ANT GROUP

Causal World Modeling for Robot Control

Lin Li* Qihang Zhang*† Yiming Luo* Shuai Yang Ruilin Wang Fei Han
Mingrui Yu Zelin Gao Nan Xue Xing Zhu Yujun Shen Yinghao Xu‡

*Equal Contribution †Project Lead ‡Corresponding Author

arXiv:submit/7211108 [cs.CV] 29 Jan 2026

This work highlights that video world modeling, alongside vision-language pre-training, establishes a fresh and independent foundation for robot learning. Intuitively, video world models provide the ability to "imagine" the near future by understanding the causality between actions and visual dynamics. Inspired by this, we introduce LingBot-VA, an autoregressive diffusion framework that learns frame prediction and policy execution simultaneously. Our model features three carefully crafted designs: (1) ***a shared latent space***, integrating vision and action tokens, driven by a Mixture-of-Transformers (MoT) architecture, (2) ***a closed-loop rollout mechanism***, allowing for ongoing acquisition of environmental feedback with ground-truth observations, (3) ***an asynchronous inference pipeline***, parallelizing action prediction and motor execution to support efficient control. We evaluate our model on both simulation benchmarks and real-world scenarios, where it shows significant promise in long-horizon manipulation, data efficiency in post-training, and strong generalizability to novel configurations. The code and model are made publicly available to facilitate the community.

Website: https://technology.robbyant.com/lingbot-va
Github: https://github.com/robbyant/lingbot-va
Checkpoints: https://huggingface.co/robbyant/lingbot-va

Robbyant

1 Introduction

Vision-Language-Action (VLA) models have emerged as a promising paradigm for general-purpose robotic manipulation [7, 11, 12, 34], demonstrating impressive capabilities in grounding linguistic instructions into visual perceptions across diverse objects and unstructured environments. However, beneath their apparent success lies a significant challenge: *representation entanglement*. Most existing VLAs adopt a feedforward paradigm that maps current observations to action sequences [17, 90], requiring a single neural network to simultaneously learn visual scene understanding, physical dynamics, and motor control from a unified supervision signal. This entanglement can create a bottleneck—the model must compress heterogeneous knowledge, ranging from high-dimensional visual semantics to low-dimensional motor commands, into a shared representation space. This often leads to limited sample efficiency and suboptimal generalization. Without explicit modeling of environmental evolution [25, 26, 81], reactive policies may rely on pattern matching rather than a principled understanding of physical dynamics.

Recent attempts to bring world modeling into robotic policies span interactive neural simulators (e.g., UniSim [85]), chunk-based video-action diffusion models (e.g., UVA [40] and UWM [96]), and offline video generators for subgoal synthesis (e.g. Gen2Act [4], Act2Goal [94]). While conceptually appealing, these approaches face three primary limitations for effective closed-loop control. First, *the reactivity gap*: chunk/open-loop generation often rolls out long segments without incorporating real-time feedback, making it hard to adapt to disturbances. Second, *limited long-term memory*: chunk-wise generation can introduce inconsistencies over long horizons when history is not persistently cached. Third, *causality*: bidirectional attention within a segment allows future tokens to influence past predictions, which diverges from the causal nature of physical reality where the present depends only on the past. These observations motivate an autoregressive formulation for robust closed-loop reasoning.

We propose LingBot-VA, an *autoregressive diffusion* world model that addresses these limitations through a unified

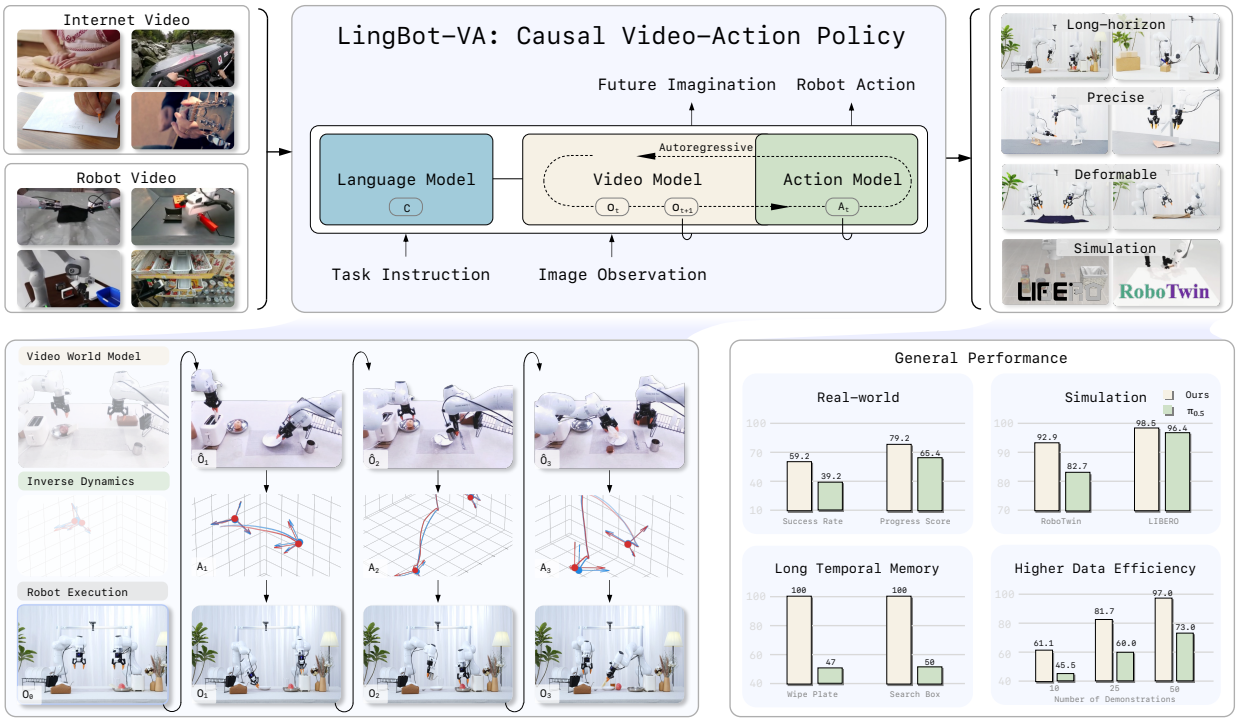


Figure 1. LingBot-VA **: An Autoregressive World Model for Robotic Manipulation.** (1) **Pretraining:** LingBot-VA is pretrained on diverse in-the-wild videos and robot action data, enabling strong generalization across scenes and objects. (2) **Comprehensive Evaluation:** We conduct extensive experiments on real-world tasks (long-horizon, deformable objects, and precision manipulation) and simulation benchmarks, significantly outperforming state-of-the-art methods including $\pi_{0.5}$. (3) **Versatile Capabilities:** Beyond policy learning, our model supports visual dynamics prediction and inverse dynamics inference from robot videos. (4) **Emergent Properties:** Our causal world modeling approach exhibits long-range temporal memory and strong few-shot adaptation ability.

video-action framework. Unlike autoregressive language models that predict discrete tokens, our model operates in a continuous latent space via flow matching [46, 50], autoregressively generating chunks of video and action representations through iterative denoising. While our approach conceptually separates visual dynamics prediction and action decoding [22, 27], the key architectural insight is to *interleave* video and action tokens into a single autoregressive sequence. Both modalities are jointly processed through a Mixture-of-Transformers (MoT) architecture [43] with shared attention. Within this unified autoregressive generation process, latent imagination and action inference occur jointly: at each autoregressive step, the model generates predicted future visual states through iterative denoising while simultaneously decoding the corresponding actions, allowing both streams to mutually condition on one another. This integration, built upon a large-scale pretrained video diffusion backbone [78], offers several advantages: *(i) Reactive AR loop*: because video and action tokens form a unified sequence, each autoregressive step allows the system to recalibrate based on the latest real-world observation, enabling timely adjustments to both the predicted future and motor commands; *(ii) Persistent context through KV-cache*: the cached key-value pairs preserve the interleaved video-action trajectory, providing a rich context that helps mitigate temporal drift; *(iii) Causal consistency*: causal attention masking over the unified sequence ensures that both predicted visual states and action commands are governed by preceding states, respecting the temporal arrow of physical dynamics. By incorporating real-world observations at each step, this formulation helps mitigate the *distribution drift* that often affects open-loop methods in long-horizon tasks.

A primary challenge in deploying large-scale autoregressive video-action models is inference latency; generating high-fidelity video tokens through iterative denoising is computationally intensive. We address this through two complementary strategies. First, we introduce *Noisy History Augmentation*, a training scheme that enables *partial denoising* at inference time. The key insight is that action decoding does not always require pixel-perfect reconstruction; instead, it can rely on robust semantic structures. By training the action decoder to predict from partially noisy latent representations, we significantly reduce the computational overhead while maintaining precise action prediction. Second, we design an *asynchronous coordination* pipeline that overlaps computation with execution: while the

robot executes current actions, the world model predicts future visual states and plans subsequent sequences. This parallelized architecture, combined with variable chunk-size training, facilitates high-frequency closed-loop control without compromising prediction quality.

We evaluate `LingBot-VA` across diverse manipulation tasks in both simulation and real-world environments. Our method demonstrates competitive performance compared to state-of-the-art VLA policies, particularly in long-horizon tasks requiring temporal consistency. Our contributions are summarized as follows:

- **Autoregressive Video-Action World Modeling:** We introduce an autoregressive diffusion framework that *architecturally unifies* visual dynamics prediction and action inference within a single interleaved sequence while maintaining their *conceptual distinction*. This formulation supports persistent memory through KV cache and causal consistency via attention masking.
- **Mixture-of-Transformers Architecture with Asynchronous Execution:** We design a dual-stream MoT architecture with asymmetric capacity and introduce a partial denoising strategy combined with asynchronous coordination to enable efficient robotic control.
- **Superior Long-Horizon and Precision Performance:** Extensive real-world and simulation experiments demonstrate consistent state-of-the-art performance, with particularly strong improvements on long-horizon and high-precision manipulation tasks. Our method also achieves significantly improved sample efficiency and strong generalization to novel scenes and object configurations.

2 Preliminary

2.1 Flow Matching

Flow matching [46, 50, 74] is a continuous-time generative modeling framework that learns to transform a simple source distribution (e.g., Gaussian noise) to a target data distribution through a continuous flow. Given a data sample x_1 and a noise sample $\epsilon \sim \mathcal{N}(0, I)$, flow matching defines a time-dependent vector field $v_s : \mathbb{R}^d \times [0, 1] \to \mathbb{R}^d$ that describes the instantaneous velocity of particles flowing from ϵ to x_1. The trajectory $x^{(s)}$ evolves according to the ordinary differential equation (ODE):

$$\frac{dx^{(s)}}{ds} = v_s(x^{(s)}), \quad x^{(0)} = \epsilon \sim \mathcal{N}(0, I), \tag{1}$$

where $s \in [0, 1]$ denotes the flow time.

The model is trained to predict this vector field by minimizing:

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{s,\epsilon,x_1}\left[\|v_\theta(x^{(s)}, s) - \dot{x}^{(s)}\|^2\right], \tag{2}$$

where $\dot{x}^{(s)}$ is the true velocity along the interpolation path, typically defined as $x^{(s)} = (1 - s)\epsilon + sx_1$, giving $\dot{x}^{(s)} = x_1 - \epsilon$.

At inference, samples are generated by solving the learned ODE from $s = 0$ to $s = 1$:

$$x_1 = \epsilon + \int_0^1 v_\theta(x^{(s)}, s)\, ds. \tag{3}$$

2.2 Video Generation with Conditional Flow Matching

Recent video generation models [23, 35, 54, 78] leverage flow matching to generate videos conditioned on text or images. These models operate in the latent space of pretrained video autoencoders, where visual observations are encoded as latent representations $z_t = E(o_t)$ using encoder E (e.g., from video diffusion models).

Given a conditioning signal c (text prompt or initial image), the flow matching model learns to generate a sequence of latent video frames $\mathbf{z} = \{z_1, \ldots, z_T\}$ by predicting the vector field:

$$v_\theta(\mathbf{z}^{(s)}, s \mid c) = \frac{d}{ds}\mathbf{z}^{(s)}, \tag{4}$$

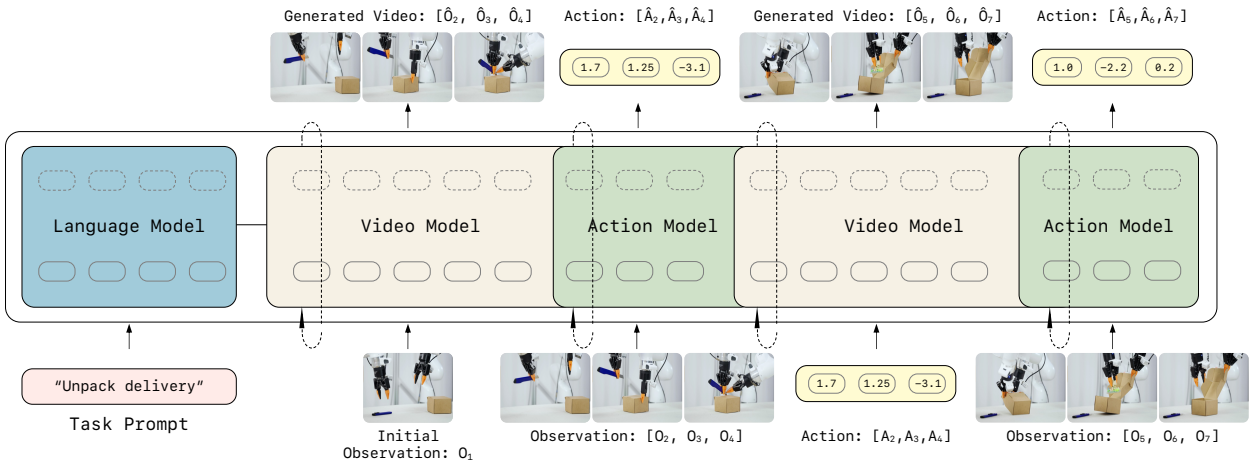


Figure 2. Framework overview: LingBot-VA is conditioned by *autoregressive diffusion* for unified *video-action world modeling*. We leverage a dual-stream *Mixture-of-Transformers (MoT)* architecture that interleaves video and action tokens within a single sequence. At each autoregressive step, the video stream (initialized from Wan2.2-5B) first predicts future latent visual states via *flow matching*. Then the action stream decodes corresponding actions through *inverse dynamics* conditioning on the predicted visual transitions.

where $s \in [0, 1]$ is the flow time and $\mathbf{z}^{(s)}$ represents the latent video at flow step s. The generation process starts from noise $\mathbf{z}^{(0)} = \epsilon \sim \mathcal{N}(0, I)$ and integrates the learned vector field to obtain the final latent video $\mathbf{z}^{(1)}$, which is then decoded to pixel space. This bidirectional generation framework enables flexible synthesis from text descriptions or seed images.

3 Method

3.1 Problem Statement & Approach Overview

We study robotic manipulation as a sequential decision-making problem under partial observability. At each timestep t, the agent receives a visual observation $o_t \in \mathcal{O}$ and executes an action $a_t \in \mathcal{A}$, which induces a transition in the underlying physical world and produces the next observation o_{t+1}.

Vision-Language-Action (VLA) Policies. Most existing VLA policies learn a direct, reactive mapping from observation history to actions:

$$a_t \sim \pi_\theta(\cdot \mid o_t), \tag{5}$$

through imitation learning on robot demonstration data. While this end-to-end approach has shown impressive results, it suffers from a fundamental coupling problem: the model must simultaneously learn visual scene understanding, physical dynamics, and motor control from a single supervision signal of paired observations and actions. This entanglement leads to poor sample efficiency and limited generalization, as the model struggles to disentangle visual reasoning from action prediction without explicit dynamics modeling.

Our Approach. Unlike VLA policies that directly learn action distributions, we adopt a world modeling perspective: instead of learning $\pi(a_t \mid o_t)$, we predict how the visual world will evolve, then infer actions based on these predictions.· Our approach operates in two stages:

$$\begin{aligned} &\text{(Stage 1) Visual dynamics prediction:} && o_{t+1} \sim p_\theta(\cdot \mid o_{\le t}), \\ &\text{(Stage 2) Inverse dynamics:} && a_t \sim g_\psi(\cdot \mid o_t, o_{t+1}). \end{aligned} \tag{6}$$

Stage 1 learns to predict future visual observations given observation history. Stage 2 uses an inverse dynamics model to decode actions from desired visual transitions. This decomposition enables Stage 1 to leverage large-scale video data for learning physical priors, while Stage 2 only requires robot demonstrations to ground visual predictions in executable actions.

Method Overview. Figure 2 illustrates the details of our framework. Our method consists of three key components, detailed in the following subsections: *(§3.2) Autoregressive Video-Action World Modeling* describes how we model visual dynamics in latent space and decode actions from predicted state transitions—this is the *core formulation* of our approach; *(§3.3) LingBot-VA: Unified Architecture & Training* presents our unified model for video-action pretraining, including the architecture design and training objective—this is the *instantiation* of our formulation; *(§3.4) Real-time Deployment & Asynchronous Inference* introduces our deployment strategy that enables real-time control through parallelized prediction and execution—this is the *practical realization* for robotic control.

3.2 Autoregressive Video-Action World Modeling

Previous video world models either focus on open-ended video prediction [54] or learn action-conditioned interactive environments [13, 56] primarily for game or simulation domains, which may not directly transfer to precise robotic manipulation. To leverage rich visual dynamics priors from video data for robot manipulation, we propose a unified video-action world modeling framework that jointly models visual observations and robot actions within a single autoregressive process. Unlike prior approaches that either decouple video prediction from action inference [16, 27] or rely on bidirectional diffusion within segments [96], our method unifies video and action within a single *causal autoregressive* framework, enabling persistent memory through KV cache and seamless integration of real-time observations.

World Dynamics with Autoregressive Modeling. Recent world models for robotics often adopt bidirectional video generation approaches [4, 20, 24, 42] or learn interactive simulators [85], which face fundamental limitations for closed-loop control. Open-loop methods that generate entire long sequences in one shot incur prohibitive computational cost and cannot incorporate real-time feedback for error correction. Chunk-based diffusion methods that generate video segments sequentially [22, 96] suffer from two critical issues: (1) they lack persistent memory across chunks, as each chunk is generated independently without access to the full history, leading to temporal inconsistencies and drift over long horizons; (2) the bidirectional attention within each chunk violates causality, preventing seamless integration with real-time observations during execution.

The physical world, however, is inherently causal and autoregressive: the present state depends only on the past, and we cannot observe the future before it occurs. This fundamental property motivates our autoregressive world modeling approach, which offers three critical advantages over chunk-based diffusion for robotic control: *(1) Persistent Memory*: by explicitly conditioning on the complete observation history through causal attention and KV cache, the model maintains long-term context and temporal coherence across the entire trajectory, avoiding the "amnesia" problem of chunk-based methods; *(2) Causal Consistency*: the unidirectional dependency structure naturally aligns with closed-loop execution, where new observations can be seamlessly incorporated as they arrive; *(3) Efficiency*: chunk-wise prediction with parallel generation within each chunk balances computational efficiency with autoregressive flexibility, enabling high-frequency control with real-time error correction.

We formalize this as an autoregressive process: at each step, the world model predicts the next chunk of K video frames using conditional flow matching:

$$o_{t+1:t+K} \sim p_\theta(\cdot \mid o_{\leq t}), \tag{7}$$

where tokens within each chunk are generated in parallel via bidirectional attention, while maintaining causal structure across chunks. This chunk-wise formulation balances generation efficiency with autoregressive flexibility for closed-loop correction.

Video-Action State Encoding. Operating directly on pixel-level video observations is computationally prohibitive due to the high dimensionality and redundancy of raw visual data. We leverage a causal video VAE [78] to compress visual observations into compact latent tokens $z_t = E(o_t \mid o_{<t}) \in \mathbb{R}^{N\times 4}$, where N is the number of spatial tokens after passing into video VAE. By conditioning on previous latent states, the encoder maintains temporal coherence while processing observations sequentially, naturally aligning with our autoregressive world modeling framework. To align robot actions with visual tokens, we project action vectors to token embeddings $a_t \in \mathbb{R}^D$ via a lightweight MLP $\phi(\cdot)$ where D is the dimension of the video token after patchfication, enabling unified interleaving of visual and action tokens as in prior approaches [5, 22].

Latent Video State Transition. While standard video generation models predict future frames based solely on visual history, robotic manipulation requires accounting for the embodiment's physical state and interaction with the

environment. During deployment, the robot's state evolves through continuous interaction: each action modifies the embodiment's configuration (e.g., gripper position, joint angles), which in turn influences how the scene evolves.

In many manipulation settings, actions encode absolute pose information (e.g., end-effector poses in world coordinates), so the action history $a_{<t}$ effectively captures the trajectory of the embodiment's configuration. Conditioning on action history thus provides knowledge of how the robot has moved and interacted with objects, consistent with prior action-conditioned video/world models [22, 85, 96]. We extend our autoregressive formulation to condition on both observation and action histories:

$$z_{t+1:t+K} \sim p_\theta(\cdot \mid z_{\leq t}, a_{<t}), \tag{8}$$

where z_t is the latent visual state and a_t is the action token. This enables the world model to ground predictions in the embodiment's state, ensuring that predicted observations reflect the robot's physical interaction with the scene.

Inverse Dynamics for Action Decoding. Once the world model predicts future visual states, we leverage these predictions to plan actions. Rather than directly predicting actions from current observations, we employ an inverse dynamics model that infers actions by conditioning on desired future observations, enabling the policy to reason about *what action leads to a desired visual outcome*.

However, simply conditioning on the current and next states (z_t, z_{t+1}) is insufficient for accurate action prediction. The action history $a_{<t}$ encodes the embodiment's state trajectory for determining feasible actions, while the observation history $z_{<t}$ provides temporal context for multi-step interactions (e.g., whether an object was previously grasped). We therefore formulate inverse dynamics as:

$$a_{t:t+K-1} \sim g_\psi(\cdot \mid \hat{z}_{t+1:t+K}, z_{\leq t}, a_{<t}), \tag{9}$$

where the inverse dynamics model g_ψ takes as input the predicted chunk of visual states $\hat{z}_{t+1:t+K}$ inferred by Eq. 8, observation history $z_{\leq t}$, and action history $a_{<t}$. This mirrors recent IDM-based policies [1, 20, 22, 55, 72] that leverage future targets to infer feasible actions while maintaining consistency with embodiment dynamics.

3.3 LingBot-VA: Unified Architecture & Training

Architecture. To jointly model video and action generation, we leverage a dual-stream diffusion transformer architecture that performs conditional flow matching for autoregressive prediction. Our model consists of two parallel transformer backbones: a video stream initialized from Wan2.2-5B (a large-scale pretrained video generation model with dimension d_v [78]), and an action stream with same depth but significantly smaller width $d_a \ll d_v$. This asymmetric design is motivated by the observation that action distributions are inherently simpler than visual data requiring fewer parameters to model effectively while maintaining expressive capacity for visual dynamics.

Video Sparsification. Video frames exhibit significant temporal redundancy, especially in robotic manipulation where scenes evolve gradually. We sparsify the video sequence by temporally downsampling frames by a factor of $\tau = 4$, reducing visual tokens while improving efficiency [5]. Since actions evolve at higher frequency than visual changes, we interleave the downsampled video tokens with action tokens in temporal order: for each video frame o_t, we associate τ consecutive actions $\{a_{t,1}, a_{t,2}, \ldots, a_{t,\tau}\}$, forming a unified sequence $[z_t, a_{t,1}, a_{t,2}, \ldots, a_{t,\tau}, z_{t+1}, \ldots]$ for joint modeling. This design means that predicting K video frames corresponds to generating τK actions, enabling high-frequency control while maintaining efficient video generation.

Mixture-of-Transformer Block. To enable interaction while preserving modality-specific feature spaces, we employ a Mixture-of-Transformers (MOT) architecture [5, 19, 43], where video and action tokens are processed by separate transformer blocks at each layer, then fused via cross-modal attention [5]. At each layer, the video and action streams independently compute their query, key, and value matrices using separate QKV projection matrices, maintaining distinct feature spaces for each modality. To align dimensions for cross-modal fusion, action tokens are first projected to the video dimension via a linear layer, participate in joint self-attention, then projected back to their original dimension via a residual connection that preserves the action-specific representations. This MOT design allows video and action to mutually influence each other through attention while maintaining separate parameterizations, preventing interference between modality-specific feature representations. For action decoding, the final action stream outputs are mapped to low-dimensional action vectors via a linear projection head.

Action Network Initialization. Proper initialization of the action stream is critical for training stability and convergence. We find that training the action network from scratch leads to unstable optimization and slow convergence, as the action

tokens' output distribution initially diverges significantly from the video distribution, disrupting the joint attention mechanism. To address this, we initialize the action network weights by interpolating the pretrained video weights according to the action dimension, then apply a scaling factor $\alpha = \sqrt{d_v/d_a}$ to preserve output variance, where d_v and d_a are the video and action dimensions. This initialization strategy ensures that action tokens start with output distributions comparable to video tokens, stabilizing early-stage training and accelerating convergence.

Variable Chunk Size Training. To enable flexible deployment, we randomly sample the chunk size K from a predefined range during training. By training with variable chunk sizes (e.g., $K \in [1, 8]$), the model learns to generate coherent predictions across different temporal horizons. At inference time, this allows freely selecting the chunk size to balance computational efficiency and planning horizon—larger chunks reduce the number of autoregressive steps but require longer per-step computation, while smaller chunks enable more frequent closed-loop correction. In our experiments, we use $K = 4$ for deployment as a practical trade-off.

Teacher Forcing for Unified Video-Action Training. In §3.2, we formulated both visual dynamics prediction (Eq. 7) and inverse dynamics (Eq. 8) as autoregressive modeling problems, where each prediction conditions on the history of observations and actions. This unified autoregressive formulation enables a natural training strategy: we can treat the interleaved video-action sequence as a single unified sequence and train the model using standard next-token prediction, analogous to language modeling in NLP [75].

Specifically, given an episode with interleaved tokens, we train the model to predict each token conditioned on all preceding tokens in the sequence. This is implemented via teacher forcing: during training, we use ground-truth tokens from the dataset as context for predicting subsequent tokens, rather than model-generated predictions. The causal dependency structure is enforced through attention masking (Figure 3)—each token can only attend to tokens that appear earlier in the temporal sequence.

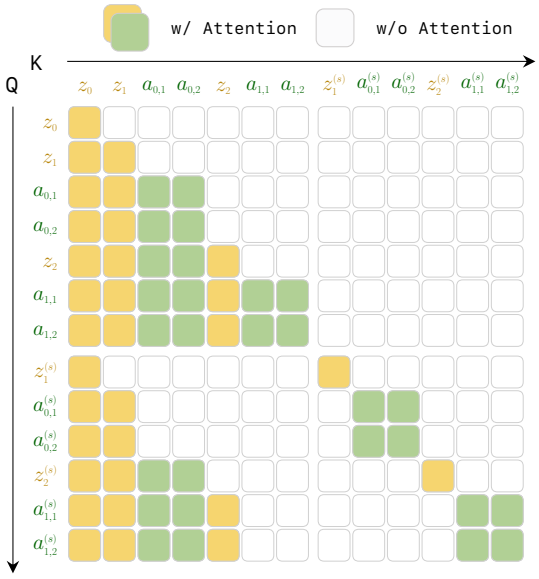


Figure 3. Teacher Forcing Attention Mask: Causal attention mask for unified video-action pretraining. Each token can only attend to preceding tokens in the temporal sequence.

Importantly, teacher forcing is particularly well-suited for robotic manipulation: unlike pure generative modeling where it leads to train-test distribution mismatch, robot policies naturally retrieve real-world observations during deployment, directly matching the training regime. This formulation offers two key benefits: (1) unifying video and action prediction under a single training objective enables end-to-end learning of world dynamics and action inference; (2) by processing episodes in parallel with causal attention masking, we efficiently optimize both components across all timesteps in a single forward pass.

Noisy History Augmentation. The primary bottleneck during inference remains video token generation—the number of video tokens are much larger than action tokens, and each requires multiple denoising steps through the flow matching process. To address this, we introduce a noise augmentation strategy during training that enables *partial denoising* at test time. The key insight is that action prediction does not require fully denoised video representations; the inverse dynamics model can learn to extract action-relevant information from partially noisy video states. Specifically, during training, we randomly augment the video history $z_{\leq t}$ with noise following the same interpolation scheme as flow matching:

$$\tilde{z}_{\leq t} = \begin{cases} (1 - s_{\text{aug}})\epsilon + s_{\text{aug}} z_{\leq t}, & p = 0.5, \quad s_{\text{aug}} \in [0.5, 1], \ \epsilon \sim \mathcal{N}(0, I) \\ z_{\leq t}, & 1 - p = 0.5 \end{cases} \tag{10}$$

This augmentation trains the action decoder to predict actions from partially noisy video representations.

At inference time, this enables a significant speedup: instead of fully denoising video tokens from $s = 0$ to $s = 1$, we only need to denoise to $s = 0.5$, halving the number of denoising steps for video generation while maintaining action prediction quality.

Algorithm 1 KV Cache Inference

Require: Initial observation o_0, chunk size K, KV cache $\mathcal{C}$

$z_0 \leftarrow E(o_0), \mathcal{C} \leftarrow \{z_0\}$
$t \leftarrow 0$
loop
  Sample $\epsilon \sim \mathcal{N}(0, I)$ ▷ Generate video chunk (integrate to $s = 0.5$)
  $\tilde{z}_{t+1:t+K} \leftarrow \epsilon + \int_0^{0.5} v_\theta(z^{(s)}_{t+1:t+K}, s \mid \mathcal{C})\, ds$
  Sample $\epsilon \sim \mathcal{N}(0, I)$ ▷ Generate action chunk (integrate to $s = 1$)
  $a_{t:t+K-1} \leftarrow \epsilon + \int_0^1 v_\psi(a^{(s)}_{t:t+K-1}, s \mid \tilde{z}_{t:t+K}, \mathcal{C})\, ds$
  for $i = t$ **to** $t + K - 1$ **do**
    Execute a_i, receive o_{i+1} ▷ Execute and collect observations
    $z_{i+1} \leftarrow E(o_{i+1})$
  end for
  $\mathcal{C} \leftarrow \mathcal{C} \cup \{z_{t+1:t+K}, a_{t:t+K-1}\}$ ▷ Update KV cache
  $t \leftarrow t + K$
end loop

Training Objective. We jointly optimize both video and action using flow matching with the noisy history augmentation described above. For video tokens z_t, the dynamics loss supervises velocity field prediction conditioned on (potentially noisy) history:

$$\mathcal{L}_{\text{dyn}} = \mathbb{E}_{t,s,z_{t+1},\epsilon}\left[\|v_\theta(z^{(s)}_{t+1}, s, \tilde{z}_{\le t}, a_{<t}|c) - \dot{z}^{(s)}_{t+1}\|^2\right], \tag{11}$$

where $s \in [0, 1]$ is flow time, $z^{(s)}_{t+1} = (1-s)\epsilon + s z_{t+1}$ with $\epsilon \sim \mathcal{N}(0, I)$, $\dot{z}^{(s)}_{t+1} = z_{t+1} - \epsilon$, $\tilde{z}_{\le t}$ is the augmented history (Eq. 10), and c is the language instruction. For action tokens a_t, the inverse dynamics loss conditions on current and next observations:

$$\mathcal{L}_{\text{inv}} = \mathbb{E}_{t,s,a_t,\epsilon}\left[\|v_\psi(a^{(s)}_t, s, \tilde{z}_{\le t+1}, a_{<t}|c) - \dot{a}^{(s)}_t\|^2\right], \tag{12}$$

where $a^{(s)}_t = (1-s)\epsilon + s a_t$ with $\epsilon \sim \mathcal{N}(0, I)$, $\tilde{z}_t, \tilde{z}_{t+1}$ are the (potentially noisy) current and next video tokens, and c is the language instruction. The complete objective is $\mathcal{L} = \mathcal{L}_{\text{dyn}} + \lambda \mathcal{L}_{\text{inv}}$.

3.4 Real-time Deployment & Asynchronous Inference

KV Cache for Efficient Autoregressive Inference. Our autoregressive formulation naturally enables KV cache acceleration during inference. Since each prediction step conditions on the history of observations and actions, we cache the key-value pairs from previous tokens to avoid redundant computation. At each autoregressive step, only the new tokens (current observation and predicted actions) require full attention computation, while cached history tokens are reused. Algorithm 1 describes the complete inference procedure with KV cache.

Asynchronous Prediction and Execution. Despite the efficiency gains from KV cache and partial denoising, autoregressive prediction still incurs non-negligible latency that can violate real-time control requirements. To address this, we introduce an asynchronous inference strategy that pipelines action prediction with execution, effectively hiding prediction latency. We illustrate the difference between synchronous and asynchronous inference in Fig. 4.

The key insight is to overlap computation with execution (Fig. 4B): While the robot executes the current action chunk a_t, the model simultaneously predicts the subsequent action chunk a_{t+1} conditioned on the most recent real observation z_{t-1} (received after the execution of a_{t-1}). For simplicity, we use z_t to denote latent observations (ignoring the video VAE compression) instead of o_t in this section. We discard all history data before timestamp $t-1$ and use the hat notation $\hat{}$ to mark predicted visual content. Consequently, the model's active context is limited to the executed action chunk a_{t-1}, the recent ground-truth observation z_{t-1}, the currently executing action a_t, and its corresponding visual forecast $\hat{z}_t$. A naive auto-regressive implementation (Fig. 4B-1) is to store these tokens into the KV cache and predict $\hat{z}_{t+1}$. However, we observed that such a design frequently leads to open-loop degradation and trajectory drift. Because the video generative model inherently favors temporal smoothness, it tends to "continue" the hallucinated video $\hat{z}_t$ while ignoring the critical physical feedback provided by the real observation z_{t-1}, eventually causing the model to lose its capacity to react to the environment.

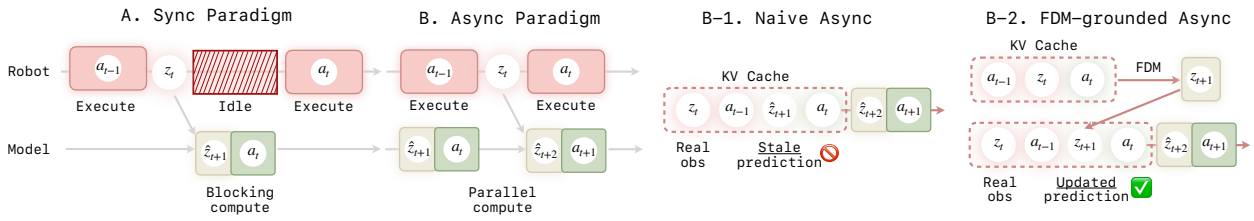


Figure 4. Asynchronous pipeline design overview: The traditional synchronous pipeline (A) suffers from delays caused by blocked computations, while the asynchronous pipeline (B) addresses this issue by enabling parallel computation and execution. However, a naive asynchronous implementation (B-1) relies on outdated visual predictions. In contrast, we improve and refine asynchronous prediction through forward dynamic prediction (B-2), which updates stale predictions with recent real-world observations.

Algorithm 2 Asynchronous Inference and Execution

Require: Initial observation o_0, chunk size K, KV cache $\mathcal{C}$

$z_0 \leftarrow E(o_0)$; $\mathcal{C} \leftarrow \{z_0\}$
$\tilde{z}_{1:K}, a_{0:K-1} \leftarrow \textsc{Predict}(\mathcal{C})$ ▷ Cold Start
ObsQueue $\leftarrow \emptyset$ ▷ Thread-safe queue for incoming real observations
$t \leftarrow 0$
loop
  parallel:
    Branch A: Robot Execution
      async $\textsc{Executor}(a_{t:t+K-1}, \text{ObsQueue})$ ▷ Execute pre-computed actions
    Branch B: Inference with FDM Grounding
      if $t > 0$ **then**
        $o_{t-K+1:t} \leftarrow \text{ObsQueue.dequeue}()$ ▷ Get real observation
        $z_{t-K+1:t} \leftarrow E(o_{t-K+1:t})$
        $\mathcal{C} \leftarrow \mathcal{C} \cup \{z_{t-K+1,t}, a_{t-K:t-1}\}$ ▷ Cache feedback
      end if
      $\mathcal{C}_{\text{tmp}} \leftarrow \mathcal{C} \cup \{a_{t:t+K-1}\}$ ▷ Cache action being executed
      $z_{t+1:t+K} \leftarrow \text{FDM}(\mathcal{C}_{\text{tmp}})$ ▷ Imagine visual outcome
      $\mathcal{C}_{\text{tmp}} \leftarrow \mathcal{C}_{\text{tmp}} \cup \{z_{t+1:t+K}\}$ ▷ Update cache
      $\tilde{z}_{t+K+1:t+2K}, a_{t+K:t+2K-1} \leftarrow \textsc{Predict}(\mathcal{C}_{\text{tmp}})$
      $t \leftarrow t + K$
end loop

To mitigate this, we introduce a Forward Dynamics Model (FDM) grounded step into our inference pipeline (Fig. 4B-2). Instead of relying on stale forecasts, we replace it by executing a forward dynamics pass: the model uses the recent feedback z_{t-1} and "imagines" the resulting visual state z_t after applying action a_t. By caching this feedback-grounded prediction instead of a stale forecast, we force the model to re-align with environmental feedback before predicting z_{t+1}. This design enhances our asynchronous algorithm into a robust closed-loop system, enabling the robot to effectively perceive and react to real-world changes.

Algorithm 2 formalizes this asynchronous pipeline. During post training, we additionally incorporate a forward dynamics prediction loss:

$$\mathcal{L}_{\text{fdm}} = \mathbb{E}_{t,s,\hat{z}_{t+1},\epsilon}\left[\|v_\psi(\tilde{z}_{t+1}, s, z_t, a_t, \tilde{z}_{<t}, \hat{a}_{<t}|c) - \dot{z}^{(s)}_{t+1}\|^2\right], \tag{13}$$

4 Experiments

4.1 Dataset Curation and Preprocessing

We curate a large-scale training corpus by aggregating existing public robot manipulation datasets. All datasets undergo preprocessing to ensure consistency in data format and annotation quality, and are split into 90% training and 10% validation per dataset to monitor training dynamics.

Unified Action Representation. To achieve cross-embodiment generalization, we define a universal action interface to adapt to different datasets. We use a dual-arm representation where each robotic arm is characterized by both end-effector pose (EEF) and joint angles. The end-effector pose consists of XYZ coordinates and a rotation quaternion (7 dimensions). For joint angles, we support a maximum of 7 degrees of freedom for single-arm embodiments; if a robot has fewer than 7 joint dimensions, we pad the missing dimensions with zeros to maintain a unified 7-dimensional representation. Each arm also has one gripper action dimension. Therefore, the total action dimensionality for dual-arm systems is: $7_{\text{EEF}} + 7_{\text{joints}} + 1_{\text{gripper}}$ per arm, resulting in $(7 + 7 + 1) \times 2 = 30$ dimensions.

Training Data Composition. We aggregate data from six sources spanning diverse embodiments, environments, and task categories:

- **Agibot** [2]: Large-scale dataset with diverse manipulation tasks from mobile manipulators.
- **RoboMind** [80]: Multi-embodiment manipulation demonstrations.
- **InternData-A1** [73]: Large-scale simulation dataset for sim-to-real transfer.
- **OXE** [53]: Multi-embodiment dataset; we use the OpenVLA subset.
- **UMI Data** [18, 45, 48, 51, 60, 91]: Human demonstration dataset collected via universal manipulation interface[1], excluding DexUMI.
- **RoboCOIN** [83]: Cross-embodiment bimanual robotics data.

In total, our training corpus comprises approximately **16K** hours of robot manipulation data across diverse tasks and environments, including internally collected demonstrations.

4.2 Implementation & Training Details

Implementation Details. We use Wan2.2-5B as the backbone for the video stream, with hidden dimension $d_v = 3072$ and 30 transformer layers. The action stream shares the same depth but uses a reduced hidden dimension $d_a = 768$ ($4\times$ smaller), resulting in approximately 350M additional parameters and a total model size of 5.3B parameters. Both streams employ RoPE positional encoding and are connected via the MoT architecture described in §3.3. We adopt the Wan2.2 causal VAE for tokenization with a $4 \times 16 \times 16$ (temporal $\times$ height $\times$ width) compression ratio, combined with a patchify operation that further reduces spatial dimensions by 2. The encoded views are concatenated along the width dimension, resulting in a total of $N = 192$ spatial tokens per frame. The action encoder ϕ and decoder are implemented as single-layer MLPs with hidden dimension 256. We normalize actions using per-dimension quantile normalization statistics computed from the training set. Task instructions are encoded using a frozen T5 text encoder [59] and injected via cross-attention. During training, chunk size K is randomly sampled from $[1, 4]$.

For inference, we use Euler solver with 3 steps for video tokens (integrating to $s = 0.6$) and 10 steps for action tokens (integrating to $s = 1.0$). Video CFG scale is set to 5.0, while action CFG scale is set to 1.0. During training, noise augmentation is applied with probability $p = 0.5$ and $s_{\text{aug}} \sim \text{Uniform}[0.5, 1.0]$. Following LLM practices, we pack multiple episodes into long sequences (up to 10K tokens) with attention masks.

[1] https://umi-data.github.io/

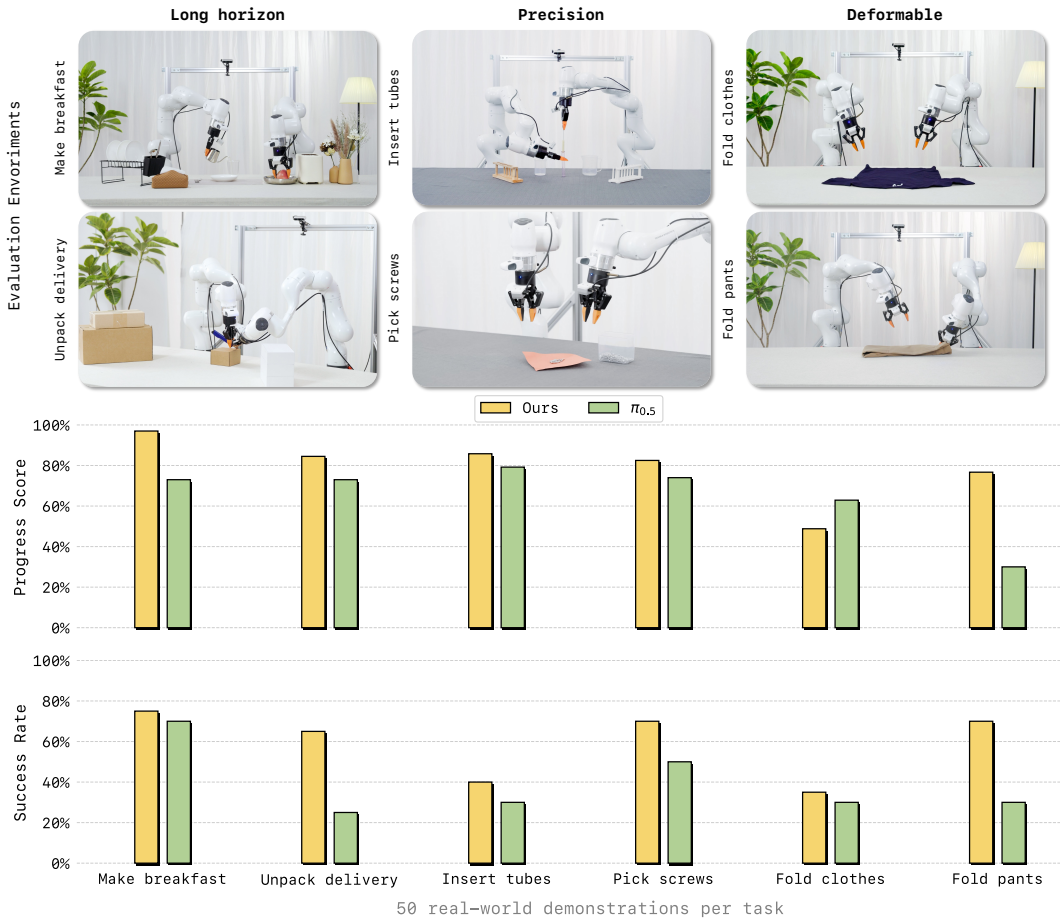


Figure 5. Real-world deployment results. We evaluate LingBot-VA on six manipulation tasks across three categories: long-horizon tasks (Make Breakfast, Pick Screws), precision tasks (Insert Tubes, Unpack Delivery), and deformable & articulated object manipulation (Fold Clothes, Fold Pants). Our method achieves state-of-the-art performance on both metrics.

Pre-Training Details. We pretrain LingBot-VA on the curated dataset for 1.4T tokens. We use the AdamW optimizer with peak learning rate 1×10^{-4}, weight decay 0.01, and cosine annealing schedule with linear warmup. Training is conducted in bfloat16 mixed precision with gradient clipping at 2.0. We apply classifier-free guidance with text dropout rate 0.1. The loss weight λ for inverse dynamics is set to 1. The dataset is sampled uniformly across all sources to ensure balanced learning. We monitor convergence using flow matching loss on the validation set. We use uniform SNR sampler for video model. For both video and action model, we use a uniform SNR sampler.

Post-Training Details. While the pretrained model exhibits zero-shot generalization to seen embodiments, adapting to novel robot platforms requires a small amount of task-specific data. We find that post-training with as few as 50 demonstrations is sufficient for effective deployment. We use a reduced learning rate of 1×10^{-5} and train for 3K steps, which yields robust performance. Alternatively, a higher learning rate of 1×10^{-4} with 1K steps also produces reasonable results, though slightly inferior, offering a faster adaptation option when computational resources are limited.

4.3 Main Results

4.3.1 Real-world Deployment

Experimental Setup. To validate the real-world effectiveness of LingBot-VA, we deploy our model on a physical robot platform and evaluate across six diverse manipulation tasks spanning three challenging categories. (1) ***Long-horizon Tasks:*** We evaluate on *Make Breakfast* and *Unpack Delivery*, which require sequential multi-step reasoning and sustained



Figure 6. Detailed task progressions and key execution steps of the six real-world tasks. Each task involves a sequence of manipulation primitives, with scoring criteria detailed in Tables S2 through S4.

task execution over extended time horizons. (2) ***Precision Tasks:*** We test on *Insert Tubes* and *Pick Screws*, demanding accurate positioning and fine-grained motor control for successful completion. (3) ***Deformable Objects:*** We include *Fold Clothes* and *Fold Pants*, which involve manipulating non-rigid materials that present unique control challenges. The detailed task procedures are summarized in Fig. 6. These tasks are only collected with **50 real-world demos** for model training. We finetune the model for 500 steps with a learning rate of 1×10^{-4} and a sequence length of 150,000.

Results. As shown in Fig. 5, `LingBot-VA` consistently achieves state-of-the-art performance across all six tasks and both evaluation metrics (success rate and progress score), substantially outperforming strong baseline $\pi_{0.5}$.

We highlight several key observations that validate our design choices: (1) The superior performance on *long-horizon tasks* demonstrates that our video-action world model possesses strong temporal memory capabilities. By jointly modeling video and action sequences, the model effectively maintains task context over extended horizons, enabling coherent multi-step reasoning without losing track of intermediate goals. (2) The strong results on *precision tasks* validate the effectiveness of our unified latent space design. By aligning video and action representations within a shared embedding space, our model achieves tighter coupling between visual perception and motor control, resulting in more accurate and fine-grained action predictions. (3) The robust performance on *deformable objects* highlights the value of video generation as implicit guidance. The generated video futures provide rich predictive signals about object dynamics and state transitions, which inform the action model to produce more physically plausible manipulation trajectories for challenging non-rigid materials.

These results collectively demonstrate that our video-action world model effectively transfers to real-world deployment, exhibiting robust performance across diverse manipulation scenarios.

4.3.2 Simulation Evaluation

Experimental Setup. We evaluate `LingBot-VA` on two widely-used simulation benchmarks: RoboTwin 2.0 [15] and LIBERO [47], covering diverse manipulation tasks across different robot embodiments.

(1) In RoboTwin 2.0, we adopt a multi-task training setup [5] where all models are trained on 2,500 demonstrations collected in clean scenes (50 per task) plus 25,000 demonstrations from heavily randomized scenes (500 per task). We downsample the original 50 Hz video to 12.5 Hz while maintaining the action frequency at 50Hz. The model is trained for 50K steps with a learning rate of 1×10^{-5}. To facilitate a clearer comparison of performance, we categorize the 50 RoboTwin tasks according to their horizons (e.g., *Place Dual Shoes* has two steps, and *Stack Blocks Three* has three steps). The detailed horizons are listed in Tab. S1.

(2) In LIBERO, we train our model on four LIBERO suites: LIBERO-Spatial, LIBERO-Object, LIBERO-Goal, and LIBERO-Long. Each suite contains 10 tasks with 50 demonstrations per task (500 total). Following OpenVLA [34], we filter unsuccessful demonstrations before training. The model is finetuned for 4K steps with a learning rate of 1×10^{-5} and a sequence length of 1×10^{5}. Specifically, we report the average success rate over three random seeds, with each seed comprising 500 evaluation trials (totally $3 \times 500 = 1500$) for every task suite.

Results RoboTwin 2.0 is a challenging bimanual manipulation benchmark featuring over 50 tasks that require coordinated dual-arm control. Unlike single-arm benchmarks, RoboTwin tasks demand precise synchronization between

Table 1. Evaluation on RoboTwin 2.0 Simulation (Easy vs Hard, 50 tasks). RoboTwin 2.0 is a challenging bimanual manipulation benchmark requiring coordinated dual-arm control. Easy uses fixed initial configurations while Hard involves randomized object poses and scene layouts. * Results for X-VLA are adopted from Motus [5]. Improvements in parentheses indicate gains over the second-best method (underlined).

	X-VLA* [92]		π_0 [7]		$\pi_{0.5}$ [29]		Motus [5]		`LingBot-VA` (Ours)	
Metric	Easy	Hard	Easy	Hard	Easy	Hard	Easy	Hard	Easy	Hard
Average $_{\text{Horizon = 1}}$	81.6	82.5	66.5	61.6	85.1	80.2	<u>91.0</u>	<u>90.6</u>	**94.18** (+3.2)	**93.56** (+3.0)
Average $_{\text{Horizon = 2}}$	59.3	55.9	66.1	54.7	79.3	73.0	<u>85.2</u>	<u>80.9</u>	**90.35** (+5.2)	**86.95** (+6.1)
Average $_{\text{Horizon = 3}}$	61.2	66.0	61.6	50.2	78.6	67.4	<u>85.0</u>	<u>84.2</u>	**93.22** (+8.2)	**93.28** (+9.1)
Average $_{\text{50 Tasks}}$	72.9	72.8	65.9	58.4	82.7	76.8	<u>88.7</u>	<u>87.0</u>	**92.93** (+4.2)	**91.55** (+4.6)

Table 2. Evaluation on LIBERO benchmarks. LIBERO tests manipulation across four task suites: Spatial, Object, Goal, and Long-horizon. Our method achieves new state-of-the-art on LIBERO-Object (99.6%), LIBERO-Long (98.5%), LIBERO-Spatial (98.5%), and overall average (98.5%). Baseline results are adopted from [92].

Methods	LIBERO				
	Spatial	Object	Goal	Long	Avg
Octo [71]	78.9	85.7	84.6	51.1	75.1
Seer [72]	-	-	-	87.7	-
MoDE [61]	-	-	-	94.0	-
SuSIE [9]	-	-	-	76.3	-
SpatialVLA [58]	88.2	89.9	78.6	55.5	78.1
TraceVLA [93]	84.6	85.2	75.1	54.1	74.8
CoT-VLA [89]	87.5	91.6	87.6	69.0	81.1
ThinkAct [28]	88.3	91.4	87.1	70.9	84.4
SmolVLA [66]	93.0	94.0	91.0	77.0	88.8
CronusVLA [37]	97.3	99.6	96.9	94.0	97.0
FLOWER [62]	97.1	96.7	95.6	93.5	95.7
GR00T-N1 [6]	94.4	97.6	93.0	90.6	93.9
π_0 [7]	96.8	98.8	95.8	85.2	94.1
π_0+FAST [57]	96.4	96.8	88.6	60.2	85.5
OpenVLA [34]	84.7	88.4	79.2	53.7	76.5
OpenVLA-OFT [32]	97.6	98.4	**97.9**	94.5	97.1
DD-VLA [44]	97.2	98.6	97.4	92.0	96.3
UniVLA [77]	95.4	98.8	93.6	94.0	95.4
X-VLA [92]	98.2	98.6	97.8	97.6	98.1
LingBot-VA (Ours)	$\mathbf{98.5 \pm 0.3}$	$\mathbf{99.6 \pm 0.3}$	97.2 ± 0.2	$\mathbf{98.5 \pm 0.5}$	**98.5**

two manipulators, making it significantly more difficult for policy learning. We evaluate under both *Easy* (fixed initial configurations) and *Hard* (varied object poses and scene layouts) settings. As shown in Tab. 1, LingBot-VA achieves an average success rate of 92.9% (Easy) and 91.6% (Hard), substantially outperforming prior methods including π_0, $\pi_{0.5}$, X-VLA, and Motus. Notably, the improvement becomes more pronounced for longer-horizon tasks: at Horizon = 3, our method achieves gains of +8.2% (Easy) and +9.1% (Hard) over the second-best approach. This suggests that our autoregressive mechanism effectively maintains long-range temporal memory, enabling more robust performance as task complexity increases.

We further evaluate on LIBERO benchmark (Tab. 2). On LIBERO, we obtain an average success rate of 98.5%, with particularly strong performance on LIBERO-Long (98.5%). These results establish new state-of-the-art performance in average success rates among foundational VLAs, demonstrating the effectiveness of our video-action world model for generalist robot control.

4.4 Ablation

Asynchronous v.s. synchronous. We compare our asynchronous video-action generation with a synchronous baseline on RoboTwin tasks. As shown in Tab. 3, both approaches achieve comparable success rates, but our asynchronous method completes tasks **2× faster** by predicting future video and action sequences while executing current actions. This validates that asynchronous generation maintains task performance while significantly improving inference efficiency.

Pretrained LingBot-VA **v.s. WAN**. To validate the design choices in our video-action architecture, we conduct a controlled ablation study comparing our pretrained LingBot-VA model with WAN (Wan2.2-5B) as the initialization baseline for fine-tuning on RoboTwin tasks. Both models are fine-tuned on the same RoboTwin dataset using identical post-training procedures (50 task-specific demonstrations, learning rate 1×10^{-5}, 3K steps).

As shown in Tab. 3, our pretrained LingBot-VA model substantially outperforms WAN fine-tuning across both Easy and Hard settings. Specifically, LingBot-VA achieves an average success rate of 92.10% (Easy) and 91.12% (Hard), while WAN fine-tuning yields significantly lower performance. This performance gap highlights the effectiveness of our joint video-action pretraining strategy, which endows the model with rich visual-motor priors that facilitate fast

Table 3. Ablation studies on RoboTwin 2.0 (Easy). We ablate three design choices: world modeling (AR vs. bidirectional), deployment mode (async vs. sync), and pretraining (Ours vs. WAN).

Ablation	Setting	Easy$_{\text{all}}$	Easy $_{\text{Horizon = 1}}$	Easy $_{\text{Horizon = 2}}$	Easy $_{\text{Horizon = 3}}$
Baseline	LingBot-VA (Ours)	92.9	94.2	90.4	93.2
Deployment	FDM-grounded Async	90.4	92.5	87.7	85.6
	Naive Async	74.3	83.3	70.3	32.9
Pretrain	WAN	80.6	84.9	76.3	67.6

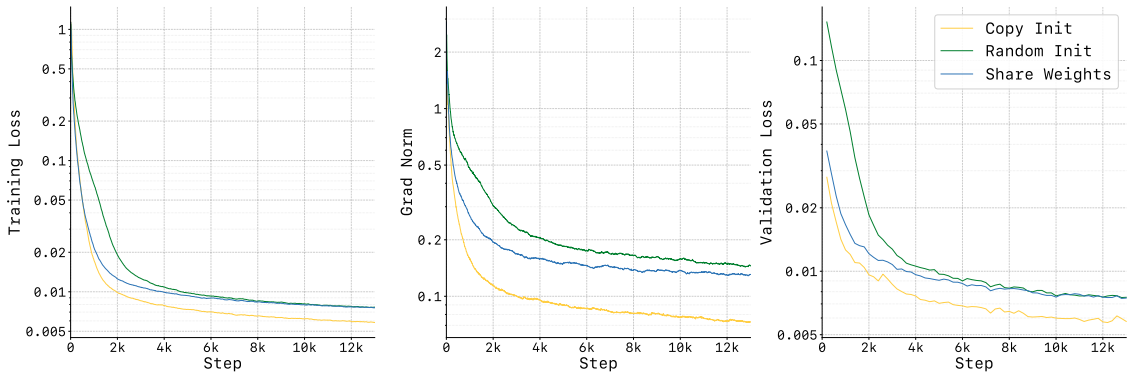


Figure 7. Training dynamic comparison between different action network initialization streategy: Random initialization leads to unstable optimization (high gradient norms) and slow convergence. Although re-using video network weights stabilizes training, the resulting performance is not optimal. Our approach, which initializes by copying pretrained video weights with proper scaling, proves to be the most effective, ensuring smooth training dynamics and faster convergence.

adaptation to complex bimanual manipulation tasks.

Action Network Initialization. Proper initialization of the action stream is critical for training stability and convergence. We compare our curated initialization strategy (Section 3.3) with naive random initialization.

As shown in Fig. 7, random initialization from scratch exhibits volatile training dynamics with significantly slower convergence. This instability arises because action tokens' output distribution initially diverges dramatically from the video distribution, disrupting the joint attention mechanism in our unified architecture. In contrast, our curated initialization strategy—where action network weights are initialized by interpolating pretrained video weights with a scaling factor $\alpha = \sqrt{d_v/d_a}$—produces smooth convergence and substantially lower loss.

4.5 Analysis

4.5.1 Sample Efficiency

We investigate the data efficiency by exploring how LingBot-VA performs with limited post-training data compared to $\pi_{0.5}$. We conduct this evaluation on both real-world and simulation settings: the "Make Breakfast" long-horizon task and RoboTwin 2.0 Easy benchmarks, allowing us to assess data efficiency across diverse manipulation scenarios.

As shown in Fig. 8, our method consistently outperforms $\pi_{0.5}$ across all data regimes on both real-world and simulation tasks. In the low-data regime (10 demonstrations), LingBot-VA achieves 15.6% higher progress score than $\pi_{0.5}$ on the "Make Breakfast" task and 10.3% higher on RoboTwin 2.0 Easy, demonstrating superior sample efficiency. These results demonstrate that our method learns more effectively from limited data across diverse manipulation scenarios.

We attribute this superior data efficiency to our video-action world model design. The jointly pretrained video generation backbone provides rich visual priors about physical dynamics and object interactions, which serve as implicit regularization during post-training. This allows the action model to leverage the world knowledge encoded in the video stream, effectively reducing the sample complexity required for adapting to new tasks. In contrast, VLA models like $\pi_{0.5}$ lack explicit modeling of visual dynamics and thus have no structured dynamics priors to guide learning, requiring

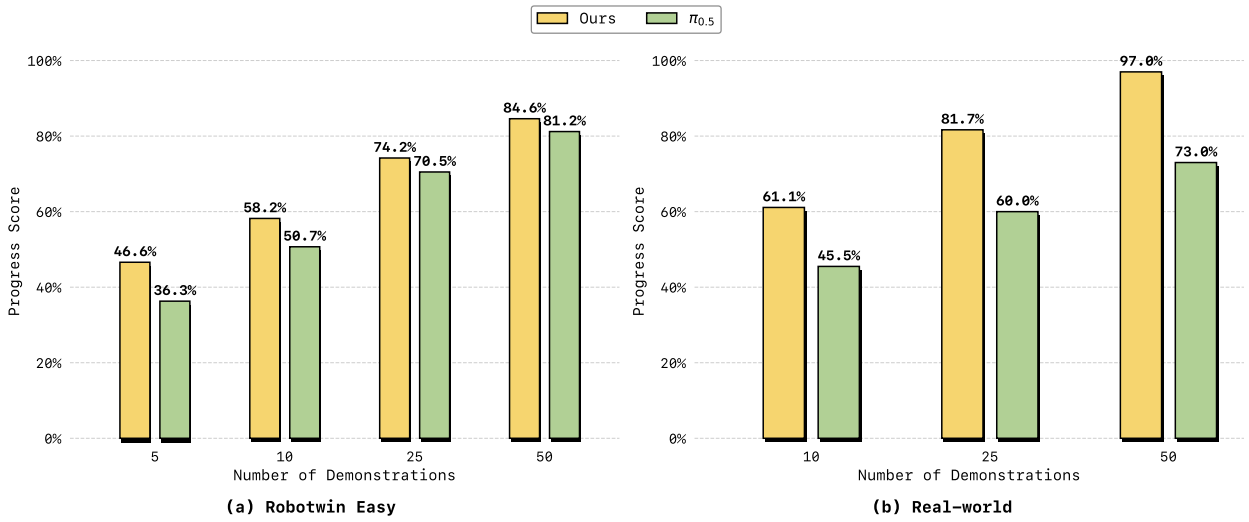


Figure 8. Sample efficiency comparison. LingBot-VA consistently outperforms $\pi_{0.5}$ across various data regimes on the "Make Breakfast" task, demonstrating superior data efficiency in the post-training stage.

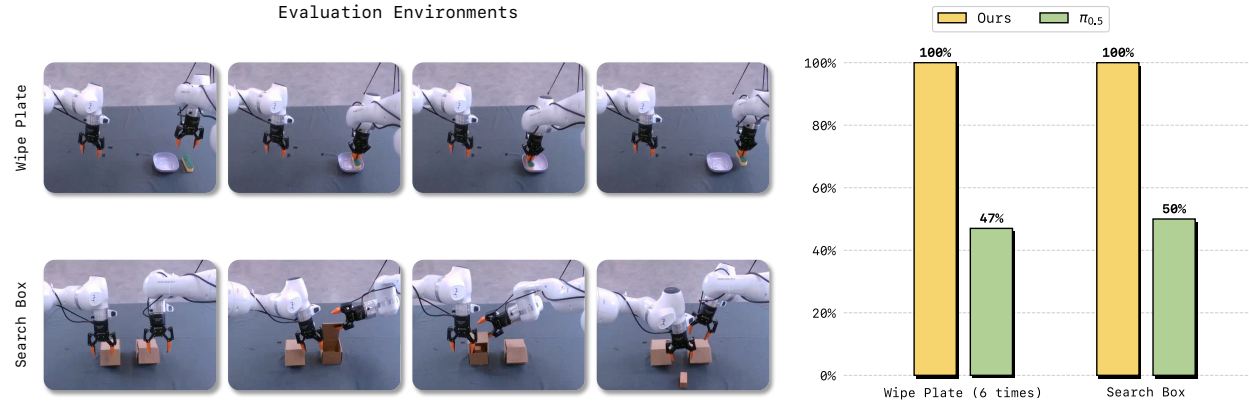


Figure 9. Temporal memory evaluation. Left: Success rates on two memory tasks (Wipe Plate and Search Box). LingBot-VA significantly outperforms $\pi_{0.5}$ on both tasks, demonstrating superior temporal state tracking ability. Right: Visualization of evaluation environments.

more demonstrations to learn task-specific behaviors from scratch.

4.5.2 Temporal Memory

We design the following tasks that explicitly require maintaining state information across time to evaluate our model's temporal memory capabilities, as shown in Figure 9.

1. ***Wipe Plate***—the robot must wipe a plate exactly six times, requiring it to count and remember repeated actions.
2. ***Search Box***—Two boxes (left and right) are in the scene, with only one containing a block. The robot opens them sequentially from right to left. In data collection, the block is equally likely to be in either box; at test time, it is always in the left box. Without memory, after finding the right box empty, the model has a 50% chance of re-opening it. With memory, it proceeds to search the left box.

As shown in Fig. 9(a), LingBot-VA substantially outperforms $\pi_{0.5}$ on both memory tasks. We attribute this to the autoregressive nature of our world model: during training, teacher forcing conditions predictions on full history; during inference, KV-cache naturally preserves all historical information for persistent memory.

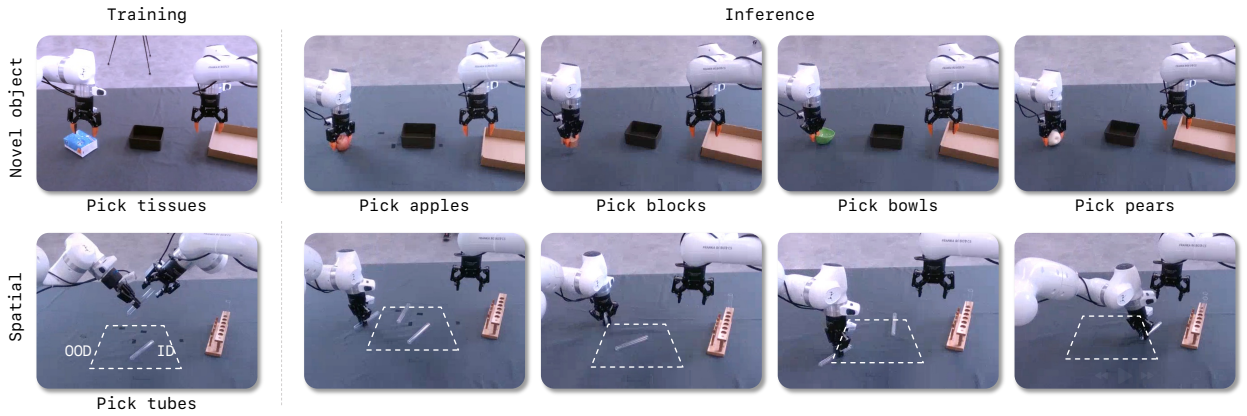


Figure 10. Novel object and spatial generalization. LingBot-VA successfully generalizes to objects with varying shapes, textures, and positions.

4.5.3 Generalization

We evaluate generalization along two axes:

1. ***Novel Object Generalization***—trained on pick-and-place with a single object, tested on different objects with varying shapes and textures;
2. ***Spatial Generalization***—trained with fixed object positions in a localized region(denoted as in-distribution (ID)), tested on random placements especially in out-of-distribution (OOD) regions.

As shown in Fig. 10, our method demonstrates a stronger generalization in both both novel object and the out-of-distribution position. The world model learns transferable visual representations through video prediction, capturing object-agnostic physical priors that transfer to novel scenarios.

5 Related Work

Vision-Language-Action Policies. Recent advancements in Embodied AI have witnessed a paradigm shift toward large-scale Vision-Language-Action (VLA) policies. By leveraging web-scale knowledge and diverse robot demonstrations, models such as $\pi_{0.5}$ [29], GR-3 [39], and GR00T-N1 [6] achieve remarkable generalizability across various manipulation tasks without relying on hand-crafted rules, modular priors, or restricted action abstractions, enabling a more direct and expressive end-to-end mapping from perception to control. These policies typically employ pre-trained Vision-Language Models (VLMs) as foundational backbones [6, 7, 11, 29, 34, 39, 86, 92], which provide superior cross-modal understanding and more generalizable action distributions compared to task-specific imitation policies like ACT [90] or Diffusion Policy [17]. Efforts have been further devoted to improving the deployability through lightweight backbones [49, 62, 66], efficient tokenization [57], real-time inference [8, 10, 69], or fine-tuning schemes [30, 32, 38]. However, despite their prowess in semantic reasoning, a fundamental limitation persists: the pre-training objectives and data distributions of standard VLMs largely overlook the fine-grained system dynamics and low-level trajectories essential for precision manipulation. While supervised fine-tuning on expensively collected large-scale robot datasets allows these models to approximate the marginal action distribution [3, 31, 53], they remain deficient in capturing the underlying transition dynamics—specifically, how the physical state of the environment should evolve and will evolve. Furthermore, most current VLA methods formulate control as a purely reactive mapping from instantaneous observations to actions. This approach inherently fails to account for the historical context necessary to resolve ambiguities in non-Markovian environments. Additionally, the static image-text pre-training inherent in VLMs fails to instill essential temporal priors. Even when augmented with memory modules [37, 64, 67], such models remain unable to reason about the causal and sequential nature of physical interactions. To address these shortcomings, recent research has pivoted towards generalist robot policies grounded in world models and generative video modeling [1, 40, 96]. Our LingBot-VA unifies autoregressive video prediction with action decoding, ensuring that robot control is explicitly conditioned on learned

video dynamics. By integrating these two objectives, our framework maintains strict causal consistency, allowing the policy to synchronize physical execution with the predicted visual evolution of the environment.

World Models for Robotic Control. Inspired by human reliance on intuitive physics to anticipate environmental changes, world models aim to facilitate effective planning by predicting future dynamics. Existing approaches are generally categorized into three groups based on their state representations. The first category operates in latent space [36, 41, 63, 79], encoding task-relevant features into compact vectors to predict evolution via probabilistic [36, 52, 82] or deterministic methods [63, 84]. The second category utilizes 3D point clouds [65, 68, 76], leveraging Graph Neural Networks (GNNs) to predict geometric evolution [87, 88], which is particularly effective for manipulating deformable objects [76, 88]. The third category focuses on 2D pixel space, directly predicting future keyframes or video sequences [21, 33, 94, 95]. Our work aligns with this third category. Within this domain, approaches range from co-training with video generation for representation learning [14, 40, 96] to serving as simulators for policy learning or evaluation [70]. Our research specifically targets methods that predict future frames during execution to condition action generation. However, prior video-conditioned methods predominantly rely on open-loop generation [21, 94], presenting two significant challenges. First, the misalignment between generated videos and real-world dynamics, coupled with cumulative drift from execution errors, often leads to suboptimal performance. Second, the computational intensity of video generation imposes high latency, severely hindering real-time inference. Our method leverages KV Cache and causal masking to continuously update the model's memory with real-world observations. This effectively transitions the system to a closed-loop control mechanism, mitigating error accumulation in long-horizon tasks. Furthermore, we introduce a partial denoising strategy, enabling action generation from intermediate representations without waiting for fully denoised frames.

6 Conclusion

We present `LingBot-VA`, an autoregressive diffusion framework that unifies video dynamics prediction and action inference for robotic manipulation. By interleaving video and action tokens within a Mixture-of-Transformers architecture, our model captures the causal structure of physical interactions while enabling closed-loop control through continuous integration of real-world observations. Extensive evaluation demonstrates strong performance across simulation benchmarks (92.0% on RoboTwin 2.0, 98.5% on LIBERO) and real-world deployment, achieving over 20% improvement on challenging tasks compared to $\pi_{0.5}$ with only 50 demonstrations for adaptation. These results suggest that autoregressive video-action world modeling provides a principled foundation for learning generalizable manipulation policies, offering a compelling alternative to reactive VLA paradigms.

Future Work. Future directions include developing more efficient video compression schemes to reduce computational overhead, and incorporating multi-modal sensory inputs (tactile, force, audio) for more robust manipulation in tasks with complex contact dynamics.

Acknowledgment. We thank Kecheng Zheng for insightful discussions and Wei Wu for valuable assistance with dataset preparation. We also thank Fangyi Xu and Yishu Shen for their help with the post-training data collection.

References

[1] 1X Technologies. 1x world model: From video to action. https://www.1x.tech/discover/world-model-self-learning, 2025. Accessed 2026-01-18.

[2] AgiBot-World-Contributors, Qingwen Bu, Jisong Cai, Li Chen, Xiuqi Cui, Yan Ding, Siyuan Feng, Shenyuan Gao, Xindong He, Xuan Hu, Xu Huang, Shu Jiang, Yuxin Jiang, Cheng Jing, Hongyang Li, et al. Agibot world colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems. *arXiv preprint arXiv:2503.06669*, 2025.

[3] Jose Barreiros, Andrew Beaulieu, Aditya Bhat, Rick Cory, Eric Cousineau, Hongkai Dai, Ching-Hsin Fang, Kunimatsu Hashimoto, Muhammad Zubair Irshad, Masha Itkina, et al. A careful examination of large behavior models for multitask dexterous manipulation. *arXiv preprint arXiv:2507.05331*, 2025.

[4] Homanga Bharadhwaj, Debidatta Dwibedi, Abhinav Gupta, Shubham Tulsiani, Carl Doersch, Ted Xiao, Dhruv Shah, Fei Xia, Dorsa Sadigh, and Sean Kirmani. Gen2act: Human video generation in novel scenarios enables generalizable robot manipulation. In *Conference on Robot Learning (CoRL)*, 2024.

[5] Hongzhe Bi, Hengkai Tan, Shenghao Xie, Zeyuan Wang, Shuhe Huang, Haitian Liu, Ruowen Zhao, Yao Feng, Chendong Xiang, Yinze Rong, Hongyan Zhao, Hanyu Liu, Zhizhong Su, Lei Ma, Hang Su, et al. Motus: A unified latent action world model. *arXiv preprint arXiv:2512.13030*, 2025.

[6] Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Jim Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

[7] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, et al. π_0: A vision-language-action flow model for general robot control. In *Robotics: Science and Systems*, 2025.

[8] Kevin Black, Manuel Y. Galliker, and Sergey Levine. Real-time execution of action chunking flow policies. *arXiv preprint arXiv:2506.07339*, 2025.

[9] Kevin Black, Mitsuhiko Nakamoto, Pranav Atreya, Homer Walke, Chelsea Finn, Aviral Kumar, and Sergey Levine. Zero-shot robotic manipulation with pretrained image-editing diffusion models. In *Int. Conf. Learn. Represent.*, 2024.

[10] Kevin Black, Allen Z Ren, Michael Equi, and Sergey Levine. Training-time action conditioning for efficient real-time chunking. *arXiv preprint arXiv:2512.05964*, 2025.

[11] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning (CoRL)*, 2023.

[12] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, et al. Rt-1: Robotics transformer for real-world control at scale. In *Robotics: Science and Systems*, 2023.

[13] 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, et al. Genie: Generative interactive environments. In *Int. Conf. Mach. Learn.*, 2024.

[14] Qingwen Bu, Yanting Yang, Jisong Cai, Shenyuan Gao, Guanghui Ren, Maoqing Yao, Ping Luo, and Hongyang Li. Learning to act anywhere with task-centric latent actions. *arXiv preprint arXiv:2502.14420*, 2025.

[15] Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, Weiliang Deng, Yubin Guo, Tian Nian, Xuanbing Xie, Qiangyu Chen, et al. Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025.

[16] Yi Chen, Yuying Ge, Weiliang Tang, Yizhuo Li, Yixiao Ge, Mingyu Ding, Ying Shan, and Xihui Liu. Moto: Latent motion token as the bridging language for robot manipulation. In *Int. Conf. Comput. Vis.*, 2025.

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

[18] Cheng Chi, Zhenjia Xu, Chuer Pan, Eric Cousineau, Benjamin Burchfiel, Siyuan Feng, Russ Tedrake, and Shuran Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. In *Robotics: Science and Systems*, 2024.

[19] Chaorui Deng, Deyao Zhu, Kunchang Li, Chenhui Gou, Feng Li, Zeyu Wang, Shu Zhong, Weihao Yu, Xiaonan Nie, Ziang Song, Guang Shi, and Haoqi Fan. Emerging properties in unified multimodal pretraining. *arXiv preprint arXiv:2505.14683*, 2025.

[20] Yilun Du, Mengjiao Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Joshua B. Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. In *Adv. Neural Inform. Process. Syst.*, 2023.

[21] 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*, 36:9156–9172, 2023.

[22] Yao Feng, Hengkai Tan, Xinyi Mao, Guodong Liu, Shuhe Huang, Chendong Xiang, Hang Su, and Jun Zhu. Vidar: Embodied video diffusion model for generalist bimanual manipulation. *arXiv preprint arXiv:2507.12898*, 2025.

[23] Google DeepMind. Veo: A text-to-video generation system. *Google DeepMind Technical Report*, 2025.

[24] 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 *Adv. Neural Inform. Process. Syst.*, 2024.

[25] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 2025.

[26] Nicklas Hansen, Hao Su, and Xiaolong Wang. Td-mpc2: Scalable, robust world models for continuous control. In *Int. Conf. Learn. Represent.*, 2024.

[27] Yucheng Hu, Yanjiang Guo, Pengchao Wang, Xiaoyu Chen, Yen-Jen Wang, Jianke Zhang, Koushil Sreenath, Chaochao Lu, and Jianyu Chen. Video prediction policy: A generalist robot policy with predictive visual representations. In *Int. Conf. Mach. Learn.*, 2025.

[28] Chi-Pin Huang, Yueh-Hua Wu, Min-Hung Chen, Yu-Chiang Frank Wang, and Fu-En Yang. Thinkact: Vision-language-action reasoning via reinforced visual latent planning. In *Adv. Neural Inform. Process. Syst.*, 2025.

[29] Physical Intelligence et al. $\pi_{0.5}$: A generalist robot policy with flow matching and world models. In *Conference on Robot Learning (CoRL)*, 2025.

[30] Dong Jing, Gang Wang, Jiaqi Liu, Weiliang Tang, Zelong Sun, Yunchao Yao, Zhenyu Wei, Yunhui Liu, Zhiwu Lu, and Mingyu Ding. Mixture of horizons in action chunking. *arXiv preprint arXiv:2511.19433*, 2025.

[31] Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, Peter David Fagan, Joey Hejna, Masha Itkina, Marion Lepert, Yecheng Jason Ma, et al. Droid: A large-scale in-the-wild robot manipulation dataset. In *Robotics: Science and Systems*, 2024.

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

[33] Moo Jin Kim, Yihuai Gao, Tsung-Yi Lin, Yen-Chen Lin, Yunhao Ge, Grace Lam, Percy Liang, Shuran Song, Ming-Yu Liu, Chelsea Finn, et al. Cosmos policy: Fine-tuning video models for visuomotor control and planning. *arXiv preprint arXiv:2601.16163*, 2026.

[34] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, et al. Openvla: An open-source vision-language-action model. In *Conference on Robot Learning (CoRL)*, 2024.

[35] Kuaishou. Kling ai. https://klingai.kuaishou.com/, 2024.

[36] Chenchang Li, Zihao Ai, Tong Wu, Xiaosa Li, Wenbo Ding, and Huazhe Xu. Deformnet: Latent space modeling and dynamics prediction for deformable object manipulation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 14770–14776. IEEE, 2024.

[37] Hao Li, Shuai Yang, Yilun Chen, Yang Tian, Xiaoda Yang, Xinyi Chen, Hanqing Wang, Tai Wang, Feng Zhao, Dahua Lin, et al. Cronusvla: Transferring latent motion across time for multi-frame prediction in manipulation. *arXiv preprint arXiv:2506.19816*, 2025.

[38] Haozhan Li, Yuxin Zuo, Jiale Yu, Yuhao Zhang, Zhaohui Yang, Kaiyan Zhang, Xuekai Zhu, Yuchen Zhang, Tianxing Chen, Ganqu Cui, et al. Simplevla-rl: Scaling vla training via reinforcement learning. *arXiv preprint arXiv:2509.09674*, 2025.

[39] Jiacheng Li, Mengzhou Sun, Bowen Zhang, Zhe Zhao, Xiu Liu, et al. Gr-3 technical report. *arXiv preprint arXiv:2507.15493*, 2025.

[40] Shuang Li, Yihuai Gao, Dorsa Sadigh, and Shuran Song. Unified video action model. In *Robotics: Science and Systems*, 2025.

[41] Yunzhu Li, Jiajun Wu, Jun-Yan Zhu, Joshua B Tenenbaum, Antonio Torralba, and Russ Tedrake. Propagation networks for model-based control under partial observation. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 1205–1211. IEEE, 2019.

[42] 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. In *Conference on Robot Learning (CoRL)*, 2024.

[43] Weixin Liang, LILI YU, Liang Luo, Srini Iyer, Ning Dong, Chunting Zhou, Gargi Ghosh, Mike Lewis, Wen tau Yih, Luke Zettlemoyer, and Xi Victoria Lin. Mixture-of-transformers: A sparse and scalable architecture for multi-modal foundation models. *Transactions on Machine Learning Research*, 2025.

[44] Zhixuan Liang, Yizhuo Li, Tianshuo Yang, Chengyue Wu, Sitong Mao, Tian Nian, Liuao Pei, Shunbo Zhou, Xiaokang Yang, Jiangmiao Pang, Yao Mu, and Ping Luo. Discrete diffusion vla: Bringing discrete diffusion to action decoding in vision-language-action policies. *arXiv preprint arXiv:2508.20072*, 2025.

[45] Fanqi Lin, Yingdong Hu, Pingyue Sheng, Chuan Wen, Jiacheng You, and Yang Gao. Data scaling laws in imitation learning for robotic manipulation. In *Int. Conf. Learn. Represent.*, 2025.

[46] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *Int. Conf. Learn. Represent.*, 2023.

[47] Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. In *Adv. Neural Inform. Process. Syst.*, 2023.

[48] Fangchen Liu, Chuanyu Li, Yihua Qin, Austin Shaw, Jing Xu, Pieter Abbeel, and Rui Chen. Vitamin: Learning contact-rich tasks through robot-free visuo-tactile manipulation interface. *arXiv preprint arXiv:2504.06156*, 2025.

[49] Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. Rdt-1b: A diffusion foundation model for bimanual manipulation. In *Int. Conf. Learn. Represent.*, 2025.

[50] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *Int. Conf. Learn. Represent.*, 2023.

[51] Zeyi Liu, Cheng Chi, Eric Cousineau, Naveen Kuppuswamy, Benjamin Burchfiel, and Shuran Song. Maniwav: Learning robot manipulation from in-the-wild audio-visual data. *arXiv preprint arXiv:2406.19464*, 2024.

[52] Bethany Lusch, J Nathan Kutz, and Steven L Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature communications*, 9(1):4950, 2018.

[53] Open X-Embodiment Collaboration. Open x-embodiment: Robotic learning datasets and rt-x models. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

[54] OpenAI. Video generation models as world simulators. *OpenAI Technical Report*, 2024.

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

[56] Jack Parker-Holder, Philip Ball, Jake Bruce, Vibhavari Dasagi, Kristian Holsheimer, Christos Kaplanis, Alexandre Moufarek, Guy Scully, Jeremy Shar, Jimmy Shi, Stephen Spencer, Jessica Yung, Michael Dennis, Sultan Kenjeyev, Shangbang Long, et al. Genie 2: A large-scale foundation world model. https://deepmind.google/discover/blog/genie-2-a-large-scale-foundation-world-model/, 2024.

[57] Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. Fast: Efficient action tokenization for vision-language-action models. In *Robotics: Science and Systems*, 2025.

[58] 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. In *Robotics: Science and Systems*, 2025.

[59] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

[60] Omar Rayyan, John Abanes, Mahmoud Hafez, Anthony Tzes, and Fares Abu-Dakka. Mv-umi: A scalable multi-view interface for cross-embodiment learning. *arXiv preprint arXiv:2509.18757*, 2025.

[61] Moritz Reuss, Jyothish Pari, Pulkit Agrawal, and Rudolf Lioutikov. Efficient diffusion transformer policies with mixture of expert denoisers for multitask learning. In *Int. Conf. Learn. Represent.*, 2025.

[62] Moritz Reuss, Hongyi Zhou, Marcel Rühle, Ömer Erdinç Yağmurlu, Fabian Otto, and Rudolf Lioutikov. Flower: Democratizing generalist robot policies with efficient vision-language-action flow policies. In *Conference on Robot Learning (CoRL)*, 2025.

[63] Bokui Shen, Zhenyu Jiang, Christopher Choy, Silvio Savarese, Leonidas J Guibas, Anima Anandkumar, and Yuke Zhu. Action-conditional implicit visual dynamics for deformable object manipulation. *The International Journal of Robotics Research*, 43(4):437–455, 2024.

[64] Hao Shi, Bin Xie, Yingfei Liu, Lin Sun, Fengrong Liu, Tiancai Wang, Erjin Zhou, Haoqiang Fan, Xiangyu Zhang, and Gao Huang. Memoryvla: Perceptual-cognitive memory in vision-language-action models for robotic manipulation. *arXiv preprint arXiv:2508.19236*, 2025.

[65] Haochen Shi, Huazhe Xu, Zhiao Huang, Yunzhu Li, and Jiajun Wu. Robocraft: Learning to see, simulate, and shape elasto-plastic objects in 3d with graph networks. *The International Journal of Robotics Research*, 43(4):533–549, 2024.

[66] Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, Simon Alibert, Matthieu Cord, Thomas Wolf, and Remi Cadene. Smolvla: A vision-language-action model for affordable and efficient robotics. *arXiv preprint arXiv:2506.01844*, 2025.

[67] Ajay Sridhar, Jennifer Pan, Satvik Sharma, and Chelsea Finn. Memer: Scaling up memory for robot control via experience retrieval. *arXiv preprint arXiv:2510.20328*, 2025.

[68] Deborah Sulsky, Shi-Jian Zhou, and Howard L Schreyer. Application of a particle-in-cell method to solid mechanics. *Computer physics communications*, 87(1-2):236–252, 1995.

[69] Jiaming Tang, Yufei Sun, Yilong Zhao, Shang Yang, Yujun Lin, Zhuoyang Zhang, James Hou, Yao Lu, Zhijian Liu, and Song Han. Vlash: Real-time vlas via future-state-aware asynchronous inference. *arXiv preprint arXiv:2512.01031*, 2025.

[70] Gemini Robotics Team, Coline Devin, Yilun Du, Debidatta Dwibedi, Ruiqi Gao, Abhishek Jindal, Thomas Kipf, Sean Kirmani, Fangchen Liu, Anirudha Majumdar, et al. Evaluating gemini robotics policies in a veo world simulator. *arXiv preprint arXiv:2512.10675*, 2025.

[71] Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Pannag Sanketi, Quan Vuong, et al. Octo: An open-source generalist robot policy. In *Robotics: Science and Systems*, 2024.

[72] Yang Tian, Sizhe Yang, Jia Zeng, Ping Wang, Dahua Lin, Hao Dong, and Jiangmiao Pang. Seer: Predictive inverse dynamics models are scalable learners for robotic manipulation. In *Int. Conf. Learn. Represent.*, 2025.

[73] Yang Tian, Yuyin Yang, Yiman Xie, Zetao Cai, Xu Shi, Ning Gao, Hangxu Liu, Xuekun Jiang, Zherui Qiu, Feng Yuan, Yaping Li, Ping Wang, Junhao Cai, Jia Zeng, Hao Dong, et al. Interndata-a1: Pioneering high-fidelity synthetic data for pre-training generalist policy. *arXiv preprint arXiv:2511.16651*, 2025.

[74] Alexander Tong, Kilian Fatras, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *Transactions on Machine Learning Research*, 2024.

[75] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Adv. Neural Inform. Process. Syst.*, 2017.

[76] Yixuan Wang, Yunzhu Li, Katherine Driggs-Campbell, Li Fei-Fei, and Jiajun Wu. Dynamic-resolution model learning for object pile manipulation. *arXiv preprint arXiv:2306.16700*, 2023.

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

[78] WanTeam. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

[79] Manuel Watter, Jost Springenberg, Joschka Boedecker, and Martin Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. *Advances in neural information processing systems*, 28, 2015.

[80] Kun Wu, Chengkai Hou, Jiaming Liu, Zhengping Che, Xiaozhu Ju, Zhuqin Yang, Meng Li, Yinuo Zhao, Zhiyuan Xu, Guang Yang, Shichao Fan, Xinhua Wang, Fei Liao, Zhen Zhao, Guangyu Li, et al. Robomind: Benchmark on multi-embodiment intelligence normative data for robot manipulation. In *Robotics: Science and Systems*, 2025.

[81] Philipp Wu, Alejandro Escontrela, Danijar Hafner, Pieter Abbeel, and Ken Goldberg. Daydreamer: World models for physical robot learning. In *Conference on Robot Learning (CoRL)*, 2022.

[82] Philipp Wu, Alejandro Escontrela, Danijar Hafner, Pieter Abbeel, and Ken Goldberg. Daydreamer: World models for physical robot learning. In *Conference on robot learning*, pages 2226–2240. PMLR, 2023.

[83] Shihan Wu, Xuecheng Liu, Shaoxuan Xie, Pengwei Wang, Xinghang Li, Bowen Yang, Zhe Li, Kai Zhu, Hongyu Wu, Yiheng Liu, Zhaoye Long, Yue Wang, Chong Liu, Dihan Wang, Ziqiang Ni, et al. Robocoin: An open-sourced bimanual robotic data collection for integrated manipulation. *arXiv preprint arXiv:2511.17441*, 2025.

[84] Zhenjia Xu, Jiajun Wu, Andy Zeng, Joshua B Tenenbaum, and Shuran Song. Densephysnet: Learning dense physical object representations via multi-step dynamic interactions. *arXiv preprint arXiv:1906.03853*, 2019.

[85] Mengjiao Yang, Yilun Du, Kamyar Ghasemipour, Jonathan Tompson, Leslie Pack Kaelbling, Dale Schuurmans, and Pieter Abbeel. Unisim: Learning interactive real-world simulators. In *Int. Conf. Learn. Represent.*, 2024.

[86] Shuai Yang, Hao Li, Yilun Chen, Bin Wang, Yang Tian, Tai Wang, Hanqing Wang, Feng Zhao, Yiyi Liao, and Jiangmiao Pang. Instructvla: Vision-language-action instruction tuning from understanding to manipulation. *arXiv preprint arXiv:2507.17520*, 2025.

[87] Kaifeng Zhang, Baoyu Li, Kris Hauser, and Yunzhu Li. Adaptigraph: Material-adaptive graph-based neural dynamics for robotic manipulation. *arXiv preprint arXiv:2407.07889*, 2024.

[88] Kaifeng Zhang, Baoyu Li, Kris Hauser, and Yunzhu Li. Particle-grid neural dynamics for learning deformable object models from rgb-d videos. *arXiv preprint arXiv:2506.15680*, 2025.

[89] Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, et al. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 1702–1713, 2025.

[90] Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *Robotics: Science and Systems*, 2023.

[91] Zhaxizhuoma, Kehui Liu, Chuyue Guan, Zhongjie Jia, Ziniu Wu, Xin Liu, Tianyu Wang, Shuai Liang, Pengan Chen, Pingrui Zhang, Haoming Song, Delin Qu, Dong Wang, Zhigang Wang, Nieqing Cao, et al. Fastumi: A scalable and hardware-independent universal manipulation interface. *arXiv preprint arXiv:2409.19499*, 2024.

[92] Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, Ya-Qin Zhang, Jiangmiao Pang, Jingjing Liu, Tai Wang, and Xianyuan Zhan. X-vla: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. *arXiv preprint arXiv:2510.10274*, 2025.

[93] Ruijie Zheng, Yongyuan Liang, Shuaiyi Huang, Jianfeng Gao, Hal Daumé III, Andrey Kolobov, Furong Huang, and Jianwei Yang. Tracevla: Visual trace prompting enhances spatial-temporal awareness for generalist robotic policies. *arXiv preprint arXiv:2412.10345*, 2024.

[94] Pengfei Zhou, Liliang Chen, Shengcong Chen, Di Chen, Wenzhi Zhao, Rongjun Jin, Guanghui Ren, and Jianlan Luo. Act2goal: From world model to general goal-conditioned policy. *arXiv preprint arXiv:2512.23541*, 2025.

[95] Siyuan Zhou, Yilun Du, Jiaben Chen, Yandong Li, Dit-Yan Yeung, and Chuang Gan. Robodreamer: Learning compositional world models for robot imagination. *arXiv preprint arXiv:2404.12377*, 2024.

[96] Chuning Zhu, Raymond Yu, Siyuan Feng, Benjamin Burchfiel, Paarth Shah, and Abhishek Gupta. Unified world models: Coupling video and action diffusion for pretraining on large robotic datasets. In *Robotics: Science and Systems*, 2025.

Appendix

A Real-world Evaluation Details

We present detailed evaluation results for all real-world manipulation tasks. Each task is evaluated with 20 trials for both our method and the baseline ($\pi_{0.5}$). To ensure fair comparison, we adopt an alternating evaluation protocol: one trial with $\pi_{0.5}$, followed by one trial with our method, and so on.

For each trial, we record the success status of every intermediate step. If a step requires a retry to succeed, we assign a score of 0.5; if it fails, the score is 0; if it succeeds on the first attempt, the score is 1. A trial is marked as successful only if all steps are completed (i.e., the total score equals the maximum possible score).

We report two metrics:

- **Progress Score (PS)**: The average score across all trials divided by the maximum possible score, expressed as a percentage: $\text{PS} = \frac{\text{Average Progress}}{\text{Max Steps}} \times 100\%$.
- **Success Rate (SR)**: The number of successful trials divided by the total number of trials, expressed as a percentage: $\text{SR} = \frac{\text{\# Successful Trials}}{N} \times 100\%$.

We evaluate on six diverse real-world tasks: **Make Breakfast** (10 steps: preparing a complete breakfast including toasting bread, pouring water, and plating), **Pick Screws** (5 steps: picking up paper, pouring screws, and inserting three screws), **Fold Clothes** (6 steps: folding a shirt including sleeves and smoothing), **Unpack Delivery** (5 steps: opening a package using a utility knife), **Insert Tubes** (2 categories: grasping and inserting 3 tubes), and **Fold Pants** (3

steps: folding pants and placing them). These tasks span long-horizon sequential manipulation, precision control, and deformable object handling. The following tables present per-trial results for each task.

Table S1. *Evaluation on RoboTwin 2.0 Simulation (Easy vs Hard, 50 tasks).* RoboTwin 2.0 is a challenging bimanual manipulation benchmark requiring coordinated dual-arm control. Easy uses fixed initial configurations while Hard involves randomized object poses and scene layouts.

Simulation Task	**Horizon**	**Ours**		π_0 [7]		$\pi_{0.5}$ [7]		X-VLA [92]		Motus [5]	
		Easy	Hard	Easy	Hard	Easy	Hard	Easy	Hard	Easy	Hard
Adjust Bottle	1	90%	94%	99%	95%	**100%**	**99%**	**100%**	**99%**	89%	93%
Beat Block Hammer	1	**96%**	98%	79%	84%	**96%**	93%	92%	88%	95%	88%
Blocks Ranking RGB	3	**99%**	98%	80%	63%	92%	85%	83%	83%	**99%**	97%
Blocks Ranking Size	3	**94%**	96%	14%	5%	49%	26%	67%	74%	75%	63%
Click Alarmclock	1	99%	**100%**	77%	68%	98%	89%	99%	99%	**100%**	**100%**
Click Bell	1	**100%**	**100%**	71%	48%	**99%**	66%	**100%**	**100%**	**100%**	**100%**
Dump Bin Bigbin	1	89%	96%	88%	83%	92%	**97%**	79%	77%	**95%**	91%
Grab Roller	1	**100%**	**100%**	98%	94%	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**
Handover Block	2	99%	78%	47%	31%	66%	57%	73%	37%	86%	73%
Handover Mic	2	94%	96%	**97%**	**97%**	**98%**	**97%**	0%	0%	78%	63%
Hanging Mug	2	**40%**	28%	14%	11%	18%	17%	23%	27%	**38%**	**38%**
Lift Pot	1	**100%**	99%	80%	72%	96%	85%	**99%**	**100%**	96%	**99%**
Move Can Pot	1	94%	97%	**68%**	48%	51%	55%	89%	86%	34%	74%
Move Pillbottle Pad	1	**99%**	99%	67%	46%	84%	61%	73%	71%	93%	**96%**
Move Playingcard Away	1	**100%**	99%	74%	65%	96%	84%	93%	**98%**	**100%**	96%
Move Stapler Pad	1	91%	79%	41%	24%	56%	42%	78%	73%	**83%**	**85%**
Open Laptop	1	92%	94%	71%	81%	90%	96%	93%	**100%**	95%	91%
Open Microwave	1	82%	86%	4%	32%	34%	**77%**	**79%**	71%	**95%**	**91%**
Pick Diverse Bottles	2	89%	82%	69%	31%	81%	71%	58%	36%	90%	**91%**
Pick Dual Bottles	2	**100%**	99%	59%	37%	93%	63%	47%	36%	96%	90%
Place A2B Left	1	**97%**	93%	43%	47%	87%	82%	48%	49%	82%	79%
Place A2B Right	1	**97%**	95%	39%	34%	87%	84%	36%	36%	90%	**87%**
Place Bread Basket	1	**97%**	95%	62%	46%	77%	64%	81%	71%	91%	**94%**
Place Bread Skillet	2	95%	90%	66%	49%	**85%**	66%	77%	67%	86%	**83%**
Place Burger Fries	2	97%	95%	81%	76%	**94%**	87%	**94%**	**94%**	**98%**	**98%**
Place Can Basket	2	81%	84%	55%	46%	**62%**	**62%**	49%	52%	**81%**	76%
Place Cans Plasticbox	2	**100%**	99%	63%	45%	94%	84%	**97%**	**98%**	**98%**	94%
Place Container Plate	1	99%	97%	**97%**	92%	**99%**	95%	**97%**	95%	98%	**99%**
Place Dual Shoes	2	94%	89%	59%	51%	75%	75%	79%	**88%**	93%	87%
Place Empty Cup	1	**100%**	**100%**	91%	85%	**100%**	99%	**100%**	98%	99%	98%
Place Fan	1	99%	93%	66%	71%	**87%**	85%	80%	75%	**91%**	87%
Place Mouse Pad	1	93%	96%	20%	20%	**60%**	39%	**70%**	**70%**	66%	68%
Place Object Basket	2	91%	88%	67%	70%	80%	76%	44%	39%	81%	**87%**
Place Object Scale	1	96%	95%	57%	52%	**86%**	80%	52%	74%	88%	85%
Place Object Stand	1	**99%**	96%	82%	68%	91%	85%	86%	88%	**98%**	**97%**
Place Phone Stand	1	**97%**	97%	49%	53%	81%	81%	**88%**	87%	87%	86%
Place Shoe	1	98%	98%	76%	76%	92%	93%	**96%**	95%	**99%**	**97%**
Press Stapler	1	85%	82%	44%	37%	87%	83%	**92%**	**98%**	93%	**98%**
Put Bottles Dustbin	3	87%	91%	65%	56%	**84%**	79%	74%	77%	81%	79%
Put Object Cabinet	2	85%	87%	73%	60%	**80%**	**79%**	46%	48%	**88%**	71%
Rotate QRcode	1	**96%**	91%	74%	70%	89%	87%	34%	33%	**89%**	73%
Scan Object	2	96%	91%	55%	42%	72%	65%	14%	36%	67%	66%
Shake Bottle Horizontally	1	**100%**	99%	98%	92%	**99%**	**99%**	**100%**	**100%**	**100%**	98%
Shake Bottle	1	**100%**	97%	94%	91%	**99%**	97%	**99%**	**100%**	**100%**	**97%**
Stack Blocks Three	3	99%	98%	72%	52%	91%	76%	6%	10%	91%	**95%**
Stack Blocks Two	2	**100%**	98%	93%	79%	97%	**100%**	92%	87%	**100%**	98%
Stack Bowls Three	3	86%	83%	**77%**	75%	**77%**	71%	76%	**86%**	79%	**87%**
Stack Bowls Two	2	94%	98%	94%	95%	95%	96%	96%	93%	**98%**	**98%**
Stamp Seal	1	96%	97%	46%	33%	79%	55%	76%	82%	93%	92%
Turn Switch	1	44%	45%	41%	42%	**62%**	54%	40%	61%	**84%**	**78%**
Average (%)	–	**92.93**	**91.55**	65.92	58.40	82.74	76.76	72.80	72.84	88.66	87.02

Table S2. Detailed evaluation results for Make Breakfast task (10 steps, max score 10).

Trial	Succ.	Grasp Plate	Grasp Bread	Grasp Fork	Place Bread	Press Toaster	Grasp Cup	Grasp Kettle	Pour	Grasp Apple	Serve	Prog.
Ours												
1	0	1	1	1	1	1	1	1	0	1	1	9
2	1	1	1	1	1	1	1	1	1	1	1	10
3	1	1	1	1	1	1	1	1	1	1	1	10
4	1	1	1	1	1	1	1	1	1	1	1	10
5	1	1	1	1	1	1	1	1	1	1	1	10
6	1	1	1	1	1	1	1	1	1	1	1	10
7	0	1	1	1	1	1	1	1	1	1	0	9
8	1	1	1	1	1	1	1	1	1	0.5	1	9.5
9	0	1	1	1	0	1	1	1	1	1	1	9
10	1	1	1	1	1	1	1	1	1	1	1	10
11	0	1	1	1	1	1	1	1	0	1	1	9
12	1	1	1	1	1	1	1	1	1	1	1	10
13	0	1	1	1	0	1	1	1	1	1	1	9
14	1	1	1	1	1	1	1	1	1	1	1	10
15	1	1	1	1	1	1	1	1	1	1	1	10
16	1	1	1	1	1	1	1	1	1	1	1	10
17	1	1	1	1	1	1	1	1	1	0.5	1	9.5
18	1	1	1	1	1	1	1	1	1	1	1	10
19	1	1	1	1	1	1	1	1	1	1	1	10
20	1	1	1	1	1	1	1	1	1	1	1	10
Avg	**0.75**	**1.00**	**1.00**	**1.00**	**0.90**	**1.00**	**1.00**	**1.00**	**0.90**	**0.95**	**0.95**	**9.70**
Progress Score	**97.0%** (= 9.70/10 × 100%)											
Success Rate	**75.0%** (= 15/20 × 100%)											
$\pi_{0.5}$												
1	0	1	1	1	1	0	0	0	0	0	0	4
2	1	1	1	1	1	1	1	1	0	1	1	9
3	1	1	1	1	1	1	1	1	0	1	1	9
4	1	1	1	1	1	0	0	1	1	1	1	8
5	0	1	1	1	0	1	0	1	1	0	1	7
6	1	1	1	1	1	1	1	1	0	1	1	9
7	1	1	1	0	1	1	1	1	0	1	1	8
8	0	1	1	1	1	1	1	1	1	1	0	9
9	0	1	1	0	1	0	1	1	0	0	0	5
10	0	1	1	1	1	0	0	0	0	0	0	4
11	1	1	1	1	1	0	0	1	1	1	1	8
12	1	1	1	1	0	0	0	1	1	1	1	7
13	1	1	1	0	1	0	0	1	1	1	1	7
14	1	1	1	1	0	1	0	1	1	1	1	8
15	1	1	1	1	1	0	0	1	1	1	1	8
16	1	1	1	1	0	1	0	1	1	1	1	8
17	1	1	1	1	1	1	0	1	1	1	1	9
18	1	1	1	1	1	1	1	0	0	1	1	8
19	1	1	1	0	1	0	0	1	0	1	1	6
20	0	1	1	1	0	1	0	0	0	1	0	5
Avg	**0.70**	**1.00**	**1.00**	**0.80**	**0.75**	**0.55**	**0.35**	**0.80**	**0.50**	**0.80**	**0.75**	**7.30**
Progress Score	**73.0%** (= 7.30/10 × 100%)											
Success Rate	**70.0%** (= 14/20 × 100%)											

Table S3. Detailed evaluation results for Pick Screws task (5 steps, max score 5).

Trial	Success	Grab Paper	Pour Screws	Screw 1	Screw 2	Screw 3	Progress
Ours							
1	1	1	1	0.5	0.5	1	4
2	0	1	0	1	1	0.5	3.5
3	1	1	1	1	0.5	0.5	4
4	1	1	1	1	1	1	5
5	1	1	1	1	1	0.5	4.5
6	1	1	1	0.5	1	0.5	4
7	1	1	1	1	1	1	5
8	1	1	1	1	1	1	5
9	0	1	0	0	0	0	1
10	1	1	1	1	1	1	5
11	1	1	1	1	1	1	5
12	0	1	0	1	1	1	4
13	0	1	0	1	1	1	4
14	1	1	1	1	0.5	0.5	4
15	0	1	0	0	0.5	1	2.5
16	1	1	1	1	1	1	5
17	1	1	1	1	1	0.5	4.5
18	1	1	1	0.5	0.5	1	4
19	0	1	1	0	1	0.5	3.5
20	1	1	1	1	1	1	5
Avg	**0.70**	**1.00**	**0.75**	**0.78**	**0.83**	**0.78**	**4.13**
Progress Score			**82.5%** (= 4.13/5 × 100%)				
Success Rate			**70.0%** (= 14/20 × 100%)				
$\pi_{0.5}$							
1	0	1	0	1	1	1	4
2	0	0.5	0	1	1	0.5	3
3	1	1	1	1	1	1	5
4	0	1	0	1	0.5	0.5	3
5	1	1	1	0.5	1	0.5	4
6	1	1	1	1	0.5	1	4.5
7	1	1	1	1	1	1	5
8	1	1	1	0.5	0.5	1	4
9	0	1	0	0.5	0.5	0	2
10	1	1	1	1	1	1	5
11	1	1	1	0.5	1	1	4.5
12	0	1	0	1	1	0.5	3.5
13	0	1	1	1	0	0.5	3.5
14	1	1	1	0.5	1	1	4.5
15	0	1	1	1	1	0	4
16	1	1	1	1	1	1	5
17	0	0	0	1	0.5	0.5	2
18	0	0	0	0	0	0	0
19	0	1	0	1	0.5	0.5	3
20	1	1	1	1	1	0.5	4.5
Avg	**0.50**	**0.88**	**0.60**	**0.83**	**0.75**	**0.65**	**3.70**
Progress Score			**74.0%** (= 3.70/5 × 100%)				
Success Rate			**50.0%** (= 10/20 × 100%)				

Table S4. Detailed evaluation results for Fold Clothes task (6 steps, max score 6).

Trial	Success	Fold Half	Left Sleeve	Right Sleeve	Fold Again	Flatten	Place	Progress
Ours								
1	1	1	1	1	1	1	1	6
2	0	1	1	0	0	0	0	2
3	1	1	1	1	1	0.5	0.5	5
4	1	1	1	1	1	1	1	6
5	0	0.5	0	0	0	0	0	0.5
6	0	0	0	0	0	0	0	0
7	0	0.5	0	0	0	0	0	0.5
8	1	1	1	1	1	0.5	0.5	5
9	1	1	1	1	1	1	1	6
10	1	1	1	1	1	1	1	6
11	0	0.5	0	0	0	0	0	0.5
12	0	0.5	0	0	0	0	0	0.5
13	0	0.5	0	0	0	0	0	0.5
14	0	1	1	1	1	1	0	5
15	0	0.5	0	0	0	0	0	0.5
16	0	1	1	1	1	0.5	0	4.5
17	0	0	0	0	0	0	0	0
18	0	1	1	1	0.5	0	0	3.5
19	0	0.5	0	0	0	0	0	0.5
20	1	1	1	1	1	1	1	6
Avg	**0.35**	**0.73**	**0.55**	**0.50**	**0.48**	**0.38**	**0.30**	**2.93**
Progress Score				**48.8%** (= 2.93/6 × 100%)				
Success Rate				**35.0%** (= 7/20 × 100%)				
$\pi_{0.5}$								
1	1	1	1	1	1	1	1	6
2	1	1	1	1	1	1	1	6
3	1	1	1	1	0.5	1	0.5	5
4	0	1	1	1	0.5	1	0	4.5
5	0	0.5	1	1	0.5	1	0	4
6	0	1	1	0.5	1	1	0	4.5
7	0	0.5	0	0	0	0	0	0.5
8	0	0.5	0	0	0	0	0	0.5
9	0	1	1	1	1	1	0	5
10	0	0.5	1	1	0.5	0	0	3
11	0	0.5	0	0	0	0	0	0.5
12	0	1	1	1	1	1	0	5
13	0	1	1	0.5	0	0	0	2.5
14	1	1	1	1	1	1	1	6
15	1	1	1	1	0.5	1	1	5.5
16	0	1	1	1	0.5	0	0	3.5
17	1	1	1	1	1	1	1	6
18	0	1	1	1	0.5	1	0	4.5
19	0	0	0	0	0	0	0	0
20	0	1	1	1	0	0	0	3
Avg	**0.30**	**0.83**	**0.80**	**0.75**	**0.53**	**0.60**	**0.28**	**3.78**
Progress Score				**62.9%** (= 3.78/6 × 100%)				
Success Rate				**30.0%** (= 6/20 × 100%)				

Table S5. Detailed evaluation results for Unpack Delivery task (5 steps, max score 5).

Trial	Success	Grab Knife	Push Blade	Handover	Cut Seal	Open Lid	Progress
Ours							
1	1	1	1	1	1	1	5
2	1	1	1	1	1	1	5
3	1	1	1	1	1	1	5
4	0	1	0	0	0	0	1
5	0	1	1	1	0.5	0	3.5
6	0	1	1	1	0	0	3
7	1	1	1	1	1	1	5
8	0	1	1	1	0	0	3
9	1	1	1	1	1	1	5
10	1	1	1	1	1	1	5
11	1	1	1	1	1	1	5
12	1	1	1	1	1	1	5
13	0	1	1	1	0	0	3
14	1	1	1	1	1	1	5
15	1	1	1	1	1	1	5
16	0	1	1	1	0	0	3
17	1	1	1	1	0.5	1	4.5
18	1	1	1	1	1	1	5
19	0	1	1	1	0.5	0	3.5
20	1	1	1	1	1	1	5
Avg	**0.65**	**1.00**	**0.95**	**0.95**	**0.68**	**0.65**	**4.23**
Progress Score	**84.5%** (= 4.23/5 × 100%)						
Success Rate	**65.0%** (= 13/20 × 100%)						
$\pi_{0.5}$							
1	0	1	1	0.5	0.5	0	3
2	0	1	1	1	0	0	3
3	0	1	1	1	0.5	0	3.5
4	0	1	1	1	0.5	0	3.5
5	0	1	1	1	0.5	0	3.5
6	0	1	1	1	0	0	3
7	0	1	1	1	0	0	3
8	1	1	1	1	0.5	1	4.5
9	0	1	1	1	0.5	0	3.5
10	1	1	1	1	1	1	5
11	0	1	1	1	0.5	0	3.5
12	0	1	1	1	0.5	0	3.5
13	1	1	1	1	1	1	5
14	0	1	1	1	0.5	0	3.5
15	1	1	1	1	0.5	1	4.5
16	0	1	1	1	0	0	3
17	0	1	1	1	0	0	3
18	0	1	1	1	0	0	3
19	0	1	1	1	0.5	0	3.5
20	1	1	1	1	1	1	5
Avg	**0.25**	**1.00**	**1.00**	**0.98**	**0.43**	**0.25**	**3.65**
Progress Score	**73.0%** (= 3.65/5 × 100%)						
Success Rate	**25.0%** (= 5/20 × 100%)						

Table S6. Detailed evaluation results for Insert Tubes task (2 categories: Grasp and Insert, max score 6).

Trial	Success	Grasp (3)	Insert (3)	Progress
Ours				
1	0	3	2	5
2	1	3	3	6
3	0	3	2	5
4	0	2	2	4
5	1	3	3	6
6	1	3	3	6
7	0	3	2	5
8	1	3	3	6
9	0	3	2	5
10	0	3	2	5
11	1	3	3	6
12	0	3	2	5
13	1	3	3	6
14	0	3	2	5
15	0	3	1	4
16	1	3	3	6
17	0	2	2	4
18	0	2	2	4
19	1	3	3	6
20	0	2	2	4
Avg	**0.40**	**2.80**	**2.35**	**5.15**
Progress Score		**85.8%** (= 5.15/6 × 100%)		
Success Rate		**40.0%** (= 8/20 × 100%)		
$\boldsymbol{\pi_{0.5}}$				
1	0	3	1	4
2	0	3	1	4
3	0	3	1	4
4	0	3	1	4
5	0	3	1	4
6	0	3	1	4
7	0	3	1	4
8	0	3	2	5
9	1	3	3	6
10	1	3	3	6
11	1	3	3	6
12	0	2	1	3
13	0	3	2	5
14	0	2	2	4
15	1	3	3	6
16	1	3	3	6
17	0	3	2	5
18	0	2	2	4
19	1	3	3	6
20	0	3	2	5
Avg	**0.30**	**2.85**	**1.90**	**4.75**
Progress Score		**79.2%** (= 4.75/6 × 100%)		
Success Rate		**30.0%** (= 6/20 × 100%)		

Table S7. Detailed evaluation results for Fold Pants task (3 steps, max score 3).

Trial	Success	Fold 1	Fold 2	Place	Progress
Ours					
1	1	1	1	1	3
2	1	1	1	1	3
3	1	1	1	1	3
4	0	0	0	0	0
5	0	1	0	0	1
6	0	1	0	0	1
7	1	1	1	1	3
8	1	1	1	1	3
9	1	1	1	1	3
10	1	1	1	1	3
11	1	1	1	1	3
12	1	1	1	1	3
13	1	1	1	1	3
14	1	1	1	1	3
15	1	1	1	1	3
16	1	1	1	1	3
17	1	1	1	1	3
18	0	1	0	0	1
19	0	1	0	0	1
20	0	0	0	0	0
Avg	**0.70**	**0.90**	**0.70**	**0.70**	**2.30**
Progress Score		**76.7%** (= 2.30/3 × 100%)			
Success Rate		**70.0%** (= 14/20 × 100%)			
$\pi_{0.5}$					
1	1	1	1	1	3
2	1	1	1	1	3
3	1	1	1	1	3
4	0	0	0	0	0
5	0	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0
11	1	1	1	1	3
12	1	1	1	1	3
13	1	1	1	1	3
14	0	0	0	0	0
15	0	0	0	0	0
16	0	0	0	0	0
17	0	0	0	0	0
18	0	0	0	0	0
19	0	0	0	0	0
20	0	0	0	0	0
Avg	**0.30**	**0.30**	**0.30**	**0.30**	**0.90**
Progress Score		**30.0%** (= 0.90/3 × 100%)			
Success Rate		**30.0%** (= 6/20 × 100%)			