ViPRA: Video Prediction for Robot Actions

Sandeep Routray[1,2] sroutra2@cs.cmu.edu
Hengkai Pan[1] hengkaip@cs.cmu.edu
Unnat Jain[2,3] unnatj@uci.edu
Shikhar Bahl[2] sbahl@skild.ai
Deepak Pathak[1,2] dpathak@cs.cmu.edu

[1]Carnegie Mellon University [2]Skild AI [3]University of California, Irvine

Abstract

Can we turn a video prediction model into a robot policy? Videos, including those of humans or teleoperated robots, capture rich physical interactions. However, most of them lack labeled actions, which limits their use in robot learning. We present **Video Prediction for Robot Actions** (ViPRA), a simple pretraining-finetuning framework that learns continuous robot control from these actionless videos. Instead of directly predicting actions, we train a video-language model to predict *both future visual observations and motion-centric latent actions*, which serve as intermediate representations of scene dynamics. We train these latent actions using perceptual losses and optical flow consistency to ensure they reflect physically grounded behavior. For downstream control, we introduce a chunked *flow matching decoder* that maps latent actions to robot-specific continuous action sequences, using only 100 to 200 teleoperated demonstrations. This approach avoids expensive action annotation, supports generalization across embodiments, and enables smooth, high-frequency continuous control upto 22 Hz via chunked action decoding. Unlike prior latent action works that treat pretraining as autoregressive policy learning, ViPRA explicitly models both what changes and how. Our method outperforms strong baselines, with a 16% gain on the SIMPLER benchmark and a 13% improvement across real world manipulation tasks. We will release models and code at `https://vipra-project.github.io`.

1 Introduction

Robots learn by doing, but collecting robot demonstrations, particularly at scale, is expensive, time-consuming, and limited by embodiment. In contrast, videos are abundant. From YouTube clips [1] of people performing tasks [2–5] to logs of teleoperated robots [6], they capture rich physical interactions, diverse objects, and long-horizon behaviors that are difficult to script or reproduce. The challenge is that most of these videos may not include action labels.

At the same time, recent advances in video prediction models [7–11] open up a new opportunity: learning directly from large corpora of *actionless videos*. Beyond preserving high-level task semantics, these generative models exhibit a strong grasp of object dynamics and fine-grained physical interactions. This naturally leads to a central question: **Can a video prediction model be transformed into a control policy for physical robots?** In this work, we explore this question through a simple and scalable pretraining-finetuning framework that adapts a powerful video-language model [7] into a robot policy capable of learning from passive videos.

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Embodied World Models for Decision Making.

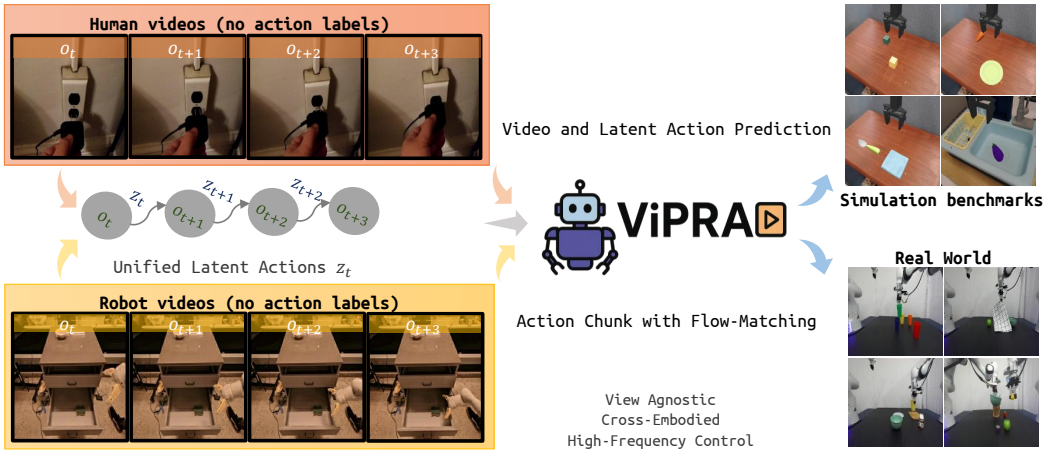


Figure 1: We present ViPRA, a framework to learn generalist robot policies from large-scale human and robot videos without action labels. It extracts motion-centric latent action sequences, pretrains a video-language model to jointly predict future visual observations and latent action chunks, and finetunes a flow matching decoder to map these latents to smooth, continuous action chunks for high-frequency, reactive control.

During pretraining, we co-train on two intuitive objectives: (i) predicting *what* happens next, in the form of *future visual observations*, and (ii) predicting *how* the scene evolves, using a compact intermediate representation known as *latent actions*[1]. By training with both objectives, the model learns to capture both semantic intent and physical dynamics. In contrast, prior latent action pretraining methods [12–15] treat pretraining purely as policy learning in latent space, without leveraging video prediction or modeling state transitions, and often use temporally coarse task-centric latent actions. Our framework instead predicts state transitions through video prediction and outputs a sequence of fine-grained *motion-centric* latent actions (3 to 6 Hz) over short horizons, capturing high-frequency dynamics critical for control. We further incorporate optical flow consistency as an additional supervision signal, promoting physically plausible and motion-aware latent representations.

Importantly, our pretraining leverages both unlabeled human and robot videos, which enables generalization across embodiments (see Figure 1, left). This broad exposure to passive visual data sets the foundation for effective finetuning with only a small number of teleoperated robot demonstrations. For finetuning on these demonstrations, we employ a *flow matching decoder* [16] that maps latent actions to smooth, continuous robot action chunks (see Figure 1, right). Unlike prior vision-language-action models (VLAs) [17–21], which required thousands of hours of labeled action trajectories[2], our decoder aligns latent transitions with embodiment-specific motor behaviors. This design amortizes inference latency via *action chunking*, enabling smooth, high-frequency control by producing multiple low-level actions in a single forward pass. Our policy can support control rates approaching 22 Hz, to our knowledge matched only by one other 7B-parameter model [22].

In summary, our contributions are as follows.

(i) A scalable method to extract fine-grained motion-centric latent actions from unlabeled human and robot videos using perceptual and optical flow consistency losses.
(ii) A novel pretraining framework for robot control that jointly predicts future visual states and motion-centric latent actions within a unified video-language model.
(iii) A data-efficient pretraining–finetuning framework that integrates flow matching and action chunking to enable smooth, high-frequency continuous control, operating at up to 22 Hz.
(iv) Demonstrate empirical gains of 16% on the SIMPLER benchmark [23] and 13% on real world tasks over the strongest prior continuous control baselines.

2 Related Work

Vision-Language-Action Models VLAs [17–21] extend vision-language models (VLMs) [24–28] by imitation learning on action-labeled robot demonstrations [6]. Recent works explore auxiliary

[1]Latent actions can be viewed as action-like tokens that summarize the transition between states without requiring access to ground-truth control commands

[2]10000 hours of pretraining OpenX data [6] and 5-100 hours of fine-tuning demonstrations

objectives including visual trace prediction [29], chain-of-thought reasoning [30], and conversational instruction tuning [31]. However, most existing VLAs require extensive labeled action data, creating a fundamental scalability bottleneck due to the prohibitive cost of data collection. Furthermore, these models focus primarily on grounding language in visual semantics while lacking explicit mechanisms for modeling physical dynamics or temporal structure in action generation. In contrast, ViPRA eliminates the labeled data requirement by leveraging unlabeled videos during pretraining and incorporates temporal dynamics through joint prediction of future visual states and multi-step latent actions, which provides robust priors for high-frequency control.

Robot Learning from Videos Videos offer a scalable source of information about object dynamics, task structure, and human behavior. Visual planning methods [32–38] use generative video models to plan in video or video-language space and rely on an inverse dynamics model to convert predicted frames into actions. While effective for long-horizon reasoning, these methods often incur high inference costs, limiting their suitability for high-frequency, dexterous control. Different from the above, policy supervision approaches [38, 39] use video models as supervision or reward sources to train policies.

Recent work explores joint training for video generation and action prediction [40, 41], with [40] also introducing decoupled action decoding to mitigate inference overhead, but evaluations are mostly on smaller scale datasets, simulation, and do not demonstrate scaling to internet-scale passive videos.

Other efforts leverage human videos to pretrain visual representations for downstream visuomotor control [42–45], or extract intermediate cues such as affordances [46–50], interactions [51], or visual traces [52–55] from unlabeled videos to guide policy learning. These approaches depend on structured priors or explicit cue extraction, which can constrain scalability. In contrast, we learn motion-centric latent actions that capture temporal dynamics and pair them with video-language grounding, enabling scalable learning directly from large action free video corpora.

Latent Action Spaces Latent action representations improve data efficiency by enabling learning from action free videos via self-supervised learning [56–60]. Recent methods impose discrete information bottlenecks with vector quantization encoders [61] and predict these tokens during policy learning [12–15, 62, 63], achieving strong real world results through imitation. Some train inverse dynamics models on limited labeled demonstrations before applying them to unlabeled video [36, 64], while others treat latent actions as abstract embodiments and jointly train policies with inverse dynamics model predictions across embodiments [65]. Another line of work uses these abstractions to build world simulators [66, 67] or plan in latent spaces [68–74] While these methods capture physical dynamics effectively, they struggle to generalize to novel settings due to limited semantic grounding. Video-language models can provide such multimodal grounding [32, 34, 35, 37, 41], but existing approaches are typically computationally heavy and slow at inference. In contrast to existing methods, ViPRA learns fine-grained, motion-centric latent actions that capture temporal dynamics while leveraging a video-language model [7] for semantic grounding. We train a unified latent space from large-scale, action free human and robot videos, enabling cross-embodiment transfer. By predicting action chunks during both latent pretraining and real-action finetuning, we amortize inference latency and achieve smooth, high-frequency control.

3 Background

We defer a detailed discussion of VQ-VAE, optical flow estimation for discrete latent actions, as well as behavior cloning and flow matching for continuous control to Appendix A.

4 Methodology

A generalist robotic agent must combine precise low-level control with environment-agnostic high-level intelligence. Video generation models are well-suited to this goal, as future-state prediction captures both physical interaction detail and task-related semantic context. Achieving this requires effectively utilizing large-scale data, architectures that expose motion-centric signals, and stable training pipelines. To this end, we present ViPRA: *(i)* learning motion-centric discrete latent actions from large-scale human and robot videos without action supervision, guided by perceptual and optical flow consistency; *(ii)* pretraining a multimodal video-language model to jointly predict future visual observations and latent action sequences, grounding temporal dynamics in semantics; and *(iii)*

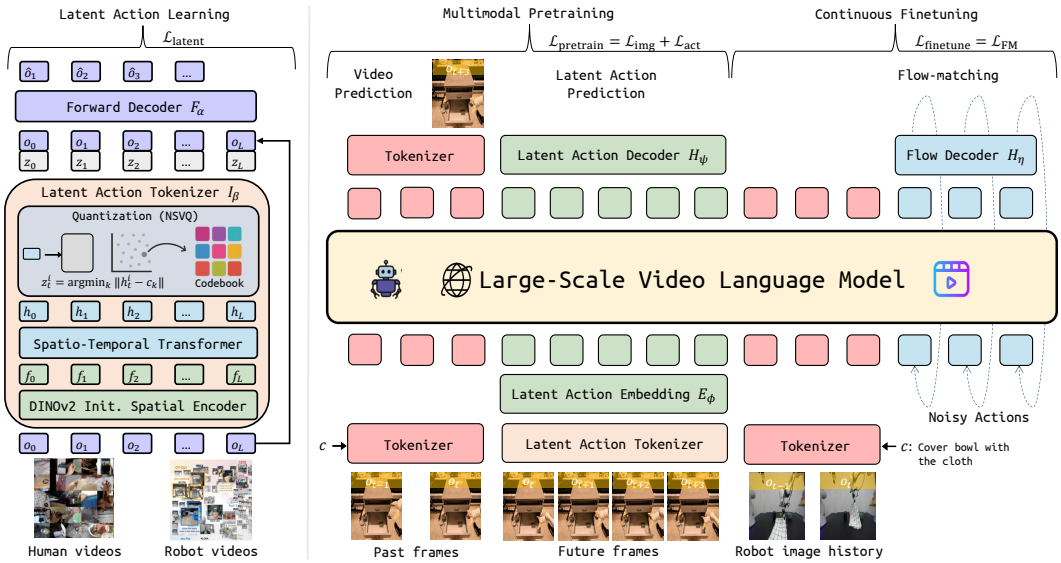


Figure 2: **ViPRA framework** comprises of: (1) **Latent Action Learning** (left): A neural quantization bottleneck extracts discrete latent actions z_t from image sequences $o_{0:L}$ in both human and robot videos, trained via reconstruction loss $\mathcal{L}_{\text{latent}}$ to capture motion-centric dynamics. (2) **Multimodal Pretraining** (center): A video-language model jointly predicts future observations o_{t+H} and latent action sequences $z_{t:t+H-1}$ from past frames (o_{t-1}, o_t) and task description c, using loss $\mathcal{L}_{\text{pretrain}}$. (3) **Continuous Finetuning** (right): A flow matching decoder maps latent actions to continuous robot actions $a_{t:t+H-1}$ using noisy action conditioning and loss $\mathcal{L}_{\text{FM}}$, enabling smooth, high-frequency control.

finetuning a flow matching decoder that maps latent actions to smooth, continuous control using only a few hundred demonstrations. This hierarchy leverages the rich physical priors of video models while ensuring the precision needed for real world robot control.

4.1 Latent Action Learning from Actionless Videos

We first train a latent action model to represent behavior in both human and robot videos without requiring action supervision. As illustrated in Figure 2 (left), this phase extracts motion-centric latent actions z_t that captures how the environment changes (inverse dynamics) and reciprocally also models the visual observations in response to these implicit actions (forward prediction).

Given a length-$(L+1)$ observation sequence $o_{0:L} = [o_0, o_1, \ldots, o_L]$ sampled from human or robot videos, our objective is to learn discrete latent action tokens $z_t = [z_t^1, z_t^2, \ldots, z_t^{N_{\text{latent}}}]$ that encode the motion dynamics at each timestep t. Here, N_{latent} denotes the number of latent action components, and each component z_t^i is quantized from a shared codebook $\mathcal{C}$ of size $|\mathcal{C}| = 8$.

We train an inverse dynamics encoder $I_\beta(z_t \mid o_{0:L})$ that predicts latent action token z_t by conditioning on the full observation sequence $o_{0:L}$. This non-causal design allows the latent action z_t to incorporate both past and future context, making it sensitive to local motion intent–for instance, distinguishing a pickup from a putdown based on surrounding frames. By providing the encoder with the entire clip, we reduce reconstruction ambiguity and force z_t to encode the minimal but sufficient information to explain the local transition.

We jointly train a forward decoder $F_\alpha(\hat{o}_{t+1} \mid o_{0:t}, z_{0:t})$ that predicts the future frame $\hat{o}_{t+1}$ given the history of observations $o_{0:t}$ and latent actions $z_{0:t}$. This reconstruction task ensures that the learned latent actions z_t contain sufficient information to explain scene dynamics. The model is optimized using three complementary loss components: pixel-level L_1 reconstruction loss $\mathcal{L}_{\text{rec}}$ for accurate frame prediction, perceptual loss $\mathcal{L}_{\text{LPIPS}}$ [75] for semantic consistency, and optical flow consistency loss $\mathcal{L}_{\text{flow}}$ to encourage physically plausible motion patterns:

$$\mathcal{L}_{\text{flow}} = \frac{1}{L} \sum_{t=2}^{L+1} \|\text{OF}(\hat{o}_t, \hat{o}_{t-1}) - \text{OF}(o_t, o_{t-1})\|_1 + \frac{1}{L} \sum_{t=1}^{L} \|\text{OF}(\hat{o}_t, \hat{o}_{t+1}) - \text{OF}(o_t, o_{t+1})\|_1 \quad (1)$$

where $\text{OF}(a, b)$ denotes optical flow between frames a and b computed via RAFT [76]. This loss encourages predicted frames to exhibit motion patterns consistent with ground truth, supporting temporally coherent dynamics. The total latent action learning loss combines all components:

$$\mathcal{L}_{\text{latent}} = \mathcal{L}_{\text{rec}} + \lambda_{\text{LPIPS}}\mathcal{L}_{\text{LPIPS}} + \mathbb{I}(\text{step} > \alpha_{\text{flow}})\lambda_{\text{flow}}\mathcal{L}_{\text{flow}}, \tag{2}$$

where the flow loss is activated only after α_{flow} warm-up steps to avoid instability from poor early reconstructions. Thus, latent actions serve as a representation of scene dynamics effectively bridging between visual observations and any embodiment-specific control commands. We provide further architecture details in Appendix B.

4.2 Leveraging Multimodal Video Models for Action Pretraining

After obtaining discrete latent actions, we design a pretraining scheme leveraging a powerful multi-modal video prediction model. Such models are trained on large-scale datasets to jointly reconstruct video tokens and predict aligned language captions, thereby encoding rich semantic cues and dynamic priors about how the world changes in their latent space. By aligning our discrete latent actions z_t with the outputs of the generative model, we can effectively pretrain a *high-level controller* that can learn from video clips.

To this end, we *jointly* predict future visual tokens and latent actions, unifying dynamic scene understanding and abstract control representation in a temporally coherent latent space. We use LWM's VQ-VAE encoder E_{VQ} to encode visual observations o_t into N_{tokens} discrete visual tokens $v_t = E_{\text{VQ}}(o_t)$. Given the most recent observation tokens (v_{t-1}, v_t) and a task description c, the model predicts a future frame tokens v_{t+H} that is H steps ahead, along with a latent action sequence $z_{t:t+H-1} = [z_t, z_{t+1}, \ldots, z_{t+H-1}]$ representing the transitions leading to future visual state o_{t+H}. This multi-step horizon encourages meaningful and distinct scene changes, providing robust conditioning for downstream action inference.

As shown in Figure 2 (center), we build on the instruction-tuned LWM-Chat-1M [7] as our base policy G_θ and extend it with two modules for latent action modeling: (i) a **Latent Action Embedding** head E_ϕ that maps each discrete latent token $z_t^i \in \mathcal{C}$ to a d_z-dimensional vector $\tilde{z}_t^i = E_\phi(z_t^i)$ in the model's token space, and (ii) a **Latent Action Token Decoder** H_ψ, a multi-layer perception (MLP) that autoregressively predicts the next latent token $\hat{z}_t^{i+1} = H_\psi\left(G_\theta\left(c, v_{t-1}, v_t, v_{t+H}, \tilde{z}_{<t}, \tilde{z}_t^{\leq i}\right)\right)$ from the transformer hidden state till position i. This allows the model to generate latent action sequences in the same autoregressive manner as language or video tokens, leveraging the multimodal token space learned during pretraining.

During training, we apply *teacher forcing* to both visual and action predictions. We use ground-truth future frame tokens $v_{t+H} = E_{\text{VQ}}(o_{t+H})$ as supervision targets for the visual prediction $\hat{v}_{t+H}^{i+1} = G_\theta\left(c, v_{t-1}, v_t, v_{t+H}^{\leq i}\right)$. The pretraining objective $\mathcal{L}_{\text{pretrain}}$ combines the visual and action components as

$$\mathcal{L}_{\text{pretrain}} = \underbrace{\sum_{i=1}^{N_{\text{tokens}}} \text{CE}(\hat{v}_{t+H}^i, v_{t+H}^i)}_{\mathcal{L}_{\text{img}}} + \underbrace{\sum_{k=t}^{t+H-1} \sum_{i=1}^{N_{\text{latent}}} \text{CE}(\hat{z}_k^i, z_k^i)}_{\mathcal{L}_{\text{act}}}, \tag{3}$$

where $\text{CE}(a, b)$ denotes the standard cross-entropy loss between logits for a and label b.

4.3 Continuous Adaptation

While the latent action pretrained video model provides robust semantic grounding, it lacks the physical precision needed for smooth, low-level robot control. To address this gap, we augment the pretrained model to output continuous actions, utilizing a flow matching decoder trained on real robot trajectories. This adaptation enables temporally smooth, physically consistent motor commands conditioned on visual and linguistic contexts.

As shown in Figure 2 (right), we augment the video model G_θ with two action-specific components: (i) an **Action Encoder** E_γ, and (ii) a **Flow Decoder** H_η. The encoder E_γ embeds continuous noisy actions $x_s \in \mathbb{R}^{H \times D}$ into the token space, while the decoder H_η predicts a flow field over the action chunk. Following the flow matching framework from Equation 7, we sample a target action sequence

$a_{t:t+H-1} \in \mathbb{R}^{H \times D}$, draw a noise sample $x_0 \sim \mathcal{N}(0, I)$, and interpolate:

$$x_s = s \cdot x_0 + (1 - s) \cdot a_{t:t+H-1}, \quad s \sim \text{Beta}(a, b).$$

We use $a = 1.5$ and $b = 1$ for sampling from Beta distribution. This noisy input is encoded via $f_s = E_\gamma(x_s, s)$, and passed into the transformer along with VQ-encoded image history tokens of (v_{t-1}, v_t) and language prompt c. The model predicts a flow field $\hat{g} = H_\eta\left(G_\theta\left(c, v_{t-1}, v_t, f_s\right)\right)$, which is supervised using the flow matching objective from Equation 9:

$$\mathcal{L}_{\text{FM}} = \left\| a_{t:t+H-1} - x_0 - (1 - s) \cdot \hat{g} \right\|_2^2 .$$

At inference time, given the visual history (o_{t-1}, o_t) and task instruction c, we iteratively solve for the continuous action chunk $a_{t:t+H-1}$ using forward Euler integration (Equation 10) of the predicted flow field from $s = 0$ to $s = 1$, over 10 uniform steps with $\Delta s = 0.1$. This continuous control refinement layer injects dynamics consistency and smoothness unavailable to the discrete latent tokens alone.

5 Experiments

To evaluate ViPRA, we conduct extensive experiments in both simulation and the real world to address the following research questions: (i) Can a generalist policy be trained to leverage both the physical dynamics and semantic understanding of video models? (ii) Does ViPRA efficiently exploit large-scale, actionless video data? (iii) Can multimodal pretraining yield strong high-level priors for downstream policy? (iv) How well does ViPRA adapt to high-frequency continuous control settings? (v) Does ViPRA outperform methods that do not exploit video foundation models?

5.1 Environments & Training

Training Dataset For learning latent actions and pretraining the video-language model, we use 198k human videos from Something-Something v2 [4] and a subset of actionless robot videos from the OpenX [6] dataset: 87k Fractal [77] videos, 25.4k BridgeV2 [78], and 85.6k Kuka [79] videos. We describe training details and hyperparameters for latent action learning in Appendix B, pretraining in Appendix C, and flow matching finetuning in Appendix D.

Simulation Benchmarks Following prior latent action works [12, 13], we benchmark ViPRA in SIMPLER [23], an open-source suite for evaluating generalist manipulation policies. We evaluate on four Bridge task with a 7-DoF WidowX arm, a benchmark designed to test generalization across diverse manipulation goals. Since SIMPLER [23] lacks finetuning data, we collect 100 diverse multi-task trajectories using a pretrained VLA model [12]. We provide details about our SIMPLER [23] tasks and LIBERO Long [80] benchmarks in Appendix E.

Real World Manipulation While SIMPLER [23] already provides a strong correlation between simulated and real world policy performance, we further strengthen our findings with rigorous evaluations on physical robots. We evaluate ViPRA on a bimanual setup with two 7-DoF Franka Panda robots. For single-arm experiments, we finetune on three multi-instruction tasks: (1) `pick up cloth and cover` $\langle$`object`$\rangle$, (2) `pick up` $\langle$`object`$_1\rangle$ `and place on` $\langle$`object`$_2\rangle$, and (3) `pick up` $\langle$`color`$_1\rangle$ `cup and stack on` $\langle$`color`$_2\rangle$ `cup`. We use GELLO [81] teleoperation to collect 180 trajectories, per task spanning 5 cup colors and 10 object types. For both simulation and real world settings, we report full success and partial success; partial success is defined as grasping the correct object, and full success requires completing the task (e.g., placing, stacking, covering). We evaluate with both seen and unseen objects, textures, colors, and shapes to test generalization. For real world evaluation, policies run using only a front-facing camera. We predict action chunks of length H=14 and replan after executing the first 7 steps. For this evaluation, we cap our policies at an effective closed-loop control rate of 3.5 Hz, though they can also operate at higher frequencies upto 22 Hz.

5.2 Baselines

We evaluate ViPRA against strong baselines across discrete and continuous action formulations.

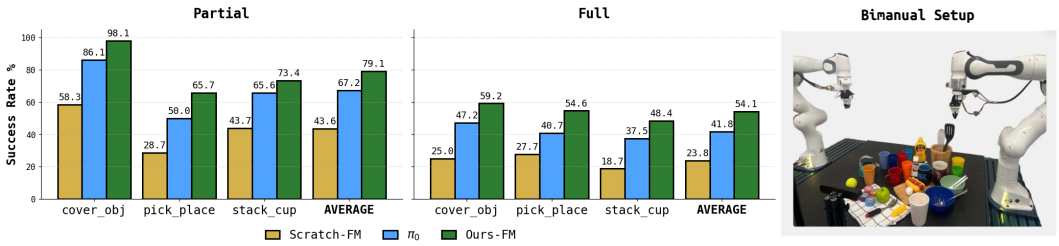


Figure 3: **Real World Evaluations** (Left) We report full and partial success rates on three manipulation tasks. ViPRA-FM significantly outperforms baselines. (Right) We show our physical robot setup and task objects.

	Discrete Actions					Continuous Actions				
Task	**Scratch-AR**	**VPT**	**OpenVLA**	**LAPA**	**ViPRA-AR**	**Scratch-FM**	**UniPI**	**π_0**	**UniVLA**	**ViPRA-FM**
					Success Rates					
StackG2Y	54.2	45.8	25.0	33.3	**66.7**	16.7	2.7	0.0	-	**54.2**
Carrot2Plate	58.3	37.5	20.8	41.7	**62.5**	33.3	2.7	20.8	-	**50.0**
Spoon2Cloth	37.5	**70.8**	50.0	66.7	66.7	50.0	0.0	4.17	-	**66.7**
Eggplant2Bask	58.3	50.0	58.3	70.8	**83.3**	66.7	0.0	**83.3**	-	79.2
AVG	52.1	51.0	38.6	53.1	**69.8**	41.7	1.7	27.1	42.7	**62.5**
					Grasp Rates					
StackG2Y	62.5	62.5	**70.8**	66.7	66.7	45.8	20.8	12.5	-	**62.5**
Carrot2Plate	54.2	54.2	37.5	**62.5**	**62.5**	45.8	33.2	25.0	-	**54.2**
Spoon2Cloth	75.0	79.2	75.0	**87.5**	75.0	62.5	22.2	16.7	-	**79.2**
Eggplant2Bask	66.7	70.8	91.7	79.2	**100**	87.5	16.0	**91.7**	-	**91.7**
AVG	65.6	66.7	68.8	73.9	**76.1**	60.4	23.1	36.5	50.0	**71.9**

Table 1: We report success rates and grasp rates on four bridge tasks in SIMPLER benchmark suite.

Scratch. As a reference, Scratch finetunes the video-language backbone (LWM) [7] directly on downstream tasks with image history and action chunking, without any pretraining. It establishes baseline performance when no latent action or video-based pretraining is used.

VLA baselines. We include OpenVLA [18] and π_0 [20]. OpenVLA [18] discretizes actions and adds a one-step autoregressive (AR) action predictor on top of a Prismatic-7B [27], while π_0 [20] augments a PaliGemma-3B [28] with a chunked flow matching (FM) decoder. Both use action-labeled robot demos from OpenX [6] containing 970k trajectories, while π_0 [20] also uses proprietary robot data.

Latent action baselines. We include LAPA [12] and UniVLA [13], both of which learn one-step temporally coarse latent actions without video prediction during pretraining. UniVLA [13] improves upon LAPA [12] by learning language-conditioned task-centric actions in DINOv2 [82] feature space. UniVLA [13] uses a Prismatic-7B [27] backbone with a L1 action decoder, whereas LAPA [12] uses an LWM [7] backbone with one-step AR prediction. Both rely on OpenX [6] demos, with UniVLA [13] additionally leveraging videos from Ego4D [2] and GNM [83].

Video learning baselines. We include UniPI [35] and VPT [36], both of which leverage videos for pretraining. UniPI [35] pretrains a text-conditioned video diffusion model and uses a learned IDM to recover actions from predicted videos. VPT [36] trains an IDM on labeled data to extract pseudo-actions that are then used as targets to pretrain a LWM [7] backbone. We report results from [12], which evaluated them on SIMPLER [23] in a comparable setting.

We include evaluations for both ViPRA-AR, aligned with discrete autoregressive baselines, and ViPRA-FM, aligned with continuous flow-matching methods.

5.3 Simulation Results

As shown in Table 1, ViPRA achieves the best average success rate in both discrete and continuous settings. In the discrete settings, ViPRA-AR surpasses LAPA [12] and OpenVLA [18] by a large margin (69.8% vs. 53.1% and 38.6%), excelling in precision-heavy tasks such as `StackG2Y`. In continuous settings, ViPRA-FM outperforms Scratch-FM by 20.8%, π_0 [20] by 35.4%, and UniVLA [13] by 19.8%, showing the benefits of motion-centric latents and multimodal video pretraining over training from scratch. Interestingly, due to the low noise and low ambiguity of the simulation setting, we find that ViPRA-AR outperforms the more expressive ViPRA-FM, which is slower to converge. However, ViPRA-FM achieves competitive performance, outperforming all other continuous/discrete baselines. ViPRA also surpasses other video learning approaches *i.e.*, UniPI [35] and VPT [36]. UniPI [35] frequently generates action sequences that diverge from the given instruction in longer-horizon

Exp.	Pretrain	Finetune	Succ.
LAPA	1-step L	1-step A	53.1
ViPRA-AR	FS + H-step L	H-step A	**69.8**
–*AC*	FS + 1-step L	1-step A	59.2
–*SP2*	H-step L	H-step A	59.4
–*LA*	FS only	H-step A	60.7
+*SP3*	H-step L	FS + H-step A	53.1
ViPRA-FM	FS + H-step L	H-step A	**62.5**
–*AC*	FS + 1-step L	1-step A	44.8
–*SP2*	H-step L	H-step A	53.2
+*SP3*	H-step L	FS + H-step A	31.3

Table 2: **Ablation study**: Average success rates in SIMPLER [84] on four Bridge tasks, evaluating the contributions of future state prediction (FS), latent action prediction (L), and action chunking (chunk size H=14) to justify ViPRA's design choices.

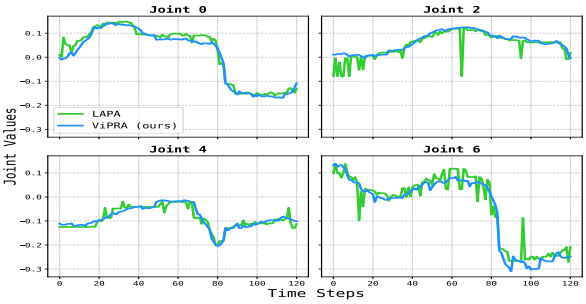


Figure 4: **Action smoothness**: ViPRA-FM (blue) produces smooth, continuous trajectories, while LAPA (green) exhibits local discontinuities and random spikes, often around contact events, despite tracking the overall trend. During real world deployment, such discontinuities triggered the emergency brake mechanism of the robot due to abrupt motor torque jumps.

settings, while VPT [36] provides only limited gains, indicating that IDM-derived pseudo-labels are sensitive to environment shifts. In contrast, ViPRA's joint use of latent action prediction and future state modeling yields stronger cross-environment transfer and more reliable task execution.

5.4 Real World Results

Figure 3 presents results on three real-world manipulation tasks. ViPRA-FM achieves the highest average success rate of 54.1%, surpassing π_0 [20] (41.8%) and Scratch-FM (23.8%). The improvement over Scratch-FM highlights the effectiveness of our latent action representations and confirms that our pretraining scheme learns transferable dynamics priors, accelerating downstream adaptation. Compared to π_0, which is pretrained on a larger corpus of action-labeled trajectories and further fine-tuned on each task, ViPRA-FM attains higher success with far less supervision. Qualitatively, it also exhibits robust retry behavior, repeatedly re-grasping after failed attempts, leading to high partial success rates, especially in `Cover-Obj`, where the cloth is consistently grasped even if placement is imperfect. These results demonstrate that ViPRA's video-based pretraining yields strong generalization and efficient adaptation to real-world control. We exclude discrete action models from real world evaluation, as their bin-based predictions exhibited unstable spikes under physical noise, often triggering emergency stops on the Franka arm. We provide additional analysis on generalization and robustness in Appendix G and evaluations of challenging bimanual tasks in Appendix H.

5.5 Ablations and Analysis

Isolating effect of future state and latent prediction. In Table 2, we disentangle the contributions of future state prediction and latent action chunk prediction. The LAPA [12] baseline (latent-only) reaches 53.1%, while adding future state prediction in ViPRA–*AC* boosts performance to 59.2%, showing that anticipating future observations improves control even with 1-step latent action. Removing future state prediction from our setup in ViPRA–*SP2* causes a drop from 69.8% to 59.4% (AR) and from 62.5% to 53.2% (FM), underscoring its importance for policy transfer. A state-only variant ViPRA–*LA* achieves 60.7%, comparable to ViPRA–*AC* but still below the full model, indicating that future state prediction alone is not sufficient. Finally, adding future state prediction at finetuning ViPRA+*SP3* degrades performance (53.1% AR, 31.3% FM), since the autoregressive structure couples action prediction with video prediction, causing compounding errors that drift irrecoverably on out-of-distribution states. ViPRA mitigates this by jointly predicting future visual tokens and latent action chunks during pretraining only.

Effect of action chunking. We apply chunking [85] in both latent and real action spaces: during pretraining the model predicts latent action sequences, and during finetuning it outputs continuous chunks via flow matching. Removing action chunking in ViPRA –*AC* from both pretraining and finetuning stages reduces performance to 59.2% (AR) and 44.8% (FM), as single-step actions fail to capture smooth temporal dynamics. By combining action chunking with future state prediction, the full ViPRA framework achieves the best results of 69.8% (AR) and 62.5% (FM), showing that the two objectives complement each other. Finally, action chunking is not only critical in pretraining but also enables robust, high-frequency control at test time. With KV caching, ViPRA 's flow matching

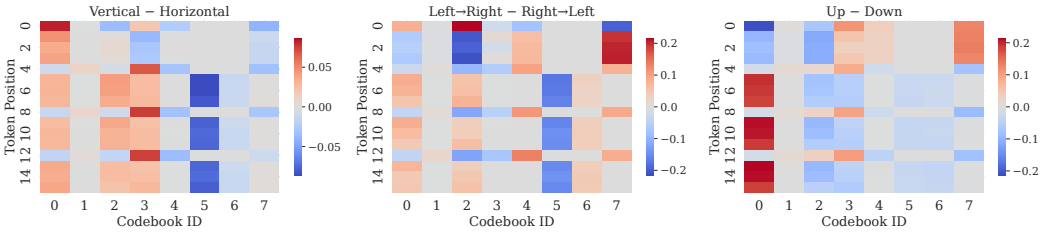
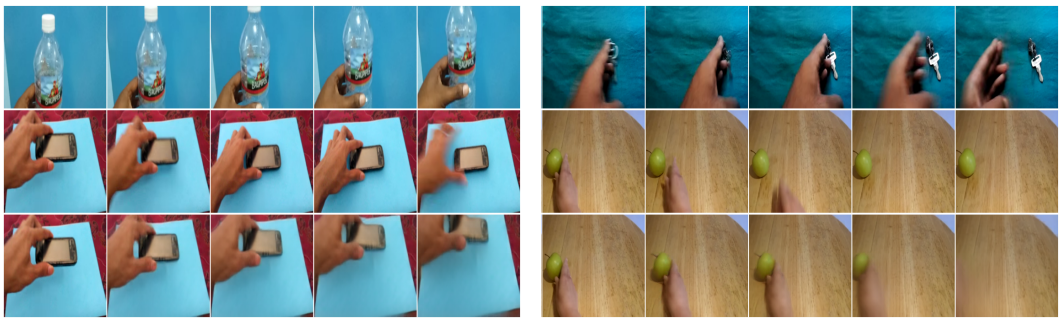


Figure 5: Positional codebook usage differences across action categories. Each heatmap shows the difference in per-position token usage between two groups: (left) vertical vs. horizontal, (middle) left → right vs. right → left, and (right) up vs. down. ViPRA learns positionally sensitive codes, with certain entries (e.g., 0, 2, 5) showing systematic variation, indicating that both token index and positions encode action dynamics.



(a) Vertical motion transfer (down → up).

(b) Horizontal motion transfer (right → left).

Figure 6: **Latent action transfer rollouts** illustrating how latent actions extracted from one video can be transferred to a different video. Each column is organized as: **Top** shows the *source clip* $o^A_{0:4}$, used to extract latent actions $z^A_{0:3} = I_\beta(o^A_{0:4})$ using the inverse model. **Middle** shows the *target clip* $o^B_{0:4}$ with different action which provides a new scene **Bottom** shows the *transferred rollout*, generated by the forward model $o^{B\to A}_{0:4} = F_\alpha(o^B_0, z^A_{0:3})$, where the latent actions are simulated from the first frame of the target clip. This produces novel behaviors in which the target scene undergoes the source motion.

decoder runs at 1.95 Hz per chunk, supporting effective control rates up to 22 Hz on hardware (chunk size $H = 14$). We provide additional discussion on the connection between action chunking and high-frequency execution in Appendix G.5.

ViPRA enables smooth continuous control. To assess action smoothness, we compare ViPRA-FM (blue) with LAPA [12] (green), a discrete policy, during closed-loop rollout. Both models are loaded into our inference pipeline and replayed over trajectories from the finetuning dataset, simulating deployment under real visual observations. As shown in Figure 4, both methods follow the intended trend, but LAPA [12] exhibits sharp local spikes at contact events or occlusions, where small perceptual shifts trigger abrupt bin flips. In contrast, ViPRA-FM's flow matching head yields smooth, demonstration-aligned commands. Since such discontinuities are unsafe on hardware, we restrict real world comparisons to continuous baselines. We provide more analysis on discrete and continuous policies in Appendix F, showing how quantization, loss design, and control space affect action smoothness and deployment.

Latent action analysis. In Figure 6, we perform cross-video latent transfer to test whether the learned latents capture action dynamics in a way that generalizes across different scenes. Concretely, given a source clip $o^A_{0:4}$ with motion type A, we extract latents $z^A_{0:3} = I_\beta\left(o^A_{0:4}\right)$ using the inverse model. We then take the first frame of an unrelated target clip o^B_0 (whose original action is motion B) and feed it into the forward model together with the source latents: $o^{B\to A}_{0:4} = F_\alpha\left(o^B_0, z^A_{0:3}\right)$. The resulting rollout exhibits motion A rather than B. For example, injecting *upward* latents into the first frame of a *downward* clip produces a reconstruction where the object moves *upward*. This demonstrates that the latents z encode transferable, dynamics-aware representations of motion, rather than simply memorizing the original video. Figure 5 complements this analysis by examining how the discrete latent codebook is used across tasks. We compute histograms over token position $\times$ code index to measure both which entries are selected and where they appear within a latent sequence. The distributions reveal structured patterns: certain code indices are consistently associated with particular motion directions, and their positions in the sequence reflect temporal dynamics. Together,

these results show that ViPRA's latent action space organizes itself around meaningful axes of control, enabling both transfer and interpretability.

6 Conclusion and Future Work

Video-language models provide a strong starting point for generalist robotic agents, as they capture both semantic intent and temporal dynamics crucial for real-world actions. Building on this, we introduced ViPRA, which learns motion-centric latent actions from large-scale actionless videos, pretrains a video-language model to jointly predict future states and latent actions, and refines these priors into smooth, high-frequency motor commands with a flow-matching decoder trained on only a few hundred demonstrations. Extensive evaluations in simulation and on real robots show that ViPRA outperforms methods relying solely on semantic pretraining, offering a scalable blueprint for general-purpose agents.

ViPRA learns efficiently and achieves strong real world results, but our current evaluation is limited to quasi-static, indoor tabletop tasks. Deploying the model in more diverse, dynamic, and unstructured settings remains an important direction for future work. Tackling such complex scenarios may also require incorporating additional sensing modalities, such as wrist-mounted cameras, proprioception, or tactile feedback. Another interesting direction involves leveraging the latent action decoder functions as a world model: given latent actions, it predicts future observations and can be conditioned on policy-sampled latents to generate multiple visual plans. This can enable alignment via reinforcement learning and test-time scaling through planning trees, where VLMs or heuristic functions act as reward models. From a societal perspective, training robotic agents involves physical and safety considerations. It is essential to ensure that such systems can operate safely around humans and in shared environments.

Acknowledgments and Disclosure of Funding

We thank Jason Liu and Tony Tao for their assistance in conducting the robot experiments, and Jim Yang, Mohan Kumar Srirama, and Tal Daniel for helpful discussions and feedback. This work was supported in part by the Air Force Office of Scientific Research (AFOSR) under Grant No. FA9550-23-1-0747 and by the Office of Naval Research (ONR) MURI under Grant No. N00014-24-1-2748.

References

[1] Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. YouTube-8M: A Large-Scale Video Classification Benchmark. *arXiv e-prints*, page arXiv:1609.08675, September 2016.

[2] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, Miguel Martin, Tushar Nagarajan, Ilija Radosavovic, Santhosh Kumar Ramakrishnan, Fiona Ryan, Jayant Sharma, Michael Wray, Mengmeng Xu, Eric Zhongcong Xu, Chen Zhao, Siddhant Bansal, Dhruv Batra, Vincent Cartillier, Sean Crane, Tien Do, Morrie Doulaty, Akshay Erapalli, Christoph Feichtenhofer, Adriano Fragomeni, Qichen Fu, Abrham Gebreselasie, Cristina Gonzalez, James Hillis, Xuhua Huang, Yifei Huang, Wenqi Jia, Weslie Khoo, Jachym Kolar, Satwik Kottur, Anurag Kumar, Federico Landini, Chao Li, Yanghao Li, Zhenqiang Li, Karttikeya Mangalam, Raghava Modhugu, Jonathan Munro, Tullie Murrell, Takumi Nishiyasu, Will Price, Paola Ruiz Puentes, Merey Ramazanova, Leda Sari, Kiran Somasundaram, Audrey Southerland, Yusuke Sugano, Ruijie Tao, Minh Vo, Yuchen Wang, Xindi Wu, Takuma Yagi, Ziwei Zhao, Yunyi Zhu, Pablo Arbelaez, David Crandall, Dima Damen, Giovanni Maria Farinella, Christian Fuegen, Bernard Ghanem, Vamsi Krishna Ithapu, C. V. Jawahar, Hanbyul Joo, Kris Kitani, Haizhou Li, Richard Newcombe, Aude Oliva, Hyun Soo Park, James M. Rehg, Yoichi Sato, Jianbo Shi, Mike Zheng Shou, Antonio Torralba, Lorenzo Torresani, Mingfei Yan, and Jitendra Malik. Ego4D: Around the World in 3,000 Hours of Egocentric Video. *arXiv e-prints*, page arXiv:2110.07058, October 2021.

[3] Kristen Grauman, Andrew Westbury, Lorenzo Torresani, Kris Kitani, Jitendra Malik, Triantafyllos Afouras, Kumar Ashutosh, Vijay Baiyya, Siddhant Bansal, Bikram Boote, Eugene Byrne, Zach Chavis, Joya Chen, Feng Cheng, Fu-Jen Chu, Sean Crane, Avijit Dasgupta, Jing Dong, Maria Escobar, Cristhian Forigua, Abrham Gebreselasie, Sanjay Haresh, Jing Huang, Md Mohaiminul Islam, Suyog Jain, Rawal Khirodkar, Devansh Kukreja, Kevin J Liang, Jia-Wei Liu, Sagnik Majumder, Yongsen Mao, Miguel Martin, Effrosyni Mavroudi, Tushar Nagarajan, Francesco Ragusa, Santhosh Kumar Ramakrishnan, Luigi Seminara, Arjun

Somayazulu, Yale Song, Shan Su, Zihui Xue, Edward Zhang, Jinxu Zhang, Angela Castillo, Changan Chen, Xinzhu Fu, Ryosuke Furuta, Cristina Gonzalez, Prince Gupta, Jiabo Hu, Yifei Huang, Yiming Huang, Weslie Khoo, Anush Kumar, Robert Kuo, Sach Lakhavani, Miao Liu, Mi Luo, Zhengyi Luo, Brighid Meredith, Austin Miller, Oluwatumininu Oguntola, Xiaqing Pan, Penny Peng, Shraman Pramanick, Merey Ramazanova, Fiona Ryan, Wei Shan, Kiran Somasundaram, Chenan Song, Audrey Southerland, Masatoshi Tateno, Huiyu Wang, Yuchen Wang, Takuma Yagi, Mingfei Yan, Xitong Yang, Zecheng Yu, Shengxin Cindy Zha, Chen Zhao, Ziwei Zhao, Zhifan Zhu, Jeff Zhuo, Pablo Arbelaez, Gedas Bertasius, David Crandall, Dima Damen, Jakob Engel, Giovanni Maria Farinella, Antonino Furnari, Bernard Ghanem, Judy Hoffman, C. V. Jawahar, Richard Newcombe, Hyun Soo Park, James M. Rehg, Yoichi Sato, Manolis Savva, Jianbo Shi, Mike Zheng Shou, and Michael Wray. Ego-Exo4D: Understanding Skilled Human Activity from First- and Third-Person Perspectives. *arXiv e-prints*, page arXiv:2311.18259, November 2023.

[4] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzyńska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, Florian Hoppe, Christian Thurau, Ingo Bax, and Roland Memisevic. The "something something" video database for learning and evaluating visual common sense. *arXiv e-prints*, page arXiv:1706.04261, June 2017.

[5] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Scaling Egocentric Vision: The EPIC-KITCHENS Dataset. *arXiv e-prints*, page arXiv:1804.02748, April 2018.

[6] Embodiment Collaboration, Abby O'Neill, Abdul Rehman, Abhinav Gupta, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, Albert Tung, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anchit Gupta, Andrew Wang, Andrey Kolobov, Anikait Singh, Animesh Garg, Aniruddha Kembhavi, Annie Xie, Anthony Brohan, Antonin Raffin, Archit Sharma, Arefeh Yavary, Arhan Jain, Ashwin Balakrishna, Ayzaan Wahid, Ben Burgess-Limerick, Beomjoon Kim, Bernhard Schölkopf, Blake Wulfe, Brian Ichter, Cewu Lu, Charles Xu, Charlotte Le, Chelsea Finn, Chen Wang, Chenfeng Xu, Cheng Chi, Chenguang Huang, Christine Chan, Christopher Agia, Chuer Pan, Chuyuan Fu, Coline Devin, Danfei Xu, Daniel Morton, Danny Driess, Daphne Chen, Deepak Pathak, Dhruv Shah, Dieter Büchler, Dinesh Jayaraman, Dmitry Kalashnikov, Dorsa Sadigh, Edward Johns, Ethan Foster, Fangchen Liu, Federico Ceola, Fei Xia, Feiyu Zhao, Felipe Vieira Frujeri, Freek Stulp, Gaoyue Zhou, Gaurav S. Sukhatme, Gautam Salhotra, Ge Yan, Gilbert Feng, Giulio Schiavi, Glen Berseth, Gregory Kahn, Guangwen Yang, Guanzhi Wang, Hao Su, Hao-Shu Fang, Haochen Shi, Henghui Bao, Heni Ben Amor, Henrik I Christensen, Hiroki Furuta, Homanga Bharadhwaj, Homer Walke, Hongjie Fang, Huy Ha, Igor Mordatch, Ilija Radosavovic, Isabel Leal, Jacky Liang, Jad Abou-Chakra, Jaehyung Kim, Jaimyn Drake, Jan Peters, Jan Schneider, Jasmine Hsu, Jay Vakil, Jeannette Bohg, Jeffrey Bingham, Jeffrey Wu, Jensen Gao, Jiaheng Hu, Jiajun Wu, Jialin Wu, Jiankai Sun, Jianlan Luo, Jiayuan Gu, Jie Tan, Jihoon Oh, Jimmy Wu, Jingpei Lu, Jingyun Yang, Jitendra Malik, João Silvério, Joey Hejna, Jonathan Booher, Jonathan Tompson, Jonathan Yang, Jordi Salvador, Joseph J. Lim, Junhyek Han, Kaiyuan Wang, Kanishka Rao, Karl Pertsch, Karol Hausman, Keegan Go, Keerthana Gopalakrishnan, Ken Goldberg, Kendra Byrne, Kenneth Oslund, Kento Kawaharazuka, Kevin Black, Kevin Lin, Kevin Zhang, Kiana Ehsani, Kiran Lekkala, Kirsty Ellis, Krishan Rana, Krishnan Srinivasan, Kuan Fang, Kunal Pratap Singh, Kuo-Hao Zeng, Kyle Hatch, Kyle Hsu, Laurent Itti, Lawrence Yunliang Chen, Lerrel Pinto, Li Fei-Fei, Liam Tan, Linxi "Jim" Fan, Lionel Ott, Lisa Lee, Luca Weihs, Magnum Chen, Marion Lepert, Marius Memmel, Masayoshi Tomizuka, Masha Itkina, Mateo Guaman Castro, Max Spero, Maximilian Du, Michael Ahn, Michael C. Yip, Mingtong Zhang, Mingyu Ding, Minho Heo, Mohan Kumar Srirama, Mohit Sharma, Moo Jin Kim, Muhammad Zubair Irshad, Naoaki Kanazawa, Nicklas Hansen, Nicolas Heess, Nikhil J Joshi, Niko Suenderhauf, Ning Liu, Norman Di Palo, Nur Muhammad Mahi Shafiullah, Oier Mees, Oliver Kroemer, Osbert Bastani, Pannag R Sanketi, Patrick "Tree" Miller, Patrick Yin, Paul Wohlhart, Peng Xu, Peter David Fagan, Peter Mitrano, Pierre Sermanet, Pieter Abbeel, Priya Sundaresan, Qiuyu Chen, Quan Vuong, Rafael Rafailov, Ran Tian, Ria Doshi, and Roberto Martín-Martín. Open X-Embodiment: Robotic Learning Datasets and RT-X Models. *arXiv e-prints*, page arXiv:2310.08864, October 2023.

[7] Hao Liu, Wilson Yan, Matei Zaharia, and Pieter Abbeel. World Model on Million-Length Video And Language With Blockwise RingAttention. *arXiv e-prints*, page arXiv:2402.08268, February 2024.

[8] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, Varun Jampani, and Robin Rombach. Stable Video Diffusion: Scaling Latent Video Diffusion Models to Large Datasets. *arXiv e-prints*, page arXiv:2311.15127, November 2023.

[9] Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, Devi Parikh, Sonal Gupta, and Yaniv Taigman. Make-A-Video: Text-to-Video Generation without Text-Video Data. *arXiv e-prints*, page arXiv:2209.14792, September 2022.

[10] Daquan Zhou, Weimin Wang, Hanshu Yan, Weiwei Lv, Yizhe Zhu, and Jiashi Feng. MagicVideo: Efficient Video Generation With Latent Diffusion Models. *arXiv e-prints*, page arXiv:2211.11018, November 2022.

[11] NVIDIA, Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, Daniel Dworakowski, Jiaojiao Fan, Michele Fenzi, Francesco Ferroni, Sanja Fidler, Dieter Fox, Songwei Ge, Yunhao Ge, Jinwei Gu, Siddharth Gururani, Ethan He, Jiahui Huang, Jacob Huffman, Pooya Jannaty, Jingyi Jin, Seung Wook Kim, Gergely Klár, Grace Lam, Shiyi Lan, Laura Leal-Taixe, Anqi Li, Zhaoshuo Li, Chen-Hsuan Lin, Tsung-Yi Lin, Huan Ling, Ming-Yu Liu, Xian Liu, Alice Luo, Qianli Ma, Hanzi Mao, Kaichun Mo, Arsalan Mousavian, Seungjun Nah, Sriharsha Niverty, David Page, Despoina Paschalidou, Zeeshan Patel, Lindsey Pavao, Morteza Ramezanali, Fitsum Reda, Xiaowei Ren, Vasanth Rao Naik Sabavat, Ed Schmerling, Stella Shi, Bartosz Stefaniak, Shitao Tang, Lyne Tchapmi, Przemek Tredak, Wei-Cheng Tseng, Jibin Varghese, Hao Wang, Haoxiang Wang, Heng Wang, Ting-Chun Wang, Fangyin Wei, Xinyue Wei, Jay Zhangjie Wu, Jiashu Xu, Wei Yang, Lin Yen-Chen, Xiaohui Zeng, Yu Zeng, Jing Zhang, Qinsheng Zhang, Yuxuan Zhang, Qingqing Zhao, and Artur Zolkowski. Cosmos World Foundation Model Platform for Physical AI. *arXiv e-prints*, page arXiv:2501.03575, January 2025.

[12] Seonghyeon Ye, Joel Jang, Byeongguk Jeon, Sejune Joo, Jianwei Yang, Baolin Peng, Ajay Mandlekar, Reuben Tan, Yu-Wei Chao, Bill Yuchen Lin, Lars Liden, Kimin Lee, Jianfeng Gao, Luke Zettlemoyer, Dieter Fox, and Minjoon Seo. Latent Action Pretraining from Videos. *arXiv e-prints*, page arXiv:2410.11758, October 2024.

[13] Qingwen Bu, Yanting Yang, Jisong Cai, Shenyuan Gao, Guanghui Ren, Maoqing Yao, Ping Luo, and Hongyang Li. UniVLA: Learning to Act Anywhere with Task-centric Latent Actions. *arXiv e-prints*, page arXiv:2505.06111, May 2025.

[14] Yi Chen, Yuying Ge, Weiliang Tang, Yizhuo Li, Yixiao Ge, Mingyu Ding, Ying Shan, and Xihui Liu. Moto: Latent Motion Token as the Bridging Language for Learning Robot Manipulation from Videos. *arXiv e-prints*, page arXiv:2412.04445, December 2024.

[15] NVIDIA, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi "Jim" Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. GR00T N1: An Open Foundation Model for Generalist Humanoid Robots. *arXiv e-prints*, page arXiv:2503.14734, March 2025.

[16] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow Matching for Generative Modeling. *arXiv e-prints*, page arXiv:2210.02747, October 2022.

[17] 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 e-prints*, page arXiv:2307.15818, July 2023.

[18] 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 e-prints*, page arXiv:2406.09246, June 2024.

[19] Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, Xiaofan Wang, Bei Liu, Jianlong Fu, Jianmin Bao, Dong Chen, Yuanchun Shi, Jiaolong Yang, and Baining Guo. CogACT: A Foundational Vision-Language-Action Model for Synergizing Cognition and Action in Robotic Manipulation. *arXiv e-prints*, page arXiv:2411.19650, November 2024.

[20] 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 e-prints*, page arXiv:2410.24164, October 2024.

[21] 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 e-prints*, page arXiv:2501.15830, January 2025.

[22] Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-Tuning Vision-Language-Action Models: Optimizing Speed and Success. *arXiv e-prints*, page arXiv:2502.19645, February 2025.

[23] Xuanlin Li, Kyle Hsu, Jiayuan Gu, Karl Pertsch, Oier Mees, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, Sergey Levine, Jiajun Wu, Chelsea Finn, Hao Su, Quan Vuong, and Ted Xiao. Evaluating Real-World Robot Manipulation Policies in Simulation. *arXiv e-prints*, page arXiv:2405.05941, May 2024.

[24] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[25] Xi Chen, Josip Djolonga, Piotr Padlewski, Basil Mustafa, Soravit Changpinyo, Jialin Wu, Carlos Riquelme Ruiz, Sebastian Goodman, Xiao Wang, Yi Tay, Siamak Shakeri, Mostafa Dehghani, Daniel Salz, Mario Lucic, Michael Tschannen, Arsha Nagrani, Hexiang Hu, Mandar Joshi, Bo Pang, Ceslee Montgomery, Paulina Pietrzyk, Marvin Ritter, AJ Piergiovanni, Matthias Minderer, Filip Pavetic, Austin Waters, Gang Li, Ibrahim Alabdulmohsin, Lucas Beyer, Julien Amelot, Kenton Lee, Andreas Peter Steiner, Yang Li, Daniel Keysers, Anurag Arnab, Yuanzhong Xu, Keran Rong, Alexander Kolesnikov, Mojtaba Seyedhosseini, Anelia Angelova, Xiaohua Zhai, Neil Houlsby, and Radu Soricut. PaLI-X: On Scaling up a Multilingual Vision and Language Model. *arXiv e-prints*, page arXiv:2305.18565, May 2023.

[26] Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. PaLM-E: An Embodied Multimodal Language Model. *arXiv e-prints*, page arXiv:2303.03378, March 2023.

[27] Siddharth Karamcheti, Suraj Nair, Ashwin Balakrishna, Percy Liang, Thomas Kollar, and Dorsa Sadigh. Prismatic VLMs: Investigating the Design Space of Visually-Conditioned Language Models. *arXiv e-prints*, page arXiv:2402.07865, February 2024.

[28] Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, Thomas Unterthiner, Daniel Keysers, Skanda Koppula, Fangyu Liu, Adam Grycner, Alexey Gritsenko, Neil Houlsby, Manoj Kumar, Keran Rong, Julian Eisenschlos, Rishabh Kabra, Matthias Bauer, Matko Bošnjak, Xi Chen, Matthias Minderer, Paul Voigtlaender, Ioana Bica, Ivana Balazevic, Joan Puigcerver, Pinelopi Papalampidi, Olivier Henaff, Xi Xiong, Radu Soricut, Jeremiah Harmsen, and Xiaohua Zhai. PaliGemma: A Versatile 3B VLM for Transfer. *arXiv e-prints*, page arXiv:2407.07726, July 2024.

[29] Dantong Niu, Yuvan Sharma, Giscard Biamby, Jerome Quenum, Yutong Bai, Baifeng Shi, Trevor Darrell, and Roei Herzig. LLARVA: Vision-Action Instruction Tuning Enhances Robot Learning. *arXiv e-prints*, page arXiv:2406.11815, June 2024.

[30] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *arXiv e-prints*, page arXiv:2201.11903, January 2022.

[31] Xiang Li, Cristina Mata, Jongwoo Park, Kumara Kahatapitiya, Yoo Sung Jang, Jinghuan Shang, Kanchana Ranasinghe, Ryan Burgert, Mu Cai, Yong Jae Lee, and Michael S. Ryoo. LLaRA: Supercharging Robot Learning Data for Vision-Language Policy. *arXiv e-prints*, page arXiv:2406.20095, June 2024.

[32] Yilun Du, Mengjiao Yang, Pete Florence, Fei Xia, Ayzaan Wahid, Brian Ichter, Pierre Sermanet, Tianhe Yu, Pieter Abbeel, Joshua B. Tenenbaum, Leslie Kaelbling, Andy Zeng, and Jonathan Tompson. Video Language Planning. *arXiv e-prints*, page arXiv:2310.10625, October 2023.

[33] 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. *arXiv e-prints*, page arXiv:2312.13139, December 2023.

[34] Po-Chen Ko, Jiayuan Mao, Yilun Du, Shao-Hua Sun, and Joshua B. Tenenbaum. Learning to Act from Actionless Videos through Dense Correspondences. *arXiv e-prints*, page arXiv:2310.08576, October 2023.

[35] Yilun Du, Mengjiao Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Joshua B. Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning Universal Policies via Text-Guided Video Generation. *arXiv e-prints*, page arXiv:2302.00111, January 2023.

[36] Bowen Baker, Ilge Akkaya, Peter Zhokhov, Joost Huizinga, Jie Tang, Adrien Ecoffet, Brandon Houghton, Raul Sampedro, and Jeff Clune. Video PreTraining (VPT): Learning to Act by Watching Unlabeled Online Videos. *arXiv e-prints*, page arXiv:2206.11795, June 2022.

[37] Junbang Liang, Ruoshi Liu, Ege Ozguroglu, Sruthi Sudhakar, Achal Dave, Pavel Tokmakov, Shuran Song, and Carl Vondrick. Dreamitate: Real-World Visuomotor Policy Learning via Video Generation. *arXiv e-prints*, page arXiv:2406.16862, June 2024.

[38] Calvin Luo, Zilai Zeng, Yilun Du, and Chen Sun. Solving New Tasks by Adapting Internet Video Knowledge. *arXiv e-prints*, page arXiv:2504.15369, April 2025.

[39] Calvin Luo, Zilai Zeng, Mingxi Jia, Yilun Du, and Chen Sun. Self-Adapting Improvement Loops for Robotic Learning. *arXiv e-prints*, page arXiv:2506.06658, June 2025.

[40] Shuang Li, Yihuai Gao, Dorsa Sadigh, and Shuran Song. Unified Video Action Model. *arXiv e-prints*, page arXiv:2503.00200, February 2025.

[41] Yanjiang Guo, Yucheng Hu, Jianke Zhang, Yen-Jen Wang, Xiaoyu Chen, Chaochao Lu, and Jianyu Chen. Prediction with Action: Visual Policy Learning via Joint Denoising Process. *arXiv e-prints*, page arXiv:2411.18179, November 2024.

[42] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3M: A Universal Visual Representation for Robot Manipulation. *arXiv e-prints*, page arXiv:2203.12601, March 2022.

[43] Sudeep Dasari, Mohan Kumar Srirama, Unnat Jain, and Abhinav Gupta. An Unbiased Look at Datasets for Visuo-Motor Pre-training. In *Conference on Robot Learning*, pages 1183–1198. PMLR, 2023.

[44] Tete Xiao, Ilija Radosavovic, Trevor Darrell, and Jitendra Malik. Masked Visual Pre-training for Motor Control. *arXiv e-prints*, page arXiv:2203.06173, March 2022.

[45] Siddharth Karamcheti, Suraj Nair, Annie S. Chen, Thomas Kollar, Chelsea Finn, Dorsa Sadigh, and Percy Liang. Language-Driven Representation Learning for Robotics. *arXiv e-prints*, page arXiv:2302.12766, February 2023.

[46] Shikhar Bahl, Russell Mendonca, Lili Chen, Unnat Jain, and Deepak Pathak. Affordances from Human Videos as a Versatile Representation for Robotics. *arXiv e-prints*, page arXiv:2304.08488, April 2023.

[47] Aditya Kannan, Kenneth Shaw, Shikhar Bahl, Pragna Mannam, and Deepak Pathak. DEFT: Dexterous Fine-Tuning for Real-World Hand Policies. *arXiv e-prints*, page arXiv:2310.19797, October 2023.

[48] Homanga Bharadhwaj, Abhinav Gupta, Shubham Tulsiani, and Vikash Kumar. Zero-Shot Robot Manipulation from Passive Human Videos. *arXiv e-prints*, page arXiv:2302.02011, February 2023.

[49] Shaowei Liu, Subarna Tripathi, Somdeb Majumdar, and Xiaolong Wang. Joint Hand Motion and Interaction Hotspots Prediction from Egocentric Videos. *arXiv e-prints*, page arXiv:2204.01696, April 2022.

[50] Mohit Goyal, Sahil Modi, Rishabh Goyal, and Saurabh Gupta. Human Hands as Probes for Interactive Object Understanding. *arXiv e-prints*, page arXiv:2112.09120, December 2021.

[51] Jia Zeng, Qingwen Bu, Bangjun Wang, Wenke Xia, Li Chen, Hao Dong, Haoming Song, Dong Wang, Di Hu, Ping Luo, Heming Cui, Bin Zhao, Xuelong Li, Yu Qiao, and Hongyang Li. Learning Manipulation by Predicting Interaction. *arXiv e-prints*, page arXiv:2406.00439, June 2024.

[52] Chuan Wen, Xingyu Lin, John So, Kai Chen, Qi Dou, Yang Gao, and Pieter Abbeel. Any-point Trajectory Modeling for Policy Learning. *arXiv e-prints*, page arXiv:2401.00025, December 2023.

[53] Homanga Bharadhwaj, Roozbeh Mottaghi, Abhinav Gupta, and Shubham Tulsiani. Track2Act: Predicting Point Tracks from Internet Videos enables Generalizable Robot Manipulation. *arXiv e-prints*, page arXiv:2405.01527, May 2024.

[54] Priyanka Mandikal and Kristen Grauman. DexVIP: Learning Dexterous Grasping with Human Hand Pose Priors from Video. *arXiv e-prints*, page arXiv:2202.00164, January 2022.

[55] Shikhar Bahl, Abhinav Gupta, and Deepak Pathak. Human-to-Robot Imitation in the Wild. *arXiv e-prints*, page arXiv:2207.09450, July 2022.

[56] Debidatta Dwibedi, Jonathan Tompson, Corey Lynch, and Pierre Sermanet. Learning Actionable Representations from Visual Observations. *arXiv e-prints*, page arXiv:1808.00928, August 2018.

[57] Anthony Liang, Pavel Czempin, Matthew Hong, Yutai Zhou, Erdem Biyik, and Stephen Tu. CLAM: Continuous Latent Action Models for Robot Learning from Unlabeled Demonstrations. *arXiv e-prints*, page arXiv:2505.04999, May 2025.

[58] 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.

[59] Dominik Schmidt and Minqi Jiang. Learning to Act without Actions. *arXiv e-prints*, page arXiv:2312.10812, December 2023.

[60] Zichen Jeff Cui, Hengkai Pan, Aadhithya Iyer, Siddhant Haldar, and Lerrel Pinto. DynaMo: In-Domain Dynamics Pretraining for Visuo-Motor Control. *arXiv e-prints*, page arXiv:2409.12192, September 2024.

[61] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural Discrete Representation Learning. *arXiv e-prints*, page arXiv:1711.00937, November 2017.

[62] Seungjae Lee, Yibin Wang, Haritheja Etukuru, H. Jin Kim, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. Behavior Generation with Latent Actions. *arXiv e-prints*, page arXiv:2403.03181, March 2024.

[63] Chenyu Yang, Davide Liconti, and Robert K. Katzschmann. VQ-ACE: Efficient Policy Search for Dexterous Robotic Manipulation via Action Chunking Embedding. *arXiv e-prints*, page arXiv:2411.03556, November 2024.

[64] Liang Xu, Ziyang Song, Dongliang Wang, Jing Su, Zhicheng Fang, Chenjing Ding, Weihao Gan, Yichao Yan, Xin Jin, Xiaokang Yang, Wenjun Zeng, and Wei Wu. ActFormer: A GAN-based Transformer towards General Action-Conditioned 3D Human Motion Generation. *arXiv e-prints*, page arXiv:2203.07706, March 2022.

[65] Joel Jang, Seonghyeon Ye, Zongyu Lin, Jiannan Xiang, Johan Bjorck, Yu Fang, Fengyuan Hu, Spencer Huang, Kaushil Kundalia, Yen-Chen Lin, Loic Magne, Ajay Mandlekar, Avnish Narayan, You Liang Tan, Guanzhi Wang, Jing Wang, Qi Wang, Yinzhen Xu, Xiaohui Zeng, Kaiyuan Zheng, Ruijie Zheng, Ming-Yu Liu, Luke Zettlemoyer, Dieter Fox, Jan Kautz, Scott Reed, Yuke Zhu, and Linxi Fan. DreamGen: Unlocking Generalization in Robot Learning through Video World Models. *arXiv e-prints*, page arXiv:2505.12705, May 2025.

[66] Shenyuan Gao, Siyuan Zhou, Yilun Du, Jun Zhang, and Chuang Gan. AdaWorld: Learning Adaptable World Models with Latent Actions. *arXiv e-prints*, page arXiv:2503.18938, March 2025.

[67] Jake Bruce, Michael Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, Yusuf Aytar, Sarah Bechtle, Feryal Behbahani, Stephanie Chan, Nicolas Heess, Lucy Gonzalez, Simon Osindero, Sherjil Ozair, Scott Reed, Jingwei Zhang, Konrad Zolna, Jeff Clune, Nando de Freitas, Satinder Singh, and Tim Rocktäschel. Genie: Generative Interactive Environments. *arXiv e-prints*, page arXiv:2402.15391, February 2024.

[68] David Ha and Jürgen Schmidhuber. World Models. *arXiv e-prints*, page arXiv:1803.10122, March 2018.

[69] Théophane Weber, Sébastien Racanière, David P. Reichert, Lars Buesing, Arthur Guez, Danilo Jimenez Rezende, Adria Puigdomènech Badia, Oriol Vinyals, Nicolas Heess, Yujia Li, Razvan Pascanu, Peter Battaglia, Demis Hassabis, David Silver, and Daan Wierstra. Imagination-Augmented Agents for Deep Reinforcement Learning. *arXiv e-prints*, page arXiv:1707.06203, July 2017.

[70] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to Control: Learning Behaviors by Latent Imagination. *arXiv e-prints*, page arXiv:1912.01603, December 2019.

[71] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning Latent Dynamics for Planning from Pixels. *arXiv e-prints*, page arXiv:1811.04551, November 2018.

[72] Alex X. Lee, Anusha Nagabandi, Pieter Abbeel, and Sergey Levine. Stochastic Latent Actor-Critic: Deep Reinforcement Learning with a Latent Variable Model. *arXiv e-prints*, page arXiv:1907.00953, July 2019.

[73] Philipp Wu, Alejandro Escontrela, Danijar Hafner, Ken Goldberg, and Pieter Abbeel. DayDreamer: World Models for Physical Robot Learning. *arXiv e-prints*, page arXiv:2206.14176, June 2022.

[74] Ramanan Sekar, Oleh Rybkin, Kostas Daniilidis, Pieter Abbeel, Danijar Hafner, and Deepak Pathak. Planning to Explore via Self-Supervised World Models. *arXiv e-prints*, page arXiv:2005.05960, May 2020.

[75] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. *arXiv e-prints*, page arXiv:1801.03924, January 2018.

[76] Zachary Teed and Jia Deng. RAFT: Recurrent All-Pairs Field Transforms for Optical Flow. *arXiv e-prints*, page arXiv:2003.12039, March 2020.

[77] 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 e-prints*, page arXiv:2212.06817, December 2022.

[78] Homer Walke, Kevin Black, Abraham Lee, Moo Jin Kim, Max Du, Chongyi Zheng, Tony Zhao, Philippe Hansen-Estruch, Quan Vuong, Andre He, Vivek Myers, Kuan Fang, Chelsea Finn, and Sergey Levine. BridgeData V2: A Dataset for Robot Learning at Scale. *arXiv e-prints*, page arXiv:2308.12952, August 2023.

[79] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. QT-Opt: Scalable Deep Reinforcement Learning for Vision-Based Robotic Manipulation. In *Conference on Robot Learning*, pages 651–673. PMLR, 2018.

[80] Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking Knowledge Transfer for Lifelong Robot Learning. *arXiv e-prints*, page arXiv:2306.03310, June 2023.

[81] Philipp Wu, Yide Shentu, Zhongke Yi, Xingyu Lin, and Pieter Abbeel. GELLO: A General, Low-Cost, and Intuitive Teleoperation Framework for Robot Manipulators. *arXiv e-prints*, page arXiv:2309.13037, September 2023.

[82] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning Robust Visual Features without Supervision. *arXiv e-prints*, page arXiv:2304.07193, April 2023.

[83] Jonathan Yang, Catherine Glossop, Arjun Bhorkar, Dhruv Shah, Quan Vuong, Chelsea Finn, Dorsa Sadigh, and Sergey Levine. Pushing the Limits of Cross-Embodiment Learning for Manipulation and Navigation. *arXiv e-prints*, page arXiv:2402.19432, February 2024.

[84] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles. *arXiv e-prints*, page arXiv:1612.01474, December 2016.

[85] Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware. *arXiv e-prints*, page arXiv:2304.13705, April 2023.

[86] Mohammad Hassan Vali and Tom Bäckström. NSVQ: Noise Substitution in Vector Quantization for Machine Learning. *IEEE Access*, 10:13598–13610, 2022.

[87] 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 e-prints*, page arXiv:2303.04137, March 2023.

[88] Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 Technical Report. *arXiv e-prints*, page arXiv:2412.15115, December 2024.

[89] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning Transferable Visual Models from Natural Language Supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[90] Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, Olivier Hénaff, Jeremiah Harmsen, Andreas Steiner, and Xiaohua Zhai. SigLIP 2: Multilingual Vision-Language Encoders with Improved Semantic Understanding, Localization, and Dense Features. *arXiv e-prints*, page arXiv:2502.14786, February 2025.

[91] Wilhelm Kutta. *Beitrag zur näherungsweisen Integration totaler Differentialgleichungen*. Teubner, 1901.

[92] Carl Runge. Über die numerische Auflösung von Differentialgleichungen. *Mathematische Annalen*, 46(2):167–178, 1895.

[93] Haoran He, Yang Zhang, Liang Lin, Zhongwen Xu, and Ling Pan. Pre-Trained Video Generative Models as World Simulators. *arXiv e-prints*, page arXiv:2502.07825, February 2025.

As part of the supplementary material, we include additional details about the following.

A **Background**: Covers key paradigms relevant for ViPRA: VQ-VAE for learning discrete latent actions, optical flow estimation, behavior cloning for direct action prediction and flow matching for smooth continuous control.

B **Latent Action Learning**: Architecture design, loss formulations, and training protocols for discrete action codebook learning, including hyperparameter configurations and optimization strategies.

C **Multimodal Video-Action Integration**: Implementation details for extending LWM with latent actions, including embedding architecture, decoder design, and joint training methodology.

D **Continuous Control via Flow Matching**: Complete specification of noise scheduling, action encoder/decoder architectures, and end-to-end training procedure.

E **Simulation Benchmarks**: Detailed description of SIMPLER tasks and additional LIBERO Long benchmark.

F **Action Output Analysis**: Comparative visualization and discussion of predicted action trajectories across discrete and continuous policies, highlighting the impact of quantization, loss formulations, and control space choices on smoothness and deployment behavior.

G **Real World Experiments**: Detailed description of hardware setup, task design, policy generalization, retrying behavior, and the impact of action chunking on control frequency and real-time performance in physical deployments.

H **Bimanual Manipulation Tasks**: Evaluation of ViPRA-FM on two real world dual-arm tasks requiring spatial coordination and tool use, including task setup, challenges, quantitative results, and rollout visualizations from real robot executions.

Code, checkpoints, and latent action labeled data and rollout videos will be released at: `https://vipra-project.github.io`.

A Background

We review key paradigms relevant for ViPRA: VQ-VAE for imposing information bottleneck to learn discrete latent actions, optical flow estimation, behavior cloning for direct action prediction and flow matching for smooth continuous control.

Vector-Quantised VAEs (VQ-VAE). An encoder e_ς maps an observation o_t to a cootinuou lat estimation,ent vector $\tilde{h}_t = [\tilde{h}_t^1, \ldots, \tilde{h}_t^N]$, which is quantized to a sequence of nearest codewords $z_t = [z_t^1, \ldots, z_t^N] \in \mathcal{C}$ using a shared discrete codebook. A decoder d_ς reconstructs the observation $\hat{o}_t = d_\psi(z_t + \text{Err})$, where Err is a gradient estimator used to allow backpropagation through the non-differentiable quantization operation.

In traditional VQ-VAE [61], this estimator takes the form

$$\text{Err}_{\text{STE}} = \text{sg}(\tilde{h}_t - z_t), \tag{4}$$

where $\text{sg}(\cdot)$ denotes the stop-gradient operator. The decoder input $z_t + \text{Err}_{\text{STE}}$ preserves the forward pass while enabling gradients to bypass the non-differentiable argmin. However, this approach typically requires auxiliary losses, such as the codebook and commitment losses, to stabilize training and encourage codebook usage.We adopt the NSVQ formulation [86], which replaces the deterministic STE with a stochastic noise-injected surrogate

$$\text{Err}_{\text{NSVQ}} = \|z_t - \tilde{h}_t\| \cdot \tilde{\epsilon} \tag{5}$$

where $\tilde{\epsilon} = \epsilon / \|\epsilon\|$ and $\epsilon \sim \mathcal{N}(0, I)$. The decoder thus receives $\hat{o}_t = d_\psi(z_t + \text{Err}_{\text{NSVQ}})$. Crucially, NSVQ enables gradients to flow to both the encoder and codebook using only the reconstruction loss, without requiring additional codebook loss terms. The noise-injected gradient estimator, combined with the *unused codebook replacement* technique applied during early training, significantly improves training stability and mitigates codebook collapse, a common issue in VQ-VAE training.

RAFT based Optical Flow Given two images, o_a and o_b, RAFT [76] obtains the dense displacement field $\mathbf{f}_{a \to b} \in \mathbb{R}^{H \times W \times 2}$ that maps each pixel in frame o_a to its location in o_b. It first extracts feature maps $\phi(o_a),\ \phi(o_b) \in \mathbb{R}^{H' \times W' \times d}$ (where ϕ is the feature extractor) and builds the all-pair correlation $\mathcal{R}$, with $\mathcal{R}_{ij,kl} = \langle \phi_{ij}(o_a), \phi_{kl}(o_b) \rangle$. The flow prediction is then refined iteratively: $\mathbf{f}^{(k+1)} = \mathbf{f}^{(k)} + \Delta \mathbf{f}^{(k)}(\mathcal{R})$. When applied to temporally close frames in a video, this flow field f can give a good estimate of motion consistency.

Behavior Cloning (BC) is a supervised learning paradigm in robotics that learns policies directly from expert demonstrations. Given a dataset $\mathcal{D} = \{(o_t, a_t)\}_{t=1}^T$ of observation-action pairs from expert trajectories, BC trains a parameterized policy $\pi_{\text{BC}}(a_t | o_t; \theta)$ to minimize a distance metric between predicted and ground-truth actions:

$$\min_\theta \ \mathbb{E}_{(o_t, a_t) \sim \mathcal{D}} \left[d(\pi_{\text{BC}}(a_t | o_t; \theta), a_t) \right], \tag{6}$$

where $d(\cdot, \cdot)$ is typically the L1 or L2 distance for continuous actions or cross-entropy for discrete actions. This framework has been extended with high-capacity architectures: diffusion models [85, 87] parameterize $\pi_{\text{BC}}(a_t | o_t; \theta)$ as a denoising process that learns $p(a_t | o_t)$ through iterative refinement, while VLAs leverage pretrained language models [24, 88] and visual encoders [82, 89, 90] as the backbone architecture for θ, enabling multimodal grounding of actions in visual and linguistic contexts.

Flow Matching [16] provides an alternative to diffusion models for learning continuous normalizing flows. While diffusion models learn the full denoising process, flow matching directly learns the vector field that transports samples from a source distribution to a target distribution. This approach offers computational advantages for robotics applications where real-time inference is critical.

Flow matching trains a neural network g_θ to predict the velocity field along a straight-line interpolation path. Given a source sample x_0 (typically Gaussian noise) and target sample x_1 (e.g., robot actions), the interpolation creates a path:

$$u_s = s \cdot x_0 + (1 - s) \cdot x_1, \quad \text{where} \quad s \in [0, 1] \text{ parameterizes the interpolation} \tag{7}$$

The model learns to predict the velocity field that guides samples along this path:

$$\frac{\partial}{\partial s} u_s = g_\theta(u_s, s | y), \quad \text{where} \quad y \text{ represents conditioning inputs} \tag{8}$$

In robotics applications, y typically includes visual observations and language commands that specify the desired behavior.

The training objective teaches the model to predict the correct velocity by minimizing the difference between predicted and true flow direction:

$$\mathcal{L}_{\mathrm{FM}} = \mathbb{E}_{(y,x_1)\sim\mathcal{D},\, s\sim\mathcal{U}[0,1]} \left\| \underbrace{x_1 - x_0}_{\text{true direction}} - (1-s) \cdot \underbrace{g_\theta(u_s, s \mid y)}_{\text{predicted velocity}} \right\|_2^2 . \tag{9}$$

At inference time, samples are generated by integrating the predicted velocity field from noise ($s = 0$) to the target ($s = 1$):

$$u_{s+\Delta s} = u_s + \Delta s \cdot g_\theta(u_s, s|y), \quad \text{where} \quad \Delta s \text{ is the integration step size} \tag{10}$$

This produces the final sample $x_1 \approx u_1$. Forward Euler integration is commonly used due to its efficiency [20], though more sophisticated solvers like Heun's method or Runge-Kutta can improve stability for high-dimensional control tasks [91, 92]. Flow matching has demonstrated superior smoothness and precision compared to direct action prediction, particularly for temporally extended manipulation tasks [15, 20].

B Latent Action Learning

We detail our latent action learning framework in Algorithm 1, which extracts discrete action tokens from video sequences using a combination of reconstruction, perceptual, and optical flow losses. A detailed diagram of this procedure is shown in Figure 7, and the complete training configuration–including model architecture and optimization hyperparameters–is provided in Table 3.

The inverse dynamics encoder maps each frame o_t into a sequence of spatial features using a DINOv2 [82]-initialized backbone. These features are enriched with clip-level context through factorized spatio-temporal attention layers, where the temporal branch employs bidirectional attention to aggregate information across the full sequence. The contextualized features are then discretized via Noise-Substitution Vector Quantization (NSVQ) [86], producing N_{latent} discrete codes selected from the shared codebook $\mathcal{C}$, which serve as the latent action tokens z_t.

The forward decoder mirrors this architecture with a factorized spatio-temporal transformer, but applies causal temporal attention so that predictions depend only on the past. It jointly attends to the latent action sequence $z_{0:t}$ and the observation history $o_{0:t}$ to reconstruct the next frame o_{t+1}. In addition, we integrate the action-conditioning modules proposed by [93] before each spatio-temporal block in the decoder to better align action tokens with visual dynamics.

Algorithm 1 Latent Action Learning (Training Step)

Require: Video clip of $L+1$ observations $o_{0:L} \in \mathbb{R}^{(L+1)\times H\times W\times 3}$
Require: Hyperparameters: LPIPS weight λ_{LPIPS}, Flow weight λ_{flow}, Flow start step α_{flow}
Require: Codebook $\mathcal{C} \in \mathbb{R}^{K\times D}$ with K codes of dimension D

Compute local motion aware embeddings: $h_{0:L} \leftarrow I_\beta(o_{0:L})$
for $t = 0$ **to** $L-1$ **do**
  Quantize embedding to latent: $z_t \leftarrow \text{NSVQ}(h_t, \mathcal{C})$
  Decode next frame: $\hat{o}_{t+1} \leftarrow F_\alpha(o_{0:t}, z_{0:t})$
end for

$\mathcal{L}_{\text{rec}} \leftarrow \sum_{t=0}^{L-1} \|\hat{o}_{t+1} - o_{t+1}\|_1$
$\mathcal{L}_{\text{LPIPS}} \leftarrow \sum_{t=0}^{L-1} \text{LPIPS}(\hat{o}_{t+1}, o_{t+1})$
if step $> \alpha_{\text{flow}}$ **then**

  $\mathcal{L}_{\text{flow}} \leftarrow \frac{1}{L}\sum_{t=1}^{L} \Big(\|\text{OF}(\hat{o}_t, \hat{o}_{t-1}) - \text{OF}(o_t, o_{t-1})\|_1 + \|\text{OF}(\hat{o}_t, \hat{o}_{t+1}) - \text{OF}(o_t, o_{t+1})\|_1 \Big)$
else
  $\mathcal{L}_{\text{flow}} \leftarrow 0$
end if

$\mathcal{L}_{\text{latent}} \leftarrow \mathcal{L}_{\text{rec}} + \lambda_{\text{LPIPS}}\mathcal{L}_{\text{LPIPS}} + \lambda_{\text{flow}}\mathcal{L}_{\text{flow}}$
Update parameters via AdamW optimizer

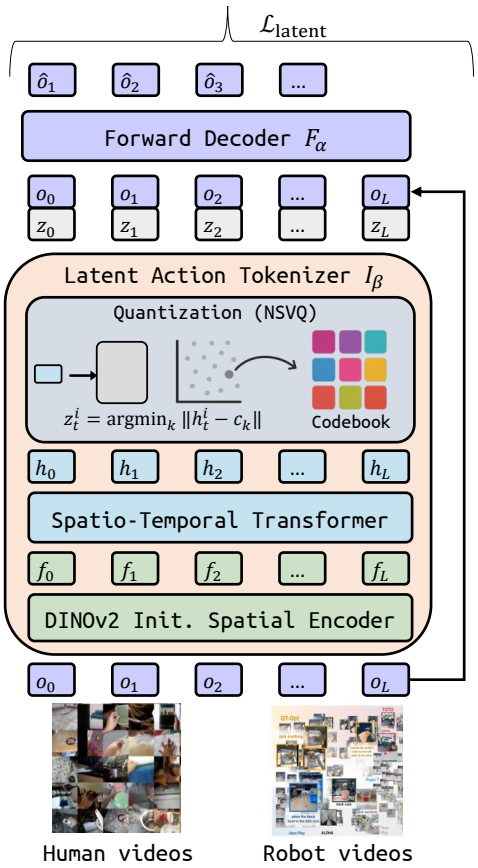


Figure 7: **Latent action learning framework.** Given a sequence of frames $o_{0:L}$, an inverse dynamics encoder $I_\beta(z_t \mid o_{0:L})$ maps the observation clip into discrete latent tokens z_t via vector quantization. A forward decoder $F_\alpha(\hat{o}_{t+1} \mid o_{0:t}, z_{0:t})$ then reconstructs the next frame $\hat{o}_{t+1}$ conditioned on the observation history and latent action sequence. Training combines reconstruction, perceptual (LPIPS), and optical flow consistency losses to ensure that the latent tokens capture physically grounded and temporally localized action information.

Hyperparameter	**Value**
Training Configuration	
Optimizer	AdamW
Base Learning Rate	1e-4
DINO Enc. Learning Rate	1e-5
Optimizer Momentum	$\beta_1, \beta_2 = 0.9, 0.99$
Batch Size	128
Grad. Norm Clip	4.0
Total Steps	240000
Image Augmentation	RandomResizeCrop
Flow Start Step α_{flow}	60000
Losses	$\mathcal{L}_{\text{rec}} + \lambda_{\text{LPIPS}}\mathcal{L}_{\text{LPIPS}} + \lambda_{\text{flow}}\mathcal{L}_{\text{flow}}$
LPIPS Weight λ_{LPIPS}	0.5
Flow Weight λ_{flow}	0.1 (after α_{flow})
GPU	8 Nvidia H100 (168 hours)
Inverse Dynamics Encoder I	
Backbone Init.	DINOv2 [82]
Embedding Dim	768
Spatio-temporal Layers	6
Attention Heads	16
Attention Head Dim	64
Latent Action Quantization	
Codebook Size $\lvert\mathcal{C}\rvert$	8
Quantized Token Dim	32
Quantization Method	NSVQ [86]
Codebook Refresh Interval	Every 10 till 100, every 100 till 1000, every 1000 till 10000
Codebook Refresh Strategy	Re-init Unused, Re-shuffle Used
Forward Decoder F_α	
Embedding Dim	768
Spatio-temporal Layers	8
Attention Heads	16
Attention Head Dim	64

Table 3: Hyperparameters for latent action learning.

C Multimodal Video Pretraining with Latent Actions

We augment a pretrained multimodal video model G_θ (LWM-Chat-1M [7]) with an embedding layer E_ϕ and a decoder H_ψ for latent action processing. The model jointly predicts future visual tokens and latent action sequences, conditioned on past frames and task context.

The latent action embedding head E_ϕ maps each code $z_t \in \mathcal{C}$ into the token space of G_θ, and the decoder head H_ψ predicts next-token logits over the latent vocabulary. Training uses teacher forcing for both video tokens (from a frozen VQ-VAE) and latent tokens with cross-entropy loss. The complete training procedure and hyperparameters are detailed in Algorithm 2 and Table 4.

Algorithm 2 Multimodal Video Pretraining via Video and Latent Action Prediction

Require: History frames: (o_{t-1}, o_t)
Require: Target frame: o_{t+H}
Require: Task description: c (text string)
Require: Labels *i.e.* latent action chunk $z_{t:t+H-1} \in \mathcal{C}^H$
Require: Pretrained models: VQ-VAE encoder E_{VQ}, video model G_θ
Require: Trainable components: random initialized embedding layer E_ϕ, decoder head H_ψ

Tokenize input frames: $v_{t-1}, v_t \leftarrow E_{\text{VQ}}(o_{t-1}), E_{\text{VQ}}(o_t)$
Tokenize target frame: $v_{t+H} \leftarrow E_{\text{VQ}}(o_{t+H})$
Encode text prompt: $\tilde{c} \leftarrow \texttt{Tokenizer}(c)$
Embed latent actions: $\tilde{z}_{t:t+H-1} \leftarrow E_\phi\left(z_{t:t+H-1}\right)$
for $i = 1$ **to** N_{tokens} **do**
  $\hat{v}^i_{t+H} \leftarrow G_\theta\left(\tilde{c}, v_{t-1}, v_t, \hat{v}^{<i}_{t+H}\right)$ {Autoregressive prediction}
end for
$\mathcal{L}_{\text{img}} \leftarrow \sum_{i=1}^{N_{\text{tokens}}} \text{CE}\left(\hat{v}^i_{t+H}, v^i_{t+H}\right)$ {Image token loss}
for $k = t$ **to** $t + H - 1$ **do**
  for $i = 1$ **to** N_{latent} **do**
    $\hat{z}^i_k \leftarrow H_\psi\left(G_\theta\left(\tilde{c}, v_{t-1}, v_t, v_{t+H}, z_{<k}, z^{<i}_k\right)\right)$
  end for
end for
$\mathcal{L}_{\text{act}} \leftarrow \sum_{k=t}^{t+H-1} \sum_{i=1}^{N_{\text{latent}}} \text{CE}\left(\hat{z}^i_k, z^i_k\right)$ {Action token loss}
$\mathcal{L}_{\text{pretrain}} \leftarrow \mathcal{L}_{\text{img}} + \mathcal{L}_{\text{act}}$ {Total loss}
Update G_θ, E_ϕ, and H_ψ using gradient descent

Hyperparameter	Value
Model Setup	
Video Model	LWM-Chat-1M [7] (initialized)
Tokenization Backbone	VQ-VAE (frozen)
Prompt Tokenizer	BPE tokenizer
Latent Action Vocabulary Size $\|\mathcal{C}\|$	8
Latent Embedding Dim (E_ϕ)	4096
Latent Decoder Type (H_ψ)	MLP
Latent Decoder Layers	1
Latent Decoder Hidden Dim	2048
Training Configuration	
Optimizer	AdamW
Learning Rate	4e-5
Weight Decay	0.0
Optimizer Betas	$(0.9, 0.95)$
Batch Size	512
Total Steps	50,000
Dropout	0.1
Gradient Clipping	1.0
Mixed Precision	`bfloat16`
GPU	8 Nvidia H100 (144 hours)
Prediction Targets	
Prediction Horizon H	14
Image Token Loss	Cross Entropy
Latent Action Loss	Cross Entropy

Table 4: Hyperparameters for multimodal video pretraining to jointly predict future visual state and latent action sequence. The video model is initialized from LWM-Chat-1M and trained jointly with lightweight latent action modules.

D Flow Matching Decoder for Continuous Control

We extend the pretrained video model G_θ with two components for continuous control. Specifically, we introduce an action encoder head E_γ that maps each continuous noisy action x_s into the model's embedding space, and an action decoder head H_η that predicts the flow field used to recover the full action chunk.

The resulting model, denoted g_ω, consists of three key components: the pretrained video model G_θ, the action encoder E_γ, and the flow decoder H_η. During training, we sample a noisy interpolation between a standard Gaussian vector and the ground-truth action chunk, and supervise the predicted flow toward the true actions using visual and task context. The full training procedure is detailed in Algorithm 3. Corresponding neural design choices and hyperparameters are include in Table 5.

Algorithm 3 Flow Matching for Continuous Control

Require: Image history frames (o_{t-1}, o_t),
Require: Task description: c (text string)
Require: Action chunk $a_{t:t+H-1} \in \mathbb{R}^{H\times D}$ {D-dimensional actions}
Require: Pretrained models: VQ-VAE encoder E_{VQ}, video model G_θ
Require: Trainable components: action encoder E_γ, flow decoder H_η

Sample timestep $s \sim \text{Beta}(1.5, 1.0)$
Sample noise $x_0 \sim \mathcal{N}(0, I)$
Compute interpolation: $x_s \leftarrow s \cdot x_0 + (1-s) \cdot a_{t:t+H-1}$
Tokenize input frames: $v_{t-1}, v_t \leftarrow E_{\text{VQ}}(o_{t-1}), E_{\text{VQ}}(o_t)$
Encode text prompt: $\tilde{c} \leftarrow \texttt{Tokenizer}(c)$
Encode noisy actions: $f_s \leftarrow E_\gamma(x_s, s)$
Predict flow field: $\hat{g} \leftarrow H_\eta\left(G_\theta\left(\tilde{c}, v_{t-1}, v_t, f_s\right)\right)$
$\mathcal{L}_{\text{FM}} \leftarrow \|a_{t:t+H-1} - x_0 - (1-s) \cdot \hat{g}\|_2^2$ {Flow matching loss}
Update parameters of E_γ, G_η, and G_θ via gradient descent

Hyperparameter	**Value**
Noisy Action Encoder Head (E_γ)	
Architecture	2-layer MLP
Hidden Dim	4096
Embedding Dim (d_a)	4096
Activation	GELU
Dropout	0.1
Flow Decoder Head (G_η)	
Input Dim	7 (End-Effector Deltas) or 8 (Absolute Joint States)
Architecture	Single linear projection
Flow Matching Setup	
Interpolation Timestep s	Beta$(1.5,\ 1.0)$
Noise Distribution $\mathbf{x}_0$	Standard normal $\mathcal{N}(0, I)$
Prediction Horizon H	14
Integration Method (Inference)	Forward Euler, $N = 10$ steps
Training Configuration (follows Table 4)	
Total Steps	12000 (SIMPLER)

Table 5: Architecture and hyperparameters used for continuous control. Training settings (optimizer, schedule, etc.) match those used during pretraining.

E Simulation Benchmarks

E.1 SIMPLER Benchmark

Following prior latent action works [12, 13], we benchmark ViPRA in SIMPLER [23], an open-source suite for evaluating generalist manipulation policies. We evaluate on four Bridge tasks with a 7-DoF WidowX arm, a benchmark designed to test generalization across diverse manipulation goals. Since SIMPLER does not provide finetuning data, we collect 100 diverse multi-task trajectories using a pretrained VLA model [12] to adapt policies before evaluation. The tasks are as follows:

- **Spoon2Cloth**: The instruction is `put the spoon on the towel`. The spoon is placed on a vertex of a 15 cm square on the tabletop, and the towel on another vertex. The spoon's orientation alternates between horizontal and vertical, requiring the robot to re-orient its gripper. This task evaluates both grasp selection and orientation adjustment.
- **Carrot2Plate**: The instruction is `put carrot on plate`. Same setup as Spoon2Cloth, but with a carrot and a plate. While similar in layout, this introduces a different geometry and surface, requiring adaptation in grasping and placement.
- **StackG2Y**: The instruction is `stack the green block on the yellow block`. A green block is placed on a vertex of a tabletop square (10 cm and 20 cm edges) and a yellow block on another. Success requires precise alignment and careful release, making it a fine-grained manipulation task that stresses stability and accuracy.
- **Eggplant2Bask**: The instruction is `put eggplant into yellow basket`. An eggplant is dropped into the right basin of a sink and a yellow basket in the left basin. The eggplant is randomized in pose but ensured to be graspable. This task evaluates robustness to shape variability and placement under uncertainty, as the object must be reliably picked and transferred across workspace regions.

We evaluate performance using two metrics: **success rate** and **partial success rate (grasp rate)**. Success rate measures whether the full task goal is completed (e.g., spoon placed on towel, block stacked without falling, eggplant deposited into basket). Grasp rate captures whether the robot is at least able to establish a successful grasp on the object, even if the subsequent placement or stacking is not achieved. This distinction is important: grasping reflects a fundamental capability for initiating manipulation, while successful completion requires the integration of grasping with precise transport and placement. Together, these metrics provide a more comprehensive view of policy competence, distinguishing between failures due to perception/grasping versus those arising from downstream control and placement.

E.2 SIMPLER Results

We report both end-to-end *success rate* and *grasp rate* in Table 1. Across **discrete actions** setting, **ViPRA-AR** attains the best average success (69.8%), exceeding LAPA (53.1%), VPT (51.0%) and OpenVLA (38.6%). It leads on precision-heavy `StackG2Y` (66.7% vs. 54.2% Scratch-AR, 45.8% VPT) and `Carrot2Plate` (62.5%), and remains competitive on `Spoon2Cloth` (66.7%, near VPT's 70.8%). On `Eggplant2Bask`, ViPRA-AR (83.3%) significantly outperforms other methods, demonstrating strong transport and placement.

In the **continuous setting**, **ViPRA-FM** achieves the highest average success (62.5%), outperforming Scratch-FM (41.7%), π_0 (27.1%), and UniVLA (42.7%). It is the strongest continuous model on `StackG2Y` (54.2%), `Carrot2Plate` (50.0%) and `Spoon2Cloth` (66.7%) while remaining competitive (79.2%) with π_0 (83.3%) on `Eggplant2Bask`. UniPI frequently generates action sequences that diverge from the given instruction in longer-horizon settings, while VPT provides only limited gains, indicating that IDM-derived pseudo-labels are sensitive to environment shifts. In contrast, ViPRA's joint use of latent action prediction and future state modeling yields stronger cross-environment transfer and more reliable task execution.

ViPRA converts grasps into task completion more reliably. On `StackG2Y`, OpenVLA achieves 70.8% grasp but only 25.0% success (a 45.8 pt gap), indicating post-grasp placement failures. ViPRA-AR maintains 66.7% grasp and 66.7% success (0 pt gap), and ViPRA-FM 62.5% grasp vs. 54.2% success (8.3 pt gap), evidencing stable transport and release. On `Eggplant2Bask`, OpenVLA's 91.7% grasp

Method	Success Rate
UniPI [35]	0.00
OpenVLA [18]	0.54
π_0-FAST [20]	0.60
π_0 [20]	0.85
UniVLA (human) [13]	0.79
UniVLA (all) [13]	0.92
ViPRA	0.79

Table 6: Success rates on LIBERO-10 benchmark.

falls to 58.3% success (33.4 pt drop), whereas ViPRA-AR (100% $\rightarrow$ 83.3%) and ViPRA-FM (91.7% $\rightarrow$ 79.2%) show markedly smaller drops, consistent with smoother post-grasp control and accurate instruction following.

E.3 LIBERO Long Benchmark

We also evaluate on LIBERO Long (a.k.a LIBERO 10) [80], the most challenging subset of the LIBERO simulation benchmark. Unlike the Spatial, Object, or Goal subsets, LIBERO Long focuses on long-horizon manipulation tasks that require sequencing multiple sub-goals with heterogeneous objects, layouts, and task dependencies. This setting stresses robustness and temporal compositionality, since errors can accumulate across long horizons.

The evaluation consists of a suite of 10 long-horizon tasks, each paired with a natural language goal description. For example, one of the task instructions includes `"put the white mug on the plate and put the chocolate pudding to the right of the plate"`, requiring reasoning over both symbolic relations (object identities, spatial references) and low-level control. For each task, the environment is initialized with objects placed in varied locations, increasing the difficulty of generalization.

Each task is evaluated across 10 runs with 5 different random seeds, and results are reported as the average reward over all 10 tasks (500 episodes in total). This protocol provides a stringent test of both semantic grounding and long-horizon policy execution, making LIBERO Long a valuable complement to SIMPLER's shorter-horizon manipulation tasks.

E.4 LIBERO Long Results

On LIBERO-10, which emphasizes long-horizon, multi-stage manipulation, ViPRA achieves a 79% success rate. This is substantially higher than OpenVLA (54%) and π_0-FAST (60%), and close to UniVLA (90%), which is specifically optimized for LIBERO. These results demonstrate that ViPRA's motion-centric latent pretraining transfers effectively to simulated long-horizon tasks, outperforming methods trained primarily with labeled actions or direct policy supervision.

We observe that ViPRA performs reliably on coarse manipulations (e.g., cups, bowls, books), which are easy to grasp, but struggles with precision grasps such as cylindrical cans that require diameter-aligned control. We attribute this to *delta-EEF drift*: since LIBERO's action space is delta end-effector, small prediction biases can accumulate over time, leading to imprecise grasps in the absence of absolute cues to re-anchor the trajectory. For instance, π_0 mitigates this issue by conditioning on proprioceptive state history and wrist-camera inputs. Despite lacking such additional signals, ViPRA surpasses OpenVLA under the same sensing setup (image-only, delta-EEF), underscoring the benefits of motion-centric latent pretraining for long-horizon manipulation.

F Action Output Analysis

We provide a more in-depth analysis of the action outputs of various policies introduced in Section 5.5, highlighting their differences in smoothness, consistency, and suitability for real world deployment. Policies differ in their action representations and control spaces:

- **Absolute Joint Space.** ViPRA-FM and LAPA [12] output full 7D joint positions (Franka), directly supervised in joint space.
- **Delta End-Effector Space.** OpenVLA [18], π_0 [20], and operate in 7D Cartesian delta commands (position, Euler rotations, gripper), decoded from visual inputs.
- **Continuous vs Discrete.** ViPRA-FM and π_0 [20] predict continuous actions via a flow matching decoder, whereas LAPA [12] and OpenVLA [18] use quantized logits over discretized action bins.

To better understand the behavioral differences between discrete and continuous policies, we analyze the predicted action trajectories across different models during closed-loop visual rollout on real robot observations. We evaluate policies by loading their finetuned checkpoints into our inference pipeline and simulating replay on the training trajectories from the finetuning dataset. This allows us to visualize their motor command trends without introducing new generalization factors. In particular, we compare ViPRA-FM (ours), LAPA [12], OpenVLA [18], and π_0 [20].

LAPA [12] and OpenVLA [18] rely on a discretization scheme in which each dimension of the robot's action space is uniformly quantized into 255 bins using equal-sized quantiles over the training distribution. That is, for each joint or end-effector dimension, bin boundaries are chosen so that each bin contains roughly the same number of training points. This quantile-based discretization ensures equal data coverage across bins but introduces two key limitations in how actions are represented and learned:

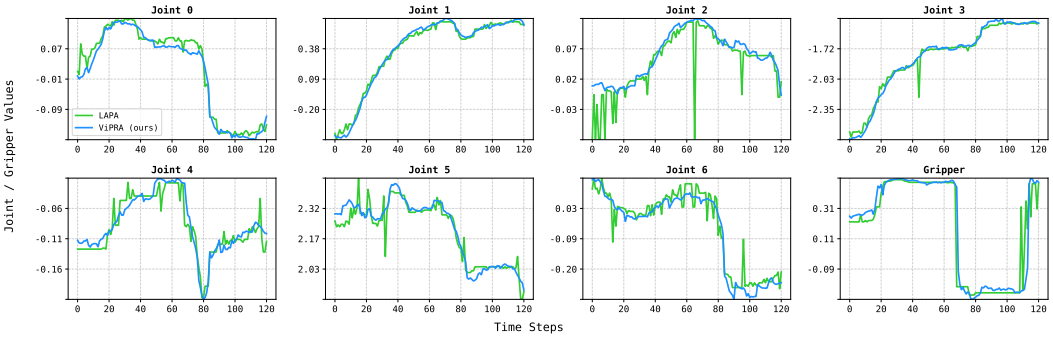
1. **Contact-Sensitive Flipping:** At test time, small perturbations in the input (e.g., due to occlusions or slight viewpoint drift) may cause the model to flip from one bin to another near the quantile boundary–especially at contact points. Since adjacent bins can correspond to different action magnitudes, these minor visual shifts can lead to abrupt discontinuities in motor output.
2. **Loss Granularity:** The cross-entropy loss used for training treats each action bin as a distinct class label. As a result, all incorrect predictions are penalized equally, regardless of how close they are to the ground-truth bin. For example, predicting bin 127 instead of 128 incurs the same loss as predicting bin 0. This is fundamentally at odds with the structure of continuous action spaces, where the cost of an error should scale with its magnitude.

We hypothesize that the combination of bin boundary flipping and non-metric loss leads to the spiky or erratic behavior seen in discrete action models, particularly around moments of contact or high-frequency motion. These effects are amplified in high-dimensional control settings, where discretization artifacts can arise independently in each action dimension–compounding into visibly unstable or jerky behaviors across the full joint trajectory.

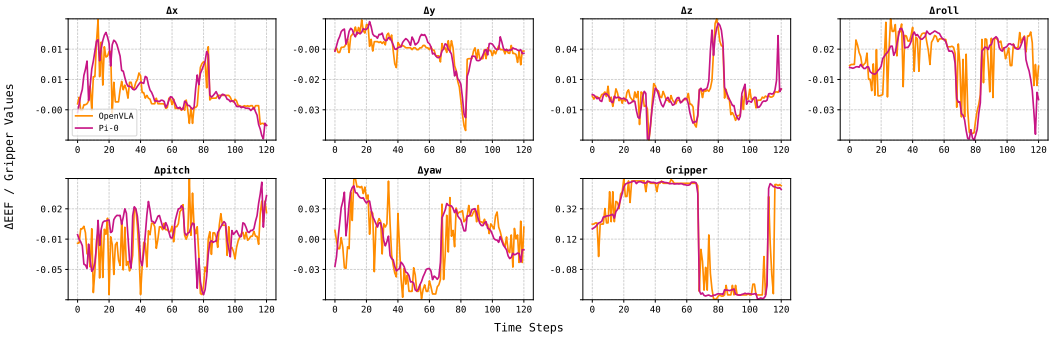
By contrast, continuous policies such as ViPRA-FM and π_0 [20] operate directly in $\mathbb{R}^D$ using flow matching. These losses naturally reflect the structure of the action space–penalizing predictions in proportion to how far they deviate from the ground truth. As a result, the output trajectories tend to be smoother, better aligned with demonstrations, and more robust to perceptual jitter.

We note that it may be possible to mitigate some of the above issues by increasing the number of bins or by using non-uniform binning schemes (e.g., higher resolution in frequently visited regions). However, these approaches increase model complexity and still inherit the fundamental limitation of using classification loss in a regression setting. Continuous decoders trained with distance-aware objectives offer a more natural and principled solution for low-level control.

To gain deeper insight into how different action representations influence control behavior, we examine the temporal structure of predicted actions across several rollout trajectories. We organize the analysis by control space–absolute joint angles vs. delta end-effector motions–and visualize per-dimension action trends across time in Figure 8.



(a) **Absolute joint space.** Predicted 7D joint positions over time for ViPRA-FM (blue) and LAPA [12] (green). ViPRA-FM produces smooth, continuous trajectories, while LAPA [12] exhibits local discontinuities and random spikes–often around contact events–despite tracking the overall trend. In real world deployment, such discontinuities *triggered Franka's emergency brake mechanism due to abrupt torque jumps.*



(b) **Delta end-effector space.** Predicted 7D delta actions (position, rotation, gripper) for π_0 [20] (magenta) and OpenVLA [18] (orange). Although delta control provides structured low-level modulation, OpenVLA exhibits sharp fluctuations due to discretized output. Notably, the gripper signal shows large, momentary switches during contact events–resulting in failed grasps or premature object drops. In contrast, π_0 maintains stable gripper behavior during fine manipulation.

Figure 8: Visualization of predicted actions across different control spaces. Discrete policies often produce sharp discontinuities due to binning artifacts and classification loss, whereas continuous policies exhibit smoother, dynamics-consistent behavior.

These visualizations support our hypothesis: *discrete policies, trained with cross-entropy over fixed bins, tend to produce abrupt transitions around perceptually sensitive regions*–especially near bin boundaries or occlusions. This manifests as random spikes, high-frequency jitter, or contact-time instability, all of which can destabilize robot behavior in deployment.

In contrast, *continuous policies like* ViPRA-FM *and* π_0*, trained with flow matching losses, yield consistently smooth, physically plausible actions that better reflect real world constraints.* The ability to interpolate naturally between states–not just classify them–proves critical for robust closed-loop performance in contact-rich manipulation.

G Real World Experiments: Setup, Challenges, and Observations

To complement our real world results in Section 5.4, we provide additional details on our hardware setup, task design, and policy behavior under realistic sensing and control constraints. We also analyze generalization to unseen objects, retry behavior, and how chunked continuous actions support efficient closed-loop control.

G.1 Hardware and Data Collection Setup

All experiments are conducted on a real world robotic platform with two 7-DOF Franka Emika Panda arms. The workspace is observed by a single front-mounted ZED stereo camera. There are no wrist-mounted or side-view cameras, so all perception is monocular and from a fixed third-person viewpoint. We use the GELLO teleoperation system [81] to collect human demonstrations at 15Hz. Demonstrations are collected directly in task-relevant environments, with each policy trained using only a single camera view.

Our decision to use image history as part of the observation is motivated by the inherently temporal nature of the video model architecture, as well as the absence of auxiliary views. Stacking observations over time allows the model to internally infer dynamics and compensate for occlusions or ambiguous single-frame cues.

G.2 Task Descriptions and Challenges

We evaluate policies on three real world single-arm tasks, each with unique control and perception challenges (Figure 9):

1. **Cover-Object:** The robot must pick up a piece of cloth and drape it over a specified object. This task is challenging due to the deformable nature of cloth, which requires reliable grasping from the table surface. Slight changes in cloth configuration or object geometry can affect dynamics drastically. Generalization requires reasoning over unseen cloth textures and novel target objects.
2. **Pick-Place:** The robot must pick up a named object (e.g., sponge, bowl, duck) and place it on a destination surface (plate or board). Object shapes vary significantly, leading to different grasp affordances. Grasping a wide bowl vs. a narrow-handled cup requires distinct motor strategies. The task is highly multimodal–there are multiple correct ways to perform the task, depending on object shape, pose, and placement surface.
3. **Stack-Cups:** The robot must follow language instructions to stack a cup of color1 onto a cup of color2. Success requires grounding object properties and executing precise stacking. Evaluation setups include unseen cup types, color shades, and geometries to test language understanding and spatial generalization.

G.3 Generalization to Novel Objects

A core goal of our real world evaluation is to assess how well the policy generalizes to unseen object instances and configurations not encountered during training. We design test-time setups that introduce meaningful variation across tasks:

- **Cover-Object:** Test scenarios include cloths of varying texture, size, and stiffness, as well as new target objects such as jars, boxes, and toys. These variations require the policy to generalize grasp strategies and adapt to deformable material dynamics.
- **Pick-Place:** We evaluate on previously unseen objects with diverse geometries and affordances (e.g., bowls, mugs, fruits), and destination surfaces of varying size and texture. The task requires flexible grasping and reliable placement across a range of object shapes and destination surfaces.
- **Stack-Cups:** Evaluation includes new cup types with unseen shapes, rim sizes, and fine-grained color variations. The policy must generalize language grounding to new color references and execute precise stacking across novel physical configurations.

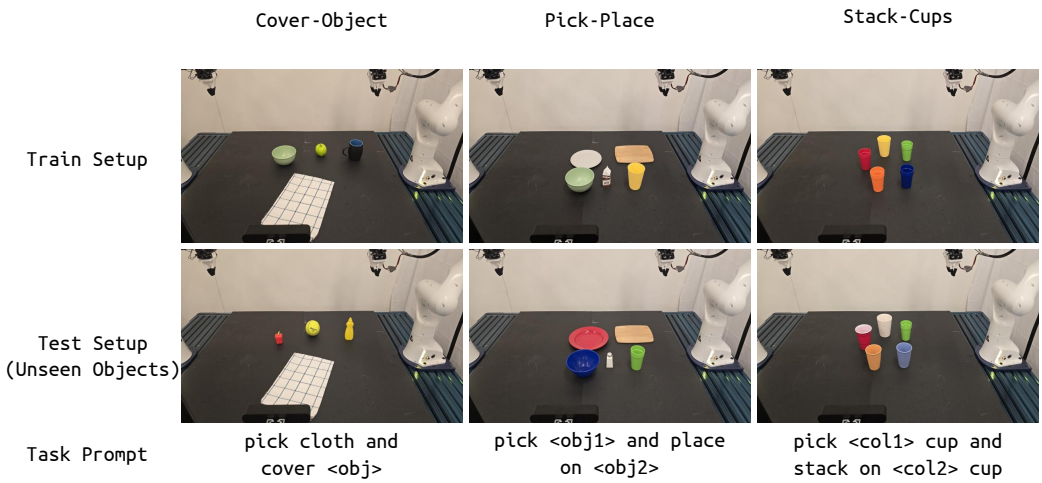


Figure 9: **Task Setup Overview.** (Top row) Training environments for each of the three single-arm manipulation tasks: `Cover-Object`, `Pick-Place`, and `Stack-Cups`. (Bottom row) Evaluation environments featuring novel objects, textures, or placements not seen during training. Note the variety in cloth shape, object geometry, plate type, and cup color/size combinations.

Despite these shifts, *our method consistently exhibits robust generalization across all tasks*. We attribute this to the combination of latent dynamics pretraining, language-conditioned perception, and a unified architecture that integrates semantic, spatial, and temporal cues. Pretraining on diverse unlabeled videos teaches the model general priors about object motion and interaction. Conditioning on task instructions guides object selection and interpretation even in ambiguous or unfamiliar contexts. Finally, the architectural design ensures that learned representations capture not just appearance, but how objects behave across time, enabling transfer to new instances that were not explicitly seen during supervised finetuning.

G.4 Retrying Behavior Enabled by Temporal Pretraining

Our method consistently *exhibits robust retry behavior*: when an initial grasp attempt fails, due to occlusion, misalignment, or object shift, the policy often reattempts until successful. This is especially evident in `Cover-Object`, where the robot frequently retries grasping if the cloth slips, and in `Pick-Place`, where wide or irregularly shaped objects like bowls may require multiple grasp attempts from different angles.

We attribute this robustness to our temporal pretraining objective. By learning to predict future video frames and latent actions over multiple steps, the model develops a sense of longer-horizon dynamics and recoverability. Rather than depending on single-step feedback, it implicitly plans through extended temporal context–enabling it to course, correct and persist through partial failures.

G.5 Action Chunking and Inference Efficiency

ViPRA produces continuous actions using a chunked flow matching decoder, generating sequences of 14 actions per inference step. At test time, we cap control frequency by evaluating two rollout strategies: **7/14 rollout**, where the first 7 actions of each chunk are executed before re-planning, and **14/14 rollout**, where all 14 actions are executed before the next inference. The former corresponds to an effective closed-loop update rate of $\sim$3.5 Hz, while the latter doubles this to 7 Hz. Because predicted action trajectories are smooth and temporally coherent, ViPRA remains stable even under open-loop execution within each chunk. This property is particularly beneficial for contact-rich phases that demand reactive yet jitter-free behavior.

KV caching for fast inference We further optimize inference with key-value (KV) caching. Language and image attention states are cached once and reused across flow matching Euler steps, so only action tokens are recomputed during integration. This reduces redundant computation, enabling the entire 14-step chunk to be produced in 510 ms ($\sim$1.95 Hz), which corresponds to a robot-side

control frequency of up to 22 Hz. Our setup can stably support control rates approaching 20 Hz, to our knowledge matched only by one other 7B-parameter model [22].

Comparison with baselines. Table 7 summarizes model sizes, action rollout lengths, and inference times. Unlike prior approaches that also use a 7B model (e.g., LAPA and OpenVLA) and operate at $\sim$200 ms per step but predict only single actions, ViPRA amortizes inference across long, smooth action chunks, enabling high frequency reactive control.

Method	**Model Size**	**Action Steps**	**Inference Time (ms)**
LAPA [12]	7B	1	220
OpenVLA [18]	7B	1	190
π_0 [20]	3.3B	16	90
UniPI [35]	–	16	24000
UVA [40]	0.5B	16	230
ViPRA (ViPRA-FM)	7B	14	510

Table 7: Inference speed comparison across models. ViPRA achieves high effective control frequencies by amortizing computation over action chunks.

H ViPRA-FM on Challenging Bimanual Tasks

Bimanual manipulation introduces significant complexity beyond single-arm control. The combined action space spans 14 degrees of freedom, and inter-arm coordination requires precise spatial alignment, collision avoidance, and timing consistency. The solution space is also highly multimodal–there are many valid ways to execute a task depending on object geometry, initial configurations, and movement variability. These challenges make bimanual tasks a strong test of a policy's ability to generalize and coordinate under real world constraints.

H.1 Bimanual Setup

We test our framework using both arms of the Franka Panda robot. While only the right arm performs active grasping, both arms are controlled jointly using a single policy conditioned on shared language instructions. The system receives monocular observations from a front-mounted ZED camera and generates chunked continuous actions for both arms in a synchronized control loop.

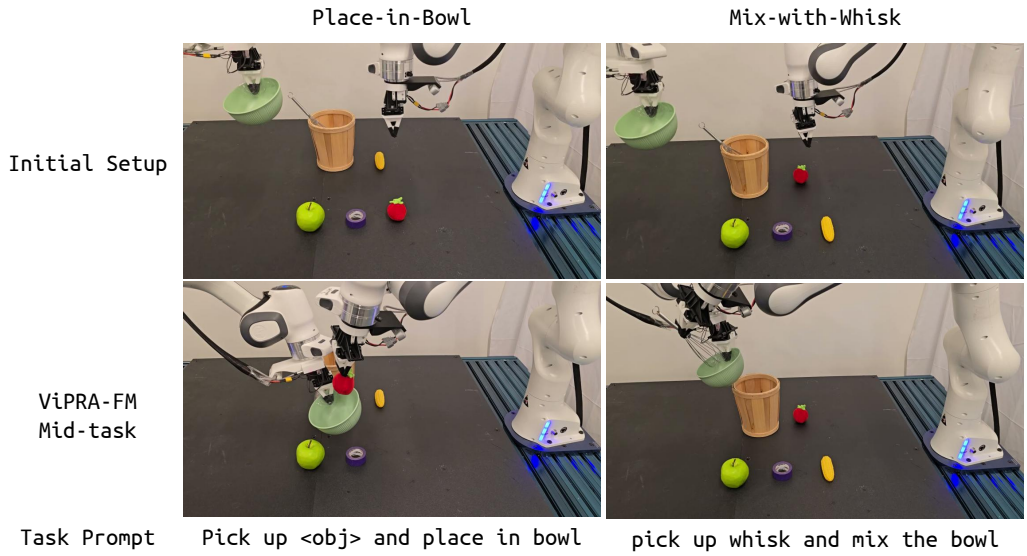


Figure 10: **Bimanual task execution by ViPRA-FM.** (Top row) Initial setup for the two tasks: placing a tomato into a bowl and mixing with a whisk. (Bottom row) Mid-execution rollout of ViPRA-FM: the right arm transports the tomato toward the bowl held by the left arm (left), and mixes the contents using the whisk while the left arm maintains bowl stability (right). These examples highlight coordinated two-arm control and fluent execution of tool- and object-handling behaviors.

We evaluate two bimanual tasks of increasing complexity:

(1) Place-in-Bowl: The right arm must grasp a target object (e.g., a fruit or kitchen item) and place it into a bowl held by the left arm. Success requires fine-grained spatial alignment above the bowl, smooth object transfer, and collision-free approach and retreat trajectories in close proximity to the support arm.

(2) Mix-with-Whisk: The right arm retrieves a whisk from a nearby basket, mixes the contents of the bowl, and returns the whisk to its original location. This task involves tool use, curved and sustained motion, and close-proximity coordination with the left arm, which dynamically maintains the bowl pose throughout the sequence.

These tasks pose significant challenges for real world bimanual coordination. Both arms must operate in close proximity, requiring precise spatial alignment to avoid collisions–especially during approach and retreat phases. With only a single fixed camera and no wrist-mounted sensors, the policy must infer depth and object interactions purely from visual input. Timing mismatches or calibration drift between the arms can further compound errors, making successful execution sensitive to both perception and control stability.

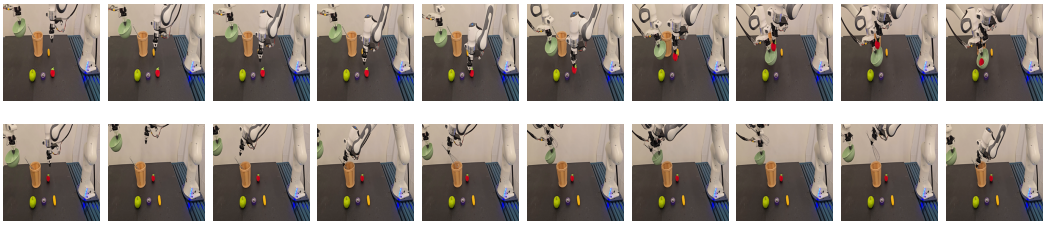


Figure 11: **ViPRA-FM rollouts in real world bimanual tasks.** Top: `Place-in-Bowl` - the robot picks up a tomato and places it into a bowl held by the left arm. Bottom: `Mix-with-Whisk` - the robot retrieves a whisk, stirs the bowl contents, and returns the tool. Each sequence shows 10 evenly spaced frames sampled from real world executions.

H.2 Bimanual Results

ViPRA-FM is deployed using 14-step action chunks, executed at 7Hz control frequency. This high-frequency chunked control allows the policy to maintain smooth, temporally coherent trajectories while remaining responsive to changing visual inputs. The model also receives short history windows as input, which helps stabilize motion during contact-heavy transitions and multi-step interactions.

In `Place-in-Bowl`, the robot completes 10 out of 18 trials. Failures were primarily due to unsuccessful grasps caused by the limited span and compliance of our custom 3D-printed gripper, not the bimanual coordination itself. In all successful grasps, the object was consistently placed into the bowl without collision or instability. This suggests that the policy reliably handles the spatial reasoning and coordination demands of the task, with grasp robustness being the primary bottleneck, a limitation that could be mitigated with a more capable gripper design.

In `Mix-with-Whisk`, the robot completes 8 out of 12 trials. The task involves sustained, curved motion in close proximity to the left arm, requiring continuous spatial alignment between the whisk and bowl. The policy leverages temporal history to stay anchored to the mixing target and uses its chunked control output to produce smooth stirring behavior. The whisk's small, symmetric handle makes it easier to grasp, allowing the policy to focus on trajectory accuracy and contact stability throughout the sequence.

Together, these results demonstrate that ViPRA-FM is capable of executing complex bimanual tasks using a single vision-conditioned policy and continuous action generation. Additional results, comparisons, and rollout videos will be shared on our project website. `https://vipra-project.github.io`.

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: We show that video prediction coupled with latent action grounding provide strong dynamics aware representations, and concretely establish this through performance gains on simulated and real world robot tasks.

   Guidelines:

   - The answer NA 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 NA 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: We include a limitation section in our paper where we elaborate on the shortcomings of our approach and how it can be improved.

   Guidelines:

   - The answer NA 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: [NA]

Justification: We don't have any theoretical results, we make use of existing models and engineer on top of that to address the issue of scaling data for robot learning.

Guidelines:

- The answer NA 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: We provide detailed explanation of our approach and parameters used to setup the framework in the paper. We will also release the code, dataset and checkpoints once the review period is done.

Guidelines:

- The answer NA 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: We will also release the code, dataset and checkpoints once the review period is done.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.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://nips.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, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We have provided training details and hyperparameter for every aspect of our method.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [No]

   Justification: Since our approach involves multiple real world evaluations with large scale transformer models, we simply did not have the compute to report error bars.

   Guidelines:

   - The answer NA 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: Yes we mention the number of GPUs and the time we run for for our method.

   Guidelines:

   - The answer NA 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: We have reviewed NeuRIPS guidelines and make utmost effort to follow it.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: Yes we have included a section discussing broader impact of our approach.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA 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., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: We don't have any sensitive models that could be misused.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: We have cited all datasets and models that we build upon.

    Guidelines:

    - The answer NA 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: [Yes]

    Justification: We mention details our methods and datasets, and will opensource our approach once the review period is up.

    Guidelines:

    - The answer NA 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: [NA]

    Justification: We do not do any research with human subjects.

    Guidelines:

    - The answer NA 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: [NA] .

    Justification: paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA 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 rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: LLM was not used in any core research.

    Guidelines:

    - The answer NA 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 (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.