Hongsi Liu, Zekun Qi, Yunnan Wang, Xinqiang Yu, Jiazhao Zhang, Runpei Dong, Jiawei He, He Wang, Zhizheng Zhang, et al. Dreamvla: a vision-language-action model dreamed with comprehensive world knowledge. *Advances in Neural Information Processing Systems*, 38: 24195–24228, 2026.

Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, et al. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1702–1713. IEEE, 2025.

Ruijie Zheng, Jing Wang, Scott Reed, Johan Bjorck, Yu Fang, Fengyuan Hu, Joel Jang, Kaushil Kundalia, Zongyu Lin, Loic Magne, et al. Flare: Robot learning with implicit world modeling. *arXiv:2505.15659*, 2025.

Chuning Zhu, Raymond Yu, Siyuan Feng, Benjamin Burchfiel, Paarth Shah, and Abhishek Gupta. Unified World Models: Coupling Video and Action Diffusion for Pretraining on Large Robotic Datasets. In *Proceedings of Robotics: Science and Systems*, 2025.

Appendix

A Model architecture details

A.1 Video backbone adaptation

To adapt the video diffusion backbone (Wan et al., 2025) to the robot video distribution, we fine-tune it using its native flow-matching objective without action labels. The adapted checkpoint initializes both the teacher and student of the same architecture. During visual-encoder training, the teacher remains frozen with bidirectional video attention and text cross-attention on all frames. The student uses the modifications below. The VAE and text encoder remain frozen throughout adaptation and visual-encoder training.

A.2 Visual encoder

Video DiT Current-frame tokens attend only within the current latent frame. To encourage the current-frame representation to encode information useful for predicting future frames, we restrict each future latent frame to attend only to the current frame and itself. Spatial attention within each allowed frame is unrestricted, including across packed camera views. Text cross-attention is applied only to current-frame tokens.

For Wan2.2-I2V-A14B, the 36 input channels comprise 16 video-latent channels, four first-frame mask channels, and 16 image-conditioning channels. Image conditioning is obtained by VAE-encoding the first image followed by zero-valued images. We replace the first frame of the noisy video-latent stream with its clean latent and assign it timestep zero, and future frames retain their sampled noise level.

Projector To match the teacher's spatiotemporal token layout, the projector expands N current-frame tokens to FN tokens, where F is the teacher's latent-frame count. Student and teacher use the same spatial token grid. LayerNorm and a linear layer project $\boldsymbol{h}_i$ to width 1024. The projected tokens are replicated F times and augmented with learned frame embeddings. Two decoder blocks apply eight-head query self-attention, eight-head cross-attention to the original N projected tokens, and a $1024 \to 4096 \to 1024$ GELU MLP, each with pre-normalization and residual connections. Final LayerNorm and a linear layer map to the teacher width: 3072 for 5B or 5120 for 14B.

Student and teacher features are extracted after complete Transformer blocks i and j, respectively. To supervise future positions only, we exclude first-frame tokens from both projector outputs and teacher targets. The loss is the mean cosine distance over the remaining $(F-1)N$ token pairs after non-affine feature-wise LayerNorm ($\epsilon = 10^{-6}$).

A.3 Action readout module

During action-module training, the visual encoder is frozen. The readout and conditioning layers are trained from scratch. The readout receives current-frame tokens from encoder block k, independently selected from distillation block i, without passing through the alignment projector. Non-affine token-wise LayerNorm and a linear layer project these tokens to width 1024. To obtain a fixed-length visual context for the action decoder, we use 64 learned queries updated by two blocks. Each block contains eight-head query self-attention, eight-head cross-attention to the projected visual tokens, and a $1024 \to 4096 \to 1024$ GELU MLP. All sublayers use pre-normalization and residual connections, with zero dropout. Output LayerNorm is followed by a separate $1024 \to 1024$ conditioning projection and affine LayerNorm. These visual tokens are appended to the projected instruction tokens as decoder cross-attention context.

A.4 State and action encoders

Separate linear layers map the normalized proprioceptive state and each noisy action to width 1024. The state contributes one prefix token, replaced with a learned mask token with probability 0.2 during training. A $4096 \to 1024 \to 1024$ GELU MLP projects instruction embeddings, with a maximum sequence length of 128.

Algorithm 1 DREAMFREE training

Require: Data $\mathcal{D}_v, \mathcal{D}_a$; fixed teacher noise t_d
Adapt video backbone on $\mathcal{D}_v$; initialize $\boldsymbol{g}$ and $\boldsymbol{u}_\theta$.
Freeze $\boldsymbol{g}$; initialize projector $\boldsymbol{p}_\psi$.
Stage 1: Visual encoder
while training **do**
  Sample $(\boldsymbol{x}_*, \boldsymbol{\ell}) \sim \mathcal{D}_v$ and $t, \boldsymbol{\epsilon}, \boldsymbol{\epsilon}_d$.
  $\boldsymbol{x}_t \leftarrow (1-t)\boldsymbol{x}_* + t\boldsymbol{\epsilon}$; $\boldsymbol{x}_t^0 \leftarrow \boldsymbol{x}_*^0$.
  $(\hat{\boldsymbol{v}}, \boldsymbol{h}_i) \leftarrow \boldsymbol{u}_\theta(\boldsymbol{x}_t, t, \boldsymbol{\ell})$. ▷ $\boldsymbol{h}_i$: current frame
  $\boldsymbol{x}_{t_d} \leftarrow (1-t_d)\boldsymbol{x}_* + t_d\boldsymbol{\epsilon}_d$.
  $\boldsymbol{y} \leftarrow [\boldsymbol{g}_j(\boldsymbol{x}_{t_d}, t_d, \boldsymbol{\ell})]^+$. ▷ $+$: future tokens only
  $\mathcal{L}_{\text{gen}} \leftarrow w(t)\,\text{MSE}(\hat{\boldsymbol{v}}^+, (\boldsymbol{\epsilon} - \boldsymbol{x}_*)^+)$.
  $\mathcal{L}_{\text{distill}} \leftarrow -\text{sim}([\boldsymbol{p}_\psi(\boldsymbol{h}_i)]^+, \boldsymbol{y})$.
  Update θ, ψ with $\mathcal{L}_{\text{gen}} + \mathcal{L}_{\text{distill}}$.
end while
Stage 2: Action module
Freeze $\boldsymbol{u}_\theta$; initialize the action module.
while training **do**
  Sample $(\boldsymbol{x}_*^0, \boldsymbol{\ell}, \boldsymbol{q}, \boldsymbol{s}, \boldsymbol{a}_*) \sim \mathcal{D}_a$ and $t, \boldsymbol{\epsilon}$.
  $\boldsymbol{h}_k \leftarrow [\boldsymbol{u}_\theta(\boldsymbol{x}_*^0, 0, \boldsymbol{\ell})]_k$.
  $\boldsymbol{c} \leftarrow \boldsymbol{r}_\omega(\text{LN}(\boldsymbol{h}_k))$. ▷ Action readout
  $\boldsymbol{a}_t \leftarrow (1-t)\boldsymbol{a}_* + t\boldsymbol{\epsilon}$.
  $\hat{\boldsymbol{v}}_a \leftarrow \boldsymbol{u}_\phi(\boldsymbol{a}_t, t \mid \boldsymbol{c}, \boldsymbol{q}, \boldsymbol{s}, \boldsymbol{\ell})$.
  $\mathcal{L}_{\text{act}} \leftarrow w(t)\,\text{MSE}(\hat{\boldsymbol{v}}_a, \boldsymbol{\epsilon} - \boldsymbol{a}_*)$.
  Update the action module with $\mathcal{L}_{\text{act}}$.
end while

Algorithm 2 DREAMFREE inference

Require: Instruction $\boldsymbol{\ell}$, steps N, execution length R
$\{t_n\}_{n=0}^N \leftarrow \text{Schedule}(N)$.
Initialize context $\boldsymbol{s} \leftarrow \varnothing$.
while task not done **do**
  $(\boldsymbol{o}, \boldsymbol{q}) \leftarrow \text{Observe}()$; $\boldsymbol{x}_*^0 \leftarrow \text{VAE}(\boldsymbol{o})$.
  $\boldsymbol{h}_k \leftarrow [\boldsymbol{u}_\theta(\boldsymbol{x}_*^0, 0, \boldsymbol{\ell})]_k$. ▷ One single pass
  $\boldsymbol{c} \leftarrow \boldsymbol{r}_\omega(\text{LN}(\boldsymbol{h}_k))$.
  Sample $\boldsymbol{a}_{t_0} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$.
  for $n = 0, \ldots, N-1$ **do**
    $\hat{\boldsymbol{v}}_n \leftarrow \boldsymbol{u}_\phi(\boldsymbol{a}_{t_n}, t_n \mid \boldsymbol{c}, \boldsymbol{q}, \boldsymbol{s}, \boldsymbol{\ell})$.
    $\boldsymbol{a}_{t_{n+1}} \leftarrow \boldsymbol{a}_{t_n} + (t_{n+1} - t_n)\hat{\boldsymbol{v}}_n$.
  end for
  Postprocess $\boldsymbol{a}_{t_N}$; execute its first R actions.
  if history enabled **then** ▷ DROID
    Update action history in $\boldsymbol{s}$.
  end if
end while

For DROID, we include eight previous actions to address unintended gripper reopening during closed-loop execution. History uses the same linear encoder as noisy actions. The input order is current state, chronological action history, then noisy future actions. Missing history entries are masked, and the entire history is independently dropped with probability 0.2 during training. State and history remain noise-free, with a fixed timestep corresponding to $\sigma = 10^{-3}$.

A.5 ACTION DECODER

The decoder uses 30 DiT blocks, hidden width 1024, feed-forward width 4096, and 24 attention heads of dimension 128. Each block contains bidirectional self-attention over state/history/action tokens, cross-attention to instruction and visual context, and a GELU MLP. Self-attention uses one-dimensional rotary position embeddings and padding masks. A 256-dimensional sinusoidal timestep embedding and an MLP provide block modulation. A final linear layer predicts the flow-matching velocity in action space, and prefix outputs are discarded, and loss is computed only at valid future-action positions.

A.6 TRAINING AND INFERENCE ALGORITHMS

Algorithms 1 and 2 summarize training and synchronous closed-loop inference. We write $\boldsymbol{x}_*$ for VAE-encoded video latents, with superscripts 0 and $+$ selecting current and future positions. Video-model calls also include the Wan-A14B image-conditioning channels when applicable. sg denotes stop-gradient, LN is non-affine token-wise LayerNorm, and sim is mean cosine similarity after this normalization. States and actions are normalized. $\boldsymbol{s}$ denotes optional action history, used only for DROID.

Here, p_{train} samples $t = 5r/(1+4r)$ with $r \sim \mathcal{U}(0,1)$, and $w(t)$ is the native scheduler's timestep weight. Losses are averaged over the batch. MSE_m excludes padded action positions. The implemented cosine loss $1 - \text{sim}$ differs from $-\text{sim}$ only by an additive constant. Native teacher conditioning keeps the first frame clean for 5B, and Wan-A14B retains a noised first frame in the video stream and separate clean image conditioning.

B Implementation Details

B.1 Baselines

For simulation benchmarks, we use reported baseline scores except for Fast-WAM on RoboCasa-Kitchen, which we train and evaluate ourselves.

Fast-WAM (Yuan et al., 2026) For RoboCasa-Kitchen, we train Fast-WAM 5B using the official implementation with learning rate 10^{-4} for 20 epochs and global batch size 128. The 14B baseline uses the Wan2.2-A14B high-noise expert with global batch size 32.

B.2 Ours

We use AdamW with a learning rate of 10^{-4} for both training stages. For LIBERO and Robocasa models, we use rank-128 LoRA to train the visual encoders initialized from the video diffusion models. For DROID 14B model, we fully train the weights without using LoRA. Input resolutions are 224×448 for LIBERO, 352×352 for RoboCasa, and 352×640 for DROID. We set the teacher noise level to $t_d = 0.7$ for simulation benchmarks and 0.8 for DROID.

We use a fixed layer configuration for each backbone size across all datasets and tasks, without dataset-specific layer selection. For the 5B backbone, we extract student features for distillation at $i = 25$ and features for action readout at $k = 19$. For the 14B backbone, we use $i = j = k = 31$. All indices are zero-based, and features are extracted after the corresponding complete Transformer block.

For LIBERO, we use global batch sizes of 128 and 64 in stages 1 and 2, respectively. Stage 1 runs for 40 epochs, and stage 2 for 10 epochs. For RoboCasa 5B, we use global batch sizes of 64 and 256 in stages 1 and 2, respectively, for four and 20 epochs. For RoboCasa 14B, we use global batch sizes of 32 and 512 in stages 1 and 2, respectively. Stage 1 runs for 10 epochs on RoboCasa-only videos or two epochs with additional videos, and stage 2 runs for 20 epochs. For DROID, we fully train the visual encoder in stage 1 without LoRA. We use global batch sizes of 64 and 4,096 in stages 1 and 2, respectively, for 6 and 16 epochs.

Inference The visual encoder runs once at timestep zero, without video denoising or classifier-free guidance (CFG). Actions are initialized from Gaussian noise and sampled with 10 Euler steps for LIBERO and RoboCasa, and 16 steps for DROID, using a flow-matching schedule with shift 5. Action sampling also omits CFG, requiring one conditional decoder evaluation per step. From each 32-action chunk, we execute 10 actions in LIBERO, 32 in RoboCasa, and 24 in DROID before replanning.

C Experiments Details

C.1 Simulation Benchmarks

LIBERO We use the same frame resolution as in the official Fast-WAM implementation during both training and inference. We report the average success rate across the four LIBERO suites, evaluated over a total of 2,000 trials covering 40 tasks with different random seeds. The baseline results are taken from Yuan et al. (2026).

Robocasa-Kitchen We use the Cosmos Policy (Kim et al., 2026) dataset with 50 human-teleoperated demonstrations per task. We train and evaluate DreamFree and Fast-WAM on this benchmark and use reported scores for the remaining baselines. Following DreamZero (Ye et al., 2026b), we pack three camera views into a single frame, with the wrist view stretched horizontally by $2\times$ above two side-by-side external views. Each of the 24 tasks is evaluated over 50 trials. The baseline results are taken from Kim et al. (2026).

For our additional-video experiments, we use a filtered and converted version of the RoboCasa365 human-teleoperation atomic-task dataset. The source dataset covers 65 atomic tasks spanning ten skill families. Each atomic task involves one basic skill executed over multiple timesteps, whereas

Table 4: Real-world evaluation tasks on DROID.

Task	Instruction	Setup and objective
Stand bottle upright	"Stand the bottle upright."	A bottle lies on its side. Stand it on its base and release it.
Fold glove	"Fold the glove."	A glove lies unfolded. Make a regular fold that turns back more than one-third of its original length and remains after release.
Cover strawberry with towel	"Use the towel to cover the strawberry."	A toy strawberry and grape lie on opposite sides of a towel. Lift the towel and release it over the strawberry, fully covering it while leaving the grape uncovered.
Put red marker in blue cup	"Put the red marker inside the blue cup."	Red and black markers lie beside blue and green cups. Release the red marker inside the upright blue cup; the black marker and green cup are distractors.
Move cup from bowl to cup	"Pick up the cup from the bowl and put it in the other cup."	One cup sits inside a bowl, with another cup and a distractor can nearby. Remove the source cup and release it inside the other cup.
Press pump	"Press the pump."	An upright pump bottle stands beside a cup holding a toothbrush. Depress the pump head while keeping the bottle upright.
Put bolts in bowl	"Put the bolts in the bowl."	Three bolts lie separately beside a bowl. Place and release all three inside the bowl.
Pour cup into pan	"Pour the contents of the cup into the pan."	A source cup and receiving pan sit on the table. Transfer the cup's contents into the pan.
Hang hat on tripod	"Hang the hat on the tripod."	A hat lies beside an upright tabletop tripod. Place and release the hat so that the tripod supports it.
Put ring around pole	"Put the ring around the pole."	A ring lies beside an upright pole on a base. Place and release the ring with the pole passing through its hole.

composite tasks combine several skills. Demonstrations use a Panda arm mounted on an Omron mobile base, with synchronized left, right, and wrist camera videos. We use the additional videos for backbone adaptation and visual-encoder training without action supervision. Action-module training uses only the Cosmos Policy demonstrations.

C.2 Real-world experiments

Tasks and evaluation We evaluate on 10 tasks in the pretrained distribution, performing 5 rollouts per task, for a total of 50 evaluation rollouts for each model. Table 4 describes the ten evaluation tasks, and Table 5 describes their progress scores. Following Barreiros et al. (2026), we apply an image overlay to the initial scene to reduce variance.

Latency Although single-pass visual encoding without iterative video denoising or CFG already makes DreamFree faster than DreamZero, we further reduce latency through engineering optimizations. These include VAE encoding of only the current frame, fused attention kernels, and fused QKV/KV projections. Visual and action Transformer blocks are compiled with `torch.compile`, and CUDA graphs replay VAE encoding, visual feature extraction, and the action sampling loop to reduce launch overhead. Instruction embeddings are cached across queries, while action-conditioning keys and values are computed once per chunk and reused across denoising steps.

We measure latency on a single NVIDIA H200 after startup warm-up, including client preprocessing, server policy execution, and client postprocessing, and excluding communication and robot execution.

Regarding task completion time, we use three rollouts per policy and block-count setting. We estimate each rollout's duration by adding nominal robot execution time to its total policy computation time. Communication and setup overhead are excluded.

Table 5: Task-progress rubrics for real-world evaluation tasks on DROID.

Task	Scoring criteria
Stand bottle upright	0.5: bottle axis exceeds 45° above horizontal; upright is 90°. 1: bottle stands on its base after release.
Fold glove	0: no fold. 0.5: a nonzero regular fold turns back at most one-third of the original glove length, or an irregular fold occurs, regardless of folded length. 1: a regular fold turns back more than one-third of the original length and remains after release.
Cover strawberry with towel	Requires lifting, then partial coverage, then full coverage. $1/3$: towel lifted. $2/3$: any partial strawberry coverage after lifting, including slight coverage. 1: strawberry fully covered after towel release, with the grape uncovered. Coverage without first lifting does not qualify.
Put red marker in blue cup	0.25: red marker picked up. 0.5: red marker contacts the blue cup. 0.75: marker enters the cup, but the cup falls or tips over. 1: marker released inside the upright blue cup. Picking up the black marker before picking up the red marker makes the entire rollout score 0, even after later completion; incidental contact does not trigger this penalty.
Move cup from bowl to cup	$1/3$: correct source cup picked up. $2/3$: source cup contacts the receiving cup without successful insertion. 1: source cup inserted and released inside the receiving cup.
Press pump	0.5: contact with the movable pump head. 1: pump head depressed while the bottle remains upright. Spring return after release is allowed.
Put bolts in bowl	$1/3$ per bolt placed and released inside the bowl. Retain intermediate credit for the highest count reached; 1 requires all three bolts to remain inside at the end.
Pour cup into pan	0.5: source cup picked up. 1: its contents transferred into the pan.
Hang hat on tripod	$1/3$: hat picked up. $2/3$: hat contacts the tripod. 1: hat remains supported by the tripod after release.
Put ring around pole	0.5: ring picked up. 1: ring placed and released with the pole passing through its hole.

Table 6: Task descriptions and ordered subgoals for task transfer from video demonstrations. Subgoals are evaluated in the listed order.

Setting	Task	Description and ordered subgoals
Atomic-to-composite	Arrange vegetables	Transfer the first vegetable from the sink to the counter $\rightarrow$ transfer the second vegetable from the sink to the counter.
Atomic-to-composite	Microwave thawing	Open the microwave door $\rightarrow$ transfer the object from the counter into the microwave $\rightarrow$ close the door $\rightarrow$ turn on the microwave.
Atomic-to-composite	Restock pantry	Transfer the first object from the counter into the cabinet $\rightarrow$ transfer the second object from the counter into the cabinet.
Cross embodiment	Place can in basket	Place the can inside the basket $\rightarrow$ lift the basket. Can placement alone constitutes partial completion.
Cross embodiment	Place cans in box	Place the first can inside the plastic box $\rightarrow$ place the second can inside the box.
Cross embodiment	Stack three bowls	Establish the base bowl $\rightarrow$ stack the second bowl on the base $\rightarrow$ stack the third bowl on the stack.

C.3 Task transfer from video demonstrations

We provide task descriptions in Table 6.

Atomic-to-composite task transfer We merge the existing LoRA weights into the frozen encoder and train fresh rank-128 adapters and the alignment projector on target-task videos at a learning rate

Table 7: LIBERO success rates (%) of DREAMFREE across the four suites.

Method	Spatial	Object	Goal	Long	Average
LingBot-VA (Li et al., 2026)	98.5	99.6	97.2	98.5	98.5
Fast-WAM (Yuan et al., 2026)	98.2	100.0	97.0	95.2	97.6
DREAMFREE (Ours)	99.0	99.8	98.8	96.6	**98.6**

Table 8: RoboCasa-Kitchen success rates (%) for DREAMFREE with the 14B backbone and additional video data. Each task is evaluated over 50 trials.

Task	SR (%)	Task	SR (%)
CloseDoubleDoor	84	PnPCounterToSink	72
CloseDrawer	98	PnPCounterToStove	74
CloseSingleDoor	98	PnPMicrowaveToCounter	58
CoffeePressButton	96	PnPSinkToCounter	76
CoffeeServeMug	72	PnPStoveToCounter	72
CoffeeSetupMug	38	TurnOffMicrowave	90
OpenDoubleDoor	96	TurnOffSinkFaucet	86
OpenDrawer	88	TurnOffStove	18
OpenSingleDoor	90	TurnOnMicrowave	94
PnPCabToCounter	56	TurnOnSinkFaucet	76
PnPCounterToCab	62	TurnOnStove	52
PnPCounterToMicrowave	50	TurnSinkSpout	90
Mean over 24 tasks			**74.4**

of 10^{-4}. We use videos of the three target RoboCasa composite tasks. Across all three settings, the instruction remains fixed throughout each rollout, without intermediate subgoal prompts. Progress is the fraction of subgoals achieved in order.

Cross-embodiment task transfer For a policy trained on ALOHA-AgileX atomic tasks, we use dual-arm Franka Panda videos to train its visual encoder on additional composite tasks. We use 50 video demonstrations for each of three selected composite tasks, totaling 150 demonstrations. We evaluate the same three tasks on ALOHA-AgileX, testing cross-embodiment transfer of task knowledge from video demonstrations.

Video-only adaptation of a trained policy Starting from a policy trained on RoboTwin, we fine-tune its visual encoder separately for each target task using 10 video demonstrations, while keeping the action module fixed. We compare fine-tuning with multiple camera views and with only the head camera: basket placement uses 60 and 40 iterations, respectively, and bowl stacking uses 100 and 80. Both variants are evaluated in RoboTwin using all three camera views.

D ADDITIONAL RESULTS

D.1 LIBERO PER-SUITE RESULTS

Table 7 reports per-suite success rates for DREAMFREE on LIBERO.

D.2 ROBOCASA KITCHEN PER-TASK RESULTS

Table 8 reports per-task success rates for the 14B model trained with additional video data.

D.3 FAILURE CASE ANALYSIS

Real-world manipulation Although DREAMFREE improves over DreamZero on tasks involving deformable objects and successive rearrangements, pump pressing remains a failure case. The policy contacts the pump head in all five trials but completes the task in only one. We hypothesize that this

reflects difficulty generalizing the motion required to actuate a new object, even when the target is reached. Whether broader training coverage of such interactions improves this generalization remains a direction for future work.

Task transfer The cabinet insertion task requires opening the cabinet before placing an object inside. The base policy did not reliably perform the opening motion, and video-only fine-tuning did not enable it to open. This failure suggests that transferring task knowledge from videos may depend on the availability of related action skills in the base policy. With the action module frozen, adapting the visual encoder alone may be insufficient to acquire a new motor skill. We leave it to future work to test whether scaling pretraining with more diverse action demonstrations alleviates this limitation.

E Further discussions

E.1 Connection to prior works

Comparison with Fast-WAM (Yuan et al., 2026) Fast-WAM uses a pretrained video diffusion backbone for direct action prediction, retaining video co-training but omitting future-video denoising at deployment. The key difference is that DreamFree directly supervises the policy features with future representation distillation in addition to the video generative objective. In our RoboCasa-Kitchen experiments (Table 2), Fast-WAM substantially trails the video-generating WAM baseline and shows little improvement from a 5B to a 14B backbone (44.4% to 44.7%). These results suggest that generative supervision through shared weights alone does not ensure that a larger VDM provides more useful predictive features for control. DreamFree supports learning and transfer from videos without action labels, while achieving performance and generalization competitive with video-generating WAMs at substantially lower inference latency.

Knowledge Insulation (Driess et al., 2025) Freezing the visual encoder in stage 2 prevents gradients from the action DiT from updating the video backbone. Knowledge Insulation and LAP (Zha et al., 2026) similarly block gradients from the continuous action expert while training VLM representations through FAST-token prediction and language-action prediction, respectively. Our method applies a similar separation to a VDM, using the native video generative objective and future representation distillation to shape the visual features. Their concurrent training also suggests an alternative to our two-stage procedure: jointly training the encoder and action module with their respective objectives, while blocking action-module gradients from the encoder. We leave this variant to future work, as we have not evaluated it due to computational constraints.

E.2 Limitations

Although DreamFree uses video-model priors to predict actions directly like VLAs, achieving lower latency than video-generating WAMs, several limitations remain.

- **Scaling and generality.** We demonstrate training a 14B video backbone with a large pretraining dataset (DROID), outperforming the existing VLA and WAM baselines in real-world success. However, training on a substantially larger video backbone across diverse datasets and embodiments remains untested. Evaluating these settings is an important direction for future work.
- **Training complexity.** Our pipeline separates backbone adaptation, predictive representation learning, and action learning. Simplifying these stages and exploring joint training of the visual encoder and action module remain directions for future work.