ViPRA: Video Prediction for Robot Actions

Anonymous Author(s)
Affiliation
Address
email

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 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 14% improvement across real world manipulation tasks. We will release models and code at https://vipra-robot.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 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.

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

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

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

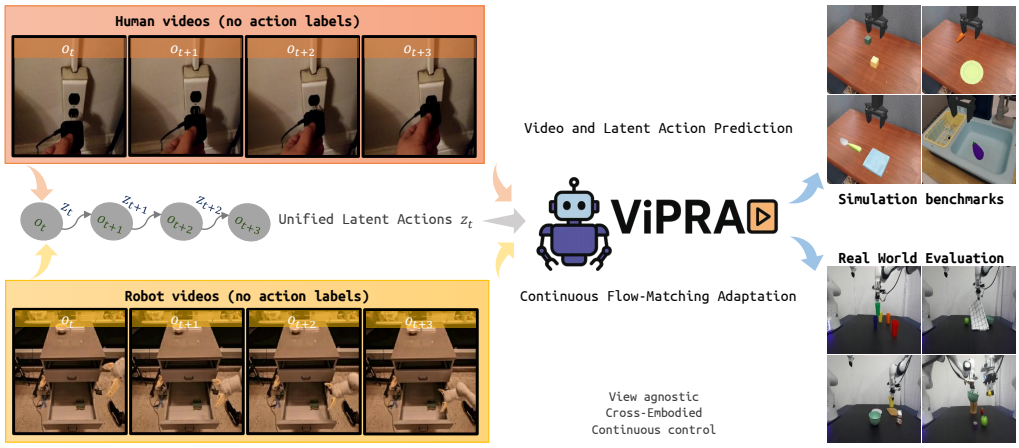


Figure 1: We present ViPRA, a framework to learn generalist robot policies from large-scale human and robot videos without action labels. It extracts motion-centric latent action sequences, pretrains a video-language model to jointly predict future visual observations and latent action chunks, and finetunes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

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 ($3 - 4$ 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 Fig. 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 Fig. 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.

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 for smooth, high-frequency continuous control.
(iv) Demonstrate empirical gains of 16% on the SIMPLER benchmark [22] and 14% on real world tasks over the strongest prior continuous control baselines.

2 Related Works

Vision–Language–Action Models Recent VLAs [17–21] extend vision–language models (VLMs) [23–27] by finetuning on large-scale, action-labeled robot datasets [6], enabling robots to interpret language commands and execute corresponding actions. Other works explore auxiliary objectives such as visual trace prediction [28], chain-of-thought reasoning [29], or conversational instruction tuning with self-supervised augmentations [30]. However, these approaches still depend heavily on labeled action data, limiting scalability due to the cost of data collection. Our method reduces this reliance by leveraging unlabeled videos during pretraining, improving scalability. Whereas

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

most VLAs focus on grounding language in visual semantics, they often lack mechanisms for modeling physical dynamics or incorporating temporal structure into action generation. We address this gap by employing a video prediction model [7] that predicts both future visual states and multi-step latent actions during pretraining. This is crucial for modeling high-frequency motions and provides robust and smooth priors for downstream policy.

Robot Learning from Videos Videos offer a scalable source of information about object dynamics, task structure, and human behavior, making them an attractive modality for robot learning. Visual planning methods [31–37] 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. Policy supervision approaches [37, 38] use video models as supervision or reward sources to train policies, though evaluations have so far focused on simulation or relatively constrained real world tasks.

Parallel work has explored joint training for video generation and action prediction [39, 40], with [39] also introducing decoupled action decoding to mitigate inference overhead. While such methods can incorporate actionless videos, they have primarily been tested on smaller-scale datasets or simulations, and their scalability to leverage large, internet-scale passive videos remains unclear.

Other efforts leverage human videos to pretrain visual representations for downstream visuomotor control [41–44], or extract intermediate cues such as affordances [45–49], interactions [50], or visual traces [51–54] from unlabeled videos to guide policy learning. These methods depend on structured visual priors or explicit cue extraction, which may limit their scalability.

Latent Action Spaces Latent action representations improve data efficiency by enabling learning from action-free videos via self-supervised learning [55–59]. Recent methods impose discrete information bottlenecks with VQ-style encoders [60] and predict these tokens during policy learning [13–15, 61–63], achieving strong real world results through imitation. Some train an inverse dynamics model (IDM) on limited labeled demonstrations before applying it to unlabeled video, improving the mapping from latent to real actions [35, 64], while others treat latent actions as an abstract embodiment and jointly train policies with IDM predictions across embodiments [65].

Another line of work uses these abstractions to build world simulators [66, 67] or plan in latent spaces [68–74], capturing physical dynamics but often struggling to generalize to novel settings due to limited semantic grounding. Video-language models can provide such multimodal grounding [31, 33, 34, 36, 40], yet existing approaches are typically computationally heavy and slow at inference.

Our method addresses these gaps by learning 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 fine-tuning, we amortize inference and achieve smooth, high-frequency control.

3 Background

Behavior Cloning (BC) is the standard supervised learning paradigm in robotics. Given a dataset $\mathcal{D} = \{(o_t, a_t)\}$ of demonstrations , the policy π_ω is trained to minimize:

$$\min_{\omega} \; \mathbb{E}_{(o_t,a_t)\sim\mathcal{D}} \, \|\pi_\omega(a_t \mid o_t) - a_t\|_2^2 \, .$$

Recent advances have extended this framework using high-capacity architectures such as diffusion models [75, 76] and VLAs [17–21] trained on large-scale robot data.

Flow Matching [16] is a technique for learning continuous normalizing flows via supervised vector field prediction. The goal is to train a neural flow field g_θ that transports samples from a source distribution x_0 (e.g., Gaussian noise) to a target distribution x_1 (e.g., actions) via linear interpolation:

$$\frac{\partial}{\partial s} u_s = g_\theta(u_s, s \mid y), \quad \text{where} \quad u_s = s x_0 + (1-s) x_1 .$$

Here, $s \in [0, 1]$ is a continuous time variable, and y denotes a general conditioning input. In the context of robotics, y can include the image history $o_{t-K+1:t}$, language command c, and other contextual inputs relevant to predicting x_1, the correct action trajectory $a_{t:t+H-1}$.

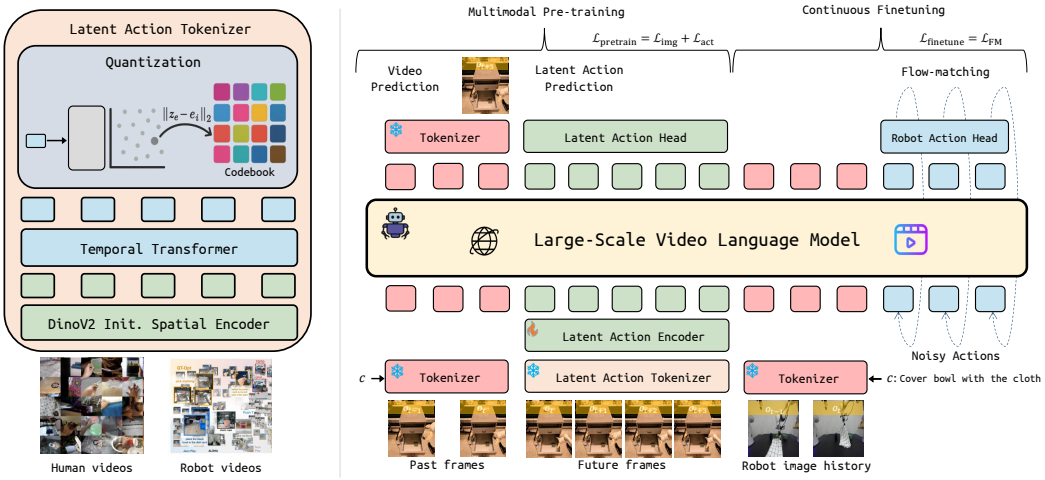


Figure 2: **Overview of our framework:** Given human and robot videos without action labels, we represent scene dynamics using a discrete latent action space z_t. A neural quantization bottleneck extracts latent actions $z_{t:t+H-1}$ from image sequences $o_{t:t+H}$, and a video-language model is pretrained to jointly predict: (i) future observations o_{t+H} and (ii) latent actions $z_{t:t+H-1}$ from past frames (o_{t-1}, o_t) and a task description c. The model is then finetuned on a small set of action-labeled robot demonstrations using flow matching to predict continuous actions $a_{t:t+H-1}$.

The flow-matching objective minimizes the difference between the actual target displacement and the flow predicted by the model:

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

At inference time, actions are generated by solving the ODE using numerical integration from $s = 0$ to $s = 1$. The most common approach is forward Euler integration:

$$u_{s+\Delta s} = u_s + \Delta s \cdot g_\theta(u_s, s \mid y), \quad s \in [0, 1].$$

This produces the final action estimate $x_1 \approx u_1$. While Euler is efficient and commonly used in practice [20], more accurate solvers such as Heun's method or Runge–Kutta (RK2) [77, 78] can be used to improve stability in high-dimensional control tasks. Flow-matching-based policies have shown strong empirical performance in robotics, offering smoother and more precise control compared to direct action prediction, especially for temporally extended or fine-grained manipulation tasks [15, 20].

4 Video Prediction for Robot Actions

A generalist robotic agent must combine precise, dexterous control with environment-agnostic high-level intelligence. We argue that video generation models are well-suited to this goal, as their future-state prediction jointly captures low-level physical detail and rich semantic context. Realizing these benefits requires more than off-the-shelf video models: one must select and scale appropriate data, adapt the architecture to expose motion-centric signals, and design training and inference pipelines that are stable and efficient for real world use. To this end, we present ViPRA–a hierarchical, end-to-end action-learning framework that (i) learns fine motion-centric latent actions from large-scale, action-free human and robot videos, (ii) uses multimodal pretraining to couple semantic grounding with temporal dynamics, and (iii) finetunes a continuous, chunk-level decoder that grounds the policy in real world robot physics while enabling smooth, high-frequency 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. The objective is to learn discrete latent tokens that capture how the environment changes in response to an implicit action.

Each training example is a length-$(L+1)$ observation sequence $o_{0:L}$ sampled from a video. We condition on the full sequence and train an inverse dynamics encoder $I_\beta(z_t \mid o_{0:L})$ to predict a

quantized latent token $z_t = [z_t^1, z_t^2, \ldots, z_t^{N_{\text{latent}}}]$ summarizing the transition at time t. This non-causal design allows z_t to incorporate both past and future context, making it sensitive to local motion intent, e.g., distinguishing a pickup from a putdown based on surrounding frames. Providing the encoder with the entire clip reduces reconstruction ambiguity and forces z_t to encode the minimal but sufficient information to explain the local transition. To enforce compactness, we apply a discrete bottleneck by quantizing each z_t^i into a small shared codebook $\mathcal{C}$ of size $|\mathcal{C}| = 8$.

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 latents $z_{0:t}$. The model is optimized using pixel-level L_1 loss $\mathcal{L}_{\text{rec}}$, perceptual loss $\mathcal{L}_{\text{LPIPS}}$ [79], and an optical flow consistency loss $\mathcal{L}_{\text{flow}}$ to encourage physically plausible motion:

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

where $\text{OF}(a, b)$ denotes optical flow between frames a and b computed via RAFT [80]. This encourages predicted frames to exhibit motion patterns consistent with the ground truth, supporting temporally coherent dynamics.

The total loss is:

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

with the flow loss activated only after α_{flow} warm-up steps to avoid instability from poor early reconstructions.

Architecture: The inverse dynamics encoder maps each frame o_t into a feature vector using a DINOv2 [81] backbone, followed by spatio-temporal attention layers so that each feature incorporates full-clip context. These features are then vector-quantized with NSVQ [82] into N_{latent} discrete codes from a shared codebook, yielding latent actions z_t. A spatio-temporal transformer decoder attends jointly to $z_{0:t}, , o_{0:t}$ and reconstructs o_{t+1}. In this way, the latents $z_{0:t}$ serve as a compact, context-aware summary of the actions driving the observed transition.

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.

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 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 z_t^{i+1} from the transformer hidden state at 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.

We *jointly* predict future visual tokens and latent actions, unifying dynamic scene understanding and abstract control representation in a temporally coherent latent space. Given the most recent observations (o_{t-1}, o_t) and a task description c, the model predicts a future frame o_{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 o_{t+H}. This multi-step horizon encourages meaningful and distinct scene changes, providing robust conditioning for downstream action inference.

During training, we apply *teacher forcing* to both visual and action predictions. The ground-truth future frame o_{t+H} is encoded into discrete video tokens using LWM's VQ-VAE encoder, which serve as supervision targets for the visual decoder. Simultaneously, the ground-truth latent action sequence $z_{t:t+H-1}$ is fed to the latent action decoder H_ψ one token at a time. Each token is predicted autoregressively with a cross-entropy loss over the codebook $\mathcal{C}$, conditioned on previous latent tokens, image observations, and the task context.

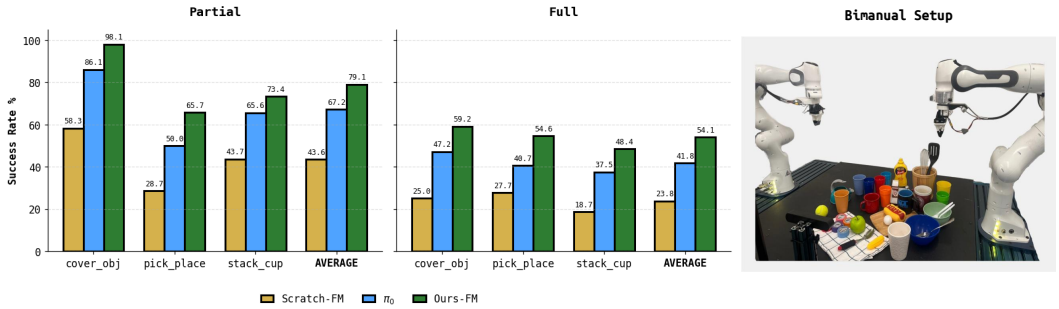


Figure 3: Real world performance on three manipulation tasks, reported as full and partial success rates. ViPRA-FM significantly outperforms baselines.

	Discrete Actions					Continuous Actions			
Task	**Scratch-AR**	**VPT** [12, 35]	**OpenVLA** [18]	**LAPA** [12]	**ViPRA-AR**	**Scratch-FM**	**UniPI** [12, 34]	π_0 [20]	**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	**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	**71.9**

Table 1: We report success rates and grasp rates on SIMPLER benchmark suite.

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.

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. At each training step, 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 tokens of (o_{t-1}, o_t) and language prompt c. The model predicts a flow field $\hat{g} = H_\eta\left(G_\theta\left(o_{t-1}, o_t, c, f_s\right)\right)$, which is supervised using the flow matching objective:

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

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 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 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 ∼180k human videos from Something-Something v2 [4] and a subset of ∼170k robot videos from the OpenX [6] dataset, including data from Fractal [83], BridgeV2 [84], and Kuka [85]. We provide further training details in Appendix A, B, C.

SIMPLER We benchmark ViPRA in SIMPLER [22], an open-source suite for evaluating generalist manipulation policies. We evaluate four tasks with a 7-DoF WidowX arm. Because SIMPLER lacks fine-tuning data, we collect 100 diverse multi-task trajectories using a pretrained VLA model [12].

Real World Manipulation We also evaluate ViPRA on a bimanual setup with two 7-DoF Franka Panda robots teleoperated via GELLO [86]. For single-arm experiments, we finetune on three multi-instruction tasks: (1) pick up cloth and cover ⟨object⟩, (2) pick up ⟨object⟩ and place on ⟨plate/board⟩, and (3) pick up ⟨color$_1$⟩ cup and stack on ⟨color$_2$⟩ cup. We 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 operate at higher frequencies. We provide additional insights about our evaluations and challenging bimanual task evaluations in Appendix E, F.

5.2 Baselines

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

1. **Scratch.** Finetunes the video-language backbone (LWM) [7] directly on downstream tasks with history and action chunking, without any pretraining.
2. **OpenVLA** [18] Pretrains a 7B VLA model on 970k real-world demonstrations from Open-X [87].
3. π_0 [20] Augments a 3B VLM [27] with a flow-matching decoder, trained on a mixture of Open-X [87] and proprietary robot data.
4. **LAPA** [12] Learns discrete latent actions via quantization between image pairs, then pretrains LWM to predict these latents given observations and instructions. We use publicly released Open-X–pretrained weights.
5. **UniPI** [34] Uses a video diffusion model for action-free pretraining, then trains an IDM to recover ground-truth actions and finetunes the diffusion model for downstream tasks.
6. **VPT** [35] Trains an IDM on labeled data to extract pseudo-actions from raw videos, then pretrains a LWM backbone with these pseudo-labels, mirroring our latent pretraining part.

LAPA, VPT and OpenVLA discretize continuous actions into bins and cast action prediction as classification; we adopt this setup for Scratch-AR and ViPRA-AR. In contrast, UniPI, π_0, Scratch-FM, and ViPRA-FM predict continuous actions directly and support action chunking. The Scratch baselines serve as a reference to evaluate benefits from our joint pretraining strategy. We include UniPI and VPT for completeness as other approaches that learn from videos. Their reported results are taken from [12], which evaluated them on SIMPLER in a comparable setting.

6 Results

6.1 Simulation Result

We evaluate all methods on four Bridge tasks from SIMPLER [22], a benchmark designed to test generalization across diverse manipulation goals. As shown in Table 1, ViPRA achieves the best average success rate in both discrete and continuous settings.

In the **discrete setting**, ViPRA-AR surpasses LAPA and OpenVLA by a large margin (69.8% vs. 53.1% and 38.6%), excelling in precision-heavy tasks such as `StackG2Y`. In the **continuous setting**, ViPRA-FM outperforms Scratch-FM by 20% and π_0 by 35%, showing the benefits of 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

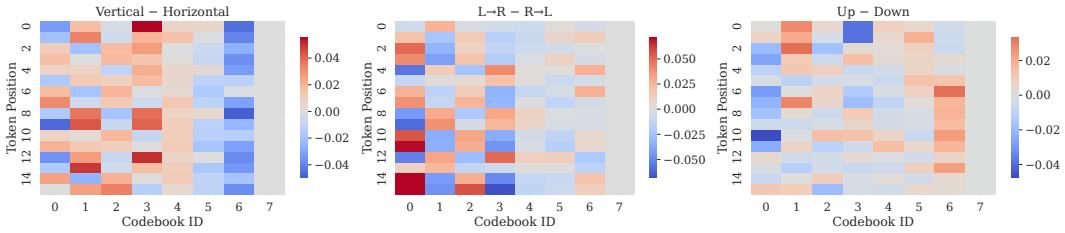


Figure 4: Positional codebook usage differences across action categories, showing that the codebook that ViPRA learn is positionally sensitive and encodes dynamics. Each heatmap shows the difference in per-position token usage between: (right) vertical actions (Up - Down), (middle) horizontal actions (Left->Right - Right->Left), and (c) averaged vertical vs. horizontal groups. Structure reveals that certain codebook entries (e.g., code 0) are positionally sensitive, indicating that token positions encode dynamics.

is slower to converge. However, ViPRA-FM achieves competitive performance, outperforming all other continuous/discrete baselines.

ViPRA also surpasses other video-pretraining approaches (UniPI and VPT) that avoid ground-truth actions. 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.

6.2 Real World Results

Figure 3 shows results on three real-world manipulation tasks. ViPRA-FM attains the best performance with a 54.1% average success rate, outperforming π_0 (40.1%) and Scratch-FM (23.8%). It also demonstrates robust retry behavior, repeatedly attempting grasps after failures, which leads to very high partial success rates—especially in `Cover-Obj`, where the cloth is reliably grasped even if not always placed correctly. We provide further analysis in Appendix E.

While π_0 benefits from task-specific fine-tuning, ViPRA-FM achieves higher success with far less labeled data by leveraging dynamics priors from unlabeled videos. 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.

6.3 Ablations & Analysis

Does video prediction pretraining allow for learning mid-level action priors ? We ablate future state prediction (*-SP2*) from pretraining and observe a clear drop in performance for both ViPRA-AR (69.8% $\rightarrow$ 59.4%) and ViPRA-FM (62.5% $\rightarrow$ 53.2%), indicating that anticipating future observations provides richer, dynamics-aware priors for control. Adding this objective at finetuning (*+SP3*) instead 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.

Experiment	Avg. Success Rate
Discrete Actions	
ViPRA-AR	**69.8**
-SP2	59.4
+SP3	53.1
-AC	59.2
Continuous Actions	
ViPRA-FM	**62.5**
-SP2	53.2
+SP3	31.3
-AC	44.8

Table 2: We verify the impact of future state prediction and action chunking on the performance.

We apply chunking 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 chunking (*-AC*) from both stages (single-step prediction) notably hurts performance, as isolated actions fail to capture smooth temporal dynamics. Our full model, combining chunking with state prediction, achieves the best results (69.8% AR), confirming their complementary benefits.

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 decoder runs at 1.95 Hz per chunk, supporting effective control rates up to 20 Hz on hardware (chunk size 14). We provide additional discussion on the connection between action chunking and high-frequency execution in Appendix E.

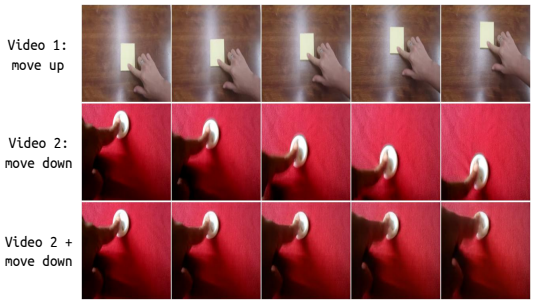


Figure 5: Cross-video latent action transfer.

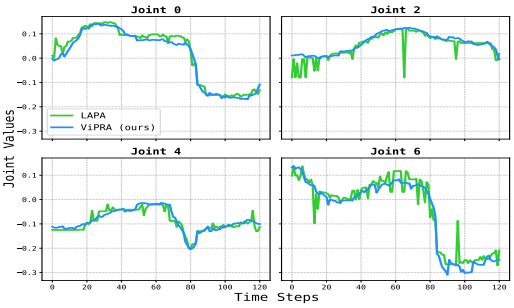


Figure 6: Action smoothness: LAPA [61] and ViPRA.

Can ViPRA learn physically grounded policies? To compare discrete and continuous policies, we analyze action smoothness during closed-loop rollout. We load finetuned ViPRA-FM (blue) and LAPA [61] (green) into our inference pipeline and replay them over trajectories from the finetuning dataset, simulating deployment with real visual observations. Figure 6 shows that while both models track the intended trend, LAPA exhibits frequent local spikes at contact events or occlusions, where small perceptual shifts cause abrupt bin flips. In contrast, ViPRA-FM's flow-matching head produces smooth, demonstration-aligned commands. Because such discontinuities proved unsafe on hardware, we restrict real-world comparisons to continuous baselines only. We provide further smoothness analysis in Appendix D.

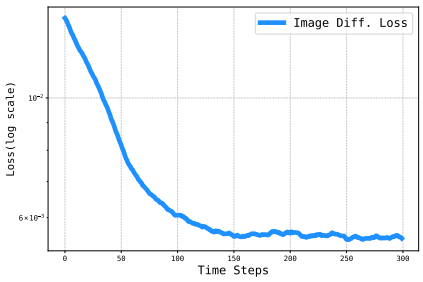


Figure 7: Linear probe loss with difference between sequential images.

Latent action analysis Figure 7 shows that a simple linear probe reveals strong correlation between our discrete latents and image differences, confirming that the codebook captures mid-level action cues. In Figure 5, a cross-video transfer test illustrates this further: when latents encoding upward motion from Video 1 are injected into the opening frames of a downward-moving Video 2, the reconstructions also move upward, demonstrating transferable, dynamics-aware semantics. Finally, Figure 4 analyzes position-wise usage across categories. We compute a token-position × code-index histogram for each category and compare usage across action pairs. The trends suggest that both the selected codebook entry and its position within the latent action sequence encode meaningful structure about motion direction and dynamics.

7 Discussion & Limitations

To train generalist robotic agents, we find that video-language models can be a strong starting point. They often contain semantic intent and temporal dynamics, which are both crucial for taking real world actions. In order to fully take advantage of such models, we present ViPRA, which first learns motion-centric latent actions from large-scale, action-free videos. Then, we pretrain a multimodal video-language model by jointly predicting future visual state and latent actions, thereby fusing semantic grounding with temporal dynamics. Finally, a continuous flow-matching head refines these priors into smooth, high-frequency motor commands using only a few hundred demonstrations. We find that ViPRA can outperform methods that only leverage semantic pretraining. We believe that this design can serve as the blueprint for training scalable general-purpose robotic agents.

Limitations and Broader Impacts 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. We only explore the real world manipulation setting in this work, though our framework could be extended to other action modalities (e.g., web agents). 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.

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, 2016. 1

[2] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18995–19012, 2022. 1

[3] Kristen Grauman, Andrew Westbury, Lorenzo Torresani, Kris Kitani, Jitendra Malik, Triantafyllos Afouras, Kumar Ashutosh, Vijay Baiyya, Siddhant Bansal, Bikram Boote, et al. Ego-exo4d: Understanding skilled human activity from first-and third-person perspectives. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19383–19400, 2024.

[4] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. The" something something" video database for learning and evaluating visual common sense. In *ICCV*, 2017. 7

[5] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, et al. Scaling egocentric vision: The epic-kitchens dataset. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 720–736, 2018. 1

[6] Abby O'Neill, Abdul Rehman, Abhinav Gupta, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023. 1, 2, 7

[7] Hao Liu, Wilson Yan, Matei Zaharia, and Pieter Abbeel. World model on million-length video and language with blockwise ringattention. *arXiv preprint arXiv:2402.08268*, 2024. 1, 3, 5, 7, 4

[8] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023.

[9] Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, et al. Make-a-video: Text-to-video generation without text-video data. *arXiv preprint arXiv:2209.14792*, 2022.

[10] Daquan Zhou, Weimin Wang, Hanshu Yan, Weiwei Lv, Yizhe Zhu, and Jiashi Feng. Magicvideo: Efficient video generation with latent diffusion models. *arXiv preprint arXiv:2211.11018*, 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, 2025. 1

[12] Seonghyeon Ye, Joel Jang, Byeongguk Jeon, Sejune Joo, Jianwei Yang, Baolin Peng, Ajay Mandlekar, Reuben Tan, Yu-Wei Chao, Bill Yuchen Lin, et al. Latent action pretraining from videos. *arXiv preprint arXiv:2410.11758*, 2024. 2, 6, 7, 8, 11

[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, 2025. 3

[14] Yi Chen, Yuying Ge, Yizhuo Li, Yixiao Ge, Mingyu Ding, Ying Shan, and Xihui Liu. Moto: Latent motion token as the bridging language for robot manipulation. *arXiv preprint arXiv:2412.04445*, 2024.

[15] Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025. 2, 3, 4

[16] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022. 2, 3

[17] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023. 2, 3

[18] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024. 6, 7, 8, 11

[19] Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, et al. Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. *arXiv preprint arXiv:2411.19650*, 2024.

[20] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. Pi-zero: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024. 4, 6, 7, 8, 11

[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, 2025. 2, 3

[22] 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 preprint arXiv:2405.05941*, 2024. 2, 7

[23] 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. 2

[24] Xi Chen, Josip Djolonga, Piotr Padlewski, Basil Mustafa, Soravit Changpinyo, Jialin Wu, Carlos Riquelme Ruiz, Sebastian Goodman, Xiao Wang, Yi Tay, et al. Pali-x: On scaling up a multilingual vision and language model. *arXiv preprint arXiv:2305.18565*, 2023.

[25] 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. In *arXiv preprint arXiv:2303.03378*, 2023.

[26] Siddharth Karamcheti, Suraj Nair, Ashwin Balakrishna, Percy Liang, Thomas Kollar, and Dorsa Sadigh. Prismatic vlms: Investigating the design space of visually-conditioned language models. In *Forty-first International Conference on Machine Learning*, 2024.

[27] Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, et al. Paligemma: A versatile 3b vlm for transfer. *arXiv preprint arXiv:2407.07726*, 2024. 2, 7

[28] 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 preprint arXiv:2406.11815*, 2024. 2

[29] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022. 2

[30] Xiang Li, Cristina Mata, Jongwoo Park, Kumara Kahatapitiya, Yoo Sung Jang, Jinghuan Shang, Kanchana Ranasinghe, Ryan Burgert, Mu Cai, Yong Jae Lee, et al. Llara: Supercharging robot learning data for vision-language policy. *arXiv preprint arXiv:2406.20095*, 2024. 2

[31] 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, 2023. 3

[32] 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 preprint arXiv:2312.13139*, 2023.

[33] Po-Chen Ko, Jiayuan Mao, Yilun Du, Shao-Hua Sun, and Joshua B Tenenbaum. Learning to act from actionless videos through dense correspondences. *arXiv preprint arXiv:2310.08576*, 2023. 3

[34] Yilun Du, Sherry Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Josh Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. *Advances in Neural Information Processing Systems*, 36, 2024. 3, 6, 7, 11

[35] Bowen Baker, Ilge Akkaya, Peter Zhokov, Joost Huizinga, Jie Tang, Adrien Ecoffet, Brandon Houghton, Raul Sampedro, and Jeff Clune. Video pretraining (vpt): Learning to act by watching unlabeled online videos. *Advances in Neural Information Processing Systems*, 35:24639–24654, 2022. 3, 6, 7

[36] 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 preprint arXiv:2406.16862*, 2024. 3

[37] Calvin Luo, Zilai Zeng, Yilun Du, and Chen Sun. Solving new tasks by adapting internet video knowledge. In *The Thirteenth International Conference on Learning Representations*, 2025. 3

[38] Calvin Luo, Zilai Zeng, Mingxi Jia, Yilun Du, and Chen Sun. Self-adapting improvement loops for robotic learning, 2025. 3

[39] Shuang Li, Yihuai Gao, Dorsa Sadigh, and Shuran Song. Unified video action model. *arXiv preprint arXiv:2503.00200*, 2025. 3, 11

[40] 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. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. 3

[41] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022. 3

[42] 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.

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

[44] Siddharth Karamcheti, Suraj Nair, Annie S Chen, Thomas Kollar, Chelsea Finn, Dorsa Sadigh, and Percy Liang. Language-driven representation learning for robotics. *arXiv preprint arXiv:2302.12766*, 2023. 3

[45] Shikhar Bahl, Russell Mendonca, Lili Chen, Unnat Jain, and Deepak Pathak. Affordances from human videos as a versatile representation for robotics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13778–13790, 2023. 3

[46] Aditya Kannan, Kenneth Shaw, Shikhar Bahl, Pragna Mannam, and Deepak Pathak. Deft: Dexterous fine-tuning for real-world hand policies. *arXiv preprint arXiv:2310.19797*, 2023.

[47] Homanga Bharadhwaj, Abhinav Gupta, Shubham Tulsiani, and Vikash Kumar. Zero-shot robot manipulation from passive human videos. *arXiv preprint arXiv:2302.02011*, 2023.

[48] Shaowei Liu, Subarna Tripathi, Somdeb Majumdar, and Xiaolong Wang. Joint hand motion and interaction hotspots prediction from egocentric videos. In *CVPR*, 2022.

[49] Mohit Goyal, Sahil Modi, Rishabh Goyal, and Saurabh Gupta. Human hands as probes for interactive object understanding. In *CVPR*, 2022. 3

[50] Jia Zeng, Qingwen Bu, Bangjun Wang, Wenke Xia, Li Chen, Hao Dong, Haoming Song, Dong Wang, Di Hu, Ping Luo, et al. Learning manipulation by predicting interaction. *arXiv preprint arXiv:2406.00439*, 2024. 3

[51] Chuan Wen, Xingyu Lin, John So, Kai Chen, Qi Dou, Yang Gao, and Pieter Abbeel. Any-point trajectory modeling for policy learning. *arXiv preprint arXiv:2401.00025*, 2023. 3

[52] Homanga Bharadhwaj, Roozbeh Mottaghi, Abhinav Gupta, and Shubham Tulsiani. Track2act: Predicting point tracks from internet videos enables diverse zero-shot robot manipulation, 2024.

[53] Priyanka Mandikal and Kristen Grauman. Dexvip: Learning dexterous grasping with human hand pose priors from video. In *Conference on Robot Learning*, pages 651–661. PMLR, 2022.

[54] Shikhar Bahl, Abhinav Gupta, and Deepak Pathak. Human-to-robot imitation in the wild. *arXiv preprint arXiv:2207.09450*, 2022. 3

[55] Debidatta Dwibedi, Jonathan Tompson, Corey Lynch, and Pierre Sermanet. Learning actionable representations from visual observations. In *2018 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 1577–1584. IEEE, 2018. 3

[56] 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 preprint arXiv:2505.04999*, 2025.

[57] 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.

[58] Dominik Schmidt and Minqi Jiang. Learning to act without actions, 2024.

[59] Zichen Cui, Hengkai Pan, Aadhithya Iyer, Siddhant Haldar, and Lerrel Pinto. Dynamo: In-domain dynamics pretraining for visuo-motor control. *Advances in Neural Information Processing Systems*, 37:33933–33961, 2024. 3

[60] Aaron van den Oord, Oriol Vinyals, et al. Neural discrete representation learning. In *NeurIPS*, pages 6309–6318, 2017. 3

[61] Seonghyeon Ye, Joel Jang, Byeongguk Jeon, Sejune Joo, Jianwei Yang, Baolin Peng, Ajay Mandlekar, Reuben Tan, Yu-Wei Chao, Bill Yuchen Lin, et al. Latent action pretraining from videos. *arXiv preprint arXiv:2410.11758*, 2024. 3, 9, 7

[62] Seungjae Lee, Yibin Wang, Haritheja Etukuru, H Jin Kim, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. Behavior generation with latent actions. *arXiv preprint arXiv:2403.03181*, 2024.

[63] Chenyu Yang, Davide Liconti, and Robert K Katzschmann. Vq-ace: Efficient policy search for dexterous robotic manipulation via action chunking embedding. *arXiv preprint arXiv:2411.03556*, 2024. 3

[64] Liang Xu, Ziyang Song, Dongliang Wang, Jing Su, Zhicheng Fang, Chenjing Ding, Weihao Gan, Yichao Yan, Xin Jin, Xiaokang Yang, et al. Actformer: A gan-based transformer towards general action-conditioned 3d human motion generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2228–2238, 2023. 3

[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, 2025. 3

[66] Shenyuan Gao, Siyuan Zhou, Yilun Du, Jun Zhang, and Chuang Gan. Adaworld: Learning adaptable world models with latent actions. In *International Conference on Machine Learning (ICML)*, 2025. 3

[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, 2024. 3

[68] David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018. 3

[69] Théophane Weber, Sébastien Racanière, David P Reichert, Lars Buesing, Arthur Guez, Danilo Jimenez Rezende, Adria Puigdomènech Badia, Oriol Vinyals, Nicolas Heess, Yujia Li, et al. Imagination-augmented agents for deep reinforcement learning. *arXiv preprint arXiv:1707.06203*, 2017.

[70] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 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 preprint arXiv:1811.04551*, 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 preprint arXiv:1907.00953*, 2019.

[73] Philipp Wu, Alejandro Escontrela, Danijar Hafner, Ken Goldberg, and Pieter Abbeel. Daydreamer: World models for physical robot learning. *arXiv preprint arXiv:2206.14176*, 2022.

[74] Ramanan Sekar, Oleh Rybkin, Kostas Daniilidis, Pieter Abbeel, Danijar Hafner, and Deepak Pathak. Planning to explore via self-supervised world models. *ICML*, 2020. 3

[75] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023. 3

[76] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023. 3

[77] Wilhelm Kutta. *Beitrag zur näherungsweisen Integration totaler Differentialgleichungen*. Teubner, 1901. 4

[78] Carl Runge. Über die numerische auflösung von differentialgleichungen. *Mathematische Annalen*, 46(2):167–178, 1895. 4

[79] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 5

[80] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 402–419. Springer, 2020. 5

[81] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. 5, 3

[82] Mohammad Hassan Vali and Tom Bäckström. Nsvq: Noise substitution in vector quantization for machine learning. *IEEE Access*, 10:13598–13610, 2022. 5, 3

[83] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022. 7

[84] Frederik Ebert, Yanlai Yang, Karl Schmeckpeper, Bernadette Bucher, Georgios Georgakis, Kostas Daniilidis, Chelsea Finn, and Sergey Levine. Bridge data: Boosting generalization of robotic skills with cross-domain datasets. *arXiv preprint arXiv:2109.13396*, 2021. 7

[85] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning*, pages 651–673. PMLR, 2018. 7

[86] Philipp Wu, Yide Shentu, Zhongke Yi, Xingyu Lin, and Pieter Abbeel. Gello: A general, low-cost, and intuitive teleoperation framework for robot manipulators, 2023. 7, 9

[87] Open X-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, 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, Roberto Mart'in-Mart'in, Rohan Baijal, Rosario Scalise, Rose Hendrix, Roy Lin, Runjia Qian, Ruohan Zhang, Russell Mendonca, Rutav Shah, Ryan Hoque, Ryan Julian, Samuel Bustamante, Sean Kirmani, Sergey Levine, Shan Lin, Sherry Moore, Shikhar Bahl, Shivin Dass, Shubham Sonawani, Shubham Tulsiani, Shuran Song, Sichun Xu, Siddhant Haldar, Siddharth Karamcheti, Simeon Adebola, Simon Guist, Soroush Nasiriany, Stefan Schaal, Stefan Welker, Stephen Tian, Subramanian Ramamoorthy, Sudeep Dasari, Suneel Belkhale, Sungjae Park, Suraj Nair, Suvir Mirchandani, Takayuki Osa, Tanmay Gupta, Tatsuya Harada, Tatsuya Matsushima, Ted Xiao, Thomas Kollar, Tianhe Yu, Tianli Ding, Todor Davchev, Tony Z. Zhao, Travis Armstrong, Trevor Darrell, Trinity Chung, Vidhi Jain, Vikash Kumar, Vincent Vanhoucke, Wei Zhan, Wenxuan Zhou, Wolfram Burgard, Xi Chen, Xiangyu Chen, Xiaolong Wang, Xinghao Zhu, Xinyang Geng, Xiyuan Liu, Xu Liangwei, Xuanlin Li, Yansong Pang, Yao Lu, Yecheng Jason Ma, Yejin Kim, Yevgen Chebotar, Yifan Zhou, Yifeng Zhu, Yilin Wu, Ying Xu, Yixuan Wang, Yonatan Bisk, Yongqiang Dou, Yoonyoung Cho, Youngwoon Lee, Yuchen Cui, Yue Cao, Yueh-Hua Wu, Yujin Tang, Yuke Zhu, Yunchu Zhang, Yunfan Jiang, Yunshuang Li, Yunzhu Li, Yusuke Iwasawa, Yutaka Matsuo, Zehan Ma, Zhuo Xu, Zichen Jeff Cui, Zichen Zhang, Zipeng Fu, and Zipeng Lin. Open X-Embodiment: Robotic learning datasets and RT-X models. https://arxiv.org/abs/2310.08864, 2023. 7

[88] Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success, 2025. 11

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

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

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

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

D **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.

E **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.

F **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-robot.github.io.

A Latent Action Learning Implementation

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 8, and the complete training configuration—including model architecture and optimization hyperparameters—is provided in Table 3.

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

Extract visual features: $f_{0:L} \leftarrow \text{DINOv2}(o_{0:L})$
Compute contextual embeddings: $h_{0:L} \leftarrow I_\beta(f_{0:L})$
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

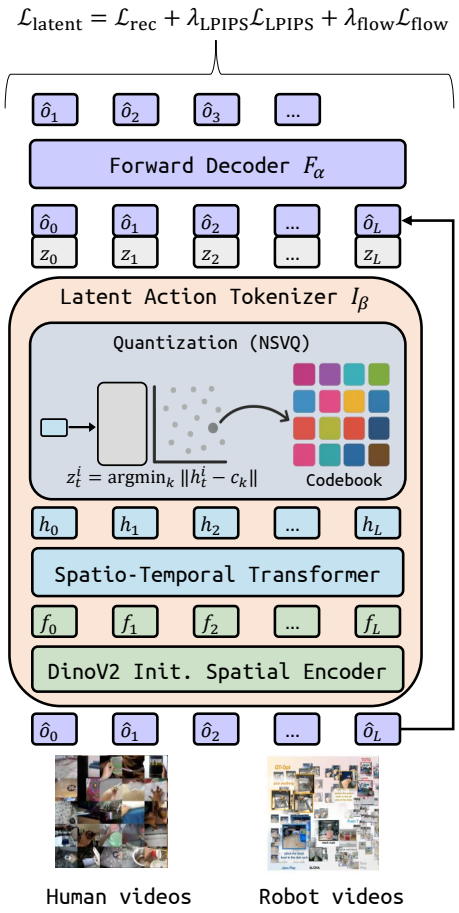


Figure 8: **Latent action learning framework.** Given a sequence of video frames, an inverse dynamics encoder processes the full observation clip to produce discrete latent tokens z_t via vector quantization. A decoder then reconstructs the future frame $\hat{o}_{t+1}$ conditioned on z_t and the observation context. The model is trained using a combination of reconstruction, perceptual (LPIPS), and optical flow losses to ensure the latent tokens capture physically grounded, 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	64
Total Steps	240000
Image Augmentation	RandomResizeCrop
Flow Start Step α_{flow}	25000
Losses	$\mathcal{L}_{\text{rec}} + \lambda_{\text{LPIPS}}\mathcal{L}_{\text{LPIPS}} + \lambda_{\text{flow}}\mathcal{L}_{\text{flow}}$
LPIPS Weight λ_{LPIPS}	1
Flow Weight λ_{flow}	0.1 (after α_{flow})
GPU	8 Nvidia L40S (120 hours)
Inverse Dynamics Encoder I	
Backbone Init.	DINOv2 [81]
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 [82]
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.

B 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: $x_{t-1}, x_t \leftarrow E_{\text{VQ}}(o_{t-1}), E_{\text{VQ}}(o_t)$
Tokenize target frame: $x_{t+H} \leftarrow E_{\text{VQ}}(o_{t+H})$
Encode text prompt: $\tilde{c} \leftarrow \texttt{Tokenizer}(c)$
Embed latent actions: $\tilde{z}_{t:t+H-1} \leftarrow E_\phi(z_{t:t+H-1})$
for $i = 1$ **to** N_{tokens} **do**
    $\hat{x}^i_{t+H} \leftarrow G_\theta(\tilde{c}, x_{t-1}, x_t, \hat{x}^{<i}_{t+H})$ {Autoregressive prediction}
end for
$\mathcal{L}_{\text{img}} \leftarrow \sum_{i=1}^{N_{\text{tokens}}} \text{CE}(\hat{x}^i_{t+H}, x^i_{t+H})$ {Image token loss}
for $k = t$ **to** $t + H - 1$ **do**
    for $i = 1$ **to** N_{latent} **do**
        $\hat{z}^i_k \leftarrow H_\psi(G_\theta(\tilde{c}, x_{t-1}, x_t, x_{t+H}, z_{<k}, z^{<i}_k))$
    end for
end for
$\mathcal{L}_{\text{act}} \leftarrow \sum_{k=t}^{t+H-1} \sum_{i=1}^{N_{\text{latent}}} \text{CE}(\hat{z}^i_k, z^i_k)$ {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	2e-5
Weight Decay	0.0
Optimizer Betas	$(0.9, 0.95)$
Batch Size	96
Total Steps	90,000
Dropout	0.1
Gradient Clipping	1.0
Mixed Precision	`bfloat16`
GPU	8 Nvidia L40S (250 hours)
Prediction Targets	
Prediction Horizon H	14
Image Token Loss	Cross Entropy
Latent Action Loss	Cross Entropy

Table 4: Hyperparameters for Stage 2 multimodal video pretraining. The video model is initialized from LWM-Chat-1M and trained jointly with lightweight latent action modules.

C 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 action a_t 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: $x_{t-1}, x_t \leftarrow E_{\text{VQ}}(o_{t-1}), E_{\text{VQ}}(o_t)$
Encode text prompt: $\tilde{c} \leftarrow \texttt{Tokenizer}(c)$
Encode noisy actions: $f_s \leftarrow E_\gamma(x_s, s)$
Predict flow field: $\hat{g} \leftarrow H_\eta\left(G_\theta(\tilde{c}, x_{t-1}, x_t, f_s)\right)$
$\mathcal{L}_{\text{FM}} \leftarrow \|a_{t:t+H-1} - x_0 - (1-s) \cdot \hat{g}\|_2^2$ {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 or 8
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

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

D Action Output Analysis

We provide a more in-depth analysis of the action outputs of various policies introduced in Section 6.3, 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 [61] 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 [61] 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 [61], 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:

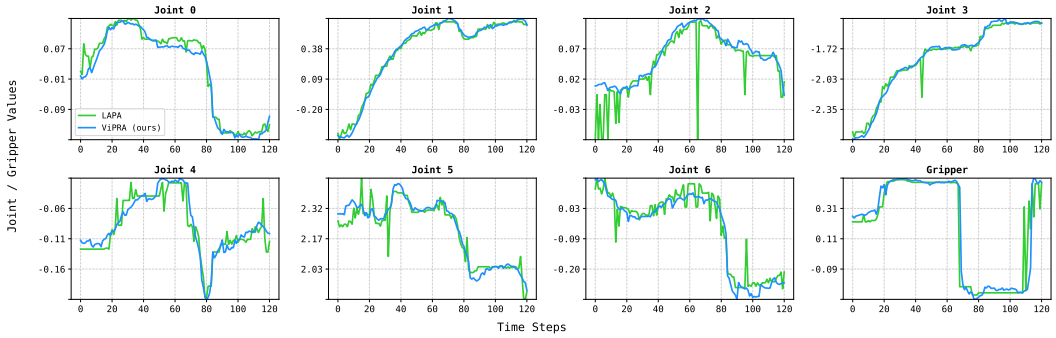
1. **Contact-Sensitive Flipping:** At test time, small perturbations in the input (e.g., due to occlusions or slight viewpoint drift) may cause the model to flip from one bin to another near the quantile boundary—especially at contact points. Since adjacent bins can correspond to different action magnitudes, 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.

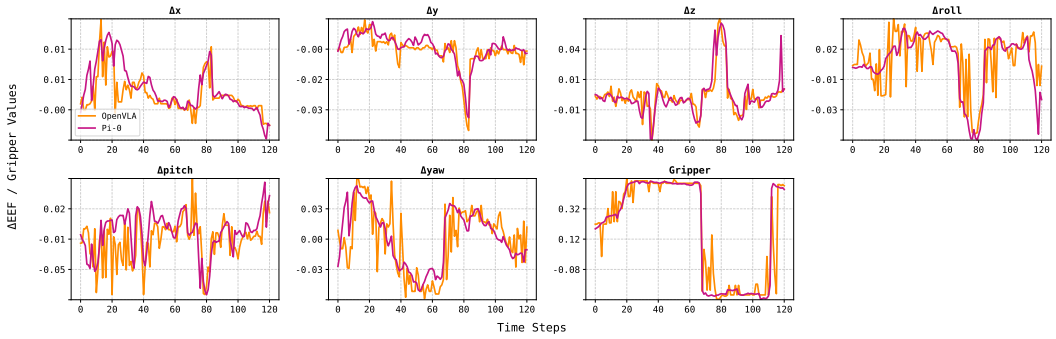
By contrast, continuous policies such as ViPRA-FM and π_0 [20] operate directly in $\mathbb{R}^D$ using flow matching. These losses naturally reflect the structure of the action space—penalizing predictions in proportion to how far they deviate from the ground truth. As a result, the output trajectories 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 9.



(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 9: 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.

E Real World Experiments: Setup, Challenges, and Observations

To complement our real world results in Section 6.2, 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.

E.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 [86] 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.

E.2 Task Descriptions and Challenges

We evaluate policies on three real world single-arm tasks, each with unique control and perception challenges (Figure 10):

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.

E.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.

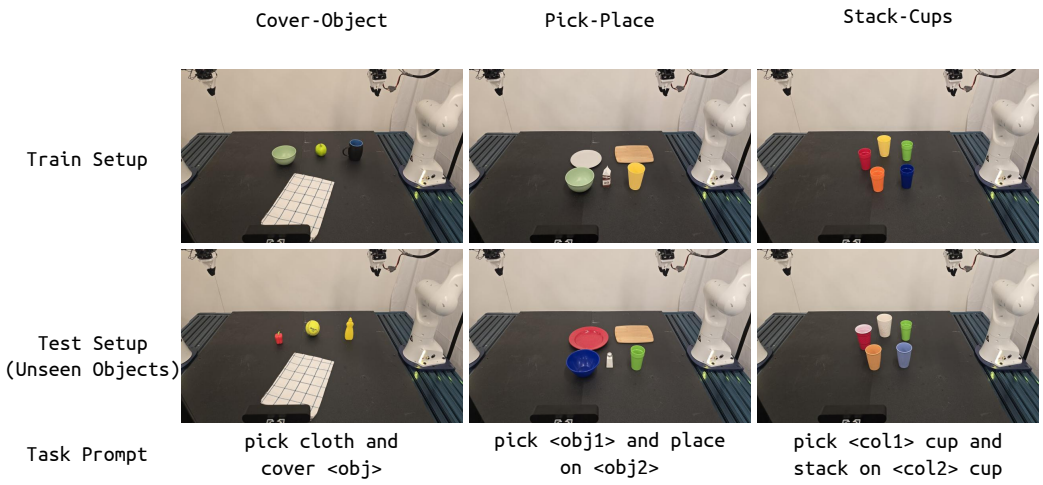


Figure 10: **Task Setup Overview.** (Top row) Training environments for each of the three single-arm manipulation tasks: `Cover-Object`, `Pick-Place`, and `Stack-Cups`. (Bottom row) Evaluation environments featuring novel objects, textures, or placements not seen during training. Note the variety in 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.

E.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.

E.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 [88].

Comparison with baselines. Table 6 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 [34]	–	16	24000
UVA [39]	0.5B	16	230
ViPRA (ViPRA-FM)	7B	14	510

Table 6: Inference speed comparison across models. ViPRA achieves high effective control frequencies by amortizing computation over action chunks.

F 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.

F.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.

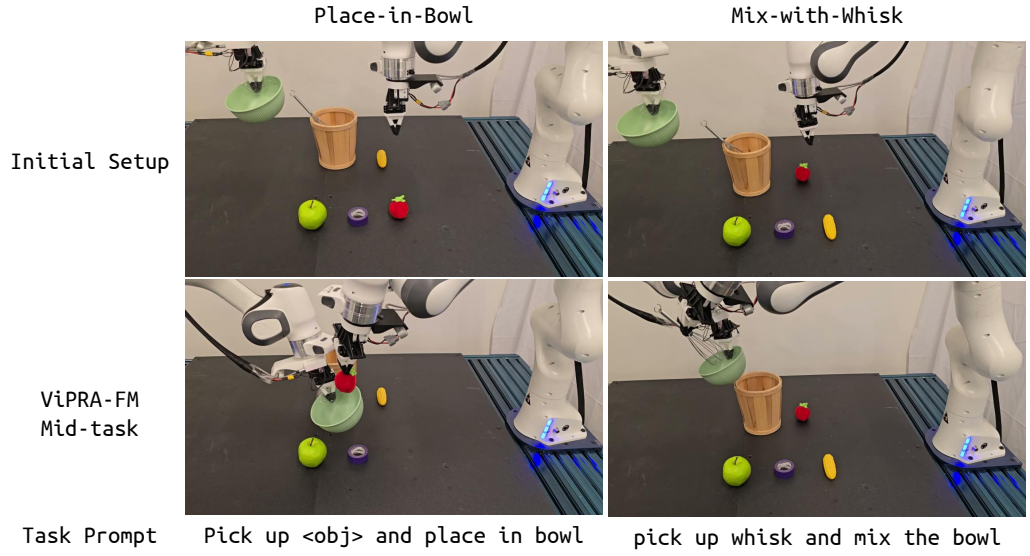


Figure 11: **Bimanual task execution by ViPRA-FM.** (Top row) Initial setup for the two tasks: placing a tomato into a bowl and mixing with a whisk. (Bottom row) Mid-execution rollout of ViPRA-FM: the right arm transports the tomato toward the bowl held by the left arm (left), and mixes the contents 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.

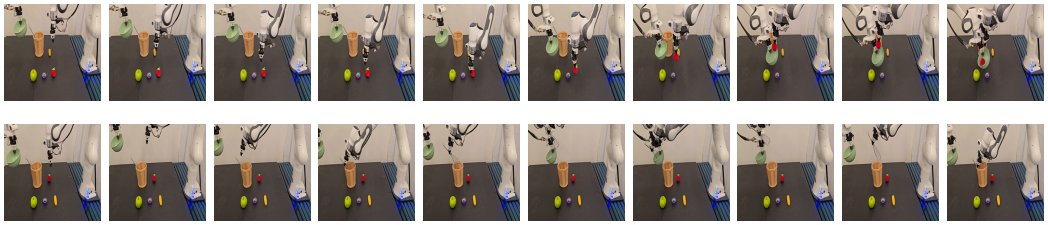


Figure 12: **ViPRA-FM rollouts in real world bimanual tasks.** Top: `Place-in-Bowl` — the robot picks up a tomato and places it into a bowl held by the left arm. Bottom: `Mix-with-Whisk` — the robot retrieves a whisk, stirs the bowl contents, and returns the tool. Each sequence shows 10 evenly spaced frames sampled from real world executions.

F.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-robot.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.