ANT GROUP

Native Video-Action Pretraining for Generalizable Robot Control

Qihang Zhang, Lin Li, Luyao Zhang, Shuai Yang, Yiming Luo, Shuaiting Li, Ruilin Wang, Junke Wang, Jiahao Shao, Gangwei Xu, Jiaming Zhou, Yishu Shen, Yudong Jin, Fangyi Xu, Shuailei Ma, Jiaqi Liao, Guanxing Lu, Zifan Shi, Yongkun Wen, Yujie Zhao, Weixuan Tang, Xinyang Wang, Chaojian Li, Jiapeng Zhu, Ka Leong Cheng, Nan Xue, Xing Zhu, Yujun Shen, Yinghao Xu[†]

[†]Project Lead

The advent of video-action models offers a promising path for robot control. Nevertheless, we argue that repurposing video generative models designed for digital content creation is inherently inadequate for physical environments. To bridge this gap, we present `LingBot-VA 2.0`, a video-action foundation model built from the ground up for embodiment. Four core design principles showcase its evolution from `LingBot-VA`. (1) Departing from traditional reconstruction-focused VAEs, we introduce ***a semantic visual-action tokenizer***, which aligns visual representations with both semantics and actions, improving instruction following and action precision in subsequent policy learning. (2) Given the strictly causal nature of temporal dynamics, we adopt ***a causal pretraining paradigm***, training from scratch to circumvent the catastrophic forgetting that frequently occurs when adapting bidirectional architectures. (3) To meet the demands of high-frequency inference, our model employs ***a sparse MoE backbone***, expanding model capacity without compromising efficiency. (4) Real-time closed-loop control is realized through ***an enhanced asynchronous inference scheme***, which predicts future latents in parallel with action execution while re-grounding each rollout on the latest observation via learned forward dynamics. Real-world deployment validates `LingBot-VA 2.0` as a robust foundation model, as evidenced by its few-shot generalization across complex manipulation tasks.

Website: https://technology.robbyant.com/lingbot-va-v2

1 Introduction

Video-action models such as LingBot-VA [50] and DreamZero [115] have recently emerged as a powerful paradigm for generalist robot manipulation. Rather than mapping observations directly to actions, as reactive vision-language-action policies do [9, 11, 43], they jointly predict how a scene will evolve and how to act within it, grounding control in physical dynamics and improving sample efficiency and generalization [35, 54, 132]. Much of this capability, however, is inherited from their video-pretrained backbones, suggesting that generalist robot control depends as much on the pretrained foundation as on the policy learned upon it.

Current video-action models are still largely built from components designed for generic video generation—a reconstruction-oriented VAE and a bidirectional video-diffusion backbone—with an action module added for robotics afterward. This starting point creates three concrete limitations. First, the representation is optimized for appearance rather than dynamics: pixel-reconstruction latents preserve visual detail but carry limited semantic and physical structure, and the separately attached action module leaves world states and actions in poorly aligned spaces. Second, inference is too slow for closed-loop control: high-dimensional video tokens and iterative denoising make first-generation video-action models costly to run at the frequencies real robots require. Third, the pretraining signal does not scale toward control: web-scale video is abundant, but generic video objectives do not teach how actions reshape the world, so the action signal remains tied to expensive robot data, limiting the control knowledge learned before downstream adaptation.

These limitations are compounded by a structural mismatch: the backbone is pretrained with bidirectional attention,

while closed-loop control unfolds strictly forward in time. LingBot-VA resolves this mismatch by finetuning the stack into a causal video-action model, but the retrofit relies on scarce robot data and may weaken the broad priors acquired from web-scale pretraining. We therefore pursue a *native* route: pretraining the full stack, including a semantic visual-action tokenizer and a causal DiT, for robot control on web-scale image and video data, so that the broad priors are acquired natively in causal form rather than inherited and then eroded.

We present `LingBot-VA 2.0`, which instantiates this native route with a design that removes the three limitations above. Rather than attaching actions to an off-the-shelf video generator, `LingBot-VA 2.0` first builds a shared semantic latent space for observations and actions, then learns causal dynamics on top of that space. The first stage is a semantic visual-action tokenizer [98]: beyond reconstruction, it aligns video latents to a frozen visual foundation model, producing compact representations that are far more semantic than pixel-reconstruction latents. It further learns latent actions through self-supervision, placing world states and actions in the same space so that even unlabeled web video carries action-relevant supervision. The second stage is a causal diffusion transformer that predicts future visual latents together with the latent actions connecting them, conditioned on language instructions through cross-attention to a frozen pretrained language model. Its causal next-latent objective provides a self-supervised signal that scales to web video while matching the temporal structure of closed-loop control, avoiding the bidirectional-to-causal retrofit that risks eroding pretrained priors; and its feed-forward layers are instantiated as a sparse mixture-of-experts, preserving model capacity while reducing the active computation per step for high-frequency control. Because adjacent chunks are often visually similar enough that next-chunk supervision alone rewards copying appearance, we additionally train with multi-chunk prediction (MCP), which supervises several future chunks at once so that the learned latents encode trajectory-level dynamics rather than short-term visual continuity.

Even with a semantic latent space and a sparse backbone, however, deployment on real hardware faces a serial bottleneck: the robot must continue acting while the world model updates its prediction; otherwise, model latency directly becomes control latency. We address this with *Foresight Reasoning*, an asynchronous inference scheme in which future visual latents are predicted in parallel with action execution. A naive asynchronous scheme can drift by conditioning on stale imagined latents. `LingBot-VA 2.0` instead re-grounds each prediction on the latest real observation through a learned forward-dynamics step of the policy itself. This shares the concurrent-inference motivation of Harmonic Reasoning in GEN-0 [31], but grounds the look-ahead in a learned world model and corrects it with every real observation.

Because its action signal no longer depends on scarce robot demonstrations, `LingBot-VA 2.0` acquires control knowledge at the scale of web video rather than at the scale of robot datasets. The resulting video-action representation improves generalization: it adapts from only 10–15 demonstrations, transfers across embodiments without large-scale re-collection, and in some settings executes new tasks zero-shot. At deployment time, sparse MoE inference, few-step consistency distillation, Foresight Reasoning, and quantized execution allow `LingBot-VA 2.0` to perform real-time closed-loop control on real robots with a peak asynchronous execution frequency of 225 Hz. Together with a hierarchical VLM planner that decomposes long-horizon goals into subtasks (Sec. 2), these gains translate to more consistent long-horizon behavior and high-precision manipulation, improving over strong baselines such as $\pi_{0.5}$ [77], LingBot-VA [50] by a substantial margin across simulation and real-world evaluations.

In summary, our contributions are:

- **Native video-action pretraining for robot control.** We pretrain the full video-action stack, including a semantic visual-action tokenizer and a causal DiT, using scalable self-supervision rather than adapting components pretrained for generic video generation.
- **A shared semantic video-action representation.** `LingBot-VA 2.0` places world states and latent actions in a single semantic latent space anchored to a language-aligned visual foundation model, tightening vision–language–action alignment for stronger instruction following and action precision.
- **General, fast, and precise robot control.** The pretrained video-action representation supports few-shot and zero-shot generalization; sparse MoE inference, few-step distillation, Foresight Reasoning, and quantized deployment enable real-time closed-loop execution; and closed-loop re-grounding with hierarchical planning sustains fine-grained, long-horizon manipulation.

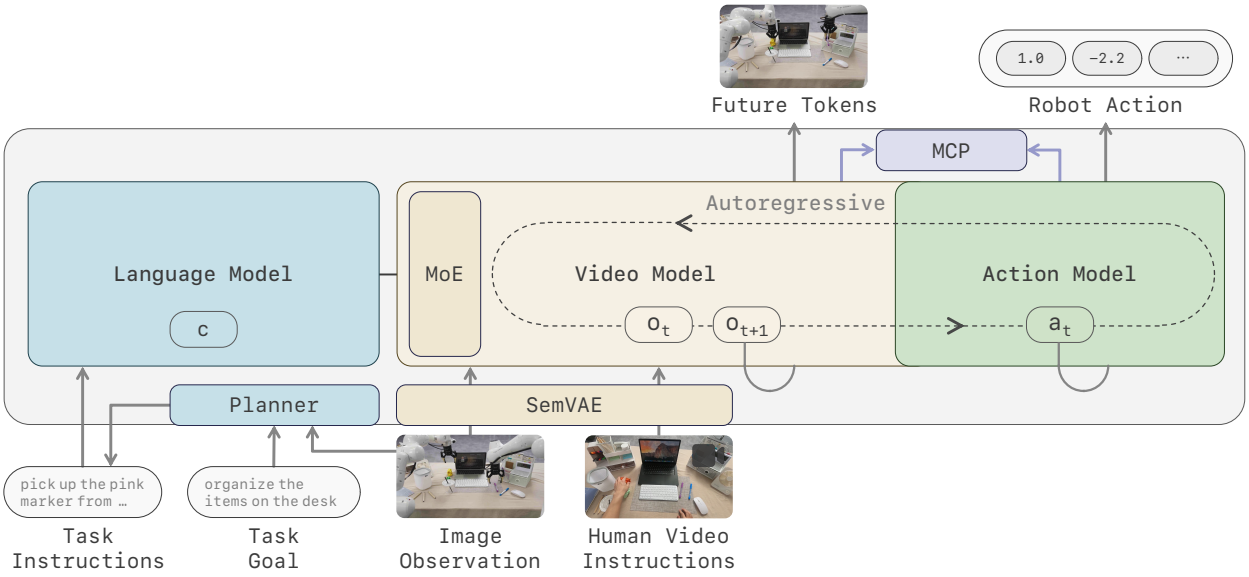

Figure 1. Overview of LingBot-VA 2.0**.** The Planner (Sec. 2.3.1) decomposes long-horizon goals into structured subtask context. The semantic visual-action tokenizer (Sec. 2.2) encodes image observations and human video prompts (Sec. 2.3.4) as visual latents. The video model with the MoE (Sec. 2.3.2) and the action model autoregressively predict future visual latents and robot actions. MCP (Sec. 2.3.3) adds future-chunk supervision, and the autoregressive rollout connects to asynchronous execution for real-time robot control. (Sec. 2.3.7).

2 Native Video-Action Model

This section presents the full LingBot-VA 2.0 stack (Fig. 1). After reviewing how prior video-action models repurpose generic video generators for control (Sec. 2.1), we build the stack natively in two stages. The first stage trains a semantic visual-action tokenizer, which aligns visual latents with a frozen foundation model and extracts latent actions from unlabeled video, yielding a shared visual-action latent space (Sec. 2.2). The second stage pretrains a causal video-action model on this space (Sec. 2.3): a high-level VLM *planner* decomposes long-horizon goals into structured subtask context (Sec. 2.3.1); the low-level policy scales its video stream with a sparse *Mixture-of-Experts* DiT (Sec. 2.3.2); *multi-chunk prediction* supervises multiple future chunks so the representation captures trajectory-level dynamics (Sec. 2.3.3); *in-context learning* admits human demonstration videos as visual task prompts (Sec. 2.3.4); and human–robot co-training brings egocentric human videos into the shared world model (Sec. 2.3.5), under the multi-task recipe of Sec. 2.3.6. At inference time, *Foresight Reasoning* overlaps prediction with execution and re-grounds each rollout on the latest real observation (Sec. 2.3.7); post-training further distills both experts into few-step samplers and applies inference acceleration for real-time control (Sec. 2.4).

2.1 Preliminary

We briefly review the video generative models that recent video-action models are built upon, together with the formulation by which they are repurposed for control.

2.1.1 Video Generative Models

Modern video generators comprise two components. First, a variational autoencoder (VAE) $(\mathcal{E}, \mathcal{D})$ compresses a raw video $o \in \mathbb{R}^{T\times H\times W\times 3}$ into a compact latent and reconstructs it,

$$z = \mathcal{E}(o) \in \mathbb{R}^{T'\times h\times w\times c}, \qquad \hat{o} = \mathcal{D}(z) \approx o, \tag{1}$$

with temporal and spatial downsampling factors $f_t = T/T'$ and $f_s = H/h = W/w$. Crucially, this VAE is trained purely for video compression: its objective is reconstruction fidelity, with no notion of actions or control. Second, a video generator—typically a flow-matching transformer [61, 93]—is trained in this latent space to synthesize z from

noise $\epsilon \sim \mathcal{N}(0, I)$ conditioned on a text prompt c_{text}. Concretely, it regresses the velocity field along the interpolation path $z^{(s)} = (1-s)\epsilon + s\, z$:

$$\mathcal{L}_{\text{gen}} = \mathbb{E}_{s,\epsilon,z}\left[\left\| v_\theta\big(z^{(s)}, s \mid c_{\text{text}}\big) - (z - \epsilon) \right\|^2 \right], \tag{2}$$

where the attention is bidirectional over all T' latent frames [32, 74, 95].

2.1.2 Repurposing Video Generative Models for Robot Control

Prior video-action models [50, 115] adapt this pretrained stack to robotics through *continued training* on robot data. Because the video tokenizer temporally downsamples the observation stream while robot actions are recorded at the original control frequency, one latent transition corresponds to a block of f_t low-level actions. We index the T' visual latents as $z_{0:N}$ with $N = T' - 1$, and write the raw action sequence as $u_{0:f_t N-1}$. We group the raw actions between two consecutive visual latents into an action chunk

$$a_t = u_{t f_t:(t+1) f_t - 1}, \qquad t = 0, \ldots, N-1, \tag{3}$$

aligned with the transition $z_t \to z_{t+1}$. Throughout the video-action model, a_t denotes this action chunk rather than an individual low-level control command. Two modifications are made. *(i) Video Causality.* The bidirectional attention of Eq. (2) is replaced by a causal mask, so that each latent frame depends only on the past, matching closed-loop control where the present cannot attend to the future:

$$p_\theta(z_{1:N} \mid z_0) = \prod_{t=0}^{N-1} p_\theta\big(z_{t+1} \mid z_{\le t}\big). \tag{4}$$

(ii) Action injection. A video-action model extends the video-only factorization of Eq. (4) to a conditional distribution over future latents and actions given the initial visual context:

$$p_\theta\big(z_{1:N}, a_{0:N-1} \mid z_0\big) = \prod_{t=0}^{N-1} \underbrace{p_\theta\big(z_{t+1} \mid z_{\le t}, a_{<t}\big)}_{\text{video generation: } (z_{\le t}, a_{<t}) \to z_{t+1}} \underbrace{p_\theta\big(a_t \mid z_{\le t}, a_{<t}, z_{t+1}\big)}_{\text{inverse dynamics: } (z_{\le t}, a_{<t}, z_{t+1}) \to a_t}. \tag{5}$$

Existing methods differ in how they parameterize Eq. (5). Joint prediction, as in DreamZero [115], concatenates video and action tokens and denoises each transition in a single shared stream. To make the transition supervision explicit, we write the video-action objective as separate video and action flow-matching losses. Let $z_{t+1}^{(s)} = (1-s)\epsilon_t^z + s z_{t+1}$ and $a_t^{(s)} = (1-s)\epsilon_t^a + s a_t$ denote their interpolated noisy targets:

$$\mathcal{L}_{\text{vid}} = \mathbb{E}_{t,s,\epsilon_t^z}\left[\left\| v_\theta^{\text{vid}}\big(z_{t+1}^{(s)}, s \mid z_{\le t}, a_{<t}\big) - \big(z_{t+1} - \epsilon_t^z\big) \right\|^2 \right], \tag{6}$$

$$\mathcal{L}_{\text{act}} = \mathbb{E}_{t,s,\epsilon_t^a}\left[\left\| v_\theta^{\text{act}}\big(a_t^{(s)}, s \mid z_{\le t}, a_{<t}, z_{t+1}\big) - \big(a_t - \epsilon_t^a\big) \right\|^2 \right]. \tag{7}$$

The complete video-action objective is $\mathcal{L}_{\text{VA}} = \mathcal{L}_{\text{vid}} + \lambda_{\text{act}} \mathcal{L}_{\text{act}}$. The Mixture-of-Transformers (MoT) of `LingBot-VA` [50, 58] assigns the two modalities to separate expert networks $\{v_\theta^{\text{vid}}, v_\theta^{\text{act}}\}$ that share a causal attention: the video expert predicts the next visual latent and the action expert decodes the action by inverse dynamics over the predicted transition,

$$\hat{z}_{t+1} = v_\theta^{\text{vid}}(z_{\le t}, a_{<t}), \qquad \hat{a}_t = v_\theta^{\text{act}}\big(z_{\le t}, a_{<t}, \hat{z}_{t+1}\big). \tag{8}$$

With a slight abuse of notation, v_θ^{vid} and v_θ^{act} denote the latent and action obtained by integrating their respective flow-matching velocity fields, rather than the velocity fields themselves.

2.2 Semantic Visual-Action Tokenizer

The compression-only VAE $(\mathcal{E}, \mathcal{D})$ of Sec. 2.1.1 is a standard pixel-reconstruction tokenizer [44, 95]: its latents are optimized for reconstruction alone, without semantic alignment or action supervision. We therefore follow RepWAM [98] and build our tokenizer as the first stage of `LingBot-VA 2.0`, augmenting reconstruction with two objectives: (1) a semantic alignment objective that pulls visual latents toward foundation-model features, and (2) a latent-action objective that extracts compact transition variables from consecutive latents.

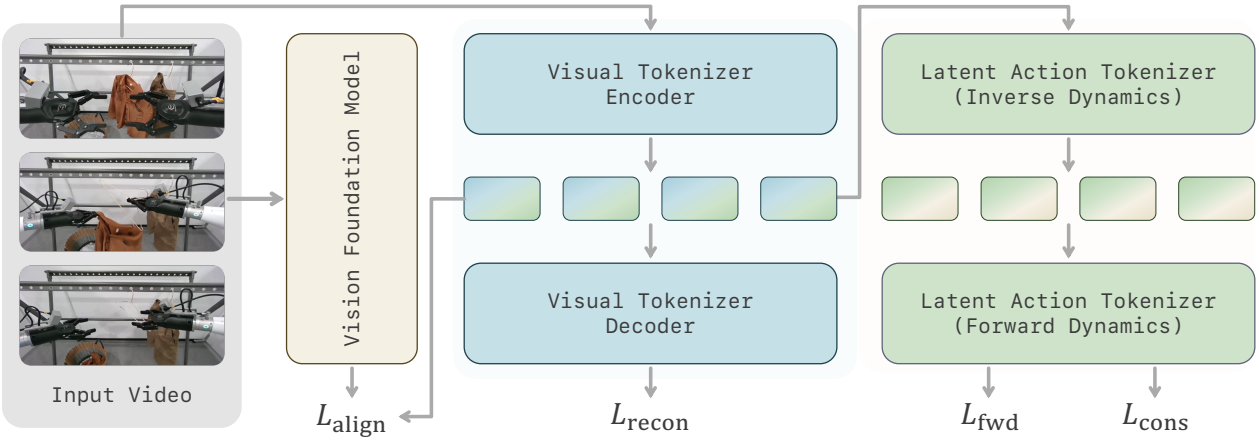

Figure 2. Semantic visual-action tokenizer. The visual tokenizer aligns reconstruction-oriented latents with visual foundation-model features, while the latent action tokenizer learns compact transition variables through inverse and forward dynamics. Adapted from RepWAM [98].

2.2.1 Visual Tokenization with Semantic Alignment

We adopt a Vision Transformer (ViT) autoencoder for visual tokenization [25, 97]. Given a video clip $o_{1:T}$, our tokenizer encodes the clip in two parts. The first frame is encoded as 16×16 spatial patches, and the subsequent frames are encoded as $4\times16\times16$ spatiotemporal tubelets. The resulting tokens are concatenated along the sequence dimension and passed to the encoder $\mathcal{E}$, which outputs visual latents $z = \mathcal{E}(o)$. Our encoder uses full spatial attention within each frame and causal attention across frames. A symmetric decoder $\mathcal{D}$ then decodes the latents back to pixel space, $\hat{o} = \mathcal{D}(z)$.

The reconstruction loss combines pixel, perceptual, and adversarial terms:

$$\mathcal{L}_{\text{rec}} = \lambda_1 \|o - \hat{o}\|_1 + \lambda_{\text{perc}} \mathcal{L}_{\text{perc}}(o, \hat{o}) + \lambda_{\text{gan}} \mathcal{L}_{\text{gan}}(\hat{o}). \tag{9}$$

For semantic alignment, we use a frozen Perception Encoder [10] as the teacher model G. Since the tokenizer latent and teacher feature may have different dimensions, a learnable projection W_{align} maps z to the teacher feature dimension. We then match the temporally pooled representations:

$$\mathcal{L}_{\text{align}} = \left\| \operatorname{avg}(W_{\text{align}}\, z) - \operatorname{avg}\left(G(o)\right) \right\|_2^2, \tag{10}$$

where avg denotes temporal average pooling. This matches the clip-level semantics of G while preserving per-frame information for reconstruction. Overall, the visual tokenization objective is:

$$\mathcal{L}_{\text{vis}} = \mathcal{L}_{\text{rec}} + \lambda_{\text{align}} \mathcal{L}_{\text{align}}, \tag{11}$$

where λ_{align} controls the strength of semantic alignment.

2.2.2 Latent Action Tokenization

We next build a latent action tokenizer upon these semantic visual latents without using action labels. Following latent action models that infer actions from observation transitions [13, 14, 20], each latent action is defined as a compact transformation between two visual latents. We freeze the visual tokenizer and train an inverse dynamics model (IDM) q_ϕ together with a forward dynamics model (FDM) f_ψ. For two consecutive visual latents, the IDM predicts:

$$\ell_t = q_\phi(z_t, z_{t+1}) \in \mathbb{R}^{d_\ell}, \qquad d_\ell \ll \dim(z_t). \tag{12}$$

The FDM decodes ℓ_t into a transport map and a residual, and reconstructs the next latent:

$$\hat{z}_{t+1} = f_\psi(z_t, \ell_t) = K_t\, z_t + \delta_t, \tag{13}$$

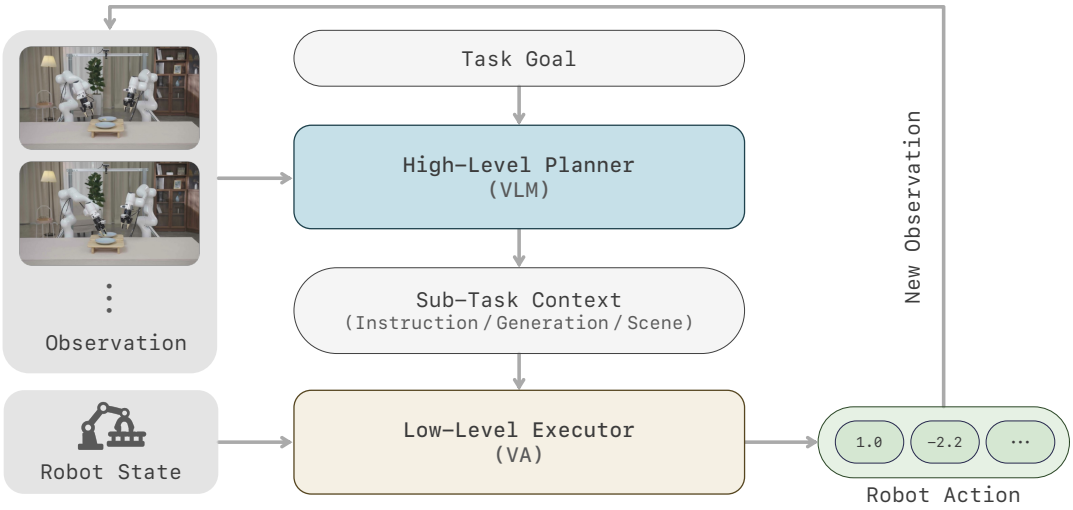

Figure 3. Dual-system hierarchical policy. A high-level VLM planner consumes the task goal and sparse recent observations to produce structured subtask context, which conditions the low-level video-action policy together with robot state to produce closed-loop actions.

where (K_t, δ_t) are predicted from ℓ_t. The transport K_t moves information across spatial latent tokens, while δ_t accounts for changes that cannot be explained by transport alone. A reverse transport $\bar{f}_\psi$ predicts the previous latent from (z_{t+1}, ℓ_t) as $\hat{z}_t = \bar{f}_\psi(z_{t+1}, \ell_t)$. Training uses forward prediction and backward consistency on unlabeled video:

$$\mathcal{L}_\ell = \sum_t \left\|\hat{z}_{t+1} - z_{t+1}\right\|_2^2 + \left\|\hat{z}_t - z_t\right\|_2^2. \tag{14}$$

The bottleneck on ℓ_t prevents it from copying the full visual state and encourages it to capture control-relevant changes. It therefore provides an action latent for modeling the transition from z_t to z_{t+1} in the video-action model. Our tokenizer thus returns paired visual-action latents $(z_{0:N}, \ell_{0:N-1})$ for each video clip, which serve as the training targets for the second-stage video-action model, in which we write the learned latent action token as $a_t \equiv \ell_t$ to match the notation in Eq. (5).

2.3 Video-action Model Pretraining and Inference

Building on the semantic visual-action tokenizer, the second stage pretrains a causal video-action model on the shared video-action latent space. We organize it into two levels: a high-level *planner* that decomposes the language goal into subtasks and tracks their progress, and a low-level *video-action policy* that realizes each subtask by predicting future visual latents and the latent actions connecting them. **The policy operates at the granularity of *chunks*, so throughout this subsection the index t ranges over chunks rather than individual frames.**

2.3.1 Hierarchical Planning

While the video-action policy handles continuous control within a single subtask, long-horizon tasks require decomposing a high-level goal into subtasks, tracking which subtask is currently active, and deciding when to transition to the next one. We implement the planner as a pretrained vision-language model (VLM) that is fine-tuned to produce structured subtask context for the policy. The VLM backbone provides visual-language grounding out of the box, allowing the planner to assess task state from sparse visual observations without learning scene understanding from scratch.

The planner and the policy run at different frequencies and are decoupled through an asynchronous shared buffer. The planner runs in the background at $\sim$2 Hz and writes structured subtask context into the buffer; the policy reads the latest context at each action-chunk boundary as its conditioning signal. This ensures that planner inference latency does not block the policy's execution loop.

Planner Output Schema. The planner's output format must exactly match the conditioning fields that the video-action policy sees during training; any mismatch causes the policy to receive out-of-distribution conditioning at inference time. Under this constraint, we define the planner output as a structured JSON with four fields:

- `done`: a completion signal indicating whether the current subtask should continue or the planner should transition to the next one.
- `instruction`: a compact execution directive for the current or next subtask, e.g., "Pick up the red bottle on the table." This is the *primary conditioning field* consumed by the policy during training.
- `generation_instruction`: a richer description of the expected robot motion and object interaction within the subtask, giving the policy a more specific execution cue.
- `local_scene_description`: an observation-grounded description of the spatial layout and object state, providing local context for the policy.

The `done` field is used only by the high-level scheduler to decide whether to keep or refresh the current subtask. The low-level policy is conditioned on the three textual fields: `instruction` provides the compact command, while `generation_instruction` and `local_scene_description` provide more detailed action and scene context when available.

Planner Data Construction. Training data for the planner is constructed from robot demonstration videos with segment-level annotations. Each video contains an episode-level task goal and a temporal sequence of segments $[g_0, g_1, \ldots, g_N]$, where each segment is annotated with the four fields above.

The core construction method is *boundary-crossing sampling*, designed to directly align with the planner's online deployment scenario. At each segment boundary, we generate two types of training samples:

- **done = false**: the observation time t^* falls inside the current segment g_i; the target is the four fields of g_i (echo the current subtask).
- **done = true**: the observation time t^* has just crossed the $g_i \to g_{i+1}$ boundary; the target is the four fields of g_{i+1} (predict the next step).

Each sample's input consists of: (i) three keyframes at t^*-2 s, t^*-1 s, and t^*, providing short-term temporal context; (ii) the episode-level task goal; (iii) the full text history of all completed prior segments $g_0, \ldots, g_{i-1}$; and (iv) the current subtask description (for done = false samples only). The `done = true` samples thus require the planner to predict the upcoming subtask from the visual observation and task history, rather than merely echoing the current one.

The training set is task-balanced to prevent high-frequency tasks from dominating.

Planner Training. The planner is trained by LoRA fine-tuning a pretrained VLM backbone with a frozen vision tower. This keeps the pretrained visual grounding ability intact while adapting the language head to the structured subtask schema. The planner is therefore treated as an interface module that supplies local language context to the video-action policy, rather than as a separately optimized policy component.

2.3.2 Architecture: Mixture-of-Experts Video Stream

The low-level video-action policy keeps the Mixture-of-Transformers (MoT) structure of Eq. (8): the video and action modalities are handled by two experts $\{v_\theta^{\text{vid}}, v_\theta^{\text{act}}\}$ that share one causal self-attention but own separate feed-forward pathways. Because visual dynamics carry most of the modeling burden, we scale the two streams asymmetrically: the *video expert* replaces the dense feed-forward network in each causal DiT block with a sparse Mixture-of-Experts (MoE) routed layer, while the *action expert* remains dense. The MoE design follows the sparse scaling principle of DeepSeekMoE and DeepSeek-V3 [23, 24], instantiated inside our causal video-action DiT. Self-attention remains causal over the visual-action history and cross-attention remains conditioned on language. Since routing is applied after causal self-attention as a token-wise feed-forward operation, increasing the number of experts increases the video stream's capacity while preserving the same causal dependency structure described in Eq. (5).

For a hidden token $h \in \mathbb{R}^d$, the router uses a gate matrix $W_g = [g_1, \ldots, g_{N_e}]^\top \in \mathbb{R}^{N_e \times d}$ to compute expert scores $r(h) \in \mathbb{R}^{N_e}$,

$$r_i(h) = \sigma(g_i^\top h), \qquad i = 1, \ldots, N_e, \tag{15}$$

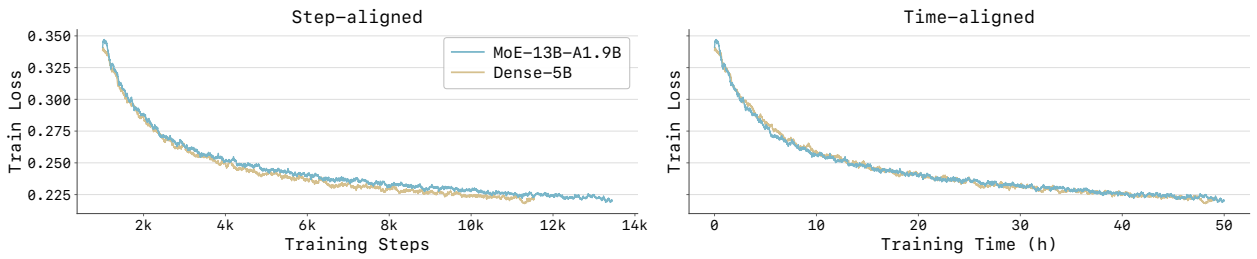

Figure 4. Training loss comparison between the MoE-13B-A1.9B video-stream model and the Dense-5B baseline. Left: loss aligned by optimization steps, where both models remain close and the dense baseline is slightly lower. Right: loss aligned by wall-clock training time, where the final losses nearly coincide.

where $\sigma(\cdot)$ is the sigmoid function and N_e is the number of routed experts. We use group-limited top-k routing. Experts are partitioned into groups; a small number of groups are first selected by their strongest router scores, and the final k experts are then chosen within those groups. The selected set is

$$\mathcal{R}(h) = \mathrm{TopK}_{\mathrm{group}}\big(r(h) + b, k\big), \qquad \mathcal{R}(h) \subset \{1, \dots, N_e\},\ |\mathcal{R}(h)| = k, \tag{16}$$

where $b \in \mathbb{R}^{N_e}$ is an expert-wise correction bias used only for expert selection. The top-k scores are normalized and scaled as

$$\alpha_i(h) = \gamma \frac{r_i(h)}{\sum_{j \in \mathcal{R}(h)} r_j(h)}, \qquad i \in \mathcal{R}(h), \tag{17}$$

where $\alpha_i(h)$ is a scalar routing weight and γ is a routed scaling factor. The MoE output is then

$$\mathrm{MoE}(h) = E_{\mathrm{shared}}(h) + \sum_{i \in \mathcal{R}(h)} \alpha_i(h) E_i(h), \tag{18}$$

where $E_{\mathrm{shared}} : \mathbb{R}^d \to \mathbb{R}^d$ captures common transformations shared by all tokens, and $E_i : \mathbb{R}^d \to \mathbb{R}^d$ are routed SwiGLU experts. In our main configuration, N_e=128 and k=8, so each token activates only a small fraction of the routed expert pool while the full parameter set remains available across the sequence.

Stable MoE training requires balanced expert utilization. We therefore use the auxiliary-loss-free load-balancing strategy of Loss-Free Balancing [99], also adopted in DeepSeek-V3 [24], and follow the auxfree bias update used in Moonlight [64]. After each optimization step, the correction bias is updated from the observed token count c_i assigned to each expert,

$$v_i = \mathrm{sign}(c_i - \bar{c}), \qquad b_i \leftarrow b_i - \eta_{\mathrm{lb}} \left(v_i - \frac{1}{N_e} \sum_{j=1}^{N_e} v_j \right), \tag{19}$$

where $\bar{c} = \frac{1}{N_e} \sum_{i=1}^{N_e} c_i$ is the mean expert load and η_{lb} is the balance update rate. This discourages overloaded experts and promotes underused experts without injecting large balancing gradients into the video-action diffusion objective. We keep a lightweight sequence-wise router regularizer and expert-utilization statistics for monitoring. The forward computation is executed with grouped expert matrix multiplications for efficient long-sequence training.

As a final check of the optimization behavior of this sparse video-stream design, we compare the training loss of the MoE-13B-A1.9B model with a Dense-5B baseline under the same training setup. As shown in Fig. 4, when compared at the same number of optimization steps, the two models follow very similar loss trajectories, with the dense baseline being slightly lower. When compared at matched wall-clock training time, their final losses nearly coincide. This indicates that the proposed MoE video stream increases the total model capacity without introducing a noticeable optimization penalty under the considered training budget.

2.3.3 Multi-chunk Prediction

Following LingBot-VA [50], we train the video-action model with teacher forcing, predicting each chunk from the ground-truth history. However, standard teacher-forced video-action training supervises the model only on the current

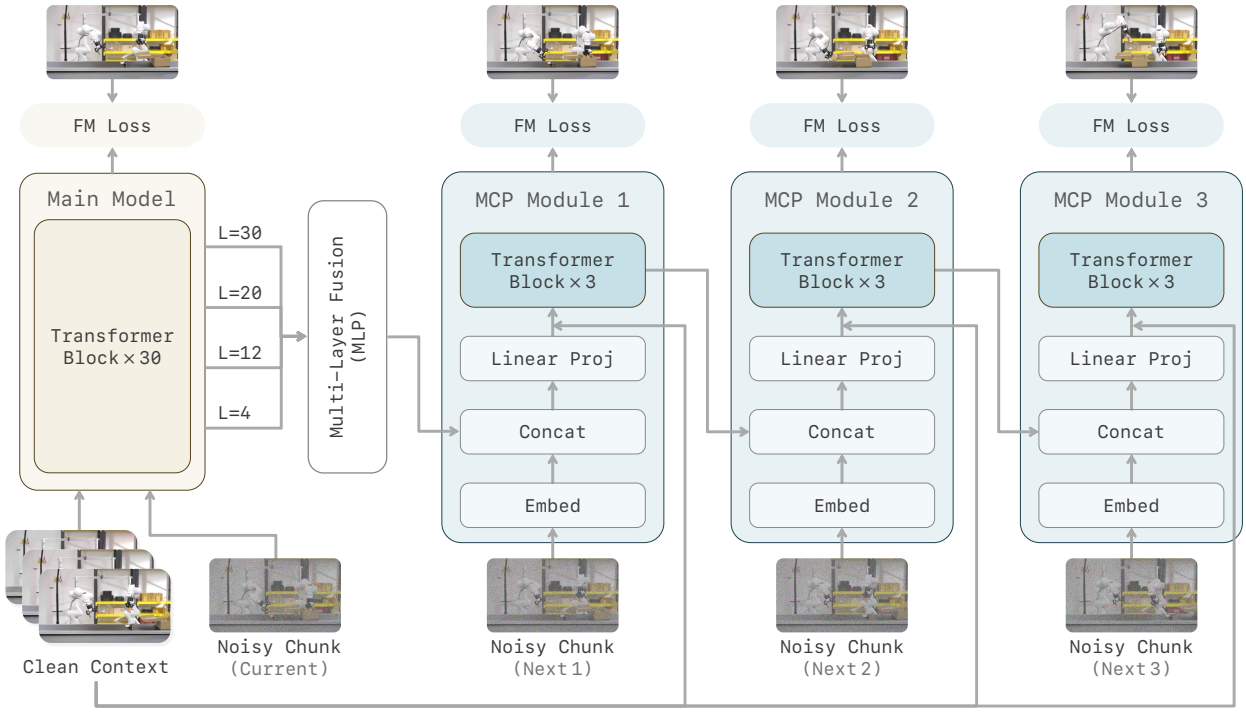

Figure 5. Multi-Chunk Prediction (MCP). Lightweight auxiliary modules predict several future latent chunks at increasing horizons from the main model's representation, providing dense temporal supervision during training. Adapted from Next Forcing [111].

chunk, creating a *myopic supervision* problem in which the learning signal becomes overly local: adjacent chunks, especially at high frame rates, are often visually similar enough that the model can reduce loss by copying appearance rather than learning the underlying dynamics.

This is misaligned with the goal of `LingBot-VA 2.0`, which is to pretrain a native video-action foundation model whose representations capture how the physical world evolves under actions, rather than only preserving short-term visual continuity. Building on the semantic visual-action latent space produced by our tokenizer, we therefore adopt Multi-Chunk Prediction (MCP), following Next Forcing [111], as an auxiliary training objective that makes the causal video-action policy's visual representations predictive of future state transitions.

Teacher forcing supervises only the next-chunk conditional of the visual stream,

$$p_\theta\big(z_{t+1} \mid z_{\le t}, a_{<t}\big), \tag{20}$$

matching each prediction against the immediate next chunk under the ground-truth visual-action prefix $(z_{\le t}, a_{<t})$. MCP instead predicts the whole block of the next K chunks from this context, supervising horizon-wise future conditionals

$$p_\theta\big(z_{t+1:t+K} \mid z_{\le t}, a_{<t}\big) \approx \prod_{k=1}^{K} p_{\theta,k}\big(z_{t+k} \mid h_t^{(k-1)}\big), \qquad h_t^{(0)} = f_\theta(z_{\le t}, a_{<t}), \tag{21}$$

so the representation at step t must encode how the scene evolves over the next K chunks, not just the next one. Here f_θ denotes the main DiT's feature extraction over the ground-truth prefix, and $h_t^{(k-1)}$ is the internal feature state passed through the stacked MCP modules, not a clean or predicted future observation. Thus the predictor for horizon $t+k$ reuses previous MCP features rather than conditioning on $z_{t+1:t+k-1}$. Each horizon k is trained with the same flow-matching objective as the main model. Specifically, with $z_{t+k}^{(s)} = (1-s)\epsilon_t^k + s z_{t+k}$, the MCP loss is

$$\mathcal{L}_k^{\mathrm{MCP}} = \mathbb{E}_{t,s,\epsilon_t^k}\Big[\big\| v_{\theta,k}^{\mathrm{MCP}}\big(z_{t+k}^{(s)}, s \mid h_t^{(k-1)}\big) - \big(z_{t+k} - \epsilon_t^k\big)\big\|^2\Big], \tag{22}$$

where $v_{\theta,k}^{\mathrm{MCP}}$ is the velocity field of the k-th MCP module.

We realize each conditional in Eq. (21) with a lightweight MCP module attached to the main DiT (Fig. 5), one per horizon. Following Next Forcing [111], we use three future chunks ($K = 3$; next^1, next^2, next^3). Each module denoises a temporally shifted visual-latent target under the same flow-matching formulation, but with a larger timestep shift, so it cannot solve the task from its own noisy input and must instead rely on the main model's representation. The modules form a causal chain: the first consumes fused intermediate features from several DiT layers, and each deeper module additionally conditions on the previous one, letting near-future predictions inform farther-future ones. Gradients from future-chunk denoising thus flow back into the backbone as dense temporal supervision, pushing it to organize its latents around trajectory-level dynamics rather than local appearance. MCP acts only on the visual stream, but the more predictive visual states also improve action decoding through the shared video-action attention and inverse-dynamics pathway.

The MCP auxiliary objective combines the horizon losses with the same weights as Next Forcing, $(w_1, w_2, w_3) = (0.5, 0.2, 0.1)$:

$$\mathcal{L}_{\text{MCP}} = \sum_{k=1}^{K} w_k \mathcal{L}_k^{\text{MCP}}. \tag{23}$$

At deployment, the same trained checkpoint supports two modes: the MCP modules can be discarded for zero-overhead inference, leaving the main policy improved purely by training-time supervision, or retained to predict the next visual chunk in parallel with the current one, reducing rollout latency for closed-loop control.

2.3.4 In-context Learning

Existing video-action models rely on language instructions as the primary task specification. The robot receives a textual command and is expected to infer the intended manipulation procedure from language alone. However, for rare-object manipulation, complex multi-object interaction, and long-horizon tasks, language descriptions are rarely exhaustive and may fail to specify the fine-grained temporal structure of the operation.

To provide a richer task specification, following Zero-WAM [60], we introduce video in-context learning (ICL). In each ICL sample, the robot video is paired with a semantically aligned human demonstration video. The human video does not need to share the same object instance, viewpoint, or scene layout as the robot video. Instead, it serves as a dynamic visual example that demonstrates how the manipulation task should unfold. Compared with language-only instructions, such a visual context provides explicit temporal and procedural guidance, helping the model follow complex task semantics and generalize to unseen open-world tasks [129].

Given a human demonstration video, we encode it with our semantic tokenizer into an in-context latent z_{icl}. Following the teacher-forced video-action pretraining objective in Eq. (5), the ICL human video augments robot prediction by conditioning every robot chunk on z_{icl}. For a paired robot trajectory with visual chunks $z_{0:N}$ and action chunks $a_{0:N-1}$, the in-context video-action objective is:

$$p_\theta\big(z_{1:N}, a_{0:N-1} \mid z_0, z_{\text{icl}}\big) = \prod_{t=0}^{N-1} p_\theta\big(z_{t+1} \mid z_{\le t}, a_{<t}, z_{\text{icl}}\big)\, p_\theta\big(a_t \mid z_{\le t}, a_{<t}, z_{t+1}, z_{\text{icl}}\big). \tag{24}$$

Here, the in-context latent z_{icl} is treated as an external task prompt and is available to all robot prediction steps, while the robot-side causal mask prevents each chunk z_t from attending to future robot chunks. Since z_{icl} comes from a separate human demonstration rather than the future of the robot trajectory, attending to the full demonstration does not violate robot-side causality. The same conditioning signal guides both future robot video prediction and the corresponding action prediction.

2.3.5 Human-Robot Data Co-training

Robot manipulation data is expensive to collect and limited in diversity, while egocentric human manipulation videos are cheap to scale and cover a much broader range of tasks and scenes. Since prior work shows that human data transfers to robot policies, especially under scarce robot data [40, 52, 96, 128], we *co-train* on human and robot data together rather than pretraining on human videos and adapting later [66, 67], allowing the shared video-action model to improve real-robot control throughout training.

Naive joint training faces two gaps: an *action-space mismatch* between human hands and robot grippers, and a *motion gap* between natural human motion and robot kinematics [52]. These gaps are especially harmful for a video-action *world model*, where past actions feed back into the prediction context (Eq. (5)) and misaligned human actions can corrupt the shared dynamics. We address them by mapping human hand poses into the robot action space while keeping embodiment-specific action encoders and decoders around a shared world model. Following [52], we additionally map the hand to a functional gripper opening rather than a placeholder, so human actions also supervise when to grasp.

We co-train two corpora that share the observation o but differ in their native actions: a robot set $\mathcal{D}_R = \{(o, a^R)\}$ and a human set $\mathcal{D}_H = \{(o, \tilde{a}^H)\}$. All actions are expressed in a unified cross-embodiment action layout in which channels missing for an embodiment are zero-padded and masked; its d_a-dimensional bimanual end-effector slice $\mathcal{A}$ holds, for each hand, a 6-DoF root pose together with a scalar gripper opening. Human hand labels are retargeted into exactly this slice by a map Φ, with the remaining channels padded and masked as for any other embodiment:

$$a^H = \Phi(\tilde{a}^H) \in \mathcal{A}. \tag{25}$$

Here Φ keeps the two 6-DoF hand-root poses and maps each hand's finger configuration q to a single scalar gripper opening $\varphi(q)$, populating $\mathcal{A}$ exactly as a robot action does. This functional opening is what replaces the no-op gripper placeholder used in prior co-training [52]. After retargeting, the two domains share an *identical* action representation; what remains is the motion gap—the same tuple carries different distributions and physical meaning for a human hand and a robot end-effector—which the per-domain action heads absorb.

Both domains pass through one shared world model and differ only in a per-domain action encoder E_d and decoder P_d ($d \in \{R, H\}$); the video expert v^{vid}, the action expert v^{act}, and the causal attention of Eq. (8) are all shared:

$$\hat{a}_t^{(d)} = P_d\big(v^{\text{act}}(z_{\le t},\, E_d(a_{<t}),\, \hat{z}_{t+1})\big). \tag{26}$$

The objective sums both domains, each decoded through its own action head:

$$\mathcal{L}_{\text{co-train}} = \mathbb{E}_{\mathcal{D}_R}\big[\mathcal{L}_{\text{vid}} + \mathcal{L}_{\text{act}}^R\big] + \mathbb{E}_{\mathcal{D}_H}\big[\mathcal{L}_{\text{vid}} + \mathcal{L}_{\text{act}}^H\big]. \tag{27}$$

Here $\mathcal{L}_{\text{vid}}$ and $\mathcal{L}_{\text{act}}^d$ are the model's flow-matching video and action losses, the action term decoded through the domain head P_d. The shared video and action experts see both domains and learn cross-domain dynamics from human data; the domain-specific heads E_d, P_d only isolate the low-level action parametrization of each embodiment from the shared representation. At deployment we use only the robot branch (E_R, P_R), so co-training adds no inference cost.

Concretely, the hand-to-gripper map φ treats the hand as a virtual parallel gripper, with the thumb as one jaw and the four fingers as the other. From the captured finger joints we recover 3D fingertip positions by forward kinematics and take the opening as the distance between the two sides along the closing direction, yielding a metric aperture that is quantile-normalized per dataset in the same way as the robot gripper channels. Measuring this over the whole finger envelope rather than a single thumb-to-index pair keeps it stable when a finger is mistracked. The 6-DoF hand-root poses are kept as is, with no conversion. The corpus and data pipeline are described in Sec. 3.3. During training, the robot and human action heads in Eq. (26) are selected according to each sample's embodiment, while the shared video-action backbone is updated by both domains. We initialize the human action heads from the robot heads, providing a stable starting point for co-training.

2.3.6 Training Recipe

Generic video generators such as Wan [95] are pretrained for synthesis rather than embodied interaction, and their bidirectional attention conflicts with the causal regime of closed-loop control (Eq. (4)); combined with our purpose-built tokenizer, whose semantic latent space differs from that of any off-the-shelf generator, this leads us to build the pretraining recipe from scratch rather than continue-train a borrowed backbone.

Rather than training the video-action model in fully separate, sequential stages, we adopt a *multi-task co-training* recipe in which all objectives are optimized jointly throughout pretraining and only their relative mixing weights change over time. A strictly staged curriculum—first text-to-image, then text-to-video, then video-action—risks catastrophically forgetting the broad appearance and dynamics priors acquired earlier once training narrows to scarce embodied data. Co-training instead keeps every objective present at all times, so the model retains general world knowledge while

progressively specializing toward control. All five tasks are instantiated on the shared semantic latent space of our tokenizer and the same causal DiT backbone, differing only in their conditioning and supervision:

- **T2I** (text-to-image): single-frame text-conditioned generation that builds visual-semantic grounding and aligns the latent space with language.
- **T2V** (text-to-video): text-conditioned multi-frame generation that learns general temporal dynamics and motion priors from web-scale video.
- **TI2VA** (text-image to video-action): the core control task, combining future-latent *video prediction* with *inverse-dynamics modeling* of the latent actions that connect consecutive states (Eq. (5)), conditioned on a language instruction and an observation history.
- **ICL** (in-context learning): given a demonstration video as additional context, the model generates the video-action rollout for a new instance of the same task, enabling few-shot adaptation without weight updates.
- **HCT** (human-robot co-training): joint training on egocentric human videos and robot data through a shared world model with embodiment-specific action encoders and decoders (Sec. 2.3.5), transferring low-cost, diverse human demonstrations to robot control.

All tasks share parameters and are optimized jointly under a single mixed objective. Let $\mathcal{T}$ denote the task set containing T2I, T2V, TI2VA, ICL, and HCT, and let each task $i \in \mathcal{T}$ contribute a loss $\mathcal{L}_i$ on the shared latent space and DiT backbone. At training progress $\tau \in [0, 1]$ we sample tasks from a distribution $\pi(\tau)$ over $\mathcal{T}$ and minimize the weighted objective

$$\mathcal{L}(\tau) = \sum_{i \in \mathcal{T}} \pi_i(\tau)\, \mathcal{L}_i, \qquad \pi_i(\tau) \geq 0, \quad \sum_{i \in \mathcal{T}} \pi_i(\tau) = 1. \tag{28}$$

Here $\mathcal{L}_{\text{T2I}}$ and $\mathcal{L}_{\text{T2V}}$ are flow-matching generation losses (Eq. (2)); $\mathcal{L}_{\text{TI2VA}}$ combines the video-prediction and inverse-dynamics terms of Eq. (5) with the multi-chunk objective $\mathcal{L}_{\text{MCP}}$ (Eq. (23)); $\mathcal{L}_{\text{ICL}}$ is the in-context objective of Eq. (24); and $\mathcal{L}_{\text{HCT}}$ is the human-robot co-training loss of Sec. 2.3.5.

The sampling distribution $\pi(\tau)$ follows a *coarse-to-fine* schedule. Early training ($\tau \to 0$) places most mass on T2I to establish appearance and semantic alignment; the mass then shifts toward T2V to consolidate temporal dynamics; and late training ($\tau \to 1$) concentrates on the video-action tasks (TI2VA, ICL, HCT) to specialize the shared representation for action-relevant control. Because all tasks share parameters, earlier-emphasized objectives still provide a small but nonzero regularizing signal in later phases, preserving the broad image- and video-priors while the model adapts to embodied control.

2.3.7 Foresight Reasoning

In the multi-stream view of Multi-Stream LLMs [87], a single serial stream of computation is self-blocking: the model must finish computing before the robot can act, and the robot must finish acting before a new observation returns. We therefore organize the closed-loop rollout as two causally coupled streams that advance at different rates: a *prediction stream*, in which the model predicts future visual latents and the actions decoded from them, and an *execution stream*, in which the robot executes the current action chunk in the real world. Run synchronously, the two streams stall each other: after executing chunk a_t the robot idles while the model denoises the next chunk, turning model latency into control latency.

Foresight Reasoning runs the two streams *asynchronously*. While the robot executes action chunk a_t, the prediction stream prepares the next action chunk a_{t+1} before the real observation of a_t has returned. It first appends the currently executing action chunk a_t to the latest feedback-grounded cache C_t—the KV-cache entries summarizing the grounded history $(z_{\leq t}, a_{<t})$—giving the temporary context $C_{\text{tmp}} = C_t \cup \{a_t\}$. The policy's video expert then runs a forward-dynamics pass, performed online by the policy itself rather than by the tokenizer's frozen FDM f_ψ, that imagines the visual outcome of the current chunk,

$$\hat{z}_{t+1} = \text{FDM}_\theta\big(C_t \cup \{a_t\}\big) = v_\theta^{\text{vid}}\big(z_{\leq t},\, a_{\leq t}\big), \tag{29}$$

and the action expert decodes the next action chunk from $C_{\text{tmp}} \cup \{\hat{z}_{t+1}\}$. Thus a_{t+1} is ready when execution of a_t finishes, hiding prediction latency behind motion. Running ahead, however, is not free: if the prediction stream keeps feeding its own $\hat{z}_{t+1}$ back into the context, the rollout becomes open-loop; because the video backbone favors temporal

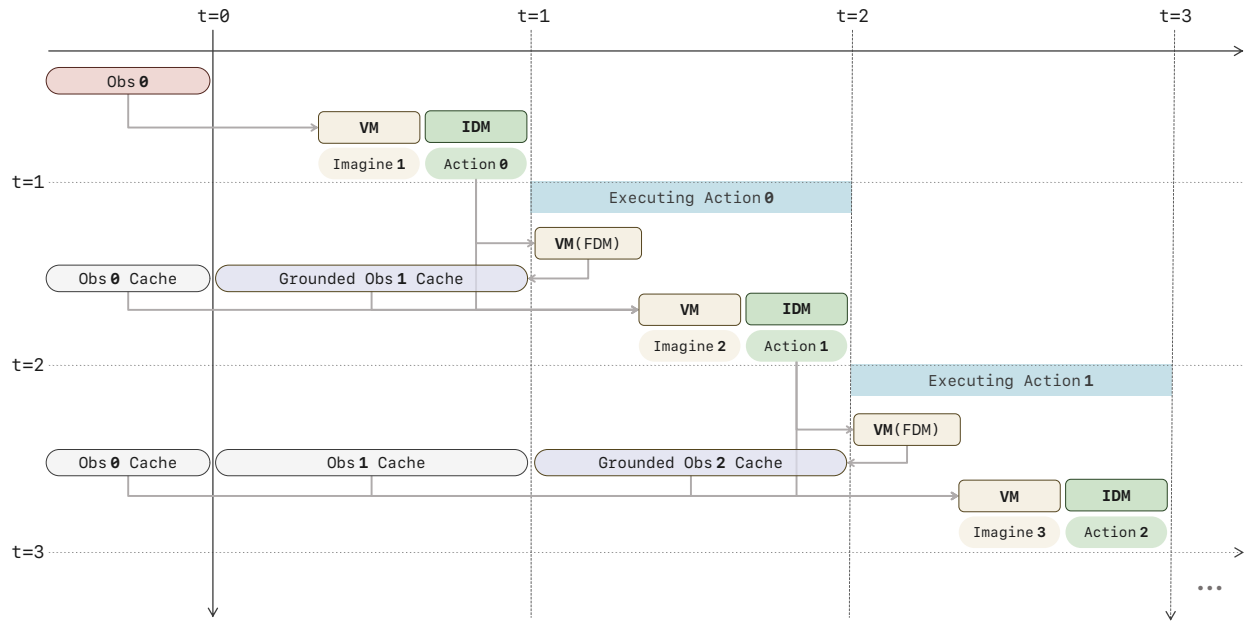

Figure 6. Foresight Reasoning as an asynchronous inference/execution flow. A cold start initializes the KV cache and precomputes the first action chunk. During the loop, the executor runs the current chunk while real observations enter ObsQueue; the inference branch re-grounds the KV cache with those observations, appends the currently executing action, imagines its visual outcome with FDM grounding, and predicts the next action chunk before execution stalls.

smoothness, it continues this hallucinated trajectory, ignores physical feedback, and drifts until the policy can no longer react.

We therefore re-ground the rollout whenever a real observation arrives. After chunk a_t completes, the returned observation is encoded into the true latent z_{t+1} and overwrites the stale predicted context $\hat{z}_{t+1}$ in the KV cache, together with the executed action chunk a_t. The next prediction step therefore starts from the feedback-grounded cache C_{t+1} rather than from an unverified imagined latent. The asynchronous rollout thus stays closed-loop—a *predict-then-correct* step in which the fast prediction stream drafts ahead while each real observation pulls it back. To train the video expert for this role, post-training adds a forward-dynamics grounding loss that supervises it to predict the next visual latent from the feedback-grounded context and executed action, following the flow-matching objective of Eq. (2):

$$\mathcal{L}_{\text{FDM}} = \mathbb{E}_{t,\, s,\, \epsilon}\left[\left\| v_\theta^{\text{vid}}\big(z_{t+1}^{(s)}, s \mid z_{\le t},\, a_{\le t}\big) - (z_{t+1} - \epsilon) \right\|^2 \right], \tag{30}$$

matching the conditioning of Eq. (29) used during asynchronous inference. Unlike the video loss $\mathcal{L}_{\text{vid}}$ (Eq. (6)), which conditions only on the past actions $a_{<t}$, this objective additionally feeds the executed action a_t (i.e. $a_{\le t}$), turning the video expert into a genuine forward-dynamics predictor of the current transition. Unlike Multi-Stream LLMs [87], whose streams carry input/output roles, our streams carry predicted future world states, so closed-loop control hinges on re-grounding them against real observations—a step with no counterpart in the language setting. Foresight Reasoning is an inference-time scheme independent of the MCP modules of Sec. 2.3.3: the two can be enabled separately.

2.4 Post-training

2.4.1 Diffusion Distillation

The post-trained policy samples each chunk by integrating the flow-matching velocity field along the probability-flow (PF) ODE; in LingBot-VA this defaults to 5 denoising steps for the video expert and 10 for the action expert. To make the policy real-time, we distill it into a few-step *consistency model* [85]. A consistency model $f_\xi(x^{(s)}, s)$ maps any point on a PF-ODE trajectory directly to its clean endpoint, satisfying the self-consistency property $f_\xi(x^{(s)}, s) = f_\xi(x^{(s')}, s')$ for any s, s' on the same trajectory and the boundary condition $f_\xi(x^{(1)}, 1) = x^{(1)}$ at the data end. To lighten notation,

x, v_θ, and f_ξ stand for either expert—the video expert acting on visual latents z or the action expert acting on action chunks a—each retaining its usual causal context conditioning $(z_{\le t}, a_{<t}, \dots)$, which we suppress here.

We take the post-trained model v_θ as the frozen teacher and discretize $s \in [0,1]$ into a grid $s_0 < \cdots < s_M{=}1$. For a clean target x—a visual latent z_{t+1} for the video expert or an action chunk a_t for the action expert—and noise ϵ, we form the noisy point $x^{(s_n)} = (1-s_n)\epsilon + s_n x$ and take one teacher Euler step toward the data end,

$$\hat{x}^{(s_{n+1})} = x^{(s_n)} + (s_{n+1} - s_n)\, v_\theta\big(x^{(s_n)},\, s_n\big). \tag{31}$$

The student is then trained to give consistent predictions at the two adjacent points,

$$\mathcal{L}_{\text{CD}} = \mathbb{E}_{x,\epsilon,n}\Big[\, d\big(f_\xi(x^{(s_n)}, s_n),\ f_{\xi^-}(\hat{x}^{(s_{n+1})}, s_{n+1})\big)\Big], \tag{32}$$

where $\xi^- = \mathrm{stopgrad}(\xi)$ is an exponential-moving-average target network and $d(\cdot,\cdot)$ is a distance metric (we use squared ℓ_2). We distill both experts, reducing the video and action samplers from 5 and 10 steps to 2 steps each, so the deployed policy produces every chunk in two function evaluations.

2.4.2 Inference Acceleration

We organize inference acceleration into three levels: model-level compilation, sequence-level attention optimization, and system-level runtime amortization. These levels correspond to different parts of the inference stack. At the model-execution level, we reduce the cost of each DiT forward pass. At the autoregressive sequence level, we optimize how historical states are stored, updated, and attended to during long-horizon rollout. At the runtime system level, we eliminate repeated host-side preparation, redundant memory allocation, and unnecessary synchronization around model invocation.

Model Level: Low-Precision Compiled Execution. The dominant computation in each inference step is the DiT forward pass for the video and action branches. We deploy the DiT through an ONNX-based TensorRT build pipeline. TensorRT converts the PyTorch computation graph into an optimized execution plan, performs hardware-specific kernel selection, and fuses compatible operations when supported by the backend. To satisfy the static interface required by TensorRT, runtime states such as the KV cache are represented as explicit engine inputs and outputs, rather than implicit Python-side module states. This interface exposes the cache capacity during engine construction and enables fixed execution profiles. FP8 quantization nodes are inserted with NVIDIA Model Optimizer. In practice, the head and tail layers remain in BF16, while the linear layers inside transformer blocks are executed in FP8.

Sequence Level: Long-Horizon Attention Optimization. Although compiled execution accelerates individual DiT calls, autoregressive rollout still faces increasing attention cost as the interaction horizon grows. We therefore optimize the cache layout and attention operator jointly. At runtime, we use paged/ragged KV-cache management to avoid repeated padding and repacking of historical tokens. The buffer capacity is determined by the sliding-window length. Cache updates are performed in place within preallocated buffers, while valid lengths, cache positions, and page metadata are tracked explicitly. To directly consume this compact representation, we replace the generic TensorRT attention subgraph with a FlashInfer-based attention plugin, which executes attention over the paged/ragged cache without materializing padded dense tensors.

System Level: Runtime Overhead Reduction. Beyond DiT computation and KV-cache management, the asynchronous inference loop contains repeated host-side preparation. Although each operation is individually small, the cumulative cost becomes non-negligible over many rollout chunks and denoising steps. We amortize these costs by caching reusable runtime objects and skipping redundant work whenever the execution state is unchanged. Specifically, we cache accepted TensorRT bindings and shape configurations, reuse runtime tensors and layer views, precompute cross-attention KV states, cache frame-id and scheduler-step metadata, and avoid redundant cache-update paths after bootstrap. We also remove unnecessary CPU-GPU synchronization points from the inference path.

Together, these three levels address complementary bottlenecks in the inference stack: transformer forward computation, long-horizon attention over historical states, and repeated runtime overhead around model execution.

3 Data Recipe

LingBot-VA 2.0 is trained with a data recipe that follows the multi-task curriculum of Sec. 2: general text-to-image pretraining, general text-to-video pretraining, and finally video-action adaptation on embodied data.

3.1 General image/video data

For the general text-to-image and text-to-video pretraining stages, we reuse the image and video corpora curated for LingBot-Video [70]. These web-scale image and video data provide broad appearance and dynamics priors before video-action adaptation.

3.2 Robot data

Compared with the first-generation LingBot-VA robot corpus [50], which contains thousands of hours of manipulation data, the robot stage in LingBot-VA 2.0 retains the same broad mixture of public, semi-public, and internally collected trajectories, while adding thousands of hours of internal robot demonstrations. The retained sources include AgiBot [2], RoboMind [107], InternData-A1 [92], Open X-Embodiment/OXE [73] including DROID [41], UMI-style datasets [22, 78, 125], and RoboCOIN [109].

All robot data are reprocessed with a unified annotation pipeline. Each long trajectory is segmented into atomic action clips, and each clip is assigned a language prompt and a global task instruction using Qwen3.5-397B. This relabeling step repairs missing or overly generic prompts in earlier versions and makes the language supervision more consistent across embodiments and datasets.

3.3 Ego-centric human data

In addition to robot data, we include an internal ego-centric human manipulation corpus as a scalable source of embodied interaction data. The corpus contains thousands of hours of first-person manipulation videos across 65.4k episodes. The data are balanced across five major tabletop-oriented environment categories: kitchens, dining tables, vanity tables, office desks, and tool benches. Across these categories, the corpus involves more than 600 operators and over 3.0k scene-task combinations, covering daily manipulation skills such as object placement, cleaning, refilling, assembly, packaging, stationery use, tool use, and cosmetic-item organization.

Each episode is captured as a single egocentric RGB stream and annotated per frame with the 6-DoF world-frame root poses of both hands and 22 finger-joint angles per hand. For ego-centric data, the key additional step is to align these hand poses with the robot gripper action space. We preserve the two 6-DoF hand root trajectories and convert the high-dimensional finger-joint trajectories into left/right scalar parallel-gripper apertures. This yields a compact robot-compatible action representation while retaining the global motion of both hands. The converted clips are then incorporated into the same unified annotation and filtering pipeline as the robot corpus before being used for the human–robot co-training of Sec. 2.3.5.

3.4 In-context learning data

To support human video in-context learning in Section 2.3.4, we utilize the data pipeline in Zero-WAM [60] that converts robot videos into human videos that have consistent task semantics. To ensure robust generalization of in-context learning across few-shot or unseen tasks, we sample videos from large-scale robotic video datasets guided by a task taxonomy, thereby constructing a large-scale in-context human-robot dataset featuring rich and diverse task semantics. For each sampled robot video, we first employ a VLM (e.g., Gemini-3.1-Pro [90]) for task analysis and to generate an image editing prompt that transforms the robot's first frame into an initial observation of human manipulation. This editing process imposes no constraints on viewpoint, scene, or object instances, requiring only the preservation of the original robot task semantics. Subsequently, given the edited initial observation image, we leverage the VLM again to generate a human video generation prompt for the corresponding manipulation task, which drives video generation models (e.g., WAN-2.6 [94] or Kling-V3 [45]) to synthesize human manipulation videos. Finally, we utilize the VLM to evaluate the generated videos by assigning scores for task semantic preservation and physical plausibility. Qualified human videos are then paired with their original robot counterparts to form individual ICL samples. The resulting ICL

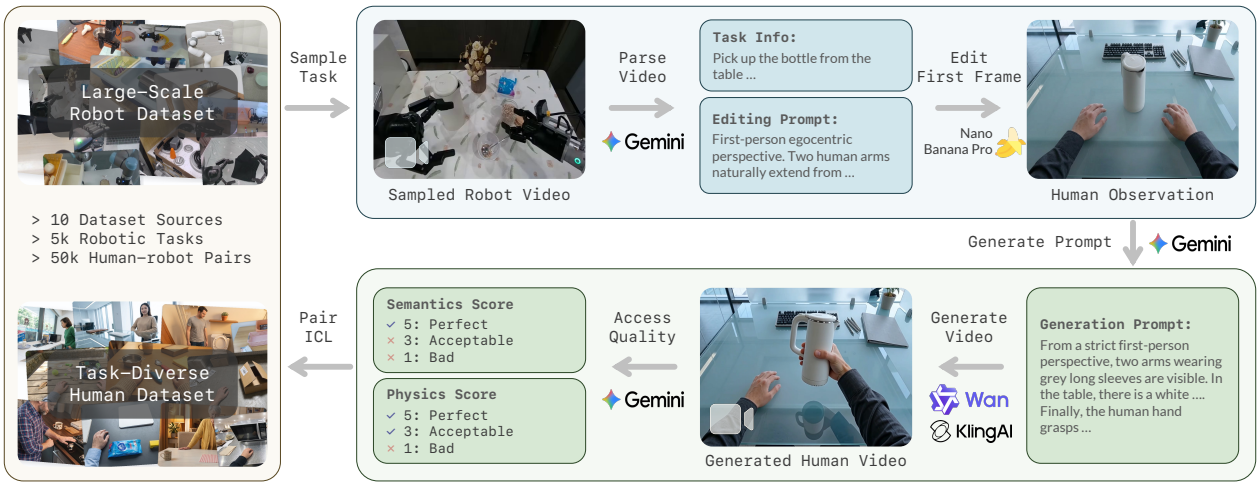

Figure 7. In-context human-robot data curation pipeline. Robot videos are sampled by task semantics, converted into human demonstration prompts with a VLM, synthesized as human videos, and filtered before pairing with the original robot trajectories as ICL samples.

dataset encompasses data from over 10 robot video pre-training datasets (e.g., AgiBot [2], OXE [73]), comprising more than 5,000 tasks and over 50,000 samples.

4 Experiments

4.1 Implementation & Training Details

4.1.1 Network Architecture

Backbone. Our model is a diffusion transformer (DiT) trained from scratch, operating on the latent space of our semantic visual-action tokenizer (96 latent channels). Video latents are patchified with a $1 \times 2 \times 2$ ($T \times H \times W$) linear patch embedding into a stream of visual tokens. The video backbone consists of 30 transformer blocks with video hidden dimension 2048; joint self-attention is performed in a 3072-dim attention space (24 attention heads $\times$ 128 head dim). Its feed-forward pathway is implemented as a sparse MoE routed layer with 128 routed SwiGLU experts, top-8 routing, one shared expert, and per-expert intermediate dimension 512, while the action stream uses a dense FFN as described below.

MoT architecture. Action tokens are processed by a parallel, narrower expert stream inside every block. Raw actions have a unified 30-dim action space (quantile-normalized per dataset; missing dimensions zero-padded and masked) and are embedded by a single bias-free linear layer into an action hidden dimension of 768 (a quarter of the shared attention width). The action FFN dimension is scaled proportionally to 3072. Video and action tokens interact through joint self-attention in the shared 3072-dim attention space: video queries/keys/values are projected $2048 \to 3072$, action queries/keys/values are lifted $768 \to 3072$, and the attention outputs are projected back to their respective stream widths. In cross-attention the action stream has its own query and output projections but shares the text key/value projections with the video stream. All remaining components are modality-specific: the sparse MoE layer for video, the dense FFN for actions, LayerNorms, AdaLN tables, the timestep/text condition embedder (a 768-dim replica), and the output heads (LayerNorm + linear to $96 \cdot 4$ latent channels per patch for video; LayerNorm + linear $768 \to 30$ for actions).

Causal chunked generation. The model generates video autoregressively at the chunk level: frames are grouped into temporal chunks, and self-attention is block-causal with a sliding window over past chunks. During training, the chunk size is resampled uniformly from 1–4 latent frames and the attention window from 1–64 chunks at every step, so a single model supports variable chunk sizes and history lengths at inference (we use chunk size 2 and a 64-chunk window for evaluation). To reduce exposure bias during autoregressive rollout, we apply diffusion-forcing-style context noise

augmentation: with probability 0.5 the clean history frames are replaced by noised versions at a randomly sampled (small) noise level.

Multi-Chunk Prediction (MCP). As an auxiliary training objective, lightweight MCP heads predict 1–3 chunks beyond the next chunk. Hidden states are collected from backbone layers $\{3, 11, 19, 29\}$, fused by a two-layer SiLU MLP ($4 \times 2048 \to 2048 \to 2048$), and fed to 3 depth groups of 3 extra transformer blocks each (identical configuration to backbone blocks, including the action expert); each depth d additionally receives the noisy latents of the $(d{+}1)$-th future chunk through a per-depth linear projection (concat $\to$ 2048). The three depths are supervised with loss weights 0.5 / 0.2 / 0.1. MCP heads are used only for training-time supervision; standard inference runs the backbone alone.

Conditioning. Text prompts are encoded by a dense UMT text encoder into 4096-dim token embeddings (up to 512 tokens), injected via cross-attention after a two-layer MLP projection. Diffusion timesteps enter through sinusoidal embeddings (256-dim) followed by an MLP that produces the AdaLN modulation signals; the video and action streams carry independent timestep conditioning, allowing decoupled noise levels for the two modalities.

Parameter counts (approx.). Video backbone $\approx$13.0B total / $\approx$1.9B active; action expert $\approx$0.6B; MCP heads $\approx$1.7B; total $\approx$15.3B trained parameters (of which $\approx$2.5B are active per token at inference).

4.1.2 Training Recipe

Objective. All stages are trained with a rectified-flow (flow-matching) objective: the network predicts the velocity between noise and data, supervised by MSE with the scheduler's timestep-dependent loss reweighting. Timesteps are sampled uniformly with a modality-specific timestep shift: image 2, web/video data 2, robot video–action data 5, MCP branch 10, and 1 (unshifted) for the action modality, whose timestep is sampled independently of the video timestep. Text embeddings are dropped with probability 0.1 for classifier-free guidance.

Optimization. We use a hybrid Muon + AdamW optimizer. All 2D weight matrices inside transformer blocks (attention projections, dense action-FFN projections, and routed MoE expert projections) are updated with Muon (with lr 2×10^{-3}, momentum 0.95, weight decay 0.1); all remaining parameters—embedders, normalization/AdaLN parameters, biases, and output heads—with AdamW (lr 1×10^{-4}, $\beta = (0.9, 0.95)$, $\epsilon = 10^{-8}$, weight decay 0.01). The learning rate follows a linear warm-up over 2000 steps and is held constant thereafter. Gradients are clipped to a global norm of 1.0.

Systems. Training is implemented with FSDP full-parameter sharding (no tensor/pipeline/context parallelism), bf16 mixed precision with FP32 gradient reduction, full activation checkpointing, and an elastic multi-node launcher. Each rank processes one packed sequence per step (local batch size 1). We hold out 1% of the data for validation.

4.2 Real-world Deployment

4.2.1 Real-world Evaluation

We evaluate real-world manipulation on an in-house robot benchmark covering a diverse set of everyday manipulation tasks. We report results on four tasks: i) *Fruit Sorting*, where the robot places an apple, a pear, and an orange into a basket; ii) *Pen Collection*, where the robot sequentially places three pens from the table into a cup; iii) *Drawer Tidying*, where the robot opens a drawer and stores objects from the tabletop inside it; and iv) *Plate Handover*, where the robot pushes a plate to the center, picks up a paper pad, places it on the plate, and then places the target object on the pad. For each task, we collect 20 teleoperated demonstrations and train a single generalist policy with multi-task supervised fine-tuning, rather than training one model per task. At evaluation time, the same checkpoint is deployed across all tasks, and we report the per-task success rate as well as the average task progress score over repeated rollouts.

We compare against representative baselines from both vision-language-action (VLA) and video-action (VA) model families. For VLA models, we use $\pi_{0.5}$ [77] as the main baseline. For VA models, we compare against LingBot-VA [50], which also models future visual dynamics for action generation. As shown in Fig. 8, `LingBot-VA 2.0` achieves the strongest overall real-world performance, improving both success rate and task progress over the VLA baseline and the prior VA baseline. Compared with $\pi_{0.5}$, the improvement is especially clear on continuous manipulation tasks that require longer-horizon visual grounding. By explicitly predicting future visual states, `LingBot-VA 2.0` can maintain

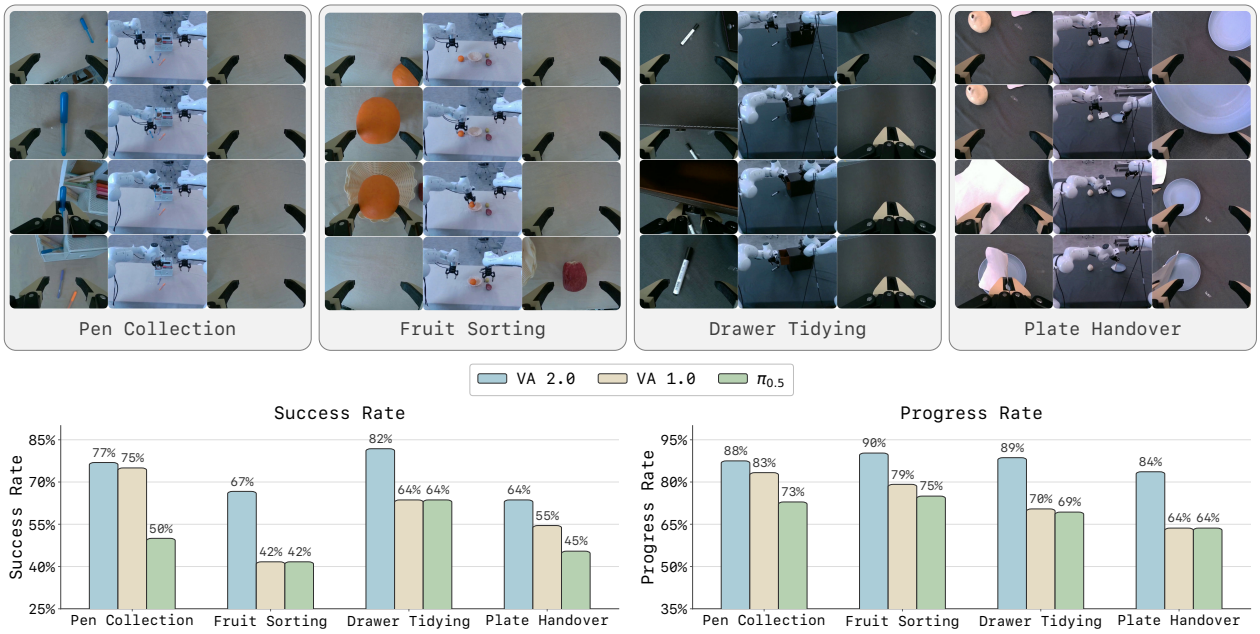

Figure 8. Real-world deployment results. Success rate and task progress rate across real-world tasks, comparing LingBot-VA 2.0 against representative baselines.

task-level context across multiple interaction steps, while its video grounding also improves the precision of local manipulation. Compared with LingBot-VA, the gains come from causal video-action pretraining, which exposes the model to more generalizable video-action correspondences before downstream policy finetuning. This broader pretraining is particularly beneficial in the more challenging multi-task setting, where the policy must generalize across diverse objects, scenes, and task procedures.

4.2.2 In-context Learning Demo

To qualitatively examine whether LingBot-VA 2.0 can use visual demonstrations as task context instead of text instructions, we conduct a separate in-context learning (ICL) study from the real-world evaluation above. As shown in Fig. 9, the tabletop scene contains three plates (green, white, and silver) and seven objects: a silver mug, two green pears, a calabash, a silver coffee cup, a yellow pear, and a white paper cup. For data collection, we only collect four training tasks: i) *put the white paper cup into the white plate*; ii) *put the silver coffee cup into the silver plate*; iii) *put the green pear in the middle into the green plate*; and iv) *put the silver mug in the left into the white plate*, with 15 demonstrations per task. We finetune our pretrained model on these four seen tasks.

At test time, we evaluate on new task compositions that are not observed in the training set. The policy receives a human reference video as the demonstration context together with the current robot observation, and then executes the task without updating model parameters. Figure 9 visualizes examples of four unseen tasks: i) *put the calabash into the green plate*, ii) *put the white paper cup into the silver plate*, iii) *put the silver coffee cup into the white plate*, and iv) *put the green pear next to the silver mug into the white plate*, all of which are absent from the training tasks. This setting evaluates whether the video-action backbone can extract procedural cues from the context video and transfer them to new task compositions, object arrangements, and object instances.

4.3 Simulation

We evaluate LingBot-VA 2.0 on RoboTwin [19], a bimanual manipulation benchmark with clean and domain-randomized settings, and report task success rate (%) against representative baselines. Following LingBot-VA [50], we adopt a multi-task training setup 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).

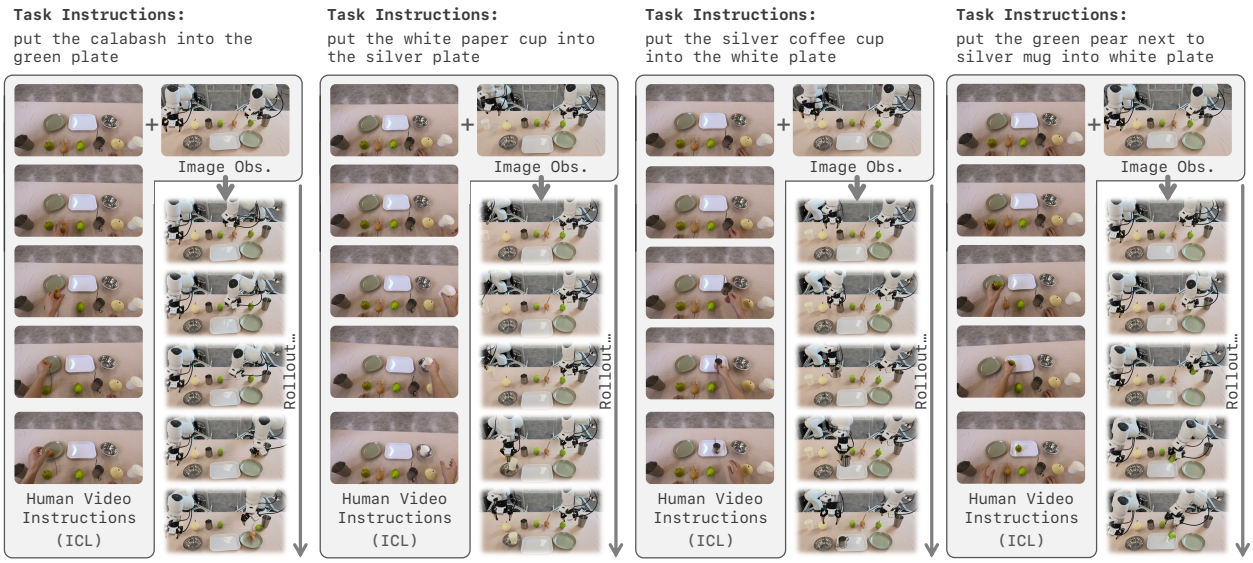

Figure 9. In-context learning demo. We demonstrate ICL rollouts on four unseen tasks. For each rollout, given a reference video demonstration as the task instruction and the robot observation, LingBot-VA 2.0 transfers the same action procedure to robot manipulation without parameter updates.

Table 1. RoboTwin 2.0. Bimanual success rate (%) under clean and domain-randomized settings. Clean/Randomized correspond to the Easy/Hard settings reported in LingBot-VA [50].

Method	Clean	Randomized	Avg.
X-VLA [127]	72.9	72.8	72.9
$\pi_{0.5}$ [77]	82.7	76.8	79.8
Motus [8]	88.7	87.0	87.9
LingBot-VA [50]	92.9	91.6	92.2
LingBot-VA 2.0 (Ours)	**93.8**	**93.4**	**93.6**

Results. As shown in Tab. 1, LingBot-VA 2.0 achieves the best performance on both clean and domain-randomized RoboTwin settings. Compared with the VLA baseline $\pi_{0.5}$, LingBot-VA 2.0 improves the average success rate by 14.0 percentage points, indicating that native video-action pretraining provides stronger physical grounding for bimanual manipulation. Among prior VA methods, LingBot-VA is the strongest baseline with a 92.2% average success rate, while LingBot-VA 2.0 further improves the average result to 93.6%. The gap between clean and randomized evaluation is also small for LingBot-VA 2.0 (0.6 percentage points), suggesting that the model preserves robust control under visual and physical domain variation.

4.4 Ablation

Semantic Visual-Action Tokenizer. To isolate the effect of the tokenizer, we train a 1.3B video-action model from scratch for each variant. The WAN2.2 VAE baseline and our tokenizer are trained with the same number of tokens and used to initialize the same downstream architecture. We then apply identical RoboTwin post-training and evaluate success rate on the official Easy and Hard splits. The tokenizer ablation in Tab. 2 shows that replacing the reconstruction-oriented WAN2.2 VAE with our tokenizer consistently improves RoboTwin performance across prediction horizons. The gain becomes larger for longer horizons, suggesting that semantic visual-action tokenization preserves state information that is more useful for world-action modeling and downstream control.

Multi-chunk prediction. Figure 10 compares post-training dynamics with and without the multi-chunk prediction (MCP) objective. We first train a 5B video-action baseline, then perform RoboTwin post-training under the same data

Table 2. Tokenizer ablation on RoboTwin 2.0. Success rate (%) of a 1.3B video-action model with different visual tokenizers on Easy and Hard splits over 50 tasks.

	WAN2.2 VAE		Our tokenizer	
Metric	Easy	Hard	Easy	Hard
$\text{Average}_{\text{hor}=1}$	81.1	78.4	**86.2**	**83.1**
$\text{Average}_{\text{hor}=2}$	75.5	73.9	**85.7**	**84.0**
$\text{Average}_{\text{hor}=3}$	67.2	68.0	**92.0**	**85.4**
$\text{Average}_{\text{50 Tasks}}$	78.0	76.0	**86.6**	**83.1**

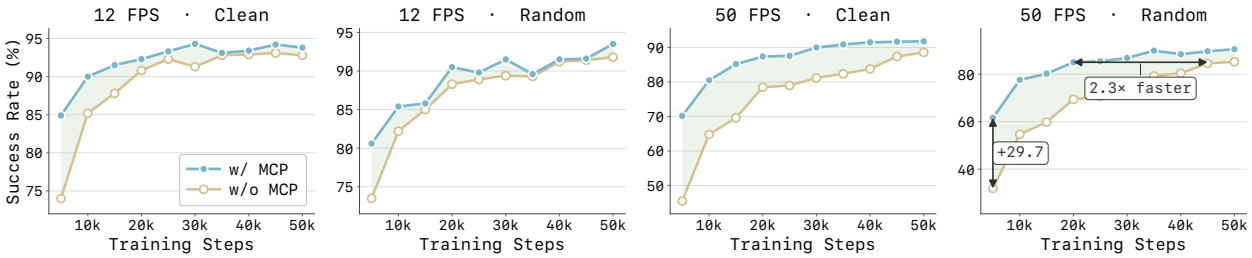

Figure 10. Multi-chunk prediction ablation. Task success rate (%) on RoboTwin across training steps. MCP converges faster and reaches higher final accuracy than the baseline at both 12 and 50 fps.

and optimization setup, with MCP enabled in only one of the two variants. MCP consistently improves optimization efficiency, converging faster and reaching higher final task success at both 12 and 50 fps. The advantage is most pronounced at 50 fps: after 5k training steps, the MCP variant already outperforms the baseline by 29.7 percentage points on the randomized setting, and it matches the baseline's 45k-step accuracy using only 20k steps, corresponding to a 2.3× training speedup.

Acceleration. Table 3 reports the cumulative effect of our inference acceleration pipeline. The Async Hz values are computed as $(1000/t_{\text{chunk}}) \times K$, where t_{chunk} is the measured inference time in milliseconds and K is the number of low-level control steps in each generated chunk; we set $K = 32$ in our experiments. Starting from a BF16 PyTorch asynchronous rollout baseline, consistency distillation first reduces the number of denoising steps required at each chunk, decreasing the inference time from 927 ms to 466 ms per chunk. We then apply the three inference optimizations described in Section 2.4.2. Low-precision compiled execution accelerates the model-execution layer by replacing eager DiT forward passes with FP8 TensorRT engines, reducing latency to 369 ms per chunk. Long-horizon attention optimization further improves the autoregressive sequence layer by using paged/ragged KV-cache execution and FlashInfer attention plugins, reducing latency to 272 ms per chunk. Finally, runtime overhead reduction amortizes repeated host-side preparation, memory allocation, and synchronization costs, bringing the end-to-end inference time to 142 ms per chunk. Overall, these optimizations improve the asynchronous control frequency from 35 Hz to 225 Hz, yielding a 6.5× end-to-end speedup.

4.5 Demonstration

Figure 11 shows four real-world demonstrations that exercise different aspects of video-action modeling. **Desk Tidying** emphasizes long-horizon planning, requiring the policy to parse a cluttered tabletop scene, maintain object state over time, and sequence a multi-step cleanup behavior. **Conveyor Belt** tests temporal grounding and dynamic prediction, where the model must anticipate moving objects and synchronize each action chunk with the conveyor motion. **Chip Picking** stresses fine-grained manipulation and control without any tactile sensors, requiring visually guided, gentle grasping of a thin object. **Air Hockey** evaluates fast inference and reactive control in a dynamic game setting, where the policy must quickly predict the puck trajectory and close the loop from recent observations.

Table 3. Inference acceleration. Inference time and asynchronous control frequency under different acceleration techniques.

Acceleration technique	Inference time (ms/chunk) ↓	Async Hz ↑
BF16 PyTorch async rollout baseline	927	35
+ Consistency distillation	466	69
+ Low-precision compiled execution	369	87
+ Long-horizon attention optimization	272	118
+ Runtime overhead reduction	**142**	**225**

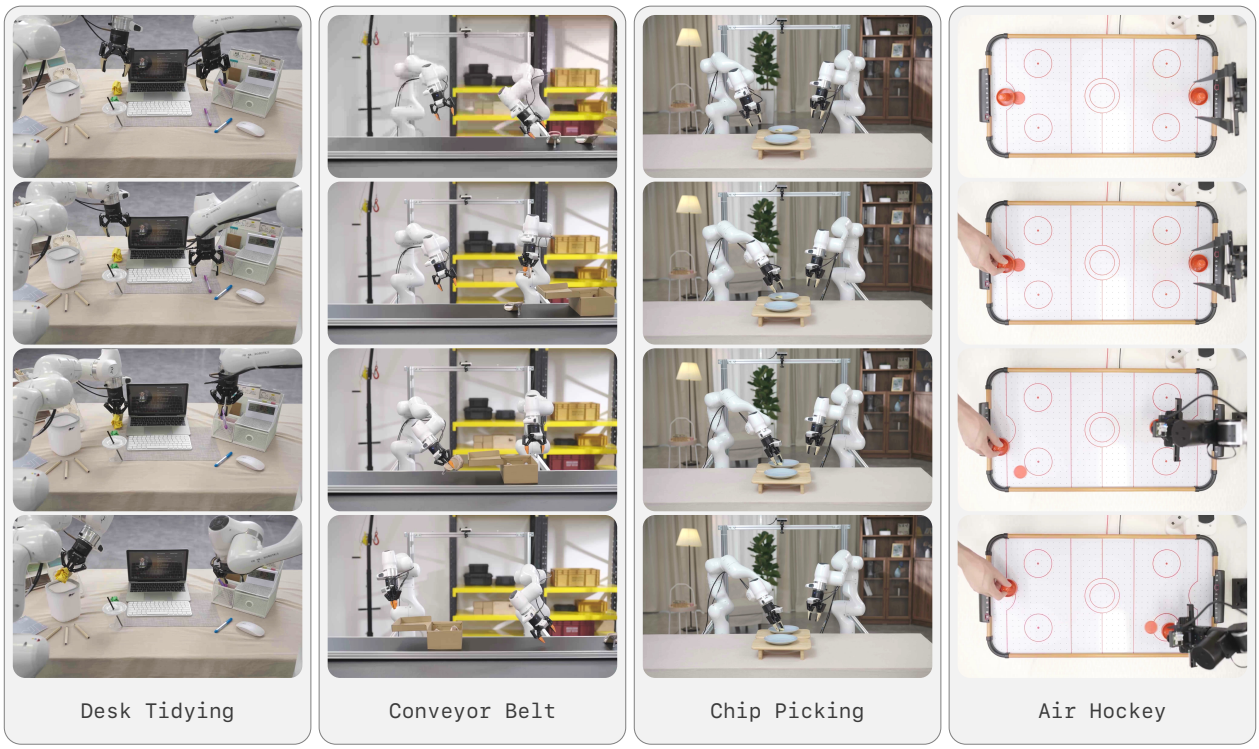

Figure 11. Real-world demonstrations. Qualitative rollouts of LingBot-VA 2.0 across four representative tasks, covering long-horizon tabletop organization, interaction with moving objects, fine-grained grasping, and reactive visual control.

5 Related Work

5.1 Generalist robot policies and VLA models

Vision-language-action (VLA) models directly map visual observations and language instructions to actions, inheriting broad semantic priors from internet-scale pretraining [6, 9, 11, 12, 30, 38, 43, 49, 51, 55, 63, 65, 71, 72, 76, 77, 103, 113, 127]. Industrial foundation models such as LingBot-VLA [110], Qwen-RobotManip [118], and WALL-OSS variants [117, 120] further push these policies toward real-world deployment. A closely related line learns visuomotor policies that regress action sequences from demonstrations, such as ACT [124] and Diffusion Policy [21], while others add visual chain-of-thought or instruction reasoning [69, 84, 114, 123] and inject temporal context through explicit memory modules [48, 82, 86]. These policies are reactive: they learn a direct observation-to-action mapping without modeling how the scene evolves, which limits long-horizon consistency and sample efficiency. In contrast, LingBot-VA 2.0 grounds control in predicted physical dynamics.

5.2 Video generation and world-action models for control

World-action models extend visual world modeling to robot control, using predicted scene evolution as an intermediate representation for action generation [5, 15, 17, 28, 36, 39, 50, 53, 59, 91, 104–106, 115, 119, 126, 133]. DreamZero [115] adapts a large pretrained video diffusion model for joint video-action modeling, while LingBot-VA [50] formulates training as causal world modeling that jointly learns frame prediction and policy execution. A broader body of work couples video prediction with policy learning from complementary angles, including video generation as policy and video-conditioned policies [7, 26, 35, 46, 57] and joint video-action modeling [54, 132]. Related efforts treat interactive or controllable video generation as a dynamics model [1, 16, 29, 75, 79] and use generated rollouts as simulators or imagined plans for policy learning and evaluation [27, 42, 62, 89, 101, 130, 131]. Beyond pixel-space prediction, world models for control also operate in compact latent spaces [33, 34, 56, 68, 81, 102, 108] or over structured 3D and particle states [47, 83, 88, 100, 112, 121, 122]. Training such causal video models also builds on autoregressive video-generation techniques that close the train–test gap, such as Diffusion Forcing [18] and Self Forcing [37]; flexible multimodal diffusion frameworks [4] offer complementary conditioning. Crucially, these methods adapt off-the-shelf, bidirectional video generators [32, 74, 95] through continued training, whereas `LingBot-VA 2.0` pretrains a native, causal video-action stack from scratch.

5.3 Representations for control: semantic tokenizers and latent actions

Another line recovers action-like variables directly from observation sequences, making unlabeled video usable for control [3]. Genie [13] makes video dynamics controllable through latent actions; LAPO [80] recovers latent-action policies, world models, and inverse dynamics purely from videos; LAPA [116] scales discrete latent actions to robot manipulation; and Moto [20] and UniVLA [14] shift the abstraction toward motion tokens. RepWAM [98] and Motus [8] pair such latent actions with representation-oriented tokenizers. `LingBot-VA 2.0` builds on this direction but places world states and actions in a single semantic latent space—aligning visual latents to a foundation model and learning latent actions jointly—rather than relying on a reconstruction-only VAE with a separately attached action module.

6 Conclusion

We presented `LingBot-VA 2.0`, a video-action foundation model built natively for robot control rather than adapted from generic video generation. Its design follows one principle throughout: every component, from representation to inference, is shaped by the demands of embodied control. A semantic visual-action tokenizer places world states and latent actions in a single latent space aligned with a visual foundation model; a causal DiT with a sparse Mixture-of-Experts video stream is pretrained from scratch on this space, with multi-chunk prediction, in-context learning, and human–robot co-training enriching the training signal; and Foresight Reasoning, together with few-step distillation and system-level acceleration, turns the pretrained model into a real-time closed-loop controller. Across simulation and real-world evaluations, `LingBot-VA 2.0` improves over strong vision-language-action and video-action baselines, adapts to new tasks from only a few demonstrations, and sustains long-horizon manipulation through closed-loop re-grounding and hierarchical planning.

Several directions remain open. The planner and the policy are currently trained separately and coupled through a fixed interface; tighter joint training may further improve long-horizon consistency. The latent-action space is learned from passive video, and interactive or reinforcement signals could sharpen it toward control. Finally, scaling the pretraining corpus and the sparse backbone further, and broadening embodiment coverage beyond bimanual manipulation, are natural next steps toward a general-purpose embodied foundation model.

7 Contributors

Core Contributors

Pretraining: Qihang Zhang, Lin Li, Junke Wang, Jiahao Shao, Gangwei Xu, Luyao Zhang, Jiaming Zhou, Yudong Jin, Shuailei Ma, Jiaqi Liao, Ka Leong Cheng
Pretraining Data: Lin Li, Luyao Zhang, Qihang Zhang, Jiaming Zhou, Yudong Jin, Xinyang Wang
Posttraining: Luyao Zhang*, Shuai Yang*, Lin Li, Qihang Zhang, Gangwei Xu, Jiapeng Zhu, Yujie Zhao, Weixuan

Tang
Acceleration: Shuaiting Li, Chaojian Li
Deployment & Demo: Yiming Luo*, Shuai Yang*, Ruilin Wang, Yishu Shen, Jiahao Shao, Fangyi Xu, Shuaiting Li, Guanxing Lu, Zifan Shi, Yongkun Wen
Project Supervision: Yujun Shen, Xing Zhu, Nan Xue
Project Lead: Yinghao Xu

Contributors

Zelin Gao, Wei Wu, Kecheng Zheng, Ruonan Zhang, Han Zhang, Jingmei Zhao, Yongtao Huang, Haitao Wang, Jingjing Wang, Chen Song

*Equal contribution

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] Bowen Baker, Ilge Akkaya, Peter Zhokov, Joost Huizinga, Jie Tang, Adrien Ecoffet, Brandon Houghton, Raul Sampedro, and Jeff Clune. Video pretraining (vpt): Learning to act by watching unlabeled online videos. In *Adv. Neural Inform. Process. Syst.*, 2022.

[4] Fan Bao, Shen Nie, Kaiwen Xue, Chongxuan Li, Shi Pu, Yaole Wang, Gang Yue, Yue Cao, Hang Su, and Jun Zhu. One transformer fits all distributions in multi-modal diffusion at scale. In *Int. Conf. Mach. Learn.*, 2023.

[5] Amir Bar, Gaoyue Zhou, Danny Tran, Trevor Darrell, and Yann LeCun. Navigation world models. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2025.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] Daniel Bolya, Po-Yao Huang, Peize Sun, Jang Hyun Cho, Andrea Madotto, Chen Wei, Tengyu Ma, Jiale Zhi, Jathushan Rajasegaran, Hanoona Bangalath, et al. Perception encoder: The best visual embeddings are not at the output of the network. In *NeurIPS*, 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. Univla: Learning to act anywhere with task-centric latent actions. In *Robotics: Science and Systems*, 2025.

[15] Jun Cen, Chaohui Yu, Hangjie Yuan, Yuming Jiang, Siteng Huang, Jiayan Guo, Xin Li, Yibing Song, Hao Luo, Fan Wang, Deli Zhao, and Hao Chen. Worldvla: Towards autoregressive action world model. *arXiv preprint arXiv:2506.21539*, 2025.

[16] Haoxuan Che, Xuanhua He, Quande Liu, Cheng Jin, and Hao Chen. Gamegen-x: Interactive open-world game video generation. In *Int. Conf. Learn. Represent.*, 2025.

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

[18] Boyuan Chen, Diego Marti Monso, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. In *Adv. Neural Inform. Process. Syst.*, 2024.

[19] 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.

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

[21] 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.

[22] 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.

[23] Damai Dai, Chengqi Deng, Chenggang Zhao, R. X. Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Y. Wu, Zhenda Xie, Y. K. Li, Panpan Huang, Fuli Luo, Chong Ruan, Zhifang Sui, and Wenfeng Liang. DeepSeekMoE: Towards ultimate expert specialization in mixture-of-experts language models. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2024.

[24] DeepSeek-AI. DeepSeek-V3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

[25] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *Int. Conf. Learn. Represent.*, 2021.

[26] 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.

[27] 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.

[28] Shenyuan Gao, Siyuan Zhou, Yilun Du, Jun Zhang, and Chuang Gan. Adaworld: Learning adaptable world models with latent actions. In *Int. Conf. Mach. Learn.*, 2025.

[29] Zelin Gao, Qiuyu Wang, Jiapeng Zhu, Jingye Chen, Zichen Liu, Qingyan Bai, et al. Infinite worlds with versatile interactions. *arXiv preprint arXiv:2607.07534*, 2026.

[30] Gemini Robotics Team. Gemini robotics: Bringing ai into the physical world. *arXiv preprint arXiv:2503.20020*, 2025.

[31] Generalist AI. Gen-0: Embodied foundation models that scale with physical interaction. https://generalistai.com/blog/nov-04-2025-GEN-0, 2025. Built on Harmonic Reasoning.

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

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

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

[35] 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.

[36] Siyuan Huang, Liliang Chen, Pengfei Liu, Yue Hu, Shengyu Zhang, Peng Gao, Hongsheng Li, Maoqing Yao, and Guanghui Ren. Enerverse: Envisioning embodied future space for robotics manipulation. *arXiv preprint arXiv:2501.01895*, 2025.

[37] Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. *arXiv preprint arXiv:2506.08009*, 2025.

[38] Chia-Yu Hung, Qi Sun, Pengfei Hong, Amir Zadeh, Chuan Li, U-Xuan Tan, Navonil Majumder, and Soujanya Poria. Nora: A small open-sourced generalist vision language action model for embodied tasks. *arXiv preprint arXiv:2504.19854*, 2025.

[39] Joel Jang, Seonghyeon Ye, Zongyu Lin, et al. Dreamgen: Unlocking generalization in robot learning through video world models. In *Conference on Robot Learning (CoRL)*, 2025.

[40] Simar Kareer, Dhruv Patel, Ryan Punamiya, Pranay Mathur, Shuo Cheng, Chen Wang, Judy Hoffman, and Danfei Xu. Egomimic: Scaling imitation learning via egocentric video. *arXiv preprint arXiv:2410.24221*, 2024.

[41] 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.

[42] 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.

[43] 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.

[44] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[45] KlingAI Team. Kling-v3. https://kling.ai/, 2026.

[46] Po-Chen Ko, Jiayuan Mao, Yilun Du, Shao-Hua Sun, and Joshua B. Tenenbaum. Learning to act from actionless videos through dense correspondences. In *Int. Conf. Learn. Represent.*, 2024.

[47] 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.

[48] 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.

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

[50] Lin Li, Qihang Zhang, Yiming Luo, Shuai Yang, Ruilin Wang, Fei Han, Mingrui Yu, Zelin Gao, Nan Xue, Xing Zhu, Yujun Shen, and Yinghao Xu. Causal world modeling for robot control. *arXiv preprint arXiv:2601.21998*, 2026.

[51] Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, Xiaofan Wang, Bei Liu, Jianlong Fu, Jianmin Bao, Dong Chen, Yuanchun Shi, Jiaolong Yang, and Baining Guo. Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. *arXiv preprint arXiv:2411.19650*, 2024.

[52] Richard Li, Aditya Prakash, Andrew Wen, Saurabh Gupta, Yilun Du, and Pulkit Agrawal. What matters when cotraining robot manipulation policies on everyday human videos? *arXiv preprint arXiv:2606.06627*, 2026.

[53] Shalfun Li, Victor Yao, Charles Yang, Truth Qu, Regis Cheng, Ryan Yu, Howard Lu, Newton Von, Vincent Chen, Yohann Tang, et al. Wall-wm: Carving world action modeling at the event joints. *arXiv preprint arXiv:2606.01955*, 2026.

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

[55] Xinghang Li, Minghuan Liu, Hanbo Zhang, Cunjun Yu, Jie Xu, Hongtao Wu, Chilam Cheang, Ya Jing, Weinan Zhang, Huaping Liu, Hang Li, and Tao Kong. Vision-language foundation models as effective robot imitators. In *Int. Conf. Learn. Represent.*, 2024.

[56] 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.

[57] 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.

[58] 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.

[59] Yue Liao, Pengfei Zhou, Siyuan Huang, Donglin Yang, Shengcong Chen, Yuxin Jiang, Yue Hu, Jingbin Cai, Si Liu, Jianlan Luo, Liliang Chen, Shuicheng Yan, Maoqing Yao, and Guanghui Ren. Genie envisioner: A unified world foundation platform for robotic manipulation. *arXiv preprint arXiv:2508.05635*, 2025.

[60] LingBot-VA Team, RobbyAnt. Zero-wam: In-context world modeling for zero-shot task generalization. https://github.com/jiaming-zhou/Zero-WAM, 2026.

[61] 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.

[62] Dongxiu Liu, Haoyi Niu, Zhihao Wang, Jinliang Zheng, Yinan Zheng, Zhonghong Ou, Jianming Hu, Jianxiong Li, and Xianyuan Zhan. Efficient robotic policy learning via latent space backward planning. In *Int. Conf. Mach. Learn.*, 2025.

[63] Jiaming Liu, Hao Chen, Pengju An, Zhuoyang Liu, Renrui Zhang, Chenyang Gu, Xiaoqi Li, Ziyu Guo, Sixiang Chen, Mengzhen Liu, Chengkai Hou, Mengdi Zhao, KC alex Zhou, Pheng-Ann Heng, and Shanghang Zhang. Hybridvla: Collaborative diffusion and autoregression in a unified vision-language-action model. *arXiv preprint arXiv:2503.10631*, 2025.

[64] Jingyuan Liu, Jianlin Su, Xingcheng Yao, Zhejun Jiang, Guokun Lai, Yulun Du, Yidao Qin, Weixin Xu, Enzhe Lu, Junjie Yan, Yanru Chen, Huabin Zheng, Yibo Liu, Shaowei Liu, Bohong Yin, Weiran He, Han Zhu, Yuzhi Wang, Jianzhou Wang, Mengnan Dong, Zheng Zhang, Yongsheng Kang, Hao Zhang, Xinran Xu, Yutao Zhang, Yuxin Wu, Xinyu Zhou, and Zhilin Yang. Muon is scalable for LLM training. *arXiv preprint arXiv:2502.16982*, 2025.

[65] 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.

[66] Hao Luo, Yicheng Feng, Wanpeng Zhang, Sipeng Zheng, Ye Wang, Haoqi Yuan, Jiazheng Liu, Chaoyi Xu, Qin Jin, and Zongqing Lu. Being-h0: Vision-language-action pretraining from large-scale human videos. *arXiv preprint arXiv:2507.15597*, 2025.

[67] Hao Luo, Wanpeng Zhang, Yicheng Feng, Sipeng Zheng, Haiweng Xu, Chaoyi Xu, Ziheng Xi, Yuhui Fu, and Zongqing Lu. Being-h0.7: A latent world-action model from egocentric videos. *arXiv preprint arXiv:2605.00078*, 2026.

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

[69] Qi Lv, Weijie Kong, Hao Li, Jia Zeng, Zherui Qiu, Delin Qu, Haoming Song, Qizhi Chen, Xiang Deng, and Jiangmiao Pang. F1: A vision-language-action model bridging understanding and generation to actions. *arXiv preprint arXiv:2509.06951*, 2025.

[70] Shuailei Ma, Jiaqi Liao, Xinyang Wang, Jingjing Wang, Chaoran Feng, Zijing Hu, Chong Bao, Zichen Xi, Yuqi Gan, Weisen Wang, Yanhong Zeng, Qin Zhao, Zifan Shi, Wei Wu, Hao Ouyang, Qiuyu Wang, Shangzhan Zhang, Jiahao Shao, Yipengjing Sun, Liangxiao Hu, Lunke Pan, Nan Xue, Kecheng Zheng, Yinghao Xu, Xing Zhu, Yujun Shen, and Ka Leong Cheng. Scaling mixture-of-experts video pretraining for embodied intelligence. *arXiv preprint arXiv:2607.07675*, 2026.

[71] NVIDIA. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

[72] Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, et al. Octo: An open-source generalist robot policy. In *Conference on Robot Learning (CoRL)*, 2024.

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

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

[75] 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.

[76] Physical Intelligence. $\pi_{0.7}$: a steerable generalist robotic foundation model with emergent capabilities. *arXiv preprint arXiv:2604.15483*, 2026.

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

[78] 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.

[79] Robbyant Team, Zelin Gao, Qiuyu Wang, Yanhong Zeng, Jiapeng Zhu, Ka Leong Cheng, et al. Advancing open-source world models. *arXiv preprint arXiv:2601.20540*, 2026.

[80] Dominik Schmidt and Minqi Jiang. Learning to act without actions. In *Int. Conf. Learn. Represent.*, 2024.

[81] 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.

[82] 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.

[83] 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.

[84] Lucy Xiaoyang Shi, Brian Ichter, Michael Equi, Liyiming Ke, Karl Pertsch, Quan Vuong, James Tanner, Anna Walling, Haohuan Wang, Niccolo Fusai, et al. Hi robot: Open-ended instruction following with hierarchical vision-language-action models. *arXiv preprint arXiv:2502.19417*, 2025.

[85] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *Int. Conf. Mach. Learn.*, 2023.

[86] 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.

[87] Guinan Su, Yanwu Yang, Xueyan Li, and Jonas Geiping. Multi-stream llms: Unblocking language models with parallel streams of thoughts, inputs and outputs. *arXiv preprint arXiv:2605.12460*, 2026.

[88] 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.

[89] 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.

[90] The Gemini Team, Google. Gemini 3.1 pro: A smarter model for your most complex tasks. https://blog.google/innovation-and-ai/models-and-research/gemini-models/gemini-3-1-pro/, 2026.

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

[92] 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.

[93] 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.

[94] WAN AI Team. Wan-2.6. https://wan.video/introduction/wan2.6, 2026.

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

[96] Chen Wang, Linxi Fan, Jiankai Sun, Ruohan Zhang, Li Fei-Fei, Danfei Xu, Yuke Zhu, and Animashree Anandkumar. Mimicplay: Long-horizon imitation learning by watching human play. In *Conference on Robot Learning (CoRL)*, 2023.

[97] Junke Wang, Yi Jiang, Zehuan Yuan, Binyue Peng, Zuxuan Wu, and Yu-Gang Jiang. Omnitokenizer: A joint image-video tokenizer for visual generation. In *NeurIPS*, 2024.

[98] Junke Wang, Qihang Zhang, Shuai Yang, Yiming Luo, Yujun Shen, Zuxuan Wu, Yu-Gang Jiang, and Yinghao Xu. Repwam: World action modeling with representation visual-action tokenizers. *arXiv preprint arXiv:2606.13674*, 2026.

[99] Lean Wang, Huazuo Gao, Chenggang Zhao, Xu Sun, and Damai Dai. Auxiliary-loss-free load balancing strategy for mixture-of-experts. *arXiv preprint arXiv:2408.15664*, 2024.

[100] 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.

[101] 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.

[102] 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.

[103] Junjie Wen, Yichen Zhu, Jinming Li, Zhibin Tang, Chaomin Shen, and Feifei Feng. Dexvla: Vision-language model with plug-in diffusion expert for general robot control. *arXiv preprint arXiv:2502.05855*, 2025.

[104] Youpeng Wen, Junfan Lin, Yi Zhu, Jianhua Han, Hang Xu, Shen Zhao, and Xiaodan Liang. Vidman: Exploiting implicit dynamics from video diffusion model for effective robot manipulation. *arXiv preprint arXiv:2411.09153*, 2024.

[105] Hongtao Wu, Ya Jing, Chilam Cheang, Guangzeng Chen, Jiafeng Xu, Xinghang Li, Minghuan Liu, Hang Li, and Tao Kong. Unleashing large-scale video generative pre-training for visual robot manipulation. In *Int. Conf. Learn. Represent.*, 2024.

[106] Jialong Wu, Shaofeng Yin, Ningya Feng, Xu He, Dong Li, Jianye Hao, and Mingsheng Long. ivideogpt: Interactive videogpts are scalable world models. In *Adv. Neural Inform. Process. Syst.*, 2024.

[107] 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.

[108] 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.

[109] 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.

[110] Wei Wu, Fan Lu, Yunnan Wang, Shuai Yang, Shi Liu, Fangjing Wang, et al. A pragmatic vla foundation model. *arXiv preprint arXiv:2601.18692*, 2026.

[111] Gangwei Xu, Qihang Zhang, Jiaming Zhou, Xing Zhu, Yujun Shen, Xin Yang, and Yinghao Xu. Next forcing: Causal world modeling with multi-chunk prediction. *arXiv preprint arXiv:2606.11187*, 2026.

[112] 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.

[113] Jianwei Yang, Reuben Tan, Qianhui Wu, Ruijie Zheng, Baolin Peng, Yongyuan Liang, Yu Gu, Mu Cai, Seonghyeon Ye, Joel Jang, Yuquan Deng, Lars Lidén, and Jianfeng Gao. Magma: A foundation model for multimodal ai agents. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2025.

[114] 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.

[115] Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, Ayaan Malik, Kyungmin Lee, William Liang, Nadun Ranawaka, Jiasheng Gu, Yinzhen Xu, Guanzhi Wang, Fengyuan Hu, Avnish Narayan, Johan Bjorck, Jing Wang, Gwanghyun Kim, Dantong Niu, Ruijie Zheng, Yuqi Xie, Jimmy Wu, Qi Wang, Ryan Julian, Danfei Xu, Yilun Du, Yevgen Chebotar, Scott Reed, Jan Kautz, Yuke Zhu, Linxi Jim Fan, and Joel Jang. World action models are zero-shot policies. *arXiv preprint arXiv:2602.15922*, 2026.

[116] Seonghyeon Ye, Joel Jang, Byeongguk Jeon, Sejune Joo, Jianwei Yang, Baolin Peng, Ajay Mandlekar, Reuben Tan, Yu-Wei Chao, Bill Yuchen Lin, Lars Liden, Kimin Lee, Jianfeng Gao, Luke Zettlemoyer, Dieter Fox, and Minjoon Seo. Latent action pretraining from videos. In *Int. Conf. Learn. Represent.*, 2025.

[117] Ryan Yu, Pushi Zhang, Starrick Liu, Brae Liu, Miracle Kang, Shalfun Li, Lights Shi, Ellie Ma, Ping Yang, Chris Pan, et al. Wall-oss-0.5 technical report. *arXiv preprint arXiv:2605.30877*, 2026.

[118] Haoqi Yuan, Zhixuan Liang, Anzhe Chen, Ye Wang, Haoyang Li, Pei Lin, Yiyang Huang, Zixing Lei, Tong Zhang, Jiazhao Zhang, et al. Qwen-robotmanip technical report: Alignment unlocks scale for robotic manipulation foundation models. *arXiv preprint arXiv:2606.17846*, 2026.

[119] Tianyuan Yuan, Zibin Dong, Yicheng Liu, and Hang Zhao. Fast-wam: Do world action models need test-time future imagination? *arXiv preprint arXiv:2603.16666*, 2026.

[120] Andy Zhai, Brae Liu, Bruno Fang, Chalse Cai, Ellie Ma, et al. Igniting vlms toward the embodied space. *arXiv preprint arXiv:2509.11766*, 2025.

[121] 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.

[122] 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.

[123] 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.

[124] 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.

[125] 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.

[126] Haoyu Zhen, Xiaowen Qiu, Peihao Chen, Jincheng Yang, Xin Yan, Yilun Du, Yining Hong, and Chuang Gan. 3d-vla: A 3d vision-language-action generative world model. In *Int. Conf. Mach. Learn.*, 2024.

[127] 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.

[128] Jiaming Zhou, Teli Ma, Kun-Yu Lin, Zifan Wang, Ronghe Qiu, and Junwei Liang. Mitigating the human-robot domain discrepancy in visual pre-training for robotic manipulation. *arXiv preprint arXiv:2406.14235*, 2024.

[129] Jiaming Zhou, Ke Ye, Jiayi Liu, Teli Ma, Zifan Wang, Ronghe Qiu, Kun-Yu Lin, Zhilin Zhao, and Junwei Liang. Exploring the limits of vision-language-action manipulation in cross-task generalization. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[130] 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.

[131] 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.

[132] 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.

[133] Fangqi Zhu, Hongtao Wu, Song Guo, Yuxiao Liu, Chilam Cheang, and Tao Kong. Irasim: Learning interactive real-robot action simulators. *arXiv preprint arXiv:2406.14540*, 2024.