Point Tracking Improves World Action Models

Jiarui Guan[1] **Wenshuai Zhao**[1,2,†] **Yue Pei**[3] **Ziliang Chen**[4,5,†]
Arno Solin[1,2] **Juho Kannala**[1,6]

[1]Aalto University [2]ELLIS Institute Finland [3]Beihang University
[4]Sun Yat-sen University [5]Peng Cheng Laboratory [6]University of Oulu

{jiarui.guan, wenshuai.zhao, juho.kannala, arno.solin}@aalto.fi
c.ziliang@yahoo.com peiyue@buaa.edu.cn

Code: https://github.com/1mather/JOPAT_WAM

Abstract

Robot policy learning benefits from world-action models that capture environment dynamics, but pixel-level prediction entangles dynamics with nuisance factors such as lighting and texture, making learned representations vulnerable to task-irrelevant visual variation. We propose JOPAT, a JOint Pixel-And-Track World-Action Model that predicts latent visual observations, 2D point tracks with visibility, and actions in a single denoising diffusion transformer. The key insight is that tracks provide an explicit representation of motion that captures long-horizon dynamics and remains robust under occlusion or partial out-of-frame motion, offering greater utility than modeling pixel appearance alone. On LIBERO and real-world LeRobot tasks, JOPAT improves over pixel-based baselines, with the largest gains on long-horizon tasks involving occlusion, object interaction, and off-screen motion.

1 Introduction

Robotic manipulation in the open world demands policies that generalize across objects, scenes, and viewpoints while remaining reliable under partial observability, occlusion, and long-horizon temporal dependencies. Vision-Language-Action (VLA) models represent a major step toward this goal, which leverages pretrained Vision-Language Models (VLMs) to map language instructions and visual observations directly into robot actions [1–9]. However, despite their strong semantic capabilities, VLAs are predominantly trained on image–language data and therefore lack an understanding of environment dynamics, which is crucial for robotic policy learning [10–14].

Recently, World Action Models (WAMs) have emerged as a promising paradigm for translating video priors into robot control [15–29]. Large-scale visual pre-training provides an effective way to acquire motion priors and physical grounding from diverse visual dynamics [30–35]. However, how to effectively translate video dynamics into generalizable robotic policies still remains an open question. Several studies suggest that pixel-level pretraining on action-free videos does not necessarily improve policy success rates [14, 36–38].

We shift the central question from *how to model the world* to *what future state should serve as the interface between video prediction and control*. Our key insight is that action-free video becomes useful for control when its supervision is expressed through a future-state abstraction that preserves visual grounding while exposing controllable correspondences. To this end, we propose the *JOint Pixel-And-Track World-Action Model* (JOPAT), which jointly denoises visual latents, point tracks with visibility, and executable robot actions. In this way, we use 2D point tracks with visibility as the motion representation: tracks encode persistent scene correspondences over time, while visibility

[†]Corresponding authors.

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

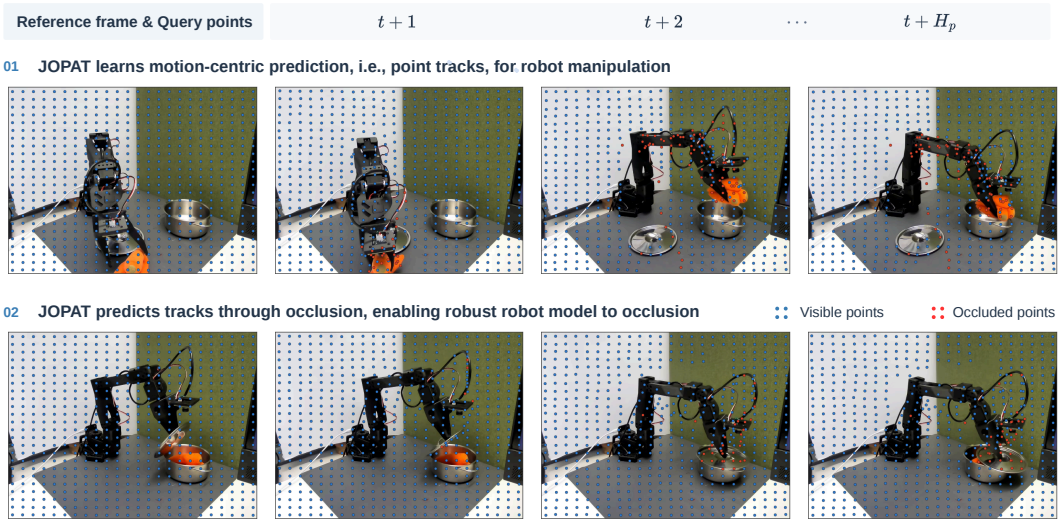

Figure 1: **JOPAT predicts structured point tracks beyond visible pixels.** Given query points in a reference frame, JOPAT predicts their long-horizon 2D trajectories and corresponding visibility logits. By modeling scene dynamics in track space, our method captures action-relevant motion while explicitly accounting for points that become occluded or move outside the field of view.

indicates when these correspondences become occluded or leave the camera view. Together with visual latents, this pixel–track future state preserves object identity, affordance, and appearance cues, while explicitly representing object displacement, contact-induced motion, occlusion, and out-of-frame transitions, as illustrated in Fig. 1.

We evaluate JOPAT on both simulated and real-world manipulation benchmarks under visual out-of-distribution variations, self-induced occlusions, and long-horizon dependencies. On 40 LIBERO tasks [39], JOPAT achieves an average success rate of 97.8%, establishing a new state of the art. It also demonstrates improved robustness on modified LIBERO settings with occlusions and disturbances. Real-world experiments and extensive ablation studies further validate the effectiveness of the proposed joint pixel-and-track modeling. Overall, integrating pixel-based visual representations with explicit point-track dynamics yields more reliable control than either signal alone, while enabling effective policy learning from limited robot demonstrations.

Our main contributions are: *(i)* we propose JOPAT, a joint pixel-and-track world-action model that integrates visual latents, 2D point tracks with visibility, and robot actions within a unified generative framework, enabling correspondence-level motion to serve as an explicit interface between perception and control; *(ii)* we achieve state-of-the-art performance on 40 LIBERO manipulation tasks with an average success rate of 97.8%, while demonstrating strong robustness under visual distribution shifts, occlusions, disturbances, and long-horizon dependencies in both simulated and real-world settings; and *(iii)* through extensive ablation studies and real-world experiments, we show that explicit point-track modeling is crucial for robust performance, as it improves long-horizon consistency, mitigates the effects of occlusion, and provides stable motion grounding that complements pixel-based representations for control.

2 Related work

Video-based world models and world action models Video-based world models learn predictive dynamics from future observations and have long been used as a route to physical interaction and control [40–42]. Recent robotic world-action models further couple video prediction with action generation, using shared generative models, diffusion objectives, or policy optimization to connect visual futures to executable controls [15, 16, 22–27]. These methods show that predictive visual objectives can improve policy learning, but most instantiate the future state in pixel, latent-image, or visual-token spaces. As a result, the motion variables that determine actions—which object moved, where it moved, and whether the evidence became hidden—must be recovered implicitly

from appearance-dominated tokens. JOPAT differs by augmenting the world-action future state with explicit point tracks and visibility, so action tokens interact with sampled correspondence-level motion variables rather than with visual latents alone.

Intermediate motion representations for control A growing line of work introduces intermediate motion representations to bridge pixels and actions. Pixel motion fields and optical-flow-like targets provide dense image-plane motion cues for policy learning [43–45], while point trajectories, 3D point tracks, and object flows introduce stronger geometric or correspondence structure [46–51]. Another line uses language-aligned keypoints, value maps, or mark-based prompts to expose task-relevant spatial structure [52–57]. These approaches demonstrate that motion and spatial abstractions are often more control-relevant than raw pixels, but they typically use motion as a policy input, an auxiliary prediction target, or a separate planning representation. JOPAT instead treats 2D tracks with visibility as part of the generative future state itself: the track variables are sampled jointly with visual latents and actions in the same denoising sequence. This design preserves semantic grounding through visual latents while using tracks as a scalable correspondence-level interface that can be obtained from ordinary videos with modern point trackers [58, 59].

Prior point-track methods commit to correspondences before the action is chosen: Track2Act [46] lifts goal-conditioned 2D tracks into an $\mathrm{SE}(3)$ plan corrected by a residual policy, 3PoinTr [48] freezes 3D tracks predicted from the initial point cloud into a static plan for a separate behavior-cloning policy, and PointWorld [49] predicts 3D scene flow for candidate trajectories and selects one with an external MPPI optimizer under a hand-specified cost. JOPAT instead samples tracks, visibility, visual latents, and actions in one denoising sequence, so the future state and the action constrain each other while both are drawn, with no goal image, depth, frozen plan, or test-time optimizer.

Learning from action-free videos Learning from action-free videos is attractive because videos are substantially easier to scale than action-labeled robot demonstrations. Prior work uses action-free or weakly labeled videos to learn visual representations, dynamics models, inverse dynamics, latent actions, or policy-relevant priors [30, 33, 35, 38, 60–62]. However, generic videos often contain uncontrolled camera motion, unknown agents, and dynamics not caused by the robot, and recent studies suggest that pixel-level pretraining alone does not always translate into improved downstream manipulation success [14, 36, 37]. JOPAT addresses this issue by using action-free videos to supervise a pixel-track future state rather than actions directly: visual latents preserve object and scene semantics, while tracks and visibility supervise realized correspondences and missing evidence. Action-labeled robot demonstrations then ground these future-state variables in executable controls.

3 Preliminaries: world-action models

We consider episodic robot manipulation with RGB observations o_t and robot actions a_t, where actions are end-effector pose deltas and gripper commands. At time t, the policy receives an observation window $\mathbf{o}_t^{\mathrm{ctx}} = o_{t-h_o+1:t}$ and predicts an executable action chunk $a_{t:t+K-1}$. An encoder maps this window to a conditioning feature $\mathbf{c}_t = E_c(\mathbf{o}_t^{\mathrm{ctx}})$. We assume access to two data sources: a large collection of action-free videos $\mathcal{V} = \{o_{1:T}\}$ and a smaller set of action-labeled demonstrations $\mathcal{D} = \{(o_{1:T}, a_{1:T})\}$. The goal is to learn a policy $\pi_\psi(a_{t:t+K-1} \mid \mathbf{c}_t)$ that transfers scalable motion priors from $\mathcal{V}$ while grounding them in executable robot actions from $\mathcal{D}$.

A world-action model (WAM) provides a generative route to this goal by coupling action prediction with future world-state prediction. Rather than learning a policy as a direct map from observations to actions, WAM models a joint distribution $p_\theta(a_{t:t+K-1}, s_{t:t+H} \mid \mathbf{c}_t)$, where $s_{t:t+H}$ denotes a future state abstraction of the scene. The policy can then be viewed as the action marginal of this larger predictive model, $\pi_\theta(a_{t:t+K-1} \mid \mathbf{c}_t) = \int p_\theta(a_{t:t+K-1}, s_{t:t+H} \mid \mathbf{c}_t) ds$. This perspective turns policy learning into a question of interface design: *what future world state should a robot predict so that the prediction is useful for action?* The answer is nontrivial because a future state for control is not the same as a future state for photorealistic reconstruction. For manipulation, the useful state should expose how robot-controllable scene elements move and whether their evidence remains visible, while still preserving enough visual semantics to identify objects and affordances.

The common instantiation is a pixel-latent WAM. Let $\mathbf{z}^o_{t+H:t+H+1} = E_o(o_{t+H:t+H+1})$ denote compact visual latents of a future observation window and $\mathbf{z}^a_{t:t+K-1}$ denote action tokens. A pixel-latent WAM uses $s^{\mathrm{pix}}_{t:t+H} = \mathbf{z}^o_{t+H:t+H+1}$ and learns $p_\theta(\mathbf{z}^a_{t:t+K-1}, \mathbf{z}^o_{t+H:t+H+1} \mid \mathbf{c}_t)$. Architecturally,

this can be implemented by concatenating noisy action and visual tokens and denoising them with a shared transformer,

$$[\hat{\boldsymbol{\epsilon}}^a, \hat{\boldsymbol{\epsilon}}^o] = T_\theta^{\mathrm{WAM}}([\tilde{\mathbf{z}}^a, \tilde{\mathbf{z}}^o, \mathbf{r}]; \mathbf{c}_t, \tau_a, \tau_o), \tag{1}$$

where $\mathbf{r}$ denotes learnable register tokens and τ_a, τ_o are modality-specific diffusion timesteps. Equation (1) defines this baseline, whose sequence holds only action and visual tokens and hence only τ_a, τ_o; JOPAT's three-modality denoiser in Section 4.1 extends it with track tokens and τ_p. This formulation enables action-free video pretraining through the observation branch, but it also inherits a limitation of pixel prediction: the state interface is dominated by appearance. For manipulation, appearance is only an indirect carrier of control-relevant change. A visual latent may encode texture, lighting, background, and camera statistics together with object motion, while the action often depends on sparse correspondence-level facts: which part moved, where it moved, whether it became occluded, and whether it left the visible workspace. Thus, the central weakness of a pixel-only WAM is not simply insufficient model capacity; it is that the predicted future state does not explicitly expose the motion variables through which actions affect the world.

4 Methods

In this section, we first present the overall architecture of the proposed JOPAT, followed by a detailed description of the critical point-track modeling component. We then introduce the training and inference pipeline.

4.1 Unified pixel-track-action architecture

Fig. 2 (b) shows the unified JOPAT architecture. Given the current observation window $o_{t-1:t}$, an observation encoder E_c produces a global conditioning feature $\mathbf{c}_t = E_c(o_{t-1:t})$, which is injected into each DiT block through AdaLN together with modality-specific diffusion timesteps τ_a, τ_o, and τ_p. JOPAT jointly denoises action tokens, future visual-latent tokens, and track tokens in a shared sequence:

$$\mathbf{Z} = [\tilde{\mathbf{z}}^a_{t:t+K-1}, \tilde{\mathbf{z}}^o_{t+H:t+H+1}, \tilde{\mathbf{z}}^p_{t+1:t+H_p}, \mathbf{r}].$$

A DiT-style transformer applies bidirectional full attention over this sequence and predicts modality-specific noise:

$$\hat{\boldsymbol{\epsilon}}^a, \hat{\boldsymbol{\epsilon}}^o, \hat{\boldsymbol{\epsilon}}^p = T_\theta(\mathbf{Z}; \mathbf{c}_t, \tau_a, \tau_o, \tau_p).$$

Separate heads decode action noise, visual-latent noise, track-coordinate noise, and visibility logits. This shared denoising design allows action generation to interact directly with imagined future pixels and point tracks, rather than relying on an isolated action decoder with auxiliary prediction heads. In our implementation, JOPAT conditions on two RGB frames, predicts a two-frame future observation window with offset $H = 16$, and uses $H_p = K = 19$ for both track and action horizons.

4.2 Track construction and modeling

Fig. 2 (a,c) illustrates the track construction and encoding pipeline. JOPAT conditions on two RGB observations $o_{t-1:t}$ and uses the last frame o_t as the reference frame for point tracking. For each sliding window, we initialize N grid query points on o_t and track them over $t+1 : t+H_p$ with an off-the-shelf point tracker [59], producing

$$\mathbf{P}^{(t)} \in \mathbb{R}^{H_p \times N \times 2}, \qquad \mathbf{V}^{(t)} \in \{0,1\}^{H_p \times N}. \tag{2}$$

The superscript (t) denotes that the query points are anchored at the reference frame o_t. We index the H_p entries of a track by a local index $k = 1, \ldots, H_p$, so that entry k corresponds to frame $t+k$; the reference frame o_t is not part of the track target.

Visibility targets $\mathbf{V}^{(t)}$ is the visibility output of the offline CoTrackerV3 predictor [59] with default settings (threshold 0.9, query-frame points forced visible, separate confidence unused), followed by an in-frame mask that sets entries whose coordinates leave $[0,\, W_{\mathrm{img}} - 1] \times [0,\, H_{\mathrm{img}} - 1]$ to 0. Occlusion and tracking failure fail the threshold and out-of-frame motion is caught by the mask, so all three map to $\mathbf{V} = 0$; we do not distinguish them.

To encode tracks compactly, we use a track-as-video representation. In our implementation, query points form a 25×25 grid, giving $N = 625$. We reshape the point dimension back to its spatial grid

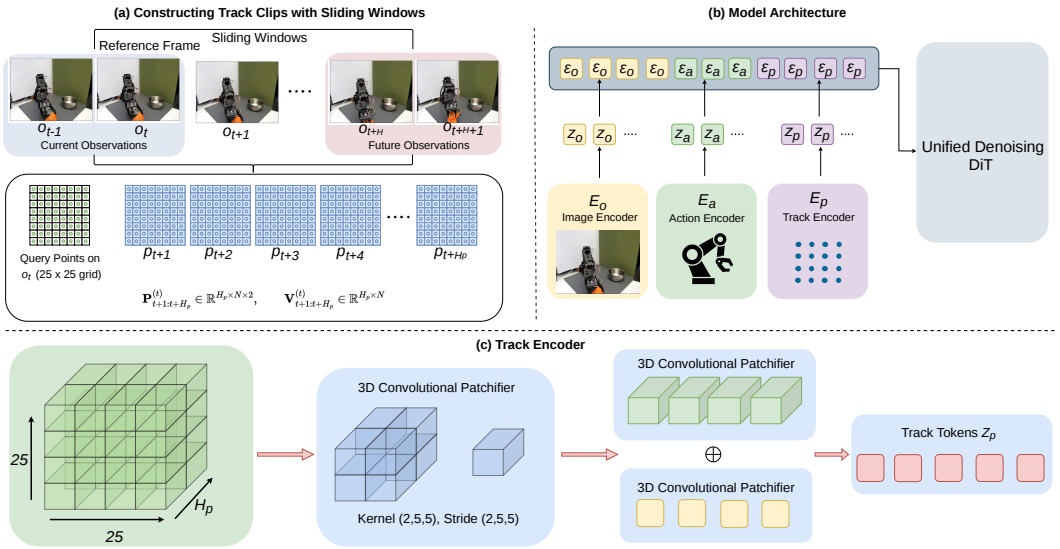

Figure 2: **Overview of JOPAT. (a)** Sliding-window track construction uses the current frame as the reference image for grid query points. **(b)** JOPAT jointly denoises future visual latents, point-track coordinates, and robot actions in a shared Transformer. **(c)** The track-as-video encoder reshapes point tracks into a spatiotemporal grid, applies 3D convolutional patchification, and predicts coordinate noise and visibility logits.

and apply a 3D convolutional patchifier:

$$\begin{aligned} \mathbf{P}^{(t)} \to \mathbf{G}^p \in \mathbb{R}^{2\times H_p\times H_g\times W_g}, \quad H_gW_g = N, \\ \mathbf{z}^p = E_p(\mathbf{P}^{(t)}) = \mathrm{Patchify}_{3D}(\mathbf{G}^p), \quad \mathbf{z}^p \in \mathbb{R}^{L_p\times d}. \end{aligned} \tag{3}$$

After shared denoising, track tokens are decoded into coordinate and visibility predictions:

$$\hat{\mathbf{P}}^{(t)} = D_p(\hat{\mathbf{z}}^p), \qquad \hat{\mathbf{V}}^{(t)} = D_v(\hat{\mathbf{z}}^p), \quad \hat{\mathbf{V}}^{(t)} \in \mathbb{R}^{H_p\times N}. \tag{4}$$

Visibility is predicted only as an output and is not provided to the track encoder, avoiding leakage of ground-truth visibility during denoising.

4.3 Training and inference

JOPAT is trained with a unified denoising objective over action, visual-latent, and track-coordinate tokens. For each modality $m \in \{a, o, p\}$, let $\mathbf{x}_0^m$ be the clean target and $\boldsymbol{\epsilon}^m \sim \mathcal{N}(0, I)$ be Gaussian noise. We independently sample a diffusion timestep τ_m for each modality and construct

$$\mathbf{x}_{\tau_m}^m = \sqrt{\bar{\alpha}_{\tau_m}}\mathbf{x}_0^m + \sqrt{1-\bar{\alpha}_{\tau_m}}\boldsymbol{\epsilon}^m, \qquad \mathcal{L}_m = \|\hat{\boldsymbol{\epsilon}}^m - \boldsymbol{\epsilon}^m\|_2^2. \tag{5}$$

The action and visual-latent losses $\mathcal{L}_a$ and $\mathcal{L}_o$ take this form directly. The track diffusion loss $\mathcal{L}_p$ is applied only to 2D coordinates and replaces the unweighted norm with a per-entry weight set by the visibility target, covering all $H_p \times N$ entries including invisible ones:

$$\mathcal{L}_p = \frac{1}{H_pN}\sum_{k=1}^{H_p}\sum_{i=1}^{N} w_{k,i}\left\|\hat{\boldsymbol{\epsilon}}_{k,i}^p - \boldsymbol{\epsilon}_{k,i}^p\right\|_2^2, \qquad w_{k,i} = \begin{cases} 1.0, & \mathbf{V}_{t+k,i} = 1, \\ 0.1, & \mathbf{V}_{t+k,i} = 0. \end{cases} \tag{6}$$

An occluded point's plausible position is worth learning, but a lost point's is not worth trusting. Since the label 0 conflates the two, we down-weight invisible entries to 0.1 rather than discarding them following CoTrackerV3 [59], which applies a similar down-weighting to invisible points. Visibility is predicted by a separate head and supervised with binary cross entropy (BCE):

$$\mathcal{L}_{\text{vis}} = \frac{1}{H_pN}\sum_{k=1}^{H_p}\sum_{i=1}^{N}\mathrm{BCE}(\hat{\mathbf{V}}_{t+k,i}, \mathbf{V}_{t+k,i}). \tag{7}$$

The final objectives for action-labeled demonstrations $\mathcal{D}$ and action-free videos $\mathcal{V}$ are

$$\begin{aligned}\mathcal{L}_{\mathcal{D}} &= \mathcal{L}_a + \lambda_o \mathcal{L}_o + \lambda_p \mathcal{L}_p + \lambda_{\text{vis}} \mathcal{L}_{\text{vis}}, \\ \mathcal{L}_{\mathcal{V}} &= \lambda_o \mathcal{L}_o + \lambda_p \mathcal{L}_p + \lambda_{\text{vis}} \mathcal{L}_{\text{vis}}.\end{aligned} \tag{8}$$

Thus, the same architecture learns visual-track dynamics from action-free videos and action-grounded control from robot demonstrations.

At inference time, JOPAT is used for receding-horizon action generation. Given the current observation context $\mathbf{c}_t$, action, visual, and track tokens are initialized from Gaussian noise and iteratively denoised in the same shared sequence. Only the action branch is decoded for execution, but the visual and track tokens are not discarded auxiliary outputs: they act as latent future-state variables that interact with action tokens through self-attention during sampling. Consequently, each action chunk is generated under an implicitly sampled pixel-track future, which is the inference-time counterpart of the track-augmented state interface used during training. The action decoder outputs a chunk $\hat{a}_{t:t+K-1}$, from which we execute the first action or a short prefix before replanning.

Implementation details We use a DiT-style Transformer with depth 12, hidden dimension 768, and 12 attention heads. The model uses 8 learnable register tokens. For tracks, we use $N = 625$ query points arranged as a 25×25 grid and a spatiotemporal patch size of $(2, 5, 5)$. We train with a DDIM scheduler using 100 diffusion steps and sample with 10 denoising steps at inference.

5 Experiments

Experiment design Our experiments are organized to examine the state-interface hypothesis introduced in Section 1. *(i)* a track-augmented future state should improve manipulation policy learning over pixel-centric world-action modeling because it exposes controllable correspondences that pixels leave implicit. *(ii)* the gains should come from the complementary roles of visual latents, tracks, and visibility: semantics, motion correspondences, and missing-evidence state. *(iii)* action-free video pretraining should be useful when it supervises the same pixel-track future state that later mediates action generation under limited robot demonstrations.

We therefore evaluate JOPAT along four axes. **(Q1) Main performance:** Does JOPAT improve success rates over prior policy-learning and world-action baselines on standard manipulation benchmarks? **(Q2) Representation design:** Are visual latents, point tracks, and visibility all necessary for this future-state interface under partial observability? **(Q3) Robustness:** Does the pixel-track future state remain reliable under long-horizon distribution shift, occlusion, and distractors? **(Q4) Scalable pretraining:** Does action-free video pretraining help under limited demonstrations, and does the benefit extend beyond robot-domain videos?

We answer Q1 on LIBERO [39], Q2 through modality and visibility ablations, Q3 with a modified LIBERO-Long out-of-distribution (OOD) stress test and real-robot LeRobot SO-101 experiments, and Q4 with action-free pretraining studies using robot-domain DROID videos [63] and generic OpenVid-1M videos [64]. For simulation, we evaluate all four LIBERO suites. Every suite follows a single two-stage protocol: an action-free Stage-I pretraining on out-of-domain DROID videos, followed by per-suite Stage-II finetuning on that suite's standard split. No LIBERO data enters Stage I. For pretraining ablations, we compare no video pretraining, DROID action-free pretraining, and a 10k-video subset from OpenVid-1M; all video-only pretraining masks the action branch and supervises only visual latents, point tracks, and visibility. Table captions state each Stage-I source, abbreviating objectives as V (visual latent), T (track), Vis (visibility), and A (action); Appendix B gives the full protocol.

5.1 Simulation results

We evaluate JOPAT on the standard LIBERO benchmark, which contains four suites: *Spatial*, *Object*, *Goal*, and *Long*, with 10 tasks and 50 demonstrations per task in each suite. Following the benchmark protocol, we report the average success rate (SR) over tasks within each suite.

Stage I pretrains for 100K steps on the RGB streams of 5,000 DROID videos [63] with action labels discarded and action denoising disabled; no LIBERO data is used. **Stage II** finetunes for 10K steps on each suite's standard split of 10 tasks $\times$ 50 demonstrations, with all objectives enabled.

Table 1: **Simulation experimental results.** Comparison of task success rates (SR) and ranks (RK) on the LIBERO benchmark across four task types. "†" indicates our reproduced results. Average SR is the mean of the four suite SRs. JOPAT's average is computed from unrounded per-task success rates, so re-averaging the four displayed suite values gives 97.7; no other row involves this discrepancy. **Stage I:** action-free DROID videos for JOPAT (5,000) and UWM, no LIBERO data; Stage II: each suite's 10 tasks $\times$ 50 demos.

Method	**Spatial**		**Object**		**Goal**		**Long**		**Average**	
	SR ↑	RK ↓	SR ↑	RK ↓	SR ↑	RK ↓	SR ↑	RK ↓	SR ↑	RK ↓
Diffusion Policy [65]	78.3	15	92.5	10	68.3	15	50.5	15	72.4	15
Octo fine-tuned [7]	78.9	14	85.7	15	84.6	12	51.1	14	75.1	14
OpenVLA [4]	84.7	11	88.4	14	79.2	13	53.7	13	76.5	13
π_0 fine-tuned [1]	96.8	5	98.8	2	95.8	4	85.2	7	94.2	7
π_0-Fast [66]	96.4	6	96.8	7	88.6	8	60.2	11	85.5	8
$\pi_{0.5}$-KI [67]	98.0	2	97.8	6	95.6	5	85.8	6	94.3	5
OpenVLA-OFT [68]	97.6	3	98.4	4	97.9	2	94.5	3	97.1	3
SpatialVLA [69]	88.2	9	89.9	13	78.6	14	55.5	12	78.1	12
PD-VLA† [70]	95.5	7	96.7	8	94.9	7	91.7	4	94.7	4
STAR [71]	95.5	7	98.3	5	95.0	6	88.5	5	94.3	5
Dita [72]	84.2	12	96.3	9	85.4	11	63.8	10	82.4	11
CoT-VLA [73]	87.5	10	91.6	12	87.6	9	69.0	9	83.9	10
CogVLA [74]	**98.6**	**1**	98.8	2	96.6	3	95.4	2	97.4	2
UWM [15]	82.3	13	92.2	11	86.8	10	77.6	8	84.7	9
JOPAT	97.2	4	**98.9**	**1**	**98.4**	**1**	**96.4**	**1**	**97.8**	**1**

Table 1 reports the main simulation results. JOPAT achieves the best average performance across the four suites and obtains the largest improvement on *Long*, the most temporally extended suite. This indicates that joint pixel-and-track prediction is particularly beneficial when policy learning requires maintaining action-relevant state over longer horizons. The gain on *Long* is consistent with the motivation of JOPAT. Compared with pixel-centric prediction, point tracks provide a compact motion-centric interface, while visual latents retain semantic and contextual information. We further isolate these components in the ablation study in Section 5.3.

Comparison with track-based methods Among track-based methods, ATM [47], TraceVLA [52], and 4D-VLA [53] report LIBERO results, with published averages from 63.4% to 88.6% against 97.8% for JOPAT; since each was obtained under its own protocol, Appendix D lists them without ranking.

5.2 Real-world experiments

We further evaluate real-world transfer on a physical LeRobot SO-101 platform. The benchmark includes four tasks spanning complementary challenges: *Cook-Soup* for long-horizon sequencing under repeated interactions, *Insert-Peg* for precision alignment and contact-sensitive execution, *Push-Tomato* for contact-rich object displacement under pose variation, and *Pick-Grocery* for object-level OOD generalization. All real-robot training, inference, evaluation, and task setup details are provided in Appendix E, and Fig. 3 there shows the four tasks.

Table 2: **Real-robot performance on LeRobot SO-101.** SR (%) over 10 rollouts per task; Succ. pools the 40 trials (pooled SR equals the task average), with two-sided 95% Wilson CI. **JOPAT Stage I:** 5,000 action-free DROID videos, no SO-101 data; Stage II: 50 demos per task.

Method	**Cook-Soup**	**Insert-Peg**	**Push-Tomato**	**Pick-Grocery**	**Avg.**	**Succ.**	**95% CI**
ACT [75]	40	0	70	50	40.0	16/40	[26.3, 55.4]
UWM [15]	10	0	80	40	32.5	13/40	[20.1, 48.0]
π_0 [1]	50	0	80	40	42.5	17/40	[28.5, 57.8]
$\pi_{0.5}$ [67]	**60**	**10**	**100**	**80**	**62.5**	**25/40**	[47.0, 75.8]
JOPAT (Ours)	**60**	**10**	**100**	60	57.5	23/40	[42.2, 71.5]

Table 2 shows that JOPAT reaches 57.5% average SR, above ACT (40.0%), UWM (32.5%), and π_0 (42.5%), while $\pi_{0.5}$ is highest at 62.5% but is pretrained on substantially larger datasets than our 5,000

action-free DROID videos. With 40 binary trials per method, one comparison is nominally significant and uncorrected for multiplicity, and it is the one our central claim rests on: JOPAT over the pixel-based world-action model UWM (two-sided Fisher's exact test, uncorrected $p = 0.042$); JOPAT over ACT ($p = 0.18$) and π_0 ($p = 0.26$), and $\pi_{0.5}$ over JOPAT ($p = 0.82$), are not significant. Because four pairwise comparisons are reported without multiple-comparison correction, the $p = 0.042$ result should be read as suggestive rather than as a corrected significance claim. JOPAT matches $\pi_{0.5}$ on every task except *Pick-Grocery* and reaches 100% on *Push-Tomato*, suggesting that explicit track prediction provides a useful motion interface for contact-rich object displacement; its gain over UWM on *Cook-Soup* is consistent with the occlusion robustness observed in simulation. *Insert-Peg* remains difficult for all methods, indicating that millimeter-level insertion may require geometric calibration or contact-aware sensing beyond RGB-based track-and-latent modeling.

5.3 Ablation on joint pixel-track modeling

We study whether the above gains come from joint pixel-and-track modeling with four variants that share the same two-frame RGB conditioning and action-denoising branch and differ only in the *predicted future state*: **Action-only**, a standard RGB-conditioned diffusion policy with no future-state head; **Latent-future**, which predicts visual latents; **Track-future**, which predicts tracks and visibility; and **Joint**, which predicts all three. All use the same 5,000-video DROID subset for Stage I, update budget, and LIBERO-Long finetuning; Stage I is action-free except for Action-only, which has no future-state objective and is pretrained on the DROID action labels instead.

Simulation ablation The results in Table 3 show that joint modeling substantially outperforms every other variant, improving over Action-only by 37.6, Latent-future by 19.0, and Track-future by 70.2 points in average SR. Predicting visual latents raises SR from 58.8% to 77.4%, and adding tracks and visibility raises it to 96.4%. This supports our hypothesis: track tokens provide correspondence-level motion constraints for action-token denoising, while visual latents preserve the semantic context needed to decide which object, affordance, or task state matters. Latent-future removes these motion constraints, leading to failures when correct object recognition is not enough to maintain precise long-horizon motion; Track-future removes the predicted visual future, leading to failures when similar motions must be tied to the correct object, affordance, or task state. Track-future falls below Action-only, so predicted tracks help only when coupled with predicted visual latents in the same denoising state.

Table 3: **Future-state ablation on LIBERO-Long.** Success rates (SR, %) across ten long-horizon tasks. All variants condition on the current RGB frames and keep the action-denoising branch; they differ only in the predicted future state. **Stage I:** the same 5,000 DROID videos; Stage II: 50 demos per task.

Task	Action-only	Latent-future	Track-future	Joint	Task	Action-only	Latent-future	Track-future	Joint
Mug-Mug	47	58	2	**92**	Tomato-Sauce	65	93	48	**100**
Soup-Cheese	49	78	14	**94**	Cream-Butter	51	91	66	**100**
Moka-Moka	82	67	26	**89**	Microwave	73	94	32	**100**
Book-Caddy	68	63	14	**100**	Chocolate-Pudding	61	73	46	**91**
Bowl-Drawer	38	95	2	**100**	Moka-Stove	54	62	12	**98**
Average SR: Action-only 58.8		Latent-future 77.4		Track-future 26.2		Joint **96.4**			

Real-robot ablation The similar effect appears in real robot experiments in Table 4. Joint modeling improves average SR from 35.0% for Latent-future and 15.0% for Track-future to 57.5%, corresponding to gains of 22.5 and 42.5 points. Latent-future retains predicted object appearance but leaves contact-relevant motion implicit, while Track-future exposes motion structure but lacks the predicted visual future needed to identify the correct object and goal region. JOPAT improves every task under the same data budget, although Insert-Peg remains difficult for all variants, suggesting that millimeter-level insertion may require additional geometry or contact information beyond RGB latents and 2D tracks.

Table 4: **Real-robot future-state ablation.** SR on four LeRobot SO-101 tasks. **Stage I:** as in Table 2.

Method	Cook-Soup	Insert-Peg	Push-Tomato	Pick-Grocery	Avg.
Latent-future	40	0	70	30	35.0
Track-future	10	0	30	20	15.0
Joint (Ours)	**60**	**10**	**100**	**60**	**57.5**

5.4 Ablation on track visibility prediction

We further ablate visibility prediction in Table 5. Removing the visibility head reduces real-robot average SR from 57.5% to 47.5%, with the largest improvement from visibility on Cook-Soup. This supports the role of visibility supervision as a missing-evidence constraint on the track stream: the model learns when a correspondence is unobservable instead of forcing every point to explain a visible coordinate. The unchanged Pick-Grocery result indicates that visibility is not merely extra capacity; it mainly helps tasks with self-occlusion, transient out-of-view motion, or long-horizon object motion where confusing hidden points with stationary points can cause drift.

Table 5: **Track visibility ablation.** We report the SR on LeRobot SO-101 tasks. The visibility head improves robustness under self-occlusion and transient out-of-view motion. **Stage I:** as in Table 2.

Method	Cook-Soup	Insert-Peg	Push-Tomato	Pick-Grocery	Avg.
w/o visibility	40	0	90	60	47.5
w/ visibility (Ours)	**60**	**10**	**100**	**60**	**57.5**

5.5 Does action-free video pretraining help?

Table 6 isolates the effect of action-free video pretraining on LIBERO-Long by fixing the architecture and varying the Stage-I source, complementing Table 3, which fixes the Stage-I source and varies the predicted future state. Both pretraining sources consistently outperform training from scratch across all demonstration budgets, showing that JOPAT's pixel-track future-state objective turns action-free videos into useful motion priors for downstream control. The gain is especially large in the low-data regime: with only 10 demonstrations, DROID pretraining improves SR from 11.9% to 64.2%, and even generic OpenVid-1M pretraining reaches 48.5%. This indicates that the benefit is not merely due to robot-domain proximity; the model can also extract transferable object-motion and occlusion priors from non-robot videos. At 50 demonstrations, the gap between DROID and OpenVid-1M nearly vanishes (96.4% vs. 95.1%), suggesting that domain-specific pretraining is most valuable when supervision is scarce, while generic video priors can be aligned effectively once sufficient action-labeled demonstrations are available.

Table 6: **Action-free video pretraining on LIBERO-Long.** SR are reported. **Stage I:** none, 5,000 DROID, or 10,000 OpenVid-1M videos (action-free; V, T, Vis).

Method	10 demos	25 demos	50 demos
JOPAT (no pretraining)	11.9%	31.6%	66.1%
JOPAT (DROID pretraining)	**64.2**%	82.7%	**96.4**%
JOPAT (OpenVid-1M pretraining)	48.5%	**84.6**%	95.1%

5.6 Robustness on OOD tasks

To evaluate robustness under stronger partial observability and distribution shift, we construct a modified LIBERO-Long setting with three changes: *(1)* the object initialization range is expanded by 5 cm; *(2)* one background object is replaced with an unseen object in each task; and *(3)* an unseen distractor is placed in front of the target object. All modifications are applied consistently across all compared methods. This setting increases failures caused by target-object occlusion, distractor interference, larger object displacement, and delayed errors in long-horizon execution.

Table 7 shows that JOPAT achieves the best average SR, improving over Diffusion Policy, UWM, and fine-tuned $\pi_{0.5}$ by 34, 32, and 13 points, respectively. Compared with UWM, JOPAT improves on all

Table 7: **OOD evaluation on modified LIBERO-Long.** Success rates (0–1 scale) on the five tasks of UWM's OOD evaluation [15]. **Stage I:** as in Table 1; modified environments at evaluation only.

Task	Diffusion Policy [65]	$\pi_{0.5}$ fine-tuned [67]	UWM [15]	JOPAT (Ours)
Book-Caddy	0.24	**0.95**	0.38	0.88
Soup-Cheese	0.21	**0.82**	0.44	0.78
Bowl-Drawer	0.56	0.12	0.52	**0.68**
Moka-Moka	0.44	**0.70**	0.22	0.48
Mug-Mug	0.18	0.08	0.14	**0.50**
Avg.	0.33	0.53	0.34	**0.66**

five tasks, with especially large gains on *Mug-Mug* (+36 points), *Moka-Moka* (+26 points), and *Book-Caddy* (+50 points). These tasks require maintaining object state across extended interactions while avoiding distractor-induced drift, suggesting that explicit correspondence and visibility prediction provides a stable future-state interface under long-horizon distribution shift.

Interestingly, fine-tuned $\pi_{0.5}$ achieves the best performance on *Book-Caddy*, *Soup-Cheese*, and *Moka-Moka*, but performs poorly on *Bowl-Drawer* and *Mug-Mug*. This leads to a lower average SR than JOPAT despite strong peak performance. The result suggests that large pretrained VLA models can be highly effective on some OOD tasks, but may be less stable across task families when distractors and occlusions alter the visual grounding conditions. In contrast, JOPAT is not always the best on every individual task, but achieves the strongest average performance and the best results on tasks where distractor-aware object tracking is critical. The largest gap to $\pi_{0.5}$ is on *Moka-Moka* (0.48 vs. 0.70), whose scene contains two identical moka pots: because grid query points cover both, the predicted future state does not consistently commit to one pot, and the policy occasionally reaches for the wrong one—a genuine limitation of grid-based tracks without object-level grouping.

6 Conclusion and discussion

In this work, we introduced JOPAT, a unified world-action model that treats 2D point tracks and visual latents as joint predictive modalities for action generation. By making motion correspondences explicit alongside pixel-level visual latents, our approach anchors predictions on geometric structures and improves robustness against occlusion. Experiments on LIBERO and real-world LeRobot tasks confirm that this formulation outperforms pixel-based baselines in data efficiency, effectively transferring motion priors from action-free video to downstream control. We release the data-processing pipeline, training code, and per-result training configurations at `https://github.com/1mather/JOPAT_WAM`.

Limitations Despite these results, limitations persist. First, grid-based point tracks are sparse and may miss fine-grained deformations needed for sub-centimeter dexterity, and without object-level grouping they do not commit to one of several identical instances (*Moka-Moka*, Section 5.6). Second, relying on an off-the-shelf tracker (e.g., CoTracker) caps what the model can learn, and our visibility targets do not separate occlusion, out-of-frame motion, and tracking failure. Third, we train the DiT from scratch with the same backbone as UWM so that the comparison isolates the future-state interface; building on a pretrained video generator such as Cosmos [31], Wan [76], or CogVideoX [77] would add confounds such as distribution shift and degradation of the backbone, and we leave it to future work, although the pretraining gains in Table 6 suggest that stronger visual priors could help further. Fourth, a JOPAT query takes 45.2 ms on an RTX 4090 (about 22 Hz; Appendix C), which limits high-frequency control. Finally, the architecture assumes a static camera: in world-action models, predicting wrist-camera video is consistently harder than predicting exterior-camera video, and we expect the same difficulty for track prediction under ego-motion in mobile manipulation.

Acknowledgments

AS and JK acknowledge funding from the Research Council of Finland (352788, 362407, 373999, 373780, 362408, and 339730). This work was supported by the Research Council of Finland Flagship programme: Finnish Center for Artificial Intelligence FCAI. This work was supported in part by

The Major Key Project of PCL (PCL2025A02, PCL2025A17). We acknowledge the computational resources provided by the Aalto Science-IT project.

References

[1] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

[2] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil J Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2023.

[3] Chilam Cheang, Sijin Chen, Zhongren Cui, Yingdong Hu, Liqun Huang, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Xiao Ma, et al. Gr-3 technical report. *arXiv preprint arXiv:2507.15493*, 2025.

[4] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

[5] Rui Shao, Wei Li, Lingsen Zhang, Renshan Zhang, Zhiyang Liu, Ran Chen, and Liqiang Nie. Large vlm-based vision-language-action models for robotic manipulation: A survey. *arXiv preprint arXiv:2508.13073*, 2025.

[6] Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, et al. Smolvla: A vision-language-action model for affordable and efficient robotics. *arXiv preprint arXiv:2506.01844*, 2025.

[7] Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Pannag Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

[8] Dapeng Zhang, Jing Sun, Chenghui Hu, Xiaoyan Wu, Zhenlong Yuan, Rui Zhou, Fei Shen, and Qingguo Zhou. Pure vision language action (vla) models: A comprehensive survey. *arXiv preprint arXiv:2509.19012*, 2025.

[9] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alexander Herzog, Jasmine Hsu, Brian Ichter, Alex Irpan, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Lisa Lee, Tsang-Wei Edward Lee, Sergey Levine, Yao Lu, Henryk Michalewski, Igor Mordatch, Karl Pertsch, Kanishka Rao, Krista Reymann, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Pierre Sermanet, Jaspiar Singh, Anikait Singh, Radu Soricut, Huong Tran, Vincent Vanhoucke, Quan Vuong, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Jialin Wu, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

[10] Zhijie Wang, Zhehua Zhou, Jiayang Song, Yuheng Huang, Zhan Shu, and Lei Ma. Vlatest: Testing and evaluating vision-language-action models for robotic manipulation. *Proceedings of the ACM on Software Engineering*, 2(FSE):1615–1638, 2025.

[11] Borong Zhang, Jiahao Li, Jiachen Shen, Yishuai Cai, Yuhao Zhang, Yuanpei Chen, Juntao Dai, Jiaming Ji, and Yaodong Yang. VLA-Arena: An open-source framework for benchmarking vision-language-action models. *arXiv preprint arXiv:2512.22539*, 2025.

[12] Aiden Swann, Lachlain McGranahan, Hugo Buurmeijer, Monroe Kennedy III, and Mac Schwager. Sparse autoencoders reveal interpretable and steerable features in vla models. *arXiv preprint arXiv:2603.19183*, 2026.

[13] Shresth Grover, Akshay Gopalkrishnan, Bo Ai, Henrik I Christensen, Hao Su, and Xuanlin Li. Enhancing generalization in vision-language-action models by preserving pretrained representations. *arXiv preprint arXiv:2509.11417*, 2025.

[14] Xinghang Li, Peiyan Li, Long Qian, Minghuan Liu, Dong Wang, Jirong Liu, Bingyi Kang, Xiao Ma, Xinlong Wang, Di Guo, et al. What matters in building vision–language–action models for generalist robots. *Nature Machine Intelligence*, pages 1–15, 2026.

[15] 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. *arXiv preprint arXiv:2504.02792*, 2025.

[16] Jun Guo, Qiwei Li, Peiyan Li, Zilong Chen, Nan Sun, Yifei Su, Heyun Wang, Yuan Zhang, Xinghang Li, and Huaping Liu. Unified 4d world action modeling from video priors with asynchronous denoising. *arXiv preprint arXiv:2604.26694*, 2026.

[17] Jiayi Chen, Wenxuan Song, Pengxiang Ding, Ziyang Zhou, Han Zhao, Feilong Tang, Donglin Wang, and Haoang Li. Unified diffusion vla: Vision-language-action model via joint discrete denoising diffusion process. *arXiv preprint arXiv:2511.01718*, 2025.

[18] Jiaming Liu, Hao Chen, Pengju An, Zhuoyang Liu, Renrui Zhang, Chenyang Gu, Xiaoqi Li, Ziyu Guo, Sixiang Chen, Mengzhen Liu, et al. Hybridvla: Collaborative diffusion and autoregression in a unified vision-language-action model. *arXiv preprint arXiv:2503.10631*, 2025.

[19] Xiufeng Song, Yiran Qin, Yan Tai, Li Kang, Heng Zhou, Siqi Luo, Jiwen Yu, Ling Yang, Philip Torr, LEI BAI, Ruimao Zhang, Xiaohong Liu, and Zhenfei Yin. DIVA: Discrete diffusion vision-language-action models for parallelized action generation, 2026. URL https://openreview.net/forum?id=mNya9d1DA2.

[20] John Won, Kyungmin Lee, Huiwon Jang, Dongyoung Kim, and Jinwoo Shin. Dual-stream diffusion for world-model augmented vision-language-action model. *arXiv preprint arXiv:2510.27607*, 2025.

[21] Yuqi Wang, Xinghang Li, Wenxuan Wang, Junbo Zhang, Yingyan Li, Yuntao Chen, Xinlong Wang, and Zhaoxiang Zhang. Unified vision-language-action model. *arXiv preprint arXiv:2506.19850*, 2025.

[22] Sandeep Routray, Hengkai Pan, Unnat Jain, Shikhar Bahl, and Deepak Pathak. ViPRA: Video prediction for robot actions. *arXiv preprint arXiv:2511.07732*, 2025.

[23] Fangqi Zhu, Zhengyang Yan, Zicong Hong, Quanxin Shou, Xiao Ma, and Song Guo. Wmpo: World model-based policy optimization for vision-language-action models. *arXiv preprint arXiv:2511.09515*, 2025.

[24] Shashank Hegde, Satyajeet Das, Gautam Salhotra, and Gaurav S Sukhatme. WARPD: World model assisted reactive policy diffusion. *arXiv preprint arXiv:2410.14040*, 2024.

[25] Wenyao Zhang, Hongsi Liu, Zekun Qi, Yunnan Wang, Xinqiang Yu, Jiazhao Zhang, Runpei Dong, Jiawei He, Fan Lu, He Wang, et al. Dreamvla: a vision-language-action model dreamed with comprehensive world knowledge. *arXiv preprint arXiv:2507.04447*, 2025.

[26] 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 preprint arXiv:2602.15922*, 2026.

[27] Liaoyuan Fan, Zetian Xu, Chen Cao, Wenyao Zhang, Mingqi Yuan, and Jiayu Chen. AIM: Intent-aware unified world action modeling with spatial value maps. *arXiv preprint arXiv:2604.11135*, 2026.

[28] Peng-Fei Zhang, Ying Cheng, Xiaofan Sun, Shijie Wang, Fengling Li, Lei Zhu, and Heng Tao Shen. A step toward world models: A survey on robotic manipulation. *arXiv preprint arXiv:2511.02097*, 2025.

[29] Xinqing Li, Xin He, Le Zhang, Min Wu, Xiaoli Li, and Yun Liu. A comprehensive survey on world models for embodied ai. *arXiv preprint arXiv:2510.16732*, 2025.

[30] 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 *International Conference on Learning Representations*, volume 2024, pages 10641–10662, 2024.

[31] Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, et al. Cosmos world foundation model platform for physical ai. *arXiv preprint arXiv:2501.03575*, 2025.

[32] Chi-Lam Cheang, Guangzeng Chen, Ya Jing, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Hongtao Wu, Jiafeng Xu, Yichu Yang, et al. Gr-2: A generative video-language-action model with web-scale knowledge for robot manipulation. *arXiv preprint arXiv:2410.06158*, 2024.

[33] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3M: A universal visual representation for robot manipulation. In *Conference on Robot Learning*, pages 892–909. PMLR, 2023.

[34] Ilija Radosavovic, Baifeng Shi, Letian Fu, Ken Goldberg, Trevor Darrell, and Jitendra Malik. Robot learning with sensorimotor pre-training. In *Conference on Robot Learning*, pages 683–693. PMLR, 2023.

[35] Simone Parisi, Aravind Rajeswaran, Senthil Purushwalkam, and Abhinav Gupta. The unsurprising effectiveness of pre-trained vision models for control. In *Proceedings of the 39th International Conference on Machine Learning*, pages 17359–17371. PMLR, 2022.

[36] Chenhao Shi, Yichen Zhu, Junjie Wen, Yefei Chen, Ziang Liu, and Faming Fang. Teleoperation, simulation, or human video? data utilization law for robot manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Findings*, pages 1388–1397, June 2026.

[37] Rhoda AI Team. Causal video models are data-efficient robot policy learners. *Rhoda AI Blog*, 2026.

[38] Haoran He, Chenjia Bai, Ling Pan, Weinan Zhang, Bin Zhao, and Xuelong Li. Learning an actionable discrete diffusion policy via large-scale actionless video pre-training. *Advances in Neural Information Processing Systems*, 37:31124–31153, 2024.

[39] Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023.

[40] Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction. *Advances in neural information processing systems*, 29, 2016.

[41] Mohammad Babaeizadeh, Chelsea Finn, Dumitru Erhan, Roy H Campbell, and Sergey Levine. Stochastic variational video prediction. *arXiv preprint arXiv:1710.11252*, 2017.

[42] Jake Bruce, Michael D Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, et al. Genie: Generative interactive environments. In *Forty-first International Conference on Machine Learning*, 2024.

[43] E-Ro Nguyen, Yichi Zhang, Kanchana Ranasinghe, Xiang Li, and Michael S Ryoo. Pixel motion diffusion is what we need for robot control. *arXiv preprint arXiv:2509.22652*, 2025.

[44] Kanchana Ranasinghe, Xiang Li, E-Ro Nguyen, Cristina Mata, Jongwoo Park, and Michael S Ryoo. Pixel motion as universal representation for robot control. *arXiv preprint arXiv:2505.07817*, 2025.

[45] Yitian Zheng, Zhangchen Ye, Weijun Dong, Shengjie Wang, Yuyang Liu, Chongjie Zhang, Chuan Wen, and Yang Gao. Translating flow to policy via hindsight online imitation. *arXiv preprint arXiv:2512.19269*, 2025.

[46] Homanga Bharadhwaj, Roozbeh Mottaghi, Abhinav Gupta, and Shubham Tulsiani. Track2act: Predicting point tracks from internet videos enables generalizable robot manipulation. In *European Conference on Computer Vision*, pages 306–324. Springer, 2024.

[47] Chuan Wen, Xingyu Lin, John So, Kai Chen, Qi Dou, Yang Gao, and Pieter Abbeel. Any-point trajectory modeling for policy learning. *arXiv preprint arXiv:2401.00025*, 2024.

[48] Adam Hung, Bardienus Pieter Duisterhof, and Jeffrey Ichnowski. 3PoinTr: 3d point tracks for robot manipulation pretraining from casual videos. *arXiv preprint arXiv:2603.08485*, 2026.

[49] Wenlong Huang, Yu-Wei Chao, Arsalan Mousavian, Ming-Yu Liu, Dieter Fox, Kaichun Mo, and Li Fei-Fei. PointWorld: Scaling 3d world models for in-the-wild robotic manipulation. *arXiv preprint arXiv:2601.03782*, 2026.

[50] Karthik Dharmarajan, Wenlong Huang, Jiajun Wu, Li Fei-Fei, and Ruohan Zhang. Dream2Flow: Bridging video generation and open-world manipulation with 3d object flow. *arXiv preprint arXiv:2512.24766*, 2025.

[51] Homanga Bharadhwaj, Debidatta Dwibedi, Abhinav Gupta, Shubham Tulsiani, Carl Doersch, Ted Xiao, Dhruv Shah, Fei Xia, Dorsa Sadigh, and Sean Kirmani. Gen2act: Human video generation in novel scenarios enables generalizable robot manipulation. *arXiv preprint arXiv:2409.16283*, 2024.

[52] Ruijie Zheng, Yongyuan Liang, Shuaiyi Huang, Jianfeng Gao, Hal Daumé III, Andrey Kolobov, Furong Huang, and Jianwei Yang. TraceVLA: Visual trace prompting enhances spatial-temporal awareness for generalist robotic policies. *arXiv preprint arXiv:2412.10345*, 2025.

[53] Jiahui Zhang, Yurui Chen, Yueming Xu, Ze Huang, Yanpeng Zhou, Yu-Jie Yuan, Xinyue Cai, Guowei Huang, Xingyue Quan, Hang Xu, and Li Zhang. 4D-VLA: Spatiotemporal vision-language-action pretraining with cross-scene calibration. *arXiv preprint arXiv:2506.22242*, 2025.

[54] Wenlong Huang, Chen Wang, Yunzhu Li, Ruohan Zhang, and Li Fei-Fei. Rekep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation. *arXiv preprint arXiv:2409.01652*, 2024.

[55] Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. VoxPoser: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023.

[56] Fangchen Liu, Kuan Fang, Pieter Abbeel, and Sergey Levine. Moka: Open-world robotic manipulation through mark-based visual prompting. *arXiv preprint arXiv:2403.03174*, 2024.

[57] Jianshu Hu, Lidi Wang, Shujia Li, Yunpeng Jiang, Xiao Li, Paul Weng, and Yutong Ban. Generalizable coarse-to-fine robot manipulation via language-aligned 3d keypoints. *arXiv preprint arXiv:2509.23575*, 2025.

[58] Carl Doersch, Ankush Gupta, Larisa Markeeva, Adrià Recasens, Lucas Smaira, Yusuf Aytar, João Carreira, Andrew Zisserman, and Yi Yang. TAP-Vid: A benchmark for tracking any point in a video. *arXiv preprint arXiv:2211.03726*, 2023.

[59] Nikita Karaev, Iurii Makarov, Jianyuan Wang, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. CoTracker3: Simpler and better point tracking by pseudo-labelling real videos. *arXiv preprint arXiv:2410.11831*, 2024.

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

[61] Younggyo Seo, Kimin Lee, Stephen L James, and Pieter Abbeel. Reinforcement learning with action-free pre-training from videos. In *International Conference on Machine Learning*, pages 19561–19579. PMLR, 2022.

[62] Xiaoyu Chen, Hangxing Wei, Pushi Zhang, Chuheng Zhang, Kaixin Wang, Yanjiang Guo, Rushuai Yang, Yucen Wang, Xinquan Xiao, Li Zhao, et al. Villa-x: enhancing latent action modeling in vision-language-action models. *arXiv preprint arXiv:2507.23682*, 2025.

[63] Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, et al. DROID: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

[64] Kepan Nan, Rui Xie, Penghao Zhou, Tiehan Fan, Zhenheng Yang, Zhijie Chen, Xiang Li, Jian Yang, and Ying Tai. OpenVid-1M: A large-scale high-quality dataset for text-to-video generation. In *International Conference on Learning Representations*, 2025.

[65] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2024.

[66] Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. FAST: Efficient action tokenization for vision-language-action models. *arXiv preprint arXiv:2501.09747*, 2025.

[67] Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

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

[69] Delin Qu, Haoming Song, Qizhi Chen, Yuanqi Yao, Xinyi Ye, Yan Ding, Zhigang Wang, JiaYuan Gu, Bin Zhao, Dong Wang, and Xuelong Li. SpatialVLA: Exploring spatial representations for visual-language-action model. *arXiv preprint arXiv:2501.15830*, 2025.

[70] Wenxuan Song, Jiayi Chen, Pengxiang Ding, Han Zhao, Wei Zhao, Zhide Zhong, Zongyuan Ge, Jun Ma, and Haoang Li. Accelerating vision-language-action model integrated with action chunking via parallel decoding. *arXiv preprint arXiv:2503.02310*, 2025.

[71] Hao Li, Qi Lv, Rui Shao, Xiang Deng, Yinchuan Li, Jianye Hao, and Liqiang Nie. STAR: Learning diverse robot skill abstractions through rotation-augmented vector quantization. *arXiv preprint arXiv:2506.03863*, 2025.

[72] Zhi Hou, Tianyi Zhang, Yuwen Xiong, Haonan Duan, Hengjun Pu, Ronglei Tong, Chengyang Zhao, Xizhou Zhu, Yu Qiao, Jifeng Dai, and Yuntao Chen. Dita: Scaling diffusion transformer for generalist vision-language-action policy. *arXiv preprint arXiv:2503.19757*, 2025.

[73] Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, Ankur Handa, Ming-Yu Liu, Donglai Xiang, Gordon Wetzstein, and Tsung-Yi Lin. CoT-VLA: Visual chain-of-thought reasoning for vision-language-action models. *arXiv preprint arXiv:2503.22020*, 2025.

[74] Wei Li, Renshan Zhang, Rui Shao, Jie He, and Liqiang Nie. CogVLA: Cognition-aligned vision-language-action model via instruction-driven routing & sparsification. *arXiv preprint, arxiv:2508.21046*, 2025.

[75] Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

[76] Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Mengyang Feng, Ningyi Zhang, Pandeng Li, Pingyu Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang, Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng Zhou, Wente Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong, Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

[77] Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, Da Yin, Yuxuan Zhang, Weihan Wang, Yean Cheng, Bin Xu, Xiaotao Gu, Yuxiao Dong, and Jie Tang. CogVideoX: Text-to-video diffusion models with an expert transformer. *arXiv preprint arXiv:2408.06072*, 2025.

[78] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

[79] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2022.

[80] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2019.

Appendices

This appendix provides implementation details and supplementary analyses that complement the main paper. We include the training protocol behind every reported result, inference latency, the real-robot experimental protocol, the real-robot action-free pretraining result, prediction-horizon sensitivity, and task-level failure modes.

A Implementation details

Architecture JOPAT uses a DiT-style transformer backbone with modality-specific encoders and decoders for visual latents, point tracks, and actions. The conditioning encoder processes the two most recent RGB observations and injects the resulting global feature into each Transformer block through AdaLN. Future observations are encoded with a frozen SDXL VAE, while point tracks are represented on a regular query grid and patchified with a 3D convolutional encoder. The Transformer output is split into modality-specific heads for action noise, visual-latent noise, track-coordinate noise, and visibility logits, matching the unified pixel-track-action architecture in Section 4.1 and the track construction in Section 4.2.

Training data For action-free videos, the action branch is masked and the model is trained only with visual-latent, track-coordinate, and visibility losses. For action-labeled robot demonstrations, all branches are supervised. Point-track and visibility targets are extracted with the offline CoTrackerV3 predictor [59] using the sliding-window construction described in Section 4.2; visibility targets use the predictor's default threshold of 0.9 followed by an in-frame mask, and invisible entries receive a coordinate-loss weight of 0.1 (Eq. (6)). Simulation and real-robot experiments use the same joint denoising formulation and, unless stated otherwise, the same Stage-I pretraining on 5,000 action-free DROID videos; no LIBERO or SO-101 data is used in Stage I. Appendix B lists the protocol behind every reported result.

Inference At test time, JOPAT performs receding-horizon action generation with DDIM sampling. Action, visual, and track tokens are initialized from Gaussian noise and denoised conditioned on the current observation history. The model can also sample future visual latents and tracks for analysis, but policy execution uses only the decoded action chunk; the first 8 actions are executed before replanning.

Compute resources. Action-free pretraining was performed on 4 NVIDIA H200 GPUs for approximately 5 days. Task-specific finetuning used 1 NVIDIA H200 GPU for approximately 1 day. During deployment, inference runs on a single NVIDIA RTX 4090 GPU, where the robot control loop replans at approximately 10 Hz (one control step every 100 ms); isolated model-query throughput is higher, at approximately 22 Hz, or 45.2 ms per query (Appendix C).

B Training protocol for each result

Every result in the paper uses a two-stage protocol. Stage I pretrains on video; it is action-free in every row except Action-only, which has no future-state objective that could be trained on action-free video and therefore uses the action labels of the same DROID episodes. Stage II finetunes on action-labeled target-domain demonstrations with all objectives available to the respective variant. Table 9 lists both stages for each table. No LIBERO data enters Stage I anywhere in the paper; in particular, the *Long* entry of Table 1 and the Joint row of Table 3 are the same model. We will release the training configuration for every row so that the source of each result can be checked directly.

C Inference latency

Table 10 reports wall-clock latency per model query on one NVIDIA RTX 4090 with batch size 1 and FP16, measured after 100 warm-up iterations as the mean $\pm$ standard deviation over 1,000 runs. The maximum model-query rate is the reciprocal of the mean latency. Adding future visual latents to Action-only costs about 6.4 ms per query, and adding tracks and visibility on top costs about 6.1 ms more; the full JOPAT model is $1.8\times$ faster than π_0 and $2.1\times$ faster than $\pi_{0.5}$. In deployment, JOPAT executes the first 8 actions of each chunk before replanning (Appendix E).

Table 8: **JOPAT hyperparameters.** Unless otherwise stated, the same configuration is used for simulation and real-robot experiments.

Parameter	Value
Data and representation	
Observation history	2 RGB frames
Image resolution	224×224
Future-observation offset	$H = 16$ steps
Action / track horizon	$K = H_p = 19$ steps
Executed action prefix	8 steps
SDXL latent shape	$28 \times 28 \times 4$
Query-point grid	25×25 ($N = 625$)
Track patch size	$(2, 5, 5)$ over time and grid dimensions
Visibility target	CoTrackerV3 default (threshold 0.9) + in-frame mask
Invisible-entry coordinate weight	0.1 (visible: 1.0)
Stage-I data	5,000 action-free DROID videos, 100K steps
Stage-II data (LIBERO)	10 tasks $\times$ 50 demos per suite, 10K steps
Model	
Conditioning encoder	ResNet-18, ImageNet initialization
Visual target encoder	Frozen SDXL VAE [78]
Transformer depth	12 layers
Hidden dimension	768
Attention heads	12
MLP ratio	4
Register tokens	8
Conditioning mechanism	AdaLN
Diffusion and optimization	
Noise schedule	squaredcos_cap_v2
Training diffusion steps	100
Inference denoising steps	10
Sampler	DDIM [79]
Optimizer	AdamW [80]
Learning rate	1×10^{-4}
Weight decay	1×10^{-6}
Adam betas	$(0.9, 0.999)$
Batch size	144 for pretraining; 72 for finetuning
LR schedule	Constant for pretraining; cosine for finetuning
Warmup steps	1000
Loss weights	$\lambda_a = \lambda_o = \lambda_p = \lambda_{\text{vis}} = 1$

D Comparison with track-based methods on LIBERO

Table 11 lists the published LIBERO results of the track-based methods that report them: ATM [47], TraceVLA [52], and 4D-VLA [53]. Each was obtained under its own protocol, so the rows are not directly rankable against one another or against Table 1. Track2Act [46], 3PoinTr [48], and PointWorld [49] do not report LIBERO results; Section 2 contrasts their design with JOPAT.

E Real-robot experimental protocol

Platform and tasks We evaluate on the LeRobot SO-101 platform with four manipulation tasks: *Cook-Soup*, which requires long-horizon sequencing and repeated object interactions; *Insert-Peg*, which requires precise geometric alignment and contact-sensitive execution; *Push-Tomato*, which tests contact-rich object displacement under pose and interaction variation; and *Pick-Grocery*, which tests object-level OOD generalization to unseen objects. Together, these tasks evaluate robustness to partial observability, arm-object occlusion, contact-induced dynamics, and distribution shift.

Training data and finetuning For real-robot transfer, JOPAT is first pretrained on 5,000 DROID action-free videos [63], which contain no SO-101 data. During action-free pretraining, the action branch is masked and only the visual-latent, track-coordinate, and visibility objectives are applied. Point tracks and visibility labels are extracted with CoTrackerV3 [59] using the same sliding-window

Table 9: **Per-result training protocol.** V = visual latent, T = track coordinates, Vis = visibility, A = action.

Result	Stage-I data	Stage-I objectives	Stage-II data
Table 1	5,000 DROID videos	V, T, Vis	LIBERO Spatial / Object / Goal / Long, 10 tasks $\times$ 50 demos
Tables 2, 4 and 5	5,000 DROID videos, no SO-101	V, T, Vis	SO-101, 50 demos per task
Table 3, Latent-future / Track-future / Joint	same 5,000 DROID videos	V / T+Vis / V+T+Vis	LIBERO-Long, 50 demos per task
Table 3, Action-only	same 5,000 DROID videos	A	LIBERO-Long, 50 demos per task
Table 6	none / 5,000 DROID / 10,000 OpenVid-1M	V, T, Vis	LIBERO-Long, 10 / 25 / 50 demos per task
Table 7 (OOD)	5,000 DROID videos	V, T, Vis	standard LIBERO-Long demos; modified environments at evaluation only

Table 10: **Inference latency on one RTX 4090** (batch size 1, FP16; mean $\pm$ s.d. over 1,000 runs). A = action, V = future visual latent, T = future track, Vis = visibility.

Method	Predicted outputs	Mean latency (ms)	Max. query rate (Hz)
ACT [75]	A	13.73 ± 1.49	72.8
UWM [15]	A + V	39.08 ± 3.19	25.6
π_0 [1]	A	83.56 ± 5.91	12.0
$\pi_{0.5}$ [67]	A	96.45 ± 6.34	10.4
Action-only	A	32.74 ± 2.35	30.5
Latent-future	A + V	39.14 ± 3.47	25.5
Joint JOPAT	A + V + T + Vis	45.23 ± 3.83	22.1

construction as in the simulation experiments. The pretrained model is then finetuned separately on each LeRobot task using 50 action-labeled demonstrations. During finetuning, all branches are supervised so that the pretrained pixel-track motion priors are aligned with executable SO-101 actions. For the VLA baselines, we finetune π_0 [1] and $\pi_{0.5}$ [67] using their open LeRobot implementations.

Inference and evaluation At test time, JOPAT uses receding-horizon action generation with DDIM sampling. The policy conditions on the two most recent RGB observations, denoises action, visual, and track tokens jointly, and executes only the decoded action chunk. Unless otherwise stated, we use a future-observation offset of $H = 16$, an action/track horizon of 19 steps, 10 DDIM denoising steps, and execute the first 8 actions before replanning. We report success rate over 10 evaluation rollouts per task for each method, giving 40 binary trials over four equally weighted tasks; with equal per-task counts, the pooled SR equals the task-average SR. We report two-sided 95% Wilson score intervals for the pooled SR and compare methods with a two-sided Fisher's exact test on the pooled 2×2 success tables.

Task setup

Fig. 3 visualizes the four LeRobot SO-101 tasks. Each column corresponds to one task, and the three rows show the initial configuration, an intermediate state, and a successful final state.

F Additional ablations

The main paper reports the LIBERO-Long joint-versus-single-modality ablation in Table 3, the real-robot modality ablation in Table 4, the visibility ablation in Table 5, and the LIBERO-Long pretraining scaling study in Table 6. Here we provide two supplementary real-robot analyses: the effect of DROID action-free video pretraining and the sensitivity to the future-observation offset. Both follow the real-robot protocol in Appendix E.

Table 11: **Track-based methods on LIBERO.** Published SR (%), each under its own protocol; rows are not directly rankable. The two ATM averages and the TraceVLA average are means of the displayed suite scores; the 4D-VLA average is quoted as published, which is 0.1 above the mean of its four displayed suite scores (88.5).

Method	Spatial	Object	Goal	Long	Avg.
ATM [47]	68.5	68.0	77.8	39.3	63.4
ATM Diffusion Policy [47]	79.0	81.0	58.7	44.0	65.7
TraceVLA [52]	84.6	85.2	75.1	54.1	74.8
4D-VLA [53]	88.9	95.2	90.9	79.1	88.6
JOPAT	**97.2**	**98.9**	**98.4**	**96.4**	**97.8**

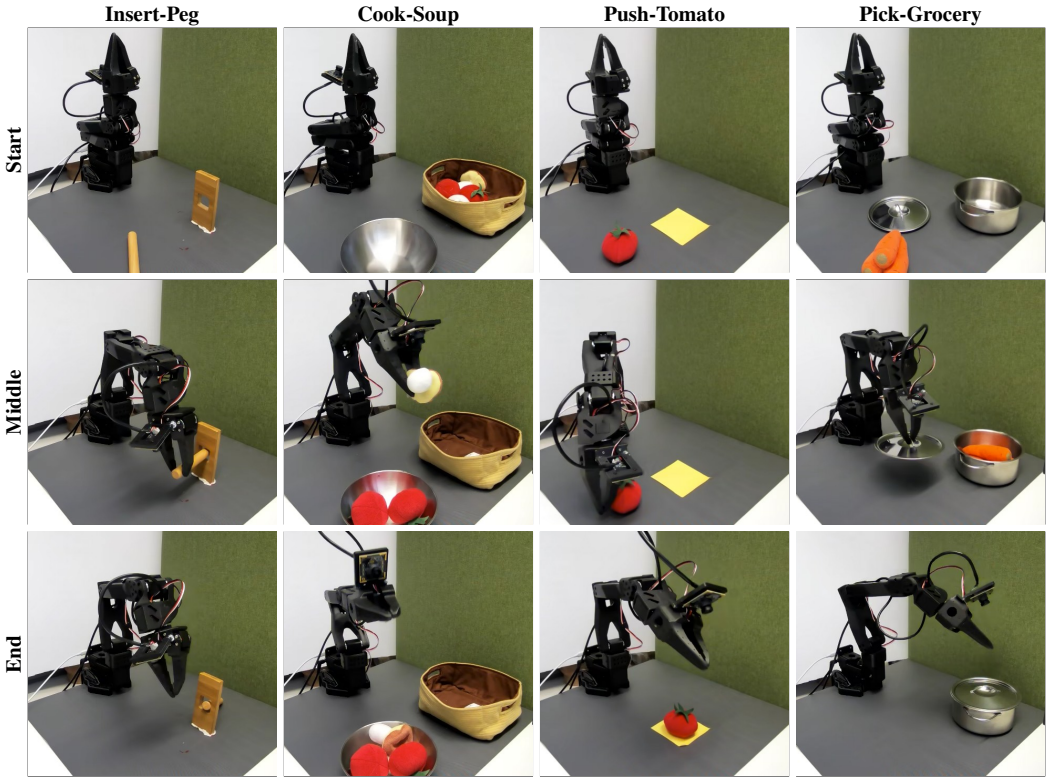

Figure 3: **Real-robot task setup.** Insert-Peg, Cook-Soup, Push-Tomato, and Pick-Grocery on the LeRobot SO-101 platform. The first row shows the initial configuration, the second row shows an intermediate state, and the third row shows successful task completion.

F.1 Action-free pretraining

Fig. 4 summarizes the effect of action-free video pretraining. With the same 50 real-robot finetuning demonstrations per task, DROID pretraining improves the average success rate from 10.0% to 57.5%. The largest gains occur on Push-Tomato and Cook-Soup, suggesting that action-free videos provide useful priors for object motion, occlusion handling, and contact-induced dynamics. Insert-Peg remains difficult, indicating that high-precision insertion also requires fine-grained 3D geometry or contact sensing beyond 2D tracks and RGB latents.

F.2 Prediction-horizon sensitivity

Fig. 5 shows that moderate future-observation offsets already capture the dynamics needed for most tasks. The $H = 16$ setting matches the performance of longer offsets on average, while avoiding the higher uncertainty associated with very long future-track prediction. Longer offsets can help multi-stage tasks such as Cook-Soup, but may hurt precision-sensitive tasks such as Insert-Peg, where future uncertainty produces noisy supervision for fine contact alignment.

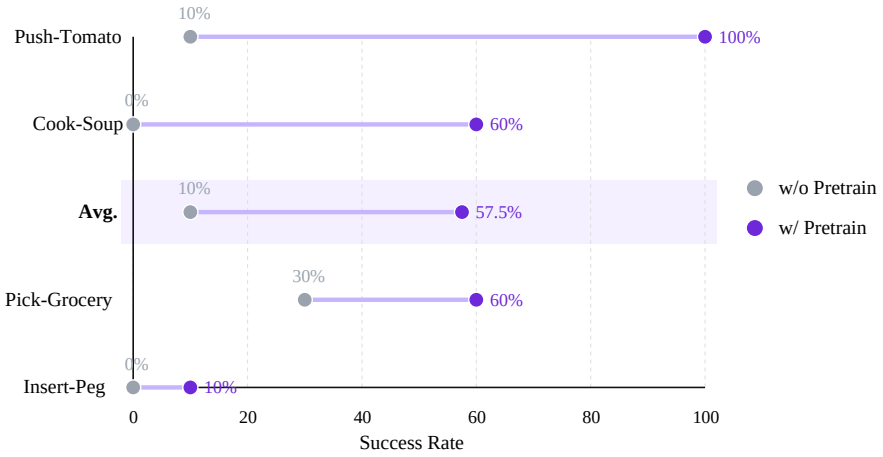

Figure 4: **Action-free pretraining ablation.** Average real-robot success rate with and without DROID action-free video pretraining.

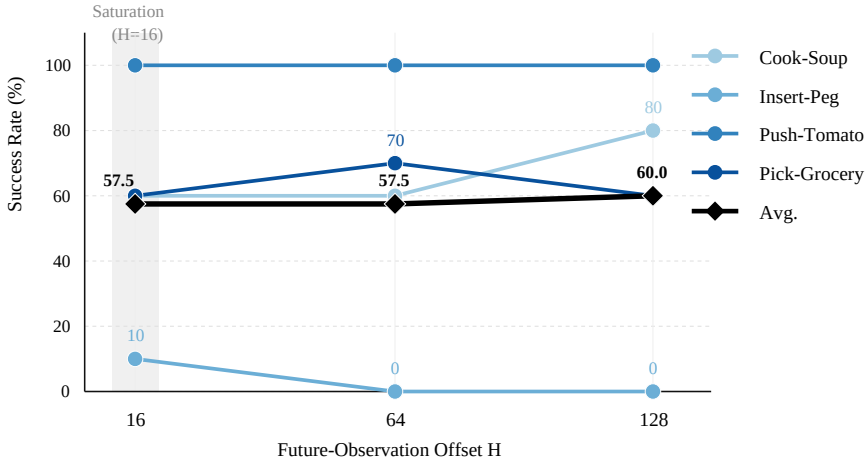

Figure 5: **Horizon sensitivity.** Average real-robot success rate for different future-observation offsets.

G Qualitative behavior and failure modes

Occlusion and off-screen motion In qualitative rollouts, the predicted tracks often remain temporally coherent when the robot arm occludes the target or when objects move near the image boundary. This behavior is consistent with the quantitative visibility ablation: the model learns to maintain a belief over hidden object motion rather than relying only on currently visible pixels. Such object permanence is particularly useful for long-horizon tasks in which the policy must continue acting through temporary perceptual dropout.

Task-level failure modes The remaining failures are task dependent:

- **Cook-Soup.** Failures often occur around the pot lid, where specular reflections and unusual lid orientations weaken visual grounding. Stronger illumination augmentation or depth cues may improve robustness.
- **Insert-Peg.** The task requires millimeter-level 3D alignment between the peg and socket. Two-dimensional tracks reduce perceptual drift but do not fully resolve contact geometry, leading to jamming or repeated failed insertions.

- **Push-Tomato.** Failures are usually caused by small localization or approach-angle errors that compound during pushing. Better contact dynamics or closed-loop state estimation may further improve reliability.
- **Pick-Grocery.** Failures mainly reflect limited spatial generalization in clutter. The policy can overfit to frequent grasp locations and struggle when object layouts differ from the demonstration distribution.

Overall, JOPAT is strongest when success depends on temporal consistency, object permanence, and robustness to visual dropout. Its main limitation is fine-grained contact geometry, suggesting that future extensions should combine the proposed pixel-track world-action model with explicit 3D sensing, tactile feedback, or contact-aware objectives.

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 proposed joint pixel-and-track world-action model and summarize the empirical claims. These claims are supported by experiments on LIBERO, modified OOD LIBERO tasks, real-world LeRobot tasks, and ablations in Sections 5–5.6.

   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 includes a Limitations paragraph in Section 6 and additional failure-mode discussion in Appendix G. These discuss limitations related to sparse 2D tracks, dependence on off-the-shelf trackers, contact geometry, inference speed, and static-camera assumptions.

   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 formal theoretical results requiring assumptions or proofs. The mathematical expressions are used to define the model, objectives, and inference procedure.

   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 describes the model architecture, training objectives, point-track construction, datasets, evaluation protocols, and key hyperparameters in Sections 4–5 and Appendix A. Real-robot task setup and evaluation details are provided in Appendix E.

   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: [Yes]

   Justification: The experiments use public datasets and benchmarks such as LIBERO, DROID, and OpenVid-1M, which are cited in the paper, and Appendix B specifies the training data and objectives behind every result. The data-processing pipeline, training code, and per-result training configurations are released at https://github.com/1mather/JOPAT_WAM.

   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 LIBERO and LeRobot protocols, action-free pretraining setup, finetuning data budgets, evaluation rollouts, model architecture, optimizer, diffusion settings, horizons, and other hyperparameters in Sections 4–5 and Appendix A.

   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: [Yes]

Justification: For the real-robot results, which have the smallest sample size, Table 2 reports success counts over 40 binary trials per method together with two-sided 95% Wilson score intervals, and Section 5.2 reports two-sided Fisher's exact tests between methods; Appendix E describes how both are computed. Latency in Appendix C is reported as mean $\pm$ standard deviation over 1,000 runs. Simulation results are reported as success rates under each benchmark's evaluation protocol without error bars.

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 GPU types and approximate wall-clock time for action-free pretraining, task-specific finetuning, and deployment-time inference, and Appendix C reports per-query inference latency on an RTX 4090.

   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 work studies robotic manipulation policies using public datasets, simulation benchmarks, and controlled real-robot experiments. The experiments do not involve human subjects, sensitive personal data, or unsafe deployment.

   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: [No]

    Justification: The paper focuses on technical contributions and discusses limitations and failure modes, but it does not currently include a dedicated discussion of both positive and negative societal impacts. Such a discussion could be added in a future version.

    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 high-risk pretrained language model, image generator, or scraped dataset. The work focuses on robotic manipulation policies evaluated in controlled simulation and laboratory settings.

    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: [No]

    Justification: The paper cites the main existing assets used, including LIBERO, DROID, OpenVid-1M, CoTrackerV3, and SDXL VAE. However, the current manuscript does not explicitly list the license and terms of use for each asset.

    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 introduces a method and reports real-robot experiments, but the current submission does not release a new dataset, benchmark, or model asset as a standalone artifact.

    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 involve crowdsourcing experiments or research with human subjects. The experiments are conducted on simulation benchmarks and controlled robotic manipulation tasks.

    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 involve human-subject research or crowdsourcing experiments, 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 does not use LLMs as an important, original, or non-standard component. Any writing or formatting assistance, if used, does not affect the methodology, experiments, or scientific claims.

    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.