Action Images: End-to-End Policy Learning via Multiview Video Generation

Haoyu Zhen[1*] **Zixian Gao**[1*§] **Qiao Sun**[1§] **Yilin Zhao**[2] **Yuncong Yang**[1]
Yilun Du[3] **Pengsheng Guo**[4] **Tsun-Hsuan Wang**[4] **Yi-Ling Qiao**[4] **Chuang Gan**[1]
[1]UMass Amherst [2]NVIDIA [3]Harvard University [4]Genesis AI
https://ActionImages.github.io

Abstract

World action models (WAMs) have emerged as a promising direction for robot policy learning, as they can leverage powerful video backbones to model the future states. However, existing approaches often rely on separate action modules, or use action representations that are not pixel-grounded, making it difficult to fully exploit the pretrained knowledge of video models and limiting transfer across viewpoints and environments. In this work, we present Action Images, a unified world action model that formulates policy learning as multi-view video generation. Instead of encoding control as low-dimensional tokens, we translate 7-DoF robot actions into interpretable action images: multi-view action videos that are grounded in 2D pixels and explicitly track robot-arm motion. This pixel-grounded action representation allows the video backbone itself to act as a zero-shot policy, without a separate policy head or action module. Beyond control, the same unified model supports video-action joint generation, action-conditioned video generation, and action labeling under a shared representation. On RLBench and real-world evaluations, our model achieves the strongest zero-shot success rates and improves video–action joint generation quality over prior video-space world models, suggesting that interpretable action images are a promising route to policy learning.

1 Introduction

World action models [24, 31, 39, 73, 78] have made rapid progress in predicting future observations, but turning this predictive ability into policy generalization remains an open challenge. In particular, strong video generation does not automatically produce a strong policy: a model may successfully synthesize plausible future frames, yet still fail to decide how to act in unseen environments. This gap between video generalization and policy generalization is a central bottleneck for world models.

A possible reason is that action is not represented in a form that world models can naturally generalize. Existing approaches typically follow one of two paths. Some [13, 39, 73, 78, 80] attach a separate policy head or action module on top of a world model, asking an additional network to decode control from learned video features. Others [31] adapt video models to action generation using representations that are not spatially grounded in image space. In both cases, the model's predictive knowledge of the world is only indirectly connected to acting. As a result, the burden of generalization is shifted to a specialized control module, which is often exactly where transfer breaks down.

In this work, we formulate policy learning as video generation and address policy generalization at the representation level. We propose Action Images, a robotics world modeling framework built

*Equal contribution.
§Work done while remote interns at UMass Amherst.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

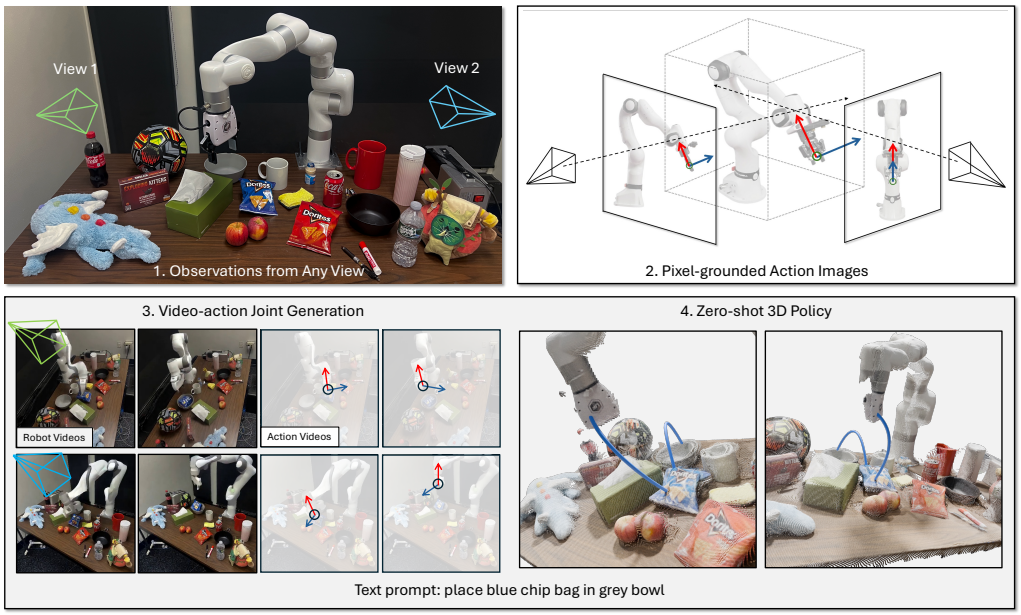

Figure 1: **Action Images** turns policy learning into multi-view video generation: 7-DoF actions are translated into pixel-grounded action images that explicitly track robot-arm motion, enabling a zero-shot policy directly from a unified video backbone.

on multi-view action videos (Fig. 1): it translates robot actions into interpretable action images and models them together with observations in a unified video-space representation of observation and action. Instead of treating 7-DoF control as low-dimensional signals or latent action codes, we convert each action into a pixel-grounded action representation that explicitly tracks robot-arm motion in image space across multiple views. This design makes action native to the video model itself: the same video backbone can observe, predict, condition on, and generate action, enabling a zero-shot policy. By grounding action in pixels rather than in an external interface, we obtain a more generalizable policy model that transfers more naturally across viewpoints and embodiments.

A key design choice is to represent these action images as multi-view videos. The motivation is not merely to add more visual observations, but to bridge the gap between 2D images and the 7-DoF robot action in the 3D space. A single view often provides only an ambiguous projection of motion, making it difficult for the model to infer the full action consistently from pixels alone [78, 82]. Using multiple views makes the pixel-grounded action image more reconstructable, while also improving robustness when some motion is partially occluded.

Beyond control, the same unified video-space representation of observation and action supports multiple tasks within a single model. Because observation and action share the same generative space, the model can perform video-action joint generation, action-conditioned video generation, and action labeling under one backbone and one training objective. These capabilities emerge without a separate policy head or action module, showing that a robotics world model can be trained not only to predict the world, but also to act in it through a common visual representation.

In summary, our contributions are as follows:

- We identify the gap between video generalization and policy generalization as a limitation of world models, and argue that this gap can be addressed at the level of action representation.
- We propose Action Images, a multi-view action representation that translates robot control into interpretable action images forming a pixel-grounded action representation, and use this representation to build a zero-shot policy without a separate policy head or action module.
- We show that this design yields a more generalizable policy model and provides a unified video-space representation of observation and action that supports video-action joint generation, action-conditioned video generation, and action labeling within a single robotics world model.

2 Related Work

Robotics World Models. Originating from Reinforcement Learning [46, 59], world models typically take actions and the current state as input and predict future states [2, 10]. In recent years, learning world models for diverse robotic applications [5, 20, 36, 67, 77] has garnered significant interest. With the success of video generation models, lots of work has developed robotics world models based on video generation [9, 13, 18, 34, 58, 78, 80]. These video-based approaches typically adopt a two-stage pipeline, where future observations are first predicted and actions are then generated based on these predictions. More recently, joint video-action generation has been explored to unify modeling and control [6, 19, 31, 37, 39, 56, 65, 71, 81]. Concurrent work [38] also investigates video-based formulations for robot policy learning. Our approach differs in representing actions as pixel-grounded multi-view images that encode full 7-DoF control, enabling a unified video-action space and eliminating the need for separate modules. In particular, DreamZero [73] demonstrates strong zero-shot generalization and cross-embodiment transfer. However, these methods encode actions with additional action modules, leaving much of the pretrained video knowledge underused; we instead use multi-view action images so the backbone itself is a zero-shot policy.

Generalist Robot Policy Models. Policy models map current states to future actions [51, 64]. Developing generalist control policies that can succeed in diverse tasks and can be lightweightly fine-tuned to adapt to downstream tasks has long been a central goal [8, 21, 40, 47, 60, 70, 75]. While multiple advances in Vision-Language-Action (VLA) models [7, 26, 32, 83], Diffusion Policy [11, 49], and Reinforcement Learning [25, 47] have greatly promoted the generalizability of policy models, their diversity is still limited to relatively narrow task distributions and they struggle to zero-shot generalize to new environments [14, 79]. In parallel, strong capabilities of video generation foundation models in predicting future frames and modeling physical dynamics have inspired policy learning approaches [24, 39, 42]. However, how to turn video prediction into transferable control remains nontrivial; our action-image representation bridges this gap by making action native to the video space.

4D Generation Models. "4D" here refers to 3D plus time. Optimization-based methods employ Score Distillation Sampling, which distills pre-trained diffusion models into specific 4D representations [3, 54, 57, 72]. Recent work [41] explores native 4D generation, which is trained directly on 4D datasets. Due to the lack of large-scale pretraining assets, a branch of research leverages the rich semantic priors in pretrained video generation models and integrates reconstruction methods to lift 2D frame sequences into 4D results [28, 48, 66, 69, 74]. However, these contributions mostly focus on single-avatar or simple scene generation. Close to our method, [4, 68] leverage multi-view generation to produce complex dynamic 4D scenes that can be replayed at any specified camera pose and timestamp. However, for robotic tasks, 4D generation is typically limited to a fixed single view [18, 78, 82]. Although [45] has leveraged multi-view inputs and introduced a geometry-consistent supervision, they still do not generalize well beyond their training scenes.

3 Method

Robotics world models have recently shown strong capability in modeling dynamics, especially when built on large pretrained video backbones. However, leveraging such video prediction capabilities for generalizable robot control remains non-trivial. To bridge this gap, we build a unified video-space representation, where robot control is translated into interpretable action images that form a pixel-grounded representation. We first introduce how 7-DoF robot actions are converted into multi-view action videos (Sec. 3.1), then describe how this representation can be decoded back into continuous control (Sec. 3.2), and finally present the training of a unified world-action model that enables a zero-shot policy (Sec. 3.3).

3.1 Action as Images

Our central idea is to represent robot actions in the same modality as robot observations. Instead of using a low-dimensional control vector with a separate policy head, we convert each 7-DoF action into interpretable action images and model it directly in video space. This yields a pixel-grounded action representation, allowing the same video backbone to process both observation and action. As shown in Fig. 2, action modeling is therefore cast as a tracking-like visual prediction problem.

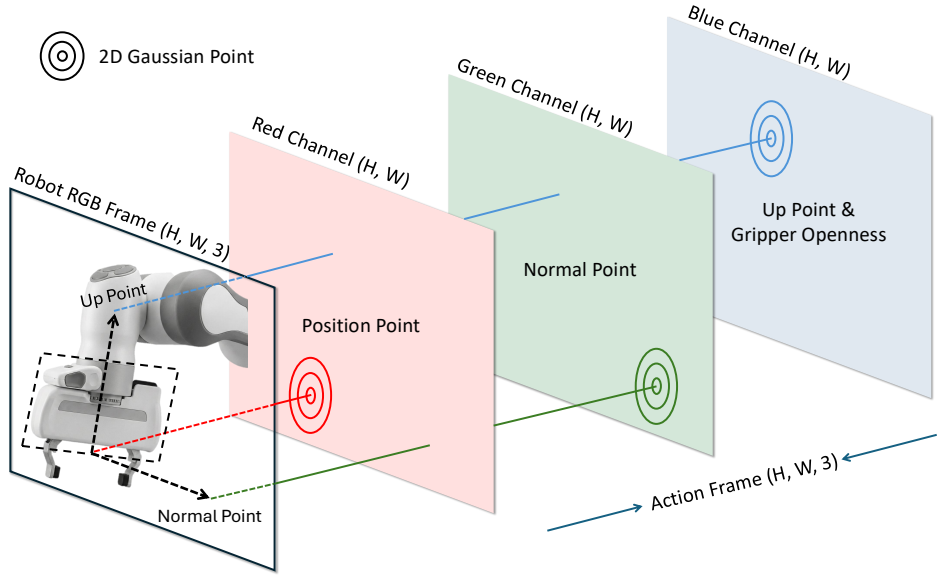

Figure 2: **Action as image.** We convert each 7-DoF robot action into three semantic 3D points (position, normal, and up), project them into image space, and render them as RGB Gaussian heatmaps. The blue channel further encodes gripper openness in the low-response background.

From 7-DoF action to semantic 3D points. At time step t, the action is $\mathbf{a}_t = [\mathbf{p}_t, \boldsymbol{\theta}_t, g_t] \in \mathbb{R}^7$, where $\mathbf{p}_t$, $\boldsymbol{\theta}_t$, and g_t denote the end-effector position, orientation, and gripper openness. We convert it into three semantic 3D points: the position point $\mathbf{q}_t^{\text{pos}} = \mathbf{p}_t$, and two orientation-aware points $\mathbf{q}_t^{\text{normal}} = \mathbf{p}_t + \ell \mathbf{R}(\boldsymbol{\theta}_t)(-\mathbf{e}_z)$ and $\mathbf{q}_t^{\text{up}} = \mathbf{p}_t + \ell \mathbf{R}(\boldsymbol{\theta}_t)\mathbf{e}_x$, where $\mathbf{R}(\boldsymbol{\theta}_t) \in SO(3)$ is the rotation matrix and ℓ is a small offset length. These three points compactly encode the end-effector pose and can be projected into image space.

Multi-view action image rendering. For each camera view v, we project the three semantic points into image space: $\mathbf{u}_t^{k,(v)} = \pi_t^{(v)}(\mathbf{q}_t^k)$, where k indexes the position, normal, and up points $k \in \{\text{pos}, \text{normal}, \text{up}\}$. We render them into an action image $\mathbf{A}_t^{(v)} \in \mathbb{R}^{H \times W \times 3}$ with 2D Gaussian maps. The red, green, and blue channels encode the position, normal, and up points, respectively. We render the three projected points as Gaussian maps:

$$\mathbf{A}_t^{(v)}(:, :, c_k) = \mathcal{G}(\cdot; \mathbf{u}_t^{k,(v)}, \sigma), \qquad k \in \{\text{pos}, \text{normal}, \text{up}\}, \tag{1}$$

where $c_{\text{pos}} = 1$, $c_{\text{normal}} = 2$, and $c_{\text{up}} = 3$ denote the red, green, and blue channels. We then overwrite low-response regions of the blue channel with the gripper openness:

$$\mathbf{A}_t^{(v)}(i, j, 3) = 0.25 g_t \quad \text{if } \mathbf{A}_t^{(v)}(i, j, 3) < 0.25 \tag{2}$$

Stacking action images over time gives an action video $\mathcal{A}^{(v)} = \{\mathbf{A}_1^{(v)}, \ldots, \mathbf{A}_T^{(v)}\} \in \mathbb{R}^{T \times H \times W \times 3}$. Since $\mathcal{A}^{(v)}$ shares the same spatial-temporal structure as the RGB observation $\mathcal{O}^{(v)}$, observation and action can be represented in a unified video space.

Benefits. Representing actions as interpretable action images provides two key benefits. First, it makes action prediction spatially grounded: the model learns control through visible robot-arm motion rather than through abstract action tokens. Second, it is naturally compatible with pretrained video backbones, allowing the same model to reason over observation and action without an action module. In this way, our zero-shot policy is obtained by turning the robot action into a visual prediction problem. Because the representation is pixel-grounded and multi-view, it transfers more naturally across viewpoints, motion patterns, and robot embodiments, leading to a more generalizable policy model.

3.2 Action Images Decoding

A useful action representation should not only be easy to generate, but easy to decode back into continuous robot control. We therefore design a decoding method that maps the generated multi-

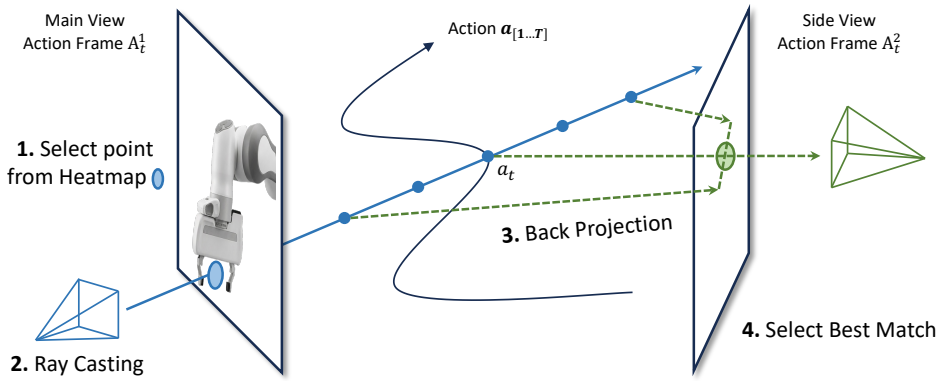

Figure 3: **Action images decoding.** A 2D heatmap point is selected in the main view, lifted to 3D by ray casting and side-view matching, and repeated for all points to recover the original 7-DoF action.

view action videos back to the original 7-DoF action. In this way, the same unified video-space representation of observation and action can be used both for generation and for control.

Decoding gripper openness. The blue channel stores both one projected semantic point and the gripper openness, where the latter is written into low-response background regions. Let $\mathbf{A}_t^{(v)}(:,:,3)$ denote the blue channel of the action image at time t and view v. We estimate gripper openness by averaging only the low-response pixels:

$$\hat{g}_t = \frac{1}{0.25} \cdot \frac{1}{|\Omega_t|} \sum_{(i,j,v)\in\Omega_t} \mathbf{A}_t^{(v)}(i,j,3), \quad \Omega_t = \{(i,j,v) \mid \mathbf{A}_t^{(v)}(i,j,3) \le 0.25\}. \tag{3}$$

Reconstructing 3D semantic points. For the remaining action information, we decode each semantic point from its corresponding heatmaps using a simple multi-view geometric procedure. As illustrated in Fig. 3, we first compute a main-view 2D anchor by heatmap-weighted averaging, then cast a ray from the main-view camera center through this anchor and sample candidate 3D points $\mathcal{R}_t^k$ between the near and far planes. The final 3D point is selected by the side-view heatmap response:

$$\hat{\mathbf{u}}_t^{k,(1)} = \frac{\sum_{i,j} \mathbf{H}_t^{k,(1)}(i,j)[i+0.5,\ j+0.5]^\top}{\sum_{i,j} \mathbf{H}_t^{k,(1)}(i,j)}, \qquad \hat{\mathbf{q}}_t^k = \arg\max_{\mathbf{x}\in\mathcal{R}_t^k} \mathbf{H}_t^{k,(2)}\left(\pi_t^{(2)}(\mathbf{x})\right). \tag{4}$$

Here, $\mathbf{H}_t^{k,(v)} = \mathbf{A}_t^{(v)}(:,:,c_k)$ denotes the heatmap of point k in the main view (v=1) or side view (v=2), and $\pi_t^{(2)}(\cdot)$ is the side-view projection function. This procedure is repeated for each $k \in \{\text{pos}, \text{normal}, \text{up}\}$, yielding the reconstructed 3D action points. The position is obtained from the reconstructed position point, $\hat{\mathbf{p}}_t = \hat{\mathbf{q}}_t^{\text{pos}}$, while the up and normal points define the local end-effector frame used to recover the orientation $\hat{\boldsymbol{\theta}}_t$. Combining it with the decoded gripper openness $\hat{g}_t$ gives the final action $\hat{\mathbf{a}}_t = [\hat{\mathbf{p}}_t, \hat{\boldsymbol{\theta}}_t, \hat{g}_t]$.

Discussion. When the predicted heatmaps are accurate, the remaining decoding error is dominated not by representation mismatch, but by discretization. In particular, the 3D reconstruction accuracy is mainly determined by (i) the sampling interval along the ray, which controls depth precision, and (ii) the spatial resolution of the heatmaps, which controls localization precision in image space. As a result, the information loss introduced by the action-image parameterization is minor and predictable: finer ray sampling and higher image resolution directly improve the fidelity of the decoded action.

3.3 Training Unified World Action Model

With robot actions represented as interpretable action images, control becomes a pixel-grounded visual signal rather than an abstract low-dimensional vector. This converts action modeling into the same video-space problem as observation modeling, yielding a unified video-space representation of observation and action. As shown in Fig. 4, we build a unified world action model by fine-tuning a pretrained video generator (Wan 2.2 [62]) to jointly model multi-view robot videos and action videos.

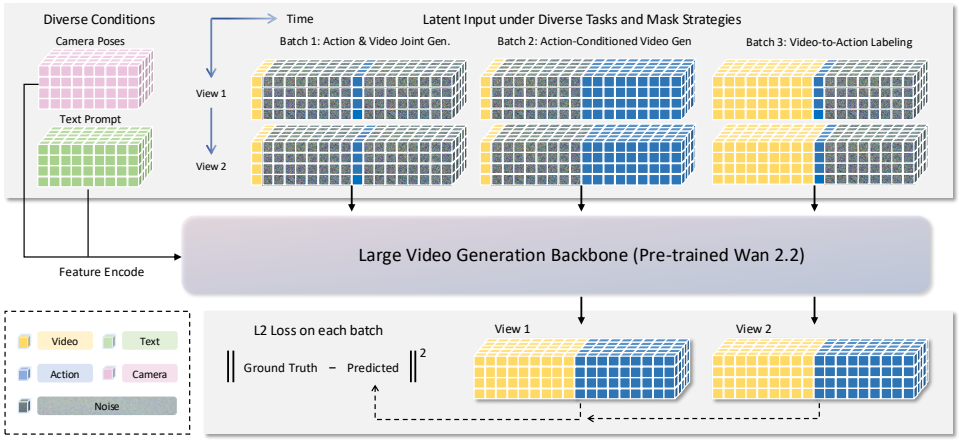

Figure 4: **Unified world-action model training.** Multi-view video and action latents are packed with text and camera conditions, and trained under diverse mask strategies.

Multi-view video-action tokenization and packing. For each camera view v, we have an RGB observation clip $\mathbf{V}_{1:T}^{(v)}$ and the aligned action-image clip $\mathbf{A}_{1:T}^{(v)}$. We first encode both streams into the backbone latent space by the 3D-VAE [33, 62], and then concatenate them temporally to form a single input sequence $\mathbf{X}_v = \left[\text{VAE}(\mathbf{V}_{1:T}^{(v)}),\ \text{VAE}(\mathbf{A}_{1:T}^{(v)})\right]$. We finally concatenate all per-view sequences along the temporal dimension and feed the packed sequence into the backbone.

Unified training via multiple mask strategies. To support multiple tasks with a single model, we adopt a multiple mask strategy in the latent token space (Fig. 4). Concretely, we randomly sample masks over the concatenated latent sequence $\mathbf{X}_k$ to instantiate different training objectives: **1)** Action & video joint generation. We mask both $\mathcal{V}$ and $\mathcal{A}$ tokens except for the first observation frame, and ask the model to generate them jointly conditioned on text and camera inputs. **2)** Action-conditioned video generation. We keep $\mathcal{A}$ visible while masking $\mathcal{V}$, training the model to synthesize future visual frames consistent with provided actions. **3)** Video-to-action labeling. We keep $\mathcal{V}$ visible while masking $\mathcal{A}$, training the model to infer action images from the input video. **4)** Video-only generation. For data without usable action, we train the model with video tokens only. This masking scheme turns the same backbone into a unified world model that can switch behaviors by changing which tokens are observed vs. predicted.

To enable camera-controlled generation, our unified model follows ReCamMaster [4] and injects camera Plücker embeddings [50] into the backbone: $\mathbf{F}_i = \mathbf{F}_o + E_c(\mathbf{cam}_t)$, where E_c is a lightweight convolutional encoder, $\mathbf{F}_o$ is the output of the spatial-attention layer, and $\mathbf{F}_i$ is the input to the subsequent 3D-attention layer.

Optimization objective. We fine-tune the pretrained backbone using a flow matching [44] objective on the masked latent tokens. The target velocity is defined as $\mathbf{v} = \boldsymbol{\epsilon} - \mathbf{X}$. We then minimize an L_2 loss between the predicted and target velocities over the masked tokens:

$$\mathcal{L} \ = \ \mathbb{E}\Big[\big\|M \odot \big(\mathbf{v} - \mathbf{v}_\theta(\mathbf{X}, \mathcal{T}, \mathbf{cam})\big)\big\|_2^2\Big], \tag{5}$$

where M is the mask, $\mathcal{T}$ is the text input, and $\mathbf{cam}$ is the camera condition.

Training datasets. Training a unified world action model requires large-scale data, but this is challenging in robotics: multi-view datasets with well-aligned action and camera annotations are limited. We therefore train on a mixture of RLBench [27], DROID [30], and BridgeV2 [61], which provide complementary supervision as shown in Tab. 1. DROID offers the most complete real-robot annotations, but its camera calibration is often noisy in practice, so we filter out low-quality samples. RLBench, although more toy-like than real-world data, provides highly accurate action and camera signals; we improve its visual diversity with Robot-Colosseum [52] background augmentation. BridgeV2 contains high-quality real-world videos, but lacks camera labels. We estimate camera annotations with VGGT [63] and use BridgeV2 for video-only generation.

Table 1: Summary of dataset for unified world action model training.

Dataset	#Traj.	#Views	Real	Action Ann.	Cam. Calib.	Cam. Motion
DROID	80k	2	✓	✓	✓	Static
RLBench	180k	4	✗	✓	✓	Diverse
BridgeV2	30k	1-4	✓	✓	✗	Static

Table 2: *Zero-shot* evaluation results on RLBench and Real-world settings.

Methods	RLBench				Real-world				
	pick cup	reach target	close drawer	close laptop	place cup	pick unseen toy	pick tissue	close drawer	close box
MV-DP	0	0	0	0	0	0	0	0	0
$\pi_{0.5}$	0	5	35	**20**	5	0	0	0	0
MolmoAct	20	25	10	0	15	5	5	10	0
TesserAct	0	0	0	0	0	0	0	0	0
Cosmos-Policy	0	5	20	0	0	0	0	0	0
Ours	**30**	**60**	**50**	15	40	**20**	**15**	**45**	10
w/ action head	10	**60**	45	15	**50**	10	5	40	**15**

4 Experiments

4.1 Text-Controlled Action & Video Joint Generation

We treat text-controlled action and video joint generation as the primary evaluation setting of this paper. Given a language instruction and the initial multi-view observations, the model jointly generates future robot videos and corresponding multi-view action videos, from which executable controls are obtained by decoding the predicted action images.

Policy evaluation setup. We compare against several representative robot policy baselines, including MV-DP [11], $\pi_{0.5}$ [26], MolmoAct [35], TesserAct [78], and Cosmos-Policy [31]. MV-DP is a multi-view extension of Diffusion Policy that encodes multi-view images. $\pi_{0.5}$ and MolmoAct are VLA-style baselines. For $\pi_{0.5}$, we use the base checkpoint and augment the model with an MLP that injects camera parameters into the VLM. MolmoAct is a reasoning-based model that can predict 2D trajectories on images; we leverage this capability by querying trajectories in multiple views and lifting them into 3D. TesserAct and Cosmos-Policy are world-model-based baselines. We reproduce both by fine-tuning the same Wan 2.2 [62] video backbone on our training set.

Besides the reconstruction-based decoder that recovers actions from generated action images, we conduct a decoder ablation with an optional learned action head on top of our model. The action head is a lightweight network conditioned on output video latents, camera parameters, text embeddings, decoded actions, and observations. It directly regresses the continuous 7-DoF action sequence. This head is used to examine whether the learned representation can support more precise low-level control. We provide further implementation details in Appendix A.

For evaluation, we use task success rate as the metric. Each task is evaluated with 20 samples, using randomly selected camera viewpoints. All experiments are conducted in the multi-view setting under open-loop evaluation. The model should complete the task from a single forward prediction. We consider two settings. **(1)** In the zero-shot setting, the model is evaluated on held-out tasks. In RLBench [27], the evaluated tasks may appear in other datasets, but they are fully removed from the training split; the robot arm and environment are seen. In real-world, the objects, environments, and robot arm are all unseen; the language instructions are similar in form to those seen during training. **(2)** We evaluate in-domain RLBench tasks, where each task is evaluated on held-out test cases.

Policy results and analysis. As shown in Tab. 2, our method delivers the best zero-shot performance, supporting our claim that the learned pixel-grounded action representation provides a more generalizable basis for policy learning. Importantly, this zero-shot capability is already achieved by the reconstruction-based action decoder, indicating that the pixel-grounded representation is an important contributor to the observed zero-shot performance.

The optional action head provides a complementary benefit. In the zero-shot setting, adding the action head yields mixed results: it improves some tasks but does not consistently outperform the reconstruction-based decoder. This suggests that the action head itself is not the primary source of

Table 3: RLBench in-domain tasks evaluation

Methods	close box	close door	open door	phone base	open bottle	close drawer	open oven	open jar	wipe desk	Avg.
MV-DP	20	40	**15**	**20**	5	50	10	0	0	17.8
MolmoAct (zero-shot)	5	10	0	0	5	10	0	0	0	3.3
$\pi_{0.5}$	10	0	5	5	**45**	65	0	0	0	14.4
TesserAct	40	25	5	15	20	70	5	**5**	0	20.6
Cosmos-Policy	40	15	0	15	30	**80**	0	0	0	20.0
Ours	55	60	0	0	5	60	5	0	0	20.6
w/ action head	**80**	**65**	**15**	**20**	40	**80**	**15**	**5**	**10**	**36.7**

Table 4: Video-and-Action Joint Generation Quality. († denotes a zero-shot model.)

Models	Video				Action	
	PSNR ↑	SSIM (%) ↑	FVD ↓	LPIPS ↓	2DErr ↓	3DErr $\times 10^3$ ↓
Cosmos-Predict2.5-14B†	17.92	50.77	208.65	0.409	-	-
Cosmos-Policy	18.29	53.41	192.58	0.418	2.11	19.4
TesserAct	20.83	59.20	154.38	0.351	1.84	19.0
TesserAct-RGB	20.31	60.19	147.83	0.372	**1.55**	14.2
Ours	**23.48**	**78.62**	**143.74**	**0.209**	1.61	**12.2**

zero-shot generalization. In contrast, Tab. 3 shows that the same action head substantially improves in-domain RLBench performance, increasing the average success rate from 20.6 to 36.7. The gains are especially clear on precision-sensitive manipulation tasks, where small errors in continuous 7-DoF control can determine success or failure. Taken together, the two policy evaluations show that our representation provides the generalizable policy, while the optional action head can further exploit this representation to achieve more accurate low-level control when in-domain supervision is available.

Joint generation quality. This experiment focuses on how accurately the model predicts both future robot videos and the corresponding actions. We compare against world models, including Cosmos-Predict [1], Cosmos-Policy [31] and TesserAct [78]. For video quality, we use PSNR and SSIM to measure pixel-level fidelity and structural similarity, with FVD and LPIPS to evaluate perceptual and temporal realism. For action quality, we report both 2D and 3D trajectory error. Since all compared models, except ours, directly predict 3D actions, we project the outputs using camera parameters to obtain 2D errors. Video generation is evaluated on in-domain RLBench, Bridge, and DROID, while action metrics are evaluated on RLBench only. Since RLBench includes dynamic cameras, this evaluation also reflects the model's ability to follow camera controls. As shown in Tab. 4, our method outperforms prior baselines on video metrics while keeping action accuracy.

4.2 Additional Unified-Model Capabilities

Action-conditioned video generation. This task tests whether the model can generate future robot videos when the action sequence is given. We compare with Tora [76], a 2D trajectory-conditioned video generation baseline. We evaluate generation quality using standard video metrics, including PSNR, SSIM, FVD, and LPIPS. As shown in Tab. 5, our method achieves better results on all metrics, suggesting that the unified video-space representation of observation and action can use action inputs more effectively for future video prediction.

Video-to-action labeling. This task tests whether the model can infer action-related motion directly from input videos. We compare with two point-tracking baselines, TAPIR [12] and CoTracker3 [29]. We use standard tracking metrics, including trajectory error, Jaccard@4, and average Jaccard. As shown in Tab. 6, our method outperforms both baselines by a clear margin. This result shows that the pixel-grounded action representation is not only useful for control and generation, but also provides a simple way to label action from video.

Table 5: Action-conditioned video quality.

Models	PSNR ↑	SSIM (%) ↑	FVD ↓	LPIPS ↓
Tora	19.76	52.43	187.41	0.396
Ours	**31.35**	**67.16**	**115.02**	**0.218**

Table 6: Video-to-action labeling results.

Models	Traj Err ↓	Jaccard@4 ↑	Avg. Jaccard ↑
TAPIR	14.80	40.26	29.77
CoTracker3	12.91	46.15	31.20
Ours	**5.79**	**64.92**	**46.71**



Figure 5: Real-world zero-shot rollouts on xArm robot.

4.3 Qualitative Results

We evaluate zero-shot rollouts on an xArm platform, where the objects and environment are unseen. As shown in Fig. 5, we visualize the input image with the predicted tracking trajectories on the left. Our model first generates future observations and multi-view action images (middle), and we then decode the predicted 2D action into a 3D trajectory for point-cloud visualization (right), where the scene geometry is reconstructed by VGGT [63]. The 3D trajectory is colored by time, from blue (earlier) to red (later). We replay the decoded trajectory on the real robot to validate executability. Separately, we include results from a strong video-generation baseline (Veo3.1 [17]) for comparison.

To further test generalization, we sample two images from the F3RM [55] room dataset and prompt the model to perform unseen tasks. Fig. 6 compares our generations with LTX-2-Fast [22, 43]. Our model produces videos with more accurate localization of targets. Notably, the model generates a BridgeV2 robot arm in these scenes together with coherent action images, although BridgeV2 provides no action supervision during training, indicating that **the learned action-generation capability transfers** across datasets and domains. We show more qualitative results in Appendix D.

5 Conclusion

We presented Action Images, a world action model that formulates policy learning as video generation through a unified video-space representation of observation and action. Our key idea is to translate 7-DoF robot control into interpretable action images, yielding a pixel-grounded action in the form of multi-view videos. This design allows the video backbone itself to serve as a zero-shot policy model, without requiring a separate policy head or action module. The same model supports video-action joint generation, action-conditioned video generation, and action labeling under a shared generative

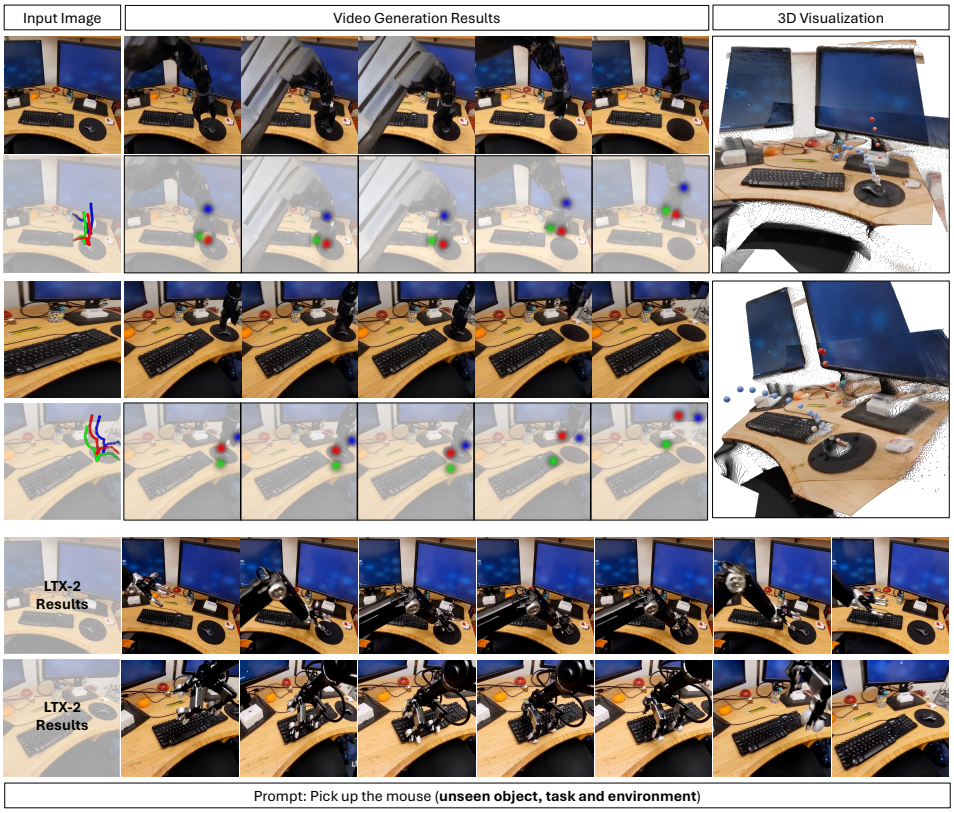

Figure 6: Zero-shot video and action-image generation on F3RM [55] rooms.

framework. We hope this work suggests that grounding action in pixels provides a promising path toward more generalizable policy learning and robotics world modeling in a common video space.

Limitations and future directions. Our current system has several limitations. First, our policy evaluation is still mainly open-loop: the model predicts an action sequence from the initial observation, but has not yet been fully integrated into a closed-loop control pipeline. Recent progress in diffusion acceleration and distillation provides a practical path toward this goal, and future work will distill the model for faster inference and deploy it with online visual feedback.

Second, our current action-image design is instantiated on a 7-DoF single-arm setting. The same pixel-grounded principle could be extended to higher-DoF embodiments, such as bi-arm robots or dexterous hands, by rendering additional semantic points for joints, fingertips, or other kinematic anchors. These richer action images could then be decoded through inverse kinematics or a learned reconstruction model to recover executable action signals.

Third, our current formulation mainly considers external-view action images. A natural extension is to support wrist-camera action prediction. In this setting, action trajectories can be rendered on the initial wrist view, while the robot-video branch predicts the future wrist-camera observations under the moving camera. This would allow the model to represent both egocentric action intent and the dynamic visual feedback induced by robot motion.

Fourth, executing the decoded actions relies on calibrated cameras, and large calibration errors degrade control precision.

Acknowledgments and Disclosure of Funding

We are extremely grateful to Zeyuan Yang, Jiaben Chen, Ziqiao Ma, Aoran Chen, Sriram Krishna, Hongxin Zhang, Zhou Xian, and Theophile Gervet for their helpful feedback and insightful discussions. This work was supported by the University of Massachusetts Amherst. The authors declare no competing interests.

References

[1] Ali, A., Bai, J., Bala, M., Balaji, Y., Blakeman, A., Cai, T., Cao, J., Cao, T., Cha, E., Chao, Y.W., et al.: World simulation with video foundation models for physical ai. arXiv preprint arXiv:2511.00062 (2025)

[2] Assran, M., Bardes, A., Fan, D., Garrido, Q., Howes, R., Muckley, M., Rizvi, A., Roberts, C., Sinha, K., Zholus, A., et al.: V-jepa 2: Self-supervised video models enable understanding, prediction and planning. arXiv preprint arXiv:2506.09985 (2025)

[3] Bahmani, S., Skorokhodov, I., Rong, V., Wetzstein, G., Guibas, L., Wonka, P., Tulyakov, S., Park, J.J., Tagliasacchi, A., Lindell, D.B.: 4d-fy: Text-to-4d generation using hybrid score distillation sampling. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7996–8006 (2024)

[4] Bai, J., Xia, M., Fu, X., Wang, X., Mu, L., Cao, J., Liu, Z., Hu, H., Bai, X., Wan, P., et al.: Recammaster: Camera-controlled generative rendering from a single video. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 14834–14844 (2025)

[5] Bar, A., Zhou, G., Tran, D., Darrell, T., LeCun, Y.: Navigation world models. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 15791–15801 (2025)

[6] Bi, H., Tan, H., Xie, S., Wang, Z., Huang, S., Liu, H., Zhao, R., Feng, Y., Xiang, C., Rong, Y., et al.: Motus: A unified latent action world model. arXiv preprint arXiv:2512.13030 (2025)

[7] Black, K., Brown, N., Driess, D., Esmail, A., Equi, M., Finn, C., Fusai, N., Groom, L., Hausman, K., Ichter, B., et al.: pi_0: A vision-language-action flow model for general robot control. arXiv preprint arXiv:2410.24164 (2024)

[8] Brohan, A., Brown, N., Carbajal, J., Chebotar, Y., Dabis, J., Finn, C., Gopalakrishnan, K., Hausman, K., Herzog, A., Hsu, J., et al.: Rt-1: Robotics transformer for real-world control at scale. arXiv preprint arXiv:2212.06817 (2022)

[9] Brooks, T., Peebles, B., Holmes, C., DePue, W., Guo, Y., Jing, L., Schnurr, D., Taylor, J., Luhman, T., Luhman, E., Ng, C., Wang, R., Ramesh, A.: Video generation models as world simulators (2024), https://openai.com/research/video-generation-models-as-world-simulators

[10] Bruce, J., Dennis, M.D., Edwards, A., Parker-Holder, J., Shi, Y., Hughes, E., Lai, M., Mavalankar, A., Steigerwald, R., Apps, C., et al.: Genie: Generative interactive environments. In: Forty-first International Conference on Machine Learning (2024)

[11] Chi, C., Xu, Z., Feng, S., Cousineau, E., Du, Y., Burchfiel, B., Tedrake, R., Song, S.: Diffusion policy: Visuomotor policy learning via action diffusion. The International Journal of Robotics Research **44**(10-11), 1684–1704 (2025)

[12] Doersch, C., Yang, Y., Vecerik, M., Gokay, D., Gupta, A., Aytar, Y., Carreira, J., Zisserman, A.: Tapir: Tracking any point with per-frame initialization and temporal refinement. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 10061–10072 (2023)

[13] Du, Y., Yang, S., Dai, B., Dai, H., Nachum, O., Tenenbaum, J., Schuurmans, D., Abbeel, P.: Learning universal policies via text-guided video generation. Advances in neural information processing systems **36**, 9156–9172 (2023)

[14] Etukuru, H., Naka, N., Hu, Z., Lee, S., Mehu, J., Edsinger, A., Paxton, C., Chintala, S., Pinto, L., Shafiullah, N.M.M.: Robot utility models: General policies for zero-shot deployment in new environments. In: 2025 IEEE International Conference on Robotics and Automation (ICRA). pp. 8275–8283. IEEE (2025)

[15] Fang, J., Zhao, S.: Usp: A unified sequence parallelism approach for long context generative ai. arXiv preprint arXiv:2405.07719 (2024)

[16] Google DeepMind: Genie 3: A new frontier for world models. Blog post (2025), https://deepmind.google/blog/genie-3-a-new-frontier-for-world-models/

[17] Google DeepMind: Veo: a text-to-video generation system. PDF (2025), https://storage.googleapis.com/deepmind-media/veo/Veo-3-Tech-Report.pdf, accessed: 2026-03-05

[18] Guo, J., Ma, X., Wang, Y., Yang, M., Liu, H., Li, Q.: Flowdreamer: A rgb-d world model with flow-based motion representations for robot manipulation. arXiv preprint arXiv:2505.10075 (2025)

[19] Guo, Y., Hu, Y., Zhang, J., Wang, Y.J., Chen, X., Lu, C., Chen, J.: Prediction with action: Visual policy learning via joint denoising process. Advances in Neural Information Processing Systems **37**, 112386–112410 (2024)

[20] Guo, Y., Shi, L.X., Chen, J., Finn, C.: Ctrl-world: A controllable generative world model for robot manipulation. arXiv preprint arXiv:2510.10125 (2025)

[21] Gupta, A., Murali, A., Gandhi, D.P., Pinto, L.: Robot learning in homes: Improving generalization and reducing dataset bias. Advances in neural information processing systems **31** (2018)

[22] HaCohen, Y., Brazowski, B., Chiprut, N., Bitterman, Y., Kvochko, A., Berkowitz, A., Shalem, D., Lifschitz, D., Moshe, D., Porat, E., et al.: LTX-2: Efficient joint audio-visual foundation model. arXiv preprint arXiv:2601.03233 (2026)

[23] Ho, J., Salimans, T.: Classifier-free diffusion guidance. arXiv preprint arXiv:2207.12598 (2022)

[24] Hu, Y., Guo, Y., Wang, P., Chen, X., Wang, Y.J., Zhang, J., Sreenath, K., Lu, C., Chen, J.: Video prediction policy: A generalist robot policy with predictive visual representations. arXiv preprint arXiv:2412.14803 (2024)

[25] Intelligence, P., Amin, A., Aniceto, R., Balakrishna, A., Black, K., Conley, K., Connors, G., Darpinian, J., Dhabalia, K., DiCarlo, J., et al.: $\pi^*_{0.6}$: a VLA that learns from experience. arXiv preprint arXiv:2511.14759 (2025)

[26] Intelligence, P., Black, K., Brown, N., Darpinian, J., Dhabalia, K., Driess, D., Esmail, A., Equi, M., Finn, C., Fusai, N., et al.: $\pi 0.5$: a vision-language-action model with open-world generalization. arXiv preprint arXiv:2504.16054 (2025)

[27] James, S., Ma, Z., Arrojo, D.R., Davison, A.J.: Rlbench: The robot learning benchmark & learning environment. IEEE Robotics and Automation Letters **5**(2), 3019–3026 (2020)

[28] Jin, Y., Peng, S., Wang, X., Xie, T., Xu, Z., Yang, Y., Shen, Y., Bao, H., Zhou, X.: Diffuman4d: 4d consistent human view synthesis from sparse-view videos with spatio-temporal diffusion models. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 11047–11057 (2025)

[29] Karaev, N., Makarov, Y., Wang, J., Neverova, N., Vedaldi, A., Rupprecht, C.: Cotracker3: Simpler and better point tracking by pseudo-labelling real videos. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6013–6022 (2025)

[30] Khazatsky, A., Pertsch, K., Nair, S., Balakrishna, A., Dasari, S., Karamcheti, S., Nasiriany, S., Srirama, M.K., Chen, L.Y., Ellis, K., et al.: Droid: A large-scale in-the-wild robot manipulation dataset. arXiv preprint arXiv:2403.12945 (2024)

[31] Kim, M.J., Gao, Y., Lin, T.Y., Lin, Y.C., Ge, Y., Lam, G., Liang, P., Song, S., Liu, M.Y., Finn, C., et al.: Cosmos policy: Fine-tuning video models for visuomotor control and planning. arXiv preprint arXiv:2601.16163 (2026)

[32] Kim, M.J., Pertsch, K., Karamcheti, S., Xiao, T., Balakrishna, A., Nair, S., Rafailov, R., Foster, E., Lam, G., Sanketi, P., et al.: Openvla: An open-source vision-language-action model. arXiv preprint arXiv:2406.09246 (2024)

[33] Kingma, D.P., Welling, M.: Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114 (2013)

[34] Ko, P.C., Mao, J., Du, Y., Sun, S.H., Tenenbaum, J.B.: Learning to act from actionless videos through dense correspondences. arXiv preprint arXiv:2310.08576 (2023)

[35] Lee, J., Duan, J., Fang, H., Deng, Y., Liu, S., Li, B., Fang, B., Zhang, J., Wang, Y.R., Lee, S., et al.: Molmoact: Action reasoning models that can reason in space. arXiv preprint arXiv:2508.07917 (2025)

[36] Li, C., Krause, A., Hutter, M.: Robotic world model: A neural network simulator for robust policy optimization in robotics. arXiv preprint arXiv:2501.10100 (2025)

[37] Li, L., Zhang, Q., Luo, Y., Yang, S., Wang, R., Han, F., Yu, M., Gao, Z., Xue, N., Zhu, X., et al.: Causal world modeling for robot control. arXiv preprint arXiv:2601.21998 (2026)

[38] Li, P., Chen, Y., Xu, Y., Yang, J., Wu, X., Guo, J., Sun, N., Qian, L., Li, X., Xiao, X., Liu, J., Liu, N., Kong, T., Huang, Y., Wang, L., Tan, T.: Multi-view video diffusion policy: A 3d spatio-temporal-aware video action model (2026), https://arxiv.org/abs/2604.03181

[39] Li, S., Gao, Y., Sadigh, D., Song, S.: Unified video action model. arXiv preprint arXiv:2503.00200 (2025)

[40] Li, Y., Luo, Z., Zhang, T., Dai, C., Kanervisto, A., Tirinzoni, A., Weng, H., Kitani, K., Guzek, M., Touati, A., et al.: Bfm-zero: A promptable behavioral foundation model for humanoid control using unsupervised reinforcement learning. arXiv preprint arXiv:2511.04131 (2025)

[41] Li, Z., Zhang, M., Wu, T., Tan, J., Wang, J., Lin, D.: Ss4d: Native 4d generative model via structured spacetime latents. ACM Transactions on Graphics (TOG) **44**(6), 1–12 (2025)

[42] Liang, J., Tokmakov, P., Liu, R., Sudhakar, S., Shah, P., Ambrus, R., Vondrick, C.: Video generators are robot policies. arXiv preprint arXiv:2508.00795 (2025)

[43] Lightricks: Ltx studio. Online (2024), https://app.ltx.studio/, accessed: 2026-02

[44] Lipman, Y., Chen, R.T., Ben-Hamu, H., Nickel, M., Le, M.: Flow matching for generative modeling. arXiv preprint arXiv:2210.02747 (2022)

[45] Liu, Z., Li, S., Cousineau, E., Feng, S., Burchfiel, B., Song, S.: Geometry-aware 4d video generation for robot manipulation. arXiv preprint arXiv:2507.01099 (2025)

[46] Ljung, L., Glad, T.: Modeling of dynamic systems. Prentice-Hall, Inc. (1994)

[47] Nakamoto, M., Mees, O., Kumar, A., Levine, S.: Steering your generalists: Improving robotic foundation models via value guidance. arXiv preprint arXiv:2410.13816 (2024)

[48] Pan, Z., Yang, Z., Zhu, X., Zhang, L.: Efficient4d: Fast dynamic 3d object generation from a single-view video. arXiv preprint arXiv:2401.08742 (2024)

[49] Pearce, T., Rashid, T., Kanervisto, A., Bignell, D., Sun, M., Georgescu, R., Macua, S.V., Tan, S.Z., Momennejad, I., Hofmann, K., et al.: Imitating human behaviour with diffusion models. arXiv preprint arXiv:2301.10677 (2023)

[50] Plucker, J.: Xvii. on a new geometry of space. Philosophical Transactions of the Royal Society of London (155), 725–791 (1865)

[51] Polydoros, A.S., Nalpantidis, L.: Survey of model-based reinforcement learning: Applications on robotics. Journal of Intelligent & Robotic Systems **86**(2), 153–173 (2017)

[52] Pumacay, W., Singh, I., Duan, J., Krishna, R., Thomason, J., Fox, D.: The colosseum: A benchmark for evaluating generalization for robotic manipulation. arXiv preprint arXiv:2402.08191 (2024)

[53] Rajbhandari, S., Rasley, J., Ruwase, O., He, Y.: Zero: Memory optimizations toward training trillion parameter models. In: SC20: international conference for high performance computing, networking, storage and analysis. pp. 1–16. IEEE (2020)

[54] Ren, J., Pan, L., Tang, J., Zhang, C., Cao, A., Zeng, G., Liu, Z.: Dreamgaussian4d: Generative 4d gaussian splatting. arXiv preprint arXiv:2312.17142 (2023)

[55] Shen, W., Yang, G., Yu, A., Wong, J., Kaelbling, L.P., Isola, P.: Distilled feature fields enable few-shot language-guided manipulation. arXiv preprint arXiv:2308.07931 (2023)

[56] Shridhar, M., Lo, Y.L., James, S.: Generative image as action models. arXiv preprint arXiv:2407.07875 (2024)

[57] Singer, U., Sheynin, S., Polyak, A., Ashual, O., Makarov, I., Kokkinos, F., Goyal, N., Vedaldi, A., Parikh, D., Johnson, J., et al.: Text-to-4d dynamic scene generation. arXiv preprint arXiv:2301.11280 (2023)

[58] Sun, Q., Yang, L., Tang, W., Huang, W., Xu, K., Chen, Y., Liu, M., Yang, J., Zhu, H., Wang, Y., et al.: Learning primitive embodied world models: Towards scalable robotic learning. arXiv preprint arXiv:2508.20840 (2025)

[59] Sutton, R.S.: Dyna, an integrated architecture for learning, planning, and reacting. ACM Sigart Bulletin **2**(4), 160–163 (1991)

[60] Team, O.M., Ghosh, D., Walke, H., Pertsch, K., Black, K., Mees, O., Dasari, S., Hejna, J., Kreiman, T., Xu, C., et al.: Octo: An open-source generalist robot policy. arXiv preprint arXiv:2405.12213 (2024)

[61] Walke, H.R., Black, K., Zhao, T.Z., Vuong, Q., Zheng, C., Hansen-Estruch, P., He, A.W., Myers, V., Kim, M.J., Du, M., et al.: Bridgedata v2: A dataset for robot learning at scale. In: Conference on Robot Learning. pp. 1723–1736. PMLR (2023)

[62] Wang, A., Ai, B., Wen, B., Mao, C., Xie, C.W., Chen, D., Yu, F., Zhao, H., Yang, J., Zeng, J., et al.: Wan: Open and advanced large-scale video generative models. arXiv preprint arXiv:2503.20314 **3**(4), 6 (2025)

[63] Wang, J., Chen, M., Karaev, N., Vedaldi, A., Rupprecht, C., Novotny, D.: Vggt: Visual geometry grounded transformer. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 5294–5306 (2025)

[64] Watkins, O., Huang, S., Frost, J., Bhatia, K., Weiner, E., Abbeel, P., Darrell, T., Plummer, B., Saenko, K., Dragan, A.: Explaining robot policies. Applied AI Letters **2**(4), e52 (2021)

[65] Won, J., Lee, K., Jang, H., Kim, D., Shin, J.: Dual-stream diffusion for world-model augmented vision-language-action model. arXiv preprint arXiv:2510.27607 (2025)

[66] Wu, H., Wu, D., He, T., Guo, J., Ye, Y., Duan, Y., Bian, J.: Geometry forcing: Marrying video diffusion and 3d representation for consistent world modeling. arXiv preprint arXiv:2507.07982 (2025)

[67] Wu, P., Escontrela, A., Hafner, D., Abbeel, P., Goldberg, K.: Daydreamer: World models for physical robot learning. In: Conference on robot learning. pp. 2226–2240. PMLR (2023)

[68] Wu, R., Gao, R., Poole, B., Trevithick, A., Zheng, C., Barron, J.T., Holynski, A.: CAT4D: Create Anything in 4D with Multi-View Video Diffusion Models. arXiv:2411.18613 (2024)

[69] Xie, Y., Yao, C.H., Voleti, V., Jiang, H., Jampani, V.: Sv4d: Dynamic 3d content generation with multi-frame and multi-view consistency. arXiv preprint arXiv:2407.17470 (2024)

[70] Xing, Y., Luo, X., Xie, J., Gao, L., Shen, H., Song, J.: Shortcut learning in generalist robot policies: The role of dataset diversity and fragmentation. arXiv preprint arXiv:2508.06426 (2025)

[71] Yang, L., Bai, Y., Eskandar, G., Shen, F., Altillawi, M., Chen, D., Liu, Z., Valada, A.: Covar: Co-generation of video and action for robotic manipulation via multi-modal diffusion. arXiv preprint arXiv:2512.16023 (2025)

[72] Yang, Z., Gao, X., Zhou, W., Jiao, S., Zhang, Y., Jin, X.: Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 20331–20341 (2024)

[73] Ye, S., Ge, Y., Zheng, K., Gao, S., Yu, S., Kurian, G., Indupuru, S., Tan, Y.L., Zhu, C., Xiang, J., et al.: World action models are zero-shot policies. arXiv preprint arXiv:2602.15922 (2026)

[74] Zhang, H., Chen, X., Wang, Y., Liu, X., Wang, Y., Qiao, Y.: 4diffusion: Multi-view video diffusion model for 4d generation. Advances in Neural Information Processing Systems **37**, 15272–15295 (2024)

[75] Zhang, W., Li, Y., Qiao, Y., Huang, S., Liu, J., Dayoub, F., Ma, X., Liu, L.: Effective tuning strategies for generalist robot manipulation policies. In: 2025 IEEE International Conference on Robotics and Automation (ICRA). pp. 7255–7262. IEEE (2025)

[76] Zhang, Z., Liao, J., Li, M., Dai, Z., Qiu, B., Zhu, S., Qin, L., Wang, W.: Tora: Trajectory-oriented diffusion transformer for video generation. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 2063–2073 (2025)

[77] Zhen, H., Qiu, X., Chen, P., Yang, J., Yan, X., Du, Y., Hong, Y., Gan, C.: 3d-vla: A 3d vision-language-action generative world model. arXiv preprint arXiv:2403.09631 (2024)

[78] Zhen, H., Sun, Q., Zhang, H., Li, J., Zhou, S., Du, Y., Gan, C.: Tesseract: learning 4d embodied world models. arXiv preprint arXiv:2504.20995 (2025)

[79] Zheng, D., Huang, S., Zhao, L., Zhong, Y., Wang, L.: Towards learning a generalist model for embodied navigation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 13624–13634 (2024)

[80] Zhou, S., Du, Y., Chen, J., Li, Y., Yeung, D.Y., Gan, C.: Robodreamer: Learning compositional world models for robot imagination. arXiv preprint arXiv:2404.12377 (2024)

[81] Zhu, C., Yu, R., Feng, S., Burchfiel, B., Shah, P., Gupta, A.: Unified world models: Coupling video and action diffusion for pretraining on large robotic datasets. arXiv preprint arXiv:2504.02792 (2025)

[82] Zhu, H., Wang, Y., Zhou, J., Chang, W., Zhou, Y., Li, Z., Chen, J., Shen, C., Pang, J., He, T.: Aether: Geometric-aware unified world modeling. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 8535–8546 (2025)

[83] Zitkovich, B., Yu, T., Xu, S., Xu, P., Xiao, T., Xia, F., Wu, J., Wohlhart, P., Welker, S., Wahid, A., et al.: Rt-2: Vision-language-action models transfer web knowledge to robotic control. In: Conference on Robot Learning. pp. 2165–2183. PMLR (2023)

A Implementation Details

Training Details. We trained our unified world-action model by fine-tuning a pretrained Wan 2.2 (5B) [62] backbone. The training data comprised Bridge [61], RLBench [27], and DROID [30], sampled with mixture ratios of 0.2, 0.5, and 0.3, respectively. Each training sample contained 41 frames for a single view and a single modality; under the full two-view setting with both robot videos and action videos, this corresponds to 164 frames in total. We sampled hierarchically: a dataset was first drawn according to the mixture ratios above, and a training task was then drawn within it. For RLBench and DROID, 85% of samples were used for joint generation, while video-only, action-label, and action-conditioned generation each accounted for 5%; BridgeV2 samples were used only for video-only generation (Sec. 3.3). Training was conducted on 32 A100 GPUs using DeepSpeed ZeRO [53], bfloat16 mixed precision, and gradient checkpointing. We used a per-device batch size of 1. The optimizer used a constant-with-warmup schedule with a learning rate of 5×10^{-7}, a warmup of 1000 steps, and gradient clipping with a maximum norm of 1.0. We trained the model for 100,000 optimization steps.

For camera conditioning, we followed the design of ReCamMaster [4], except that we used Plücker embeddings [50] as the camera representation. The camera encoder first pools the spatiotemporal camera features to a fixed resolution of $48 \times (H_l/2) \times (W_l/2)$, where H_l and W_l denote the latent spatial resolution of the Wan video backbone. The pooled features are then flattened and projected into the model hidden dimension dim by a linear projector. Here, dim is set to the default hidden dimension of the Wan video model. We initialized the encoder projection to zeros and the final projector as an identity mapping, which stabilizes optimization at the beginning of training.

Lightweight Action Head. In addition to the reconstruction-based decoder used by our main model, we also evaluate an optional lightweight action head to test whether the generated video-action representation can support more precise low-level control. The head is used only in the w/ action head ablation. It takes five modalities as input: multi-view RGB observations, camera parameters, output video latents from the world model, text embeddings, and the action sequence predicted by our model. Each modality is encoded by a lightweight branch and projected to a shared hidden dimension d_h. The resulting features are concatenated and decoded by a two-layer MLP to predict the final continuous action sequence of length T_a. Concretely, we use $V = 2$ camera views. RGB observations are encoded by a shared ResNet-18 encoder, producing a 512-dimensional feature per view. The per-view features are concatenated and projected to d_h. Camera parameters consist of a flattened 4×4 extrinsic matrix and a 3×3 intrinsic matrix for each view, giving $25V$ camera dimensions, which are encoded by an MLP. The output video latent from the Wan backbone has channel dimension d_b and is encoded by a 3D convolution followed by global spatiotemporal average pooling and an MLP. Text features are obtained by mean-pooling the Wan text-token embeddings of dimension 4096 and projecting them to d_h. The model-predicted action sequence $\mathbf{a}^{\mathrm{model}}_{1:T_a} \in \mathbb{R}^{T_a \times d_a}$ is flattened and encoded by another MLP. Finally, the five d_h-dimensional embeddings are concatenated into a $5d_h$-dimensional vector and mapped to $T_a d_a$ outputs, which are reshaped into the refined action sequence $\hat{\mathbf{a}}_{1:T_a} \in \mathbb{R}^{T_a \times d_a}$.

Table 7: Architecture of the lightweight action head. All modality embeddings are projected to the shared hidden dimension d_h before fusion. In our implementation, $V = 2$, $d_b = 1536$, and the default d_h is 512.

Modality	Input shape	Encoder
RGB observations	$B \times V \times 3 \times H \times W$	Shared ResNet-18 + MLP
Camera parameters	$B \times V \times 25$	Flatten + MLP
World-model latent	$B \times d_b \times T_l \times H_l \times W_l$	3D Conv ($d_b \to 256$) + AvgPool + MLP
Text embedding	$B \times L \times 4096$	Mean pool over tokens + MLP
Model-predicted action	$B \times T_a \times d_a$	Flatten + MLP
Fusion	$B \times 5d_h$	MLP ($5d_h \to 2d_h \to T_a d_a$)

Inference Details. At inference time, we keep the input formatting and spatial-temporal configuration same with training. We use classifier-free guidance [23] with a scale of 10.0, and perform sampling for 50 denoising steps. In all experiments, inference is executed with 8-GPU Unified Sequence Parallelism [15]. For in-the-wild images without camera annotations, we estimate camera extrinsics

and intrinsics using VGGT [63]. As shown in Table 8, we further improve inference throughput by introducing several system-level optimizations, including CFG parallelism, VAE parallelism, caching, and `torch.compile`. With these optimizations, the video backbone reaches up to 71 FPS. Note that these optimizations only illustrate the achievable speed; all results reported in the main paper use the default 50-step sampling. We also note that although DreamZero-Flash achieves extremely fast inference, it relies on highly aggressive denoising steps, which leads to a severe degradation in video quality.

Table 8: Inference efficiency.

Models	Size	GPU	Steps	#Frames	Image Res.	Inference Time (s)
TesserAct (original) [78]	5B	1 H100	50	49	(480, 640)	137.5
DreamZero [73]	14B	1 H100	16	48	(176, 320)	5.7
DreamZero-Flash	14B	2 GB200	1	48	(176, 320)	0.15
Ours	5B	1 H100	50	164	(512, 512)	49.1
+ Parallelism	5B	8 H100	50	164	(512, 512)	11.8
+ Caching	5B	8 H100	16	164	(512, 512)	2.3

Action Images Details. Following Sec. 3.1, each robot action is converted into three semantic 3D points (position, normal, and up) and projected into image space. The normal and up points are placed at a fixed distance of 0.1 m from the position point along their directions. The projected 2D points are then rasterized as Gaussian heatmaps with a standard deviation $\sigma = 0.05$ relative to the image resolution. In practice, we observe that moderate changes to these hyperparameters do not noticeably affect performance, as long as the projected points remain within the image plane.

Decoding Details. For decoding generated action images back to continuous actions, we use the same camera extrinsics and intrinsics as those used for rendering. The ray-sampling range is set to $[0.5, 1.0]$ m, with 512 uniformly spaced depth samples. These settings are kept fixed across all experiments. We additionally ablate the number of depth samples used during decoding, and report the success rate under both model-predicted and ground-truth heatmaps in Table 9.

Table 9: Ablation of the number of depth samples for action-image decoding on in-domain RLBench tasks (success rate, %).

Task	Model-predicted heatmaps				Ground-truth heatmaps			
	128	256	512	1024	128	256	512	1024
Close drawer	30	45	60	60	85	100	100	100
Pick cup	20	30	30	30	55	95	100	100

B Multi-view Consistency

We further evaluate the multi-view geometric consistency of generated videos using VGGT depth confidence. For each generated timestep, we run VGGT inference over the generated views and compute the mean depth confidence. We compare three settings. "Single view" denotes that the model independently generates each view without explicit multi-view joint generation. The metric ranges from 1 to $+\infty$, where values above 3 generally indicate strong geometric consistency. Since many robot videos contain largely static backgrounds, all methods fall within a reasonably good range; nevertheless, the differences still reveal the benefit of modeling views jointly.

TesserAct single view (reproduced on Wan 2.2) obtains a mean depth confidence of 3.309, while our single-view variant gets to 3.131. With full multi-view generation, our model increases the score to 3.718, moving closer to the ground-truth score of 4.815. These results suggest that our multi-view formulation produces more geometrically coherent generations than independently generated single-view videos.

C Action-Image and Multi-view Ablation

Table 10 ablates action images, the number of views, and the action head on two zero-shot (ZS) and two in-domain (ID) RLBench tasks. Without action images, the head directly regresses 7-DoF actions from video features. With a single action-image view, depth is underdetermined, so the geometric decoder cannot recover a 7-DoF action. Zero-shot success only emerges with action images, and the second view helps mainly when paired with action images.

Table 10: Ablation of action images, views, and the action head (success rate, %). –: not decodable from a single view.

Views	Action image	Action head	ZS: `pick cup`	ZS: `reach target`	ID: `close box`	ID: `open bottle`
1	✗	✓	0	0	40	20
2	✗	✓	0	0	35	20
1	✓	✗	–	–	–	–
2	✓	✗	30	60	55	5
1	✓	✓	0	0	50	15
2	✓	✓	10	60	80	40

D More Zero-shot Qualitative Results

We provide the complete video results on our project page (`https://ActionImages.github.io`), where readers can view the generated videos and rollout examples corresponding to our qualitative results.

First, Fig. 7 shows action labeling results given input videos, including one π_0 [7] robot video and one Genie 3 [10, 16] human-hand video, demonstrating that our model can handle both.

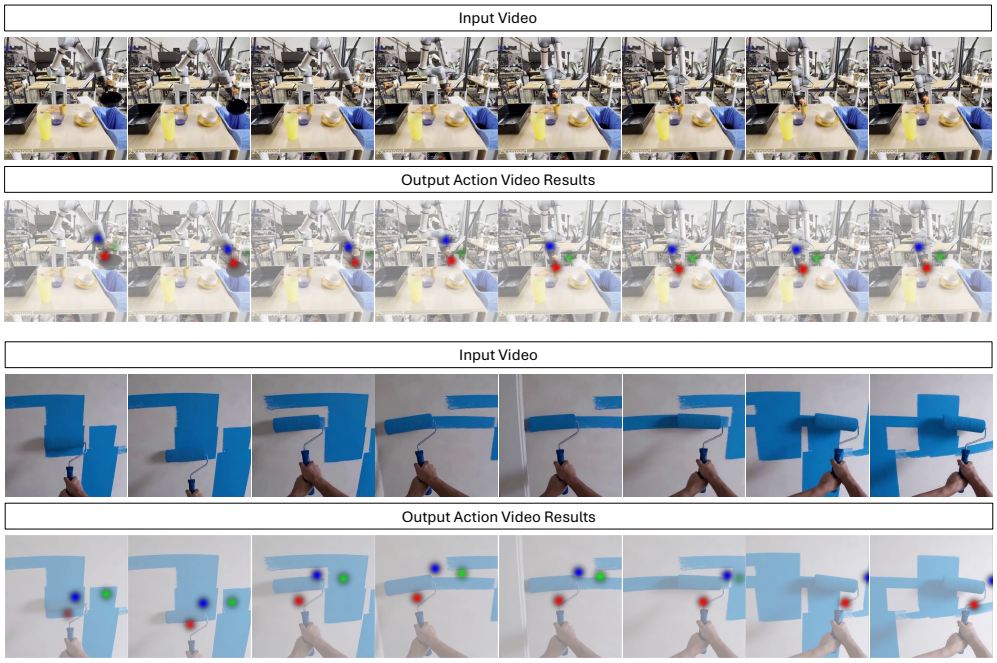

Figure 7: Action labeling results.

Fig. 8 provides more qualitative robot manipulation results, mainly on grasping tasks across diverse objects and scenes.

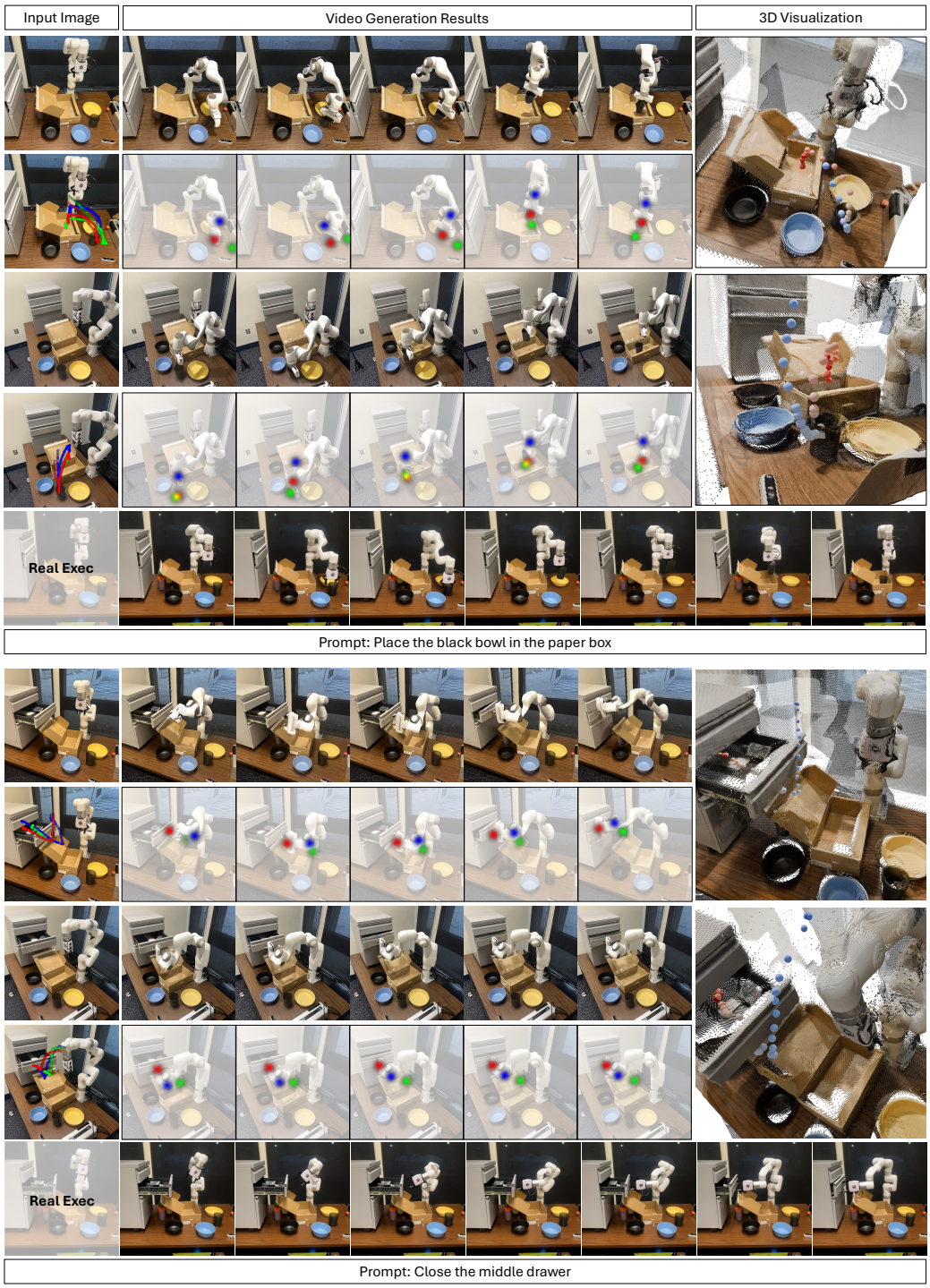

Figure 8: Additional robot manipulation results, mainly on grasping tasks.

We then show camera control results in Fig. 9, where the model is given an input image and a task, and generates videos with controlled viewpoint changes in complex scenes from the π_0 website.

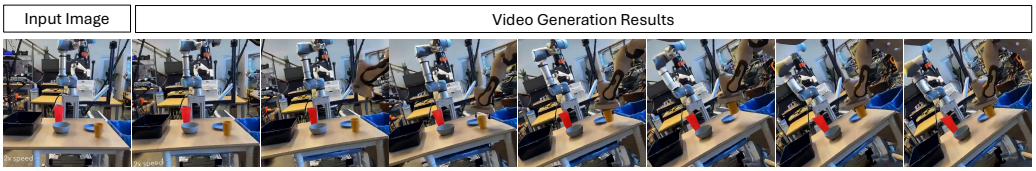

Figure 9: Camera control results in complex scenes from the π_0 website.

Finally, we show action-conditioned generation results in Fig. 10, where we use the first frame from π_0 demo videos as input to generate future videos conditioned on actions.

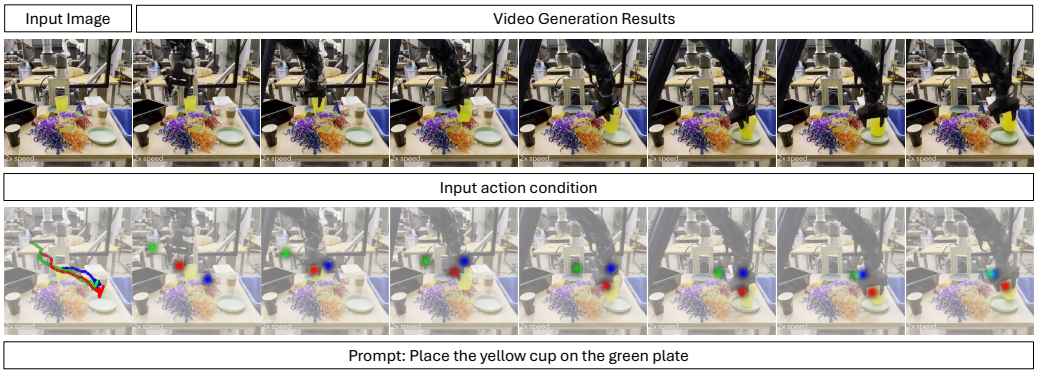

Figure 10: Action-conditioned generation results.

E Broader Impacts

This work aims to make robot policy learning more generalizable by reusing the knowledge of pretrained video models, which could lower the cost of deploying robots for assistive and household manipulation. However, policies derived from generative models can be unreliable: our evaluation is open-loop, and an erroneous predicted trajectory executed on a physical robot could damage objects or harm nearby people. Real-world deployment should therefore include safety constraints, human supervision, and careful validation. In addition, as with other video generation models, the video synthesis capability could be misused to produce misleading content.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the main contributions and scope, including the pixel-grounded action-image representation, the zero-shot policy setting, and the unified video-action generation tasks. These claims are supported by the method description and experimental results in Secs. 3–4.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses limitations of the current approach, including potential challenges in higher-DoF control, closed-loop deployment, and settings involving wrist-camera observations. It also analyzes possible future directions and mitigation strategies for addressing these limitations.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper does not present theoretical results, theorems, or formal proofs; its contributions are methodological and empirical.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper provides the main methodological details in Sec. 3 and gives detailed implementation information in Appendix A, including the training datasets and mixture ratios, input formatting, task mixture, optimization setup, training schedule, and inference configuration. Together with the evaluation protocol and metrics in Sec. 4, these details disclose the information needed to reproduce the main experimental results.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: The paper uses existing datasets and pretrained models; the code is not released with the paper. Video results are available on the project page.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper specifies the experimental setup, datasets, baselines, metrics, and evaluation protocol in Sec. 4, and provides detailed training and inference settings in Appendix. These include data mixture ratios, frame configuration, task mixture, optimizer schedule, learning rate, warmup, gradient clipping, training steps, and inference sampling settings.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

Justification: The reported tables provide single values for success rates and generation metrics, but the paper does not report error bars, confidence intervals, statistical significance tests, or the sources of experimental variability.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix A reports the main training compute setup, including training on 32 A100 GPUs with DeepSpeed ZeRO, bfloat16 mixed precision, and gradient checkpointing. It also reports inference compute and latency details, including Sequence Parallelism and the hardware/time comparison.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conforms to the NeurIPS Code of Ethics. The paper studies robot policy learning and video-based world-action modeling using existing robotics datasets and simulation or controlled real-world robot evaluations.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.

- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Appendix E discusses the broader impacts of learning generalizable robot policies from video-space action representations. Potential positive impacts include improving the generalization and usability of robot learning systems, reducing the need for task-specific policy heads, and enabling more flexible robot manipulation. Potential negative impacts include risks from deploying open-loop or imperfect robot policies in safety-critical physical environments, possible misuse of generative robot-control models, and the need for careful validation before real-world deployment.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release a new high-risk pretrained model, scraped dataset, or open-ended generative system as part of this paper. The work uses existing robotics datasets and pretrained video-generation components for research on robot policy learning. Therefore, responsible-release safeguards for newly released high-risk assets are not applicable.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.

- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The paper uses existing datasets and models, all cited in the paper: RLBench (RLBench Software Licence, non-commercial research use), DROID (CC BY 4.0), Bridge-Data V2 (data: CC BY 4.0; code: MIT), The Colosseum (no explicit license; built on RLBench), Wan 2.2 (Apache 2.0), and VGGT-1B (CC BY-NC 4.0). All assets are used for non-commercial research in accordance with their terms.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not introduce or release a new dataset, benchmark, code package, or model checkpoint as a public asset. The contribution is methodological and empirical, based on existing robotics datasets and pretrained models.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

Justification: The paper does not describe crowdsourcing experiments or research involving human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The paper does not describe crowdsourcing experiments or research involving human subjects, so IRB approval or equivalent review is not applicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The core method is based on video generation and robotics action representations; LLMs are not an important, original, or non-standard component of the research methodology.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.