WORLD ACTION MODELS ARE ZERO-SHOT POLICIES

Anonymous authors
Paper under double-blind review

ABSTRACT

State-of-the-art Vision-Language-Action (VLA) models excel at semantic generalization but struggle to generalize to unseen physical motions in novel environments. We introduce DREAMZERO, a World Action Model (WAM) built upon a pretrained video diffusion backbone. Unlike VLAs, WAMs learn physical dynamics by predicting future world states and actions, using video as a dense representation of how the world evolves. By jointly modeling video and action, DREAMZERO learns diverse skills effectively from heterogeneous robot data without relying on repetitive demonstrations. This results in over 2× improvement in generalization to new tasks in new environments compared to state-of-the-art VLAs in real-robot experiments. Crucially, through model and system optimizations, we enable a 14B autoregressive video diffusion model to perform real-time closed-loop control at 7Hz. Finally, we demonstrate cross-embodiment transfer in both directions: (1) video-only demonstrations from other robots or humans improve unseen task performance by over 40% with just 10–20 minutes of data, and (2) DREAMZERO adapts to entirely new embodiments—achieving zero-shot generalization on the YAM robot with only 30 minutes of play data.

1 INTRODUCTION

Recent robotic foundation models, termed Vision-Language Action models (VLAs), extend pretrained Vision-Language Models (VLMs) to predict discrete or continuous motor actions (Kim et al., 2024b; Black et al., 2024b; Bjorck et al., 2025; Gemini Robotics Team, 2025; Brohan et al., 2023a). While these models successfully inherit linguistic priors to generalize across diverse language instructions, especially manipulating diverse objects (Brohan et al., 2023a), their generalization to novel environments and, more critically, to new skills remains limited (Zhou et al., 2025; Guruprasad et al., 2025). For example, VLAs can successfully execute "move coke can to Taylor Swift" (Brohan et al., 2023a) by leveraging the web knowledge acquired during VLM pretraining to identify the target location, and connecting it to the learned motor skill from the robot data. However, they fail at a task like "untie the shoelace" if that specific skill was not present in the robot training data.

While VLM priors encode *what* to do at a semantic level, they lack representations of *how* actions should be executed with precise spatial awareness aligned with geometry, dynamics, and motor control (Chen et al., 2024; Feng et al., 2025). As a result, VLAs often struggle to adapt to new environments or generalize to novel tasks beyond the distribution of expert demonstrations, without explicitly collecting large-scale task- and environment-specific action data.

In this paper, we present DREAMZERO, a 14B robot foundation model built upon a pretrained image-to-video diffusion backbone (Team Wan, 2025). We introduce *World Action Model (WAM)*—a foundation model which jointly predict both actions and visual future states in an aligned manner. Initialized from video generative models trained on web-scale video data, WAMs leverages rich spatiotemporal priors to jointly generate future frames and actions conditioned on language instructions and observations. This shifts action learning from dense state–action imitation to inverse dynamics—aligning motor commands with predicted visual futures. Consequently, this enables effective learning from robot data that are heterogeneous trajectories collected during the execution of useful behaviors in real-world settings, rather than relying solely on carefully repeated demonstrations.

This approach yields two core advancements that distinguish DREAMZERO from prior work, including other WAMs (Kim et al., 2026; Liang et al., 2025; Pai et al., 2025). First, DREAMZERO breaks away from the conventional wisdom that generalist robot policies require multiple repeated

Figure 1: **Overview.** World Action Models (WAMs) inherit world physics priors that enable effective learning from diverse, non-repetitive data, open-world generalization, cross-embodiment learning from video-only data, and efficient adaptation to new robots.

demonstrations per task, demonstrating the ability to learn effectively from diverse, heterogeneous data. Although other WAMs show that priors learned from videos prediction improves sample efficiency for action learning compared to VLAs (Pai et al., 2025; Liao et al., 2025), most works still focus on repeated demonstrations for action learning. Second, it unlocks new generalization capabilities beyond traditional VLAs and previous WAMs—across environments, across tasks, and across embodiments (Figure 1 and Figure 9). Compared to the state-of-the-art pretrained VLAs, we observe more than a 2x improvement in average task progress on environment and task generalization benchmarks for bi-manual mobile manipulation robots. By leveraging this capability, DreamZero is one of the top performing models in the RoboArena Leaderboard (Atreya et al., 2025), a public distributed real-world robot benchmark, without any cross embodiment pretraining. Also, the environment generalization of DreamZero is retained even after task-specific post-training, outperforming state-of-the-art VLAs by 30% on average task progress. Furthermore, we demonstrate cross-embodiment transfer in both directions: (1) video-only demonstrations from another robot (YAM) or humans improve unseen task performance on the target robot (AgiBot G1) by over 40% with just 10–20 minutes of data, and (2) DreamZero pretrained on diverse AgiBot G1 data, adapts to an entirely new embodiment (YAM)—achieving zero-shot generalization on the YAM robot with only 30 minutes of play data.

As the core of DreamZero is a 14B autoregressive transformer trained with a teacher-forcing objective using chunk-wise video denoising, and leverages KV caching for efficient inference. Our architectural analyses reveal several key insights. We find that (1) larger pretrained video diffusion models produce higher-quality video predictions, which directly translates to superior downstream action execution—indicating that policy performance is fundamentally tied to video generation quality; (2) diverse distribution of the training data is essential for generalization, yielding better performance than training on multi-task repetitive data with the same amount of hours. (3) autoregres-

sive architectures lead to smoother robot motions and higher modality alignment between predicted videos and executed actions.

To address the computational overhead inherent to video diffusion models, we introduce a suite of optimizations spanning three categories: (1) algorithmic improvements, including decoupled video and action denoising schedules (DREAMZERO-Flash); (2) system-level parallelism and caching strategies; and (3) low-level optimizations such as quantization, CUDA kernel tuning, asynchronized inference, and action chunk smoothing. Collectively, these techniques achieve a 38× inference speedup without performance degredation, enabling DREAMZERO to generate 1.6-second action chunks at approximately 7Hz for smooth, real-time robotic control.

Our main contributions are:

- We introduce DREAMZERO, a 14B WAM that jointly predicts video and actions, enabling effective learning from diverse, non-repetitive robot data, being **one of the top ranking models**, without cross embodiment pretraining.
- We demonstrate over 2× improvement in zero-shot generalization to **unseen verbs and motions** compared to state-of-the-art VLAs, while retaining generalization across objects and environments.
- We present model and system optimizations achieving **38× inference speedup**, enabling real-time closed-loop control at **7Hz**.
- We demonstrate **bidirectional cross-embodiment transfer**: video-only data from humans (12 minutes) or other robots (20 minutes) improves performance on unseen tasks for the target embodiment (AgiBot G1) by 40%, while DREAMZERO pretrained on the target embodiment adapts to a totally new embodiment (YAM robot) with just 30 minutes of play data, enabling zero-shot generalization.

2 RELATED WORK

2.1 VISION LANGUAGE ACTION MODELS

Research into foundation models for robotics has diverged into two primary paths: modular systems and end-to-end Vision-Language-Action models (VLAs). Modular approaches leverage pre-trained models as "black-box" reasoners to generate high-level plans or affordances executed by low-level controllers (Brohan et al., 2023b; Huang et al.; Singh et al., 2023; Driess et al., 2023; Huang et al., 2023; Kumar et al., 2024; Li et al., 2025b; Lee et al., 2025; Dreczkowski et al., 2025; Bommasani et al., 2021); however, these suffer from potential compounding errors and a dependence on pre-existing skill libraries. On the other hand, VLAs integrate semantics and control into a single framework, often fine-tuned from vision-language models (VLMs) (Brohan et al., 2022; 2023a; Kim et al., 2024a; Zheng et al., 2025a; Ye et al., 2025; Yang et al., 2025; Black et al., 2024a; Bjorck et al., 2025; Physical Intelligence, 2025; Gemini Robotics Team, 2025; Bu et al., 2025). While VLAs excel at object-level and semantic generalization (Brohan et al., 2023a; Gao et al., 2025), they struggle with new physical skills in new environments without scaling human teleoperation to cover all possible physical interactions (Physical Intelligence, 2025; Gemini Robotics Team, 2025; Zhou et al., 2025; Guruprasad et al., 2025). Unlike these fixed task-based approaches, video-based world models offer a promising alternative by learning physical dynamics from continuous frame sequences.

2.2 VIDEO MODEL-BASED ROBOT POLICIES

Recent work demonstrates that video generation models can synthesize robot trajectories and actions via inverse-dynamics (Du et al., 2023; Zhou et al., 2024), optical flow (Ko et al., 2024), or high-level trajectory planning (Yang et al., 2024; Du et al., 2024). Beyond generating human motion for policy training (Liang et al., 2024; Bharadhwaj et al., 2024; Chen et al., 2025), video generation models can produce synthetic robot data for unseen behaviors in novel environments (Jang et al., 2025; Luo et al., 2025). Building on this, World Action Models (WAMs) couple video and action generation end-to-end, either learning from scratch/VLAs (Wu et al., 2024; Cheang et al., 2024; Li et al., 2025a; Zhu et al., 2025; Zhao et al., 2025; Zheng et al., 2025b; Won et al., 2025) or leveraging pretrained video diffusion backbones to inherit rich spatiotemporal priors (Kim et al., 2026; Liao et al., 2025;

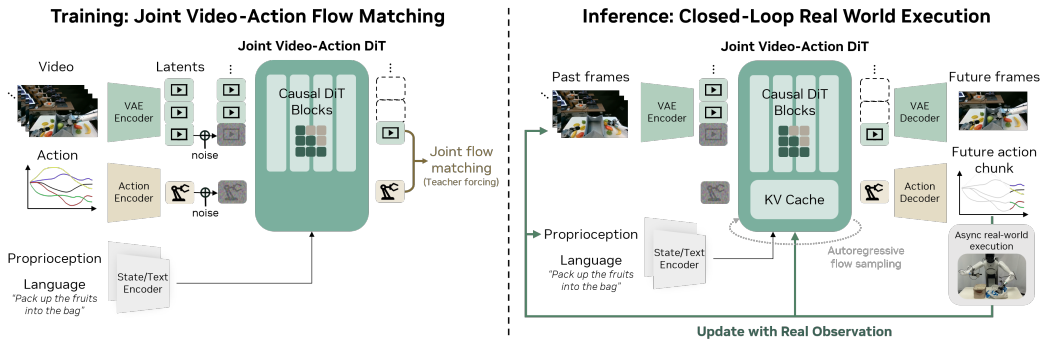


Figure 2: **Model Architecture of DreamZero.** The model takes three inputs: visual context (encoded via a VAE), language instructions (via a text encoder), and proprioceptive state (via a state encoder). These are processed by an autoregressive DiT backbone using flow matching, which jointly predicts future video frames and actions through separate decoders. During training (left), the model learns to denoise both video and action targets conditioned on clean context. During inference (right), predictions are executed asynchronously in the real world, and ground-truth observations are fed back into the KV cache to prevent error accumulation.

Hu et al., 2024; Liang et al., 2025; Pai et al., 2025). By learning the joint distribution of video and action, WAMs combine the seamless gradient flow of VLAs with dense world-modeling supervision, where video prediction serves as an implicit visual planner. Unlike prior WAMs, DreamZero utilizes an autoregressive architecture to systematically explore data scale and diversity, achieving state-of-the-art zero-shot generalization and effective learning from heterogeneous data by reducing robotic capability to a video generation challenge.

3 DreamZero

Pretrained video diffusion models offer rich spatiotemporal priors from web-scale data, making them attractive backbones for robot policies. However, converting these models into effective World Action Models (WAMs) presents three key challenges: (1) **Video-action alignment**: jointly predicting video and actions requires tight coupling between visual futures and motor commands, yet naively combining separate video and action heads can lead to misalignment; (2) **Architectural design**: it remains unclear whether bidirectional or autoregressive architectures are better suited for WAMs, with trade-offs in modality alignment, error accumulation, and inference efficiency; and (3) **Real-time inference**: video diffusion models require iterative denoising across high-dimensional latent spaces, making them prohibitively slow for closed-loop control.

DreamZero addresses these challenges through three design choices. First, we train a single end-to-end model that jointly denoises video and action with a shared objective, ensuring deep integration between modalities. Second, we adopt an autoregressive architecture and exploit the closed-loop setting: after each action chunk is executed, we replace predicted frames with ground-truth observations in the KV cache, eliminating compounding errors while enabling efficient inference via KV caching and preserving native frame rates for precise modality alignment. Third, we introduce a suite of system-, implementation-, and model-level optimizations that achieve a 38× inference speedup, enabling real-time control at 7Hz. We detail the model architecture in Section 3.1 and real-time execution in Section 3.2.

3.1 Model Architecture

Problem Formulation. DreamZero jointly predicts video $\mathbf{o}_{l:l+H}$ and actions $\mathbf{a}_{l:l+H}$ conditioned on language instruction $\mathbf{c}$, proprioceptive state $\mathbf{q}_l$ and visual observation including the current and the past history $\mathbf{o}_{0:l}$ where $H > 0$ is a fixed horizon. Note that joint prediction of video and action is a decomposition of (1) autoregressive video prediction and (2) action prediction from an inverse-

dynamics model (IDM):

$$\begin{aligned}&\underbrace{\pi_0(\mathbf{o}_{l:l+H}, \mathbf{a}_{l:l+H} \mid \mathbf{o}_{0:l}, \mathbf{c}, \mathbf{q}_l)}_{\text{DREAMZERO}} \\ &= \underbrace{\pi_0(\mathbf{o}_{l:l+H} \mid \mathbf{o}_{0:l}, \mathbf{c}, \mathbf{q}_l)}_{\text{video prediction}} \underbrace{\pi_0(\mathbf{a}_{l:l+H} \mid \mathbf{o}_{0:l+H}, \mathbf{q}_l)}_{\text{IDM}}\end{aligned} \tag{1}$$

Instead of using two separate models (video prediction model and inverse dynamics model) to model the decomposed objective (Pai et al., 2025), we train a single model end-to-end with joint prediction objective. We believe that this end-to-end design enables better video-action alignment through a deep integration between the two modalities. Since pretrained video models are already optimized on the video prediction objective on diverse web-scale video data, DREAMZERO only needs to additionally learn to predict videos for the robot embodiment videos and extract corresponding actions from the generated videos. We further hypothesize that this encourages better generalization than the conventional practice of training VLA from VLM (i.e., learning $\pi_0(\mathbf{a}_{l:l+H}|\mathbf{o}_l, \mathbf{c}, \mathbf{q}_l)$ from $\pi_0(\mathbf{c}_{l:H}|\mathbf{o}_t, \mathbf{c}_{0:l})$), as our approach explicitly learns temporal dynamics from video frames used both as conditioning inputs and prediction targets.

Model Architecture. The model architecture is shown in Figure 2. To retain the generalization capability of video models, we introduce minimal additional parameters: state encoders, action encoders, and decoders. For robot training data that contains multiple views, we concatenate all views into a single frame instead of making architectural changes to the backbone model.

In particular, DREAMZERO is trained to predict video frames and corresponding actions autoregressively. Autoregressive generation possesses the following advantages: (1) it enables faster inference speed by utilizing KV-cache, (2) the policy model can leverage the visual observation history as guidance for the next generation, and (3) it avoids the modality alignment challenges (video, action, and language alignment) inherent to bidirectional models. Concretely, bidirectional diffusion typically requires processing fixed-length sequences, which often necessitates video subsampling that distorts native FPS, potentially harming video-action alignment. On the other hand, autoregressive generation leverages KV caching to support arbitrarily long contexts within a single forward pass. This preserves the native frame rate, ensuring precise alignment between video frames and robot actions. Some details illustration of this difference is provided in Appendix A.4.

We introduce autoregressive modeling only for the video modality to avoid error propagation coming from closed-loop action prediction. DREAMZERO is trained to predict video frames in a *chunk* manner; each chunk has a fixed number of latent frames K to match the action horizon. Chunk-wise generation enables training on variable length of videos, similar to how LLMs are trained on variable length of language tokens. We provide more details on the QKV attention masking strategy in Appendix A.3.

Training Objective. Similar to recent video diffusion models and VLAs, we employ flow-matching (Liu et al., 2022; Lipman et al., 2022; Albergo et al., 2023) as the training objective (Team Wan, 2025; Ali et al., 2025; Teng et al., 2025). Unlike recent WAMs (Zhu et al., 2025; Kim et al., 2026; Li et al., 2025a; Liao et al., 2025), DREAMZERO shares the denoising timestep between video and action modality for faster convergence at the beginning of training. Also, we apply teacher forcing (Jin et al., 2024; Gao et al., 2024) as a training objective; the model is trained to denoise the noisy current chunk conditioned on the clean previous chunks.

Formally, given a specific chunk index $k > 0$ and the denoising step $t \in [0, 1]$, we denote the corresponding video latent vector of original video $\mathbf{o}^k$ as $\mathbf{z}_t^k$ and normalized actions as $\mathbf{a}_t^k$, we denoise videos in latent space and actions in their original normalized form. They are defined as linear interpolations between clean vectors and random Gaussian noises:

$$\mathbf{z}_t^k = t_k \mathbf{z}_1^k + (1 - t_k)\mathbf{z}_0^k, \quad \mathbf{a}_t^k = t_k \mathbf{a}_1^k + (1 - t_k)\mathbf{a}_0^k, \tag{2}$$

where $\mathbf{z}_0^k \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, $\mathbf{a}_0^k \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and $\mathbf{z}_1^k$ and $\mathbf{a}_1^k$ are a clean video latent vector and a normalized action, respectively. Thus, the clean context from the previous chunks can be denoted as $\mathcal{C}_k = \bigcup_{j=1}^{k-1} \{\mathbf{z}_1^j,\ \mathbf{a}_1^j\}$. We train the model $\mathbf{u}_\theta$ to predict the joint velocity for both modalities using the following flow-matching objective:

$$\mathcal{L}(\theta) = \mathbb{E}_{\mathbf{z},\mathbf{a},t}\left[\frac{1}{K}\sum_{k=1}^{K} w(t_k)\left\|\mathbf{u}_\theta([\mathbf{z}_t^k, \mathbf{a}_t^k]; \mathcal{C}_k, \mathbf{c}, \mathbf{q}_k, t) - \mathbf{v}_t^k\right\|^2\right], \tag{3}$$

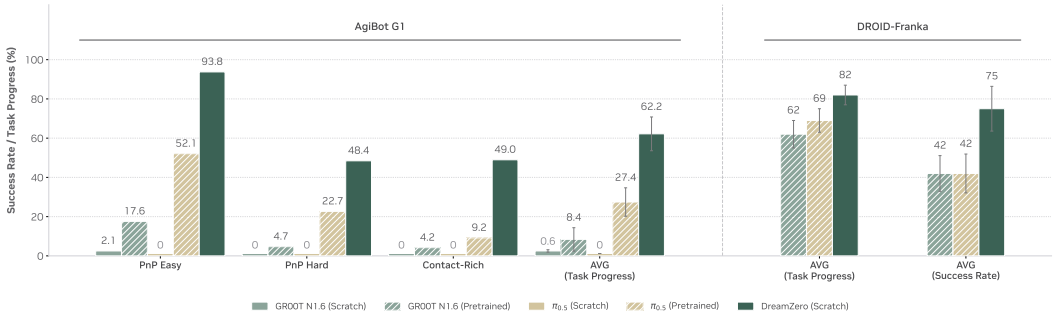


Figure 3: **Seen Task Evaluation.** DREAMZERO can effectively learn from diverse data sources compared to VLAs and generalize to new environments.

where $w(t) > 0$ is a predefined weight function for t, $\mathbf{c}$ is the text condition, $\mathbf{q}_k$ is the proprioceptive states of k-th chunk, and the velocity $\mathbf{v}_t^k := [\mathbf{z}_1^k, \mathbf{a}_1^k] - [\mathbf{z}_0^k, \mathbf{a}_0^k]$ for every t. To enable efficient training, we update the gradient on the trajectory level and apply attention masking (e.g., see Fig. 11 for details) so that the current noisy video and chunk can attend to clean context of previous chunks. Pseudo-code for the training algorithm is provided in Algorithm A.3.

Model Inference. As shown in Figure 2, during inference, DREAMZERO jointly denoises video and action chunks, leveraging KV cache for efficiency (Huang et al., 2025; Teng et al., 2025; Yin et al., 2025). Unlike pure video generation, our closed-loop setting allows ground-truth observations to replace generated frames in the KV cache after each action execution (see Fig. 11). This eliminates the compounding error problem inherent to autoregressive generation—a key advantage unique to WAMs. Additionally, as a stateful policy, DREAMZERO can leverage visual history for tasks requiring memory. Further pseudo-code of model inference is provided in Algorithm A.3.

3.2 REAL-TIME EXECUTION OF DREAMZERO

To bridge the gap between the computational demands of diffusion models and the reactivity required for robotic control, DREAMZERO employs a suite of optimizations that collectively achieve a 38× speedup, enabling 7Hz closed-loop control. The system uses **asynchronous execution** to decouple inference from actuation, so the robot executes the current action chunk while the next is computed. System-level improvements such as **CFG parallelism** and **DiT caching** reduce effective diffusion steps from 16 to 4. We further propose an **architectual optimization**, DREAMZERO-Flash, which decouples noise schedules during training—biasing video toward noise while keeping actions uniform—allowing clean actions to be predicted from noisy visual context in a single step. Combined with low-level optimizations like **NVFP4 quantization** and **CUDA graph compilation**, these techniques reduce inference latency from 6.2 seconds to approximately 150ms. See Appendix A.5 for more details.

4 EXPERIMENTAL SETUP

Embodiments and Data. We validate DREAMZERO on two embodiments: the bimanual AgiBot G1 and the single-arm Franka robot. For AgiBot, we collect roughly 500 hours of diverse teleoperation data across 22 real-world environments, emphasizing task breadth over repetition. For Franka, we train on DROID (Khazatsky et al., 2024), a highly heterogeneous public dataset. We compare against state-of-the-art VLA baselines, including GR00T (Bjorck et al., 2025) and $\pi_{0.5}$ (Physical Intelligence, 2025), using both from-scratch and pre-trained initializations.

Training and Architecture. DREAMZERO uses Wan2.1-I2V-14B (Team Wan, 2025) as the 14B image-to-video diffusion backbone. We train for 100K steps on AgiBot and 75K on DROID, updating DiT blocks, the state encoder, and action heads while freezing the text and image encoders. Training uses a joint video-action objective that predicts future frames and actions autoregressively.

Evaluation Protocol. We prioritize **zero-shot generalization** by evaluating in unseen environments with unseen objects. We report *Seen Tasks* (variants of training tasks such as pick-and-place)

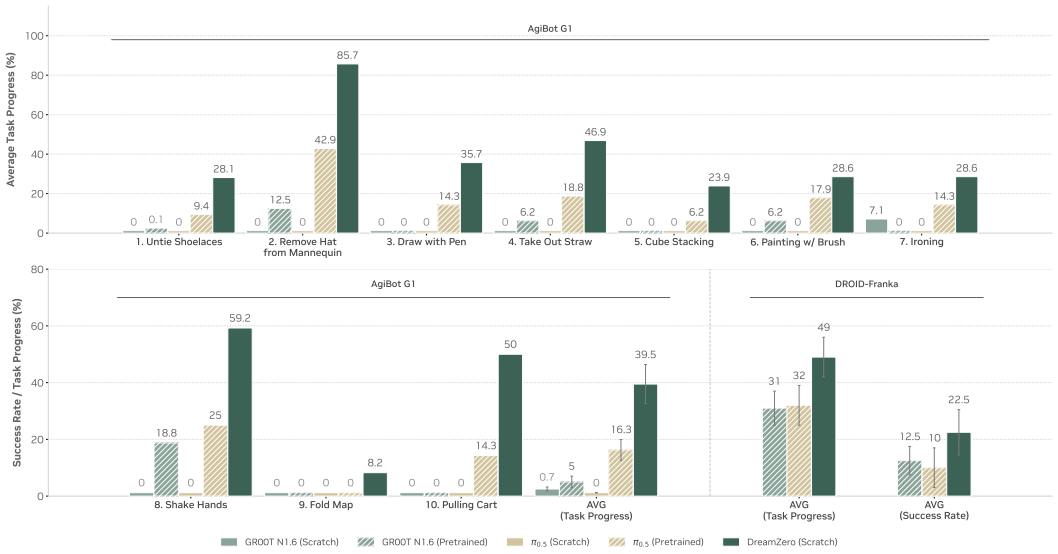


Figure 4: **Zero-shot Generalization to Unseen Tasks.** DREAMZERO achieves significant task progress on 10 tasks absent in the training data, while VLAs struggle across both embodiments.

and *Unseen Tasks* (novel behaviors such as ironing or untying shoelaces). We also assess **post-training** performance by fine-tuning on three downstream tasks with varying diversity to measure robustness under distribution shift. Additional experimental details are provided in Appendix A.6 and data collection strategy for AgiBot pretraining data is provided in Appendx A.7.

5 EXPERIMENTAL RESULTS

5.1 MAIN RESULTS

We evaluate the zero-shot generalization performance of DREAMZERO compared to baseline models and investigate the following research questions:

Q1: Do WAMs learn better from diverse, non-repetitive data? We pretrain models on diverse data and evaluate them zero-shot on tasks contained in the pretraining set on unseen objects and environments (Figure 15, Figure 3). On AgiBot G1, from-scratch VLAs achieve near-zero task progress across all categories; even on simple PnP Easy tasks, they may reach the correct novel object yet fail to interact reliably. In contrast, DREAMZERO attains 62.2% average task progress—more than 2× the best pretrained VLA baseline (27.4%), despite those baselines being pretrained on thousands of hours of cross-embodiment data before training on our mix. On DROID-Franka, DREAMZERO trained only on DROID (no cross-embodiment pretraining) also outperforms pretrained multi-embodiment baselines. As of *Jan 30, 2026*, our model ranks among the top methods on the RoboArena Leaderboard, further supporting that WAMs benefit from heterogeneous data.

We attribute this to the joint video-action formulation: VLAs must learn direct observation-to-action mappings from massive robot datasets, whereas WAMs exploit video generation as a strong prior for action prediction, improving generalization to unseen environments. We observe close alignment between generated videos and real execution, including suboptimal behaviors (Appendix A.8). Most DREAMZERO failures arise from video generation errors, not from action decoding, indicating that better video backbones should directly boost WAM performance.

Q2. Do WAMs generalize to unseen tasks? Figure 4 evaluates generalization to 10 tasks entirely absent from the pretraining distribution, including untying shoelaces, ironing, painting with a brush, and shaking hands. On AgiBot G1, from-scratch VLAs achieve near-zero task progress ($< 1\%$), while DREAMZERO reaches 39.5% on average. DREAMZERO also significantly outperforms pretrained VLA baselines (39.5% vs. 16.3%), even though those baselines may have encountered some of these tasks during cross-embodiment pretraining. On the DROID-Franka setup, DREAMZERO

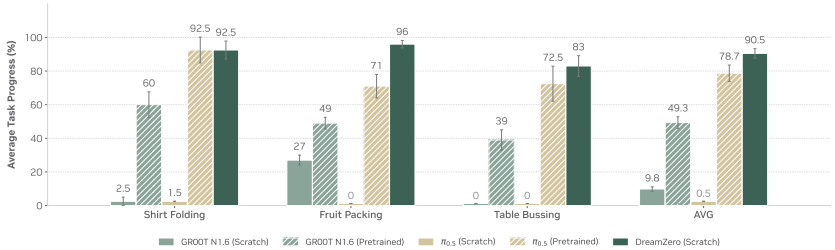


Figure 5: **Posttraining Results**. WAMs enable stronger post-training results across three tasks. We observe that the improvement of DREAMZERO over VLAs correlate with the diversity of the post-training dataset.

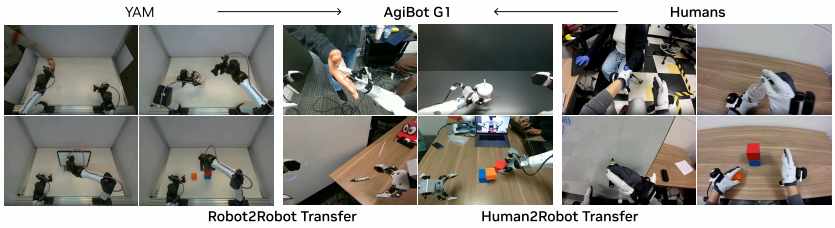


Figure 6: **Cross-Embodiment Transfer.** We explore robot-to-robot (YAM → AgiBot) and human-to-robot embodiment transfer to unseen tasks.

significantly outperforms (49% task progress, 22.5% success rate) other pretrained baselines (31% task progress, 12.5% success rate for GR00T N1.6 and 32% task progress, 10% success rate for $\pi_{0.5}$). Examples of free-form prompting evaluations are shown in Figure 10.

Q3. Do WAMs improve post-training performance? We investigate whether WAMs retain their generalization advantage after fine-tuning on task-specific data. Figure 5 shows results on three tasks with varying distribution diversity. DREAMZERO matches or outperforms VLA baselines across all tasks, with clear gains on fruit packing and comparable performance on shirt folding and table bussing. Despite lacking cross-embodiment pretraining, DREAMZERO remains competitive, indicating that its environment generalization is preserved after post-training.

Q4. Do WAMs enable cross-embodiment transfer to unseen tasks? We investigate whether video-only data from different embodiments can improve generalization to unseen tasks. We explore two settings (Figure 6): (1) robot-to-robot transfer using YAM robot (20 minutes), and (2) human-to-robot transfer using egocentric demonstrations (12 minutes). We co-train from DREAMZERO using only video prediction for cross-embodiment data (no actions) and joint video-action for pretraining data. Table 1 shows both settings improve over baseline: robot-to-robot (38.3% → 55.4%) and human-to-robot (38.3% → 54.3%).

Table 1: **Cross-Embodiment Transfer Results.** Average task progress on unseen tasks (± standard error). Both transfer settings improve over baseline (result from Table 4) using only 10–20 minutes of video-only demonstration data.

Method	Task Progress
DREAMZERO	38.3% ± 7.6%
DREAMZERO + Human2Robot Transfer	54.3% ± 10.4%
DREAMZERO + Robot2Robot Transfer	55.4% ± 9.5%

These results point to a promising property of WAMs: unlike recent VLA approaches to embodiment transfer (Kareer et al., 2025; Team, 2025), our method relies solely on visual information without action labels. While current success rates remain moderate, the consistent improvement from just 10–20 minutes of video-only data provides an early signal that cross-embodiment visual experience transfers meaningfully. This opens a potential scaling pathway: abundant human video data—orders of magnitude larger than robot datasets—could enable WAMs to acquire diverse skills without action annotation, pending further research into strengthening the transfer mechanism.

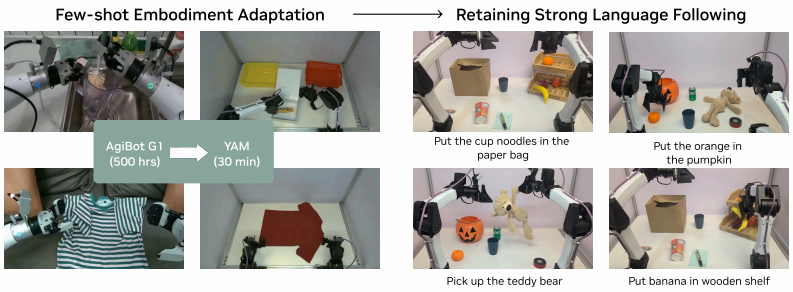


Figure 7: **Few-shot Embodiment Adaptation.** We explore few-shot embodiment adaptation by post-training on 30 minutes of new embodiment data and evaluating on novel objects unseen during training.

Table 2: **Ablations.** Task progress on PnP Easy tasks (50K steps, batch 32).

Config	Size	Data	Progress
Data Diversity			
DreamZero (AR)	14B	Repetitive	33%
DreamZero (AR)	14B	Diverse	50%
Model Scale			
DreamZero (AR)	5B	Diverse	21%
DreamZero (AR)	14B	Diverse	50%
Architecture			
DreamZero (AR)	14B	Diverse	50%
DreamZero (BD)	14B	Diverse	50%

5.2 Model and Data Ablations

We conduct ablations to isolate the contributions of data diversity, model scale, and architecture. Due to computational constraints, all ablation models are trained for 50K steps with batch size 32 and evaluated on *PnP Easy* tasks.

Data Diversity. Table 2 shows diverse data substantially improves generalization (33% $\rightarrow$ 50%) compared to repetitive demonstrations. We hypothesize that since video prediction is inherited from pretraining, the key challenge is learning inverse dynamics, which requires diverse state-action correspondences.

Model Scale. WAMs exhibit clear scaling: 14B significantly outperforms 5B (50% vs. 21%). We also trained VLAs at 5B and 14B scales, but both achieve 0% task progress, suggesting scaling alone doesn't address VLAs' difficulty with diverse data.

Architecture. Autoregressive (AR) and bidirectional (BD) achieve similar task progress, but AR produces smoother motions and enables 3–4$\times$ faster inference via KV caching. Additional ablations on few-shot embodiment adaptation and DreamZero-Flash are in Appendix A.1.

6 Discussion, Limitations, and Future Work

While we have identified that larger backbones and diverse data boost performance, formal scaling laws for WAMs—relating model size, dataset size, and compute—remain to be fully explored (Kaplan et al., 2020). Furthermore, our experiments with egocentric human data are currently limited to lab scales, where leverage massive in-the-wild datasets (Grauman et al., 2022; Hoque et al., 2025; Chen et al., 2026) will be further explored. Current WAMs operate primarily as "System 1" models with short-horizon visual memory. To achieve robust long-horizon execution, integrating "System 2" reasoning via modular dual systems (Shi et al., 2025), unified systems (Deng et al., 2025), or long-context video world models (Ball et al., 2025; HunyuanWorld, 2025) is necessary, which we leave as future work. Additionally, while we achieve 7Hz on GB200s, WAMs remain computationally expensive compared to 20Hz+ VLAs due to their iterative denoising nature. Future research into smaller, high-generalization video backbones will enable real-time inference.

REFERENCES

Michael S Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *arXiv preprint arXiv:2303.08797*, 2023.

Arslan Ali, Junjie Bai, Maciej Bala, Yogesh Balaji, Aaron Blakeman, Tiffany Cai, Jiaxin Cao, Tianshi Cao, Elizabeth Cha, Yu-Wei Chao, et al. World simulation with video foundation models for physical ai. *arXiv preprint arXiv:2511.00062*, 2025.

Pranav Atreya, Karl Pertsch, Tony Lee, Moo Jin Kim, Arhan Jain, Artur Kuramshin, Clemens Eppner, Cyrus Neary, Edward Hu, Fabio Ramos, et al. Roboarena: Distributed real-world evaluation of generalist robot policies. *arXiv preprint arXiv:2506.18123*, 2025.

Philip J. Ball, Jakob Bauer, Frank Belletti, Bethanie Brownfield, Ariel Ephrat, Shlomi Fruchter, Agrim Gupta, Kristian Holsheimer, Aleksander Holynski, Jiri Hron, Christos Kaplanis, Marjorie Limont, Matt McGill, Yanko Oliveira, Jack Parker-Holder, Frank Perbet, Guy Scully, Jeremy Shar, Stephen Spencer, Omer Tov, Ruben Villegas, Emma Wang, Jessica Yung, Cip Baetu, Jordi Berbel, David Bridson, Jake Bruce, Gavin Buttimore, Sarah Chakera, Bilva Chandra, Paul Collins, Alex Cullum, Bogdan Damoc, Vibha Dasagi, Maxime Gazeau, Charles Gbadamosi, Woohyun Han, Ed Hirst, Ashyana Kachra, Lucie Kerley, Kristian Kjems, Eva Knoepfel, Vika Koriakin, Jessica Lo, Cong Lu, Zeb Mehring, Alex Moufarek, Henna Nandwani, Valeria Oliveira, Fabio Pardo, Jane Park, Andrew Pierson, Ben Poole, Helen Ran, Tim Salimans, Manuel Sanchez, Igor Saprykin, Amy Shen, Sailesh Sidhwani, Duncan Smith, Joe Stanton, Hamish Tomlinson, Dimple Vijaykumar, Luyu Wang, Piers Wingfield, Nat Wong, Keyang Xu, Christopher Yew, Nick Young, Vadim Zubov, Douglas Eck, Dumitru Erhan, Koray Kavukcuoglu, Demis Hassabis, Zoubin Gharamani, Raia Hadsell, Aäron van den Oord, Inbar Mosseri, Adrian Bolton, Satinder Singh, and Tim Rocktäschel. Genie 3: A new frontier for world models. 2025.

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.

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

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

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024a.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. $\pi 0$: A vision-language-action flow model for general robot control. *URL https://arxiv. org/abs/2410.24164*, 2024b.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

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

Brianna Zitkovich. Rt-1: Robotics transformer for real-world control at scale. In *arXiv preprint arXiv:2212.06817*, 2022.

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

Anthony Brohan, Yevgen Chebotar, Chelsea Finn, Karol Hausman, Alexander Herzog, Daniel Ho, Julian Ibarz, Alex Irpan, Eric Jang, Ryan Julian, et al. Do as i can, not as i say: Grounding language in robotic affordances. In *Conference on robot learning*, pp. 287–318. PMLR, 2023b.

Qingwen Bu, Yanting Yang, Jisong Cai, Shenyuan Gao, Guanghui Ren, Maoqing Yao, Ping Luo, and Hongyang Li. Univla: Learning to act anywhere with task-centric latent actions. *arXiv preprint arXiv:2505.06111*, 2025.

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

Boyuan Chen, Zhuo Xu, Sean Kirmani, Brain Ichter, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14455–14465, 2024.

Boyuan Chen, Tianyuan Zhang, Haoran Geng, Kiwhan Song, Caiyi Zhang, Peihao Li, William T Freeman, Jitendra Malik, Pieter Abbeel, Russ Tedrake, et al. Large video planner enables generalizable robot control. *arXiv preprint arXiv:2512.15840*, 2025.

Delong Chen, Tejaswi Kasarla, Yejin Bang, Mustafa Shukor, Willy Chung, Jade Yu, Allen Bolourchi, Theo Moutakanni, and Pascale Fung. Action100m: A large-scale video action dataset. *arXiv preprint arXiv:2601.10592*, 2026.

Chaorui Deng, Deyao Zhu, Kunchang Li, Chenhui Gou, Feng Li, Zeyu Wang, Shu Zhong, Weihao Yu, Xiaonan Nie, Ziang Song, et al. Emerging properties in unified multimodal pretraining. *arXiv preprint arXiv:2505.14683*, 2025.

Kamil Dreczkowski, Pietro Vitiello, Vitalis Vosylius, and Edward Johns. Learning a thousand tasks in a day. *Science Robotics*, 10(108):eadv7594, 2025.

Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.

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.

Yilun Du, Sherry Yang, Pete Florence, Fei Xia, Ayzaan Wahid, brian ichter, Pierre Sermanet, Tianhe Yu, Pieter Abbeel, Joshua B. Tenenbaum, Leslie Pack Kaelbling, Andy Zeng, and Jonathan Tompson. Video language planning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=9pKtcJcMP3.

Zhiyuan Feng, Zhaolu Kang, Qijie Wang, Zhiying Du, Jiongrui Yan, Shubin Shi, Chengbo Yuan, Huizhi Liang, Yu Deng, Qixiu Li, et al. Seeing across views: Benchmarking spatial reasoning of vision-language models in robotic scenes. *arXiv preprint arXiv:2510.19400*, 2025.

Jensen Gao, Suneel Belkhale, Sudeep Dasari, Ashwin Balakrishna, Dhruv Shah, and Dorsa Sadigh. A taxonomy for evaluating generalist robot policies. *arXiv preprint arXiv:2503.01238*, 2025.

Kaifeng Gao, Jiaxin Shi, Hanwang Zhang, Chunping Wang, Jun Xiao, and Long Chen. Ca2-vdm: Efficient autoregressive video diffusion model with causal generation and cache sharing. *arXiv preprint arXiv:2411.16375*, 2024.

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

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia

Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. The llama 3 herd of models. 2024. URL https://arxiv.org/abs/2407.21783.

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

Pranav Guruprasad, Yangyue Wang, Sudipta Chowdhury, Harshvardhan Sikka, and Paul Pu Liang. Benchmarking vision, language, & action models in procedurally generated, open ended action environments. *arXiv preprint arXiv:2505.05540*, 2025.

Ryan Hoque, Peide Huang, David J Yoon, Mouli Sivapurapu, and Jian Zhang. Egodex: Learning dexterous manipulation from large-scale egocentric video. *arXiv preprint arXiv:2505.11709*, 2025.

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. *arXiv preprint arXiv:2412.14803*, 2024.

Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. Inner monologue: Embodied reasoning through planning with language models. In *6th Annual Conference on Robot Learning*.

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

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.

Team HunyuanWorld. Hy-world 1.5: A systematic framework for interactive world modeling with real-time latency and geometric consistency. *arXiv preprint*, 2025.

Sam Ade Jacobs, Masahiro Tanaka, Chengming Zhang, Minjia Zhang, Shuaiwen Leon Song, Samyam Rajbhandari, and Yuxiong He. Deepspeed ulysses: System optimizations for enabling training of extreme long sequence transformer models. *arXiv preprint arXiv:2309.14509*, 2023. URL https://arxiv.org/abs/2309.14509.

Arhan Jain, Mingtong Zhang, Kanav Arora, William Chen, Marcel Torne, Muhammad Zubair Irshad, Sergey Zakharov, Yue Wang, Sergey Levine, Chelsea Finn, et al. Polaris: Scalable real-to-sim evaluations for generalist robot policies. *arXiv preprint arXiv:2512.16881*, 2025.

Joel Jang, Seonghyeon Ye, Zongyu Lin, Jiannan Xiang, Johan Bjorck, Yu Fang, Fengyuan Hu, Spencer Huang, Kaushil Kundalia, Yen-Chen Lin, Loïc Magne, Ajay Mandlekar, Avnish Narayan, You Liang Tan, Guanzhi Wang, Jing Wang, Qi Wang, Yinzhen Xu, Xiaohui Zeng, Kaiyuan Zheng, Ruijie Zheng, Ming-Yu Liu, Luke Zettlemoyer, Dieter Fox, Jan Kautz, Scott Reed, Yuke Zhu, and Linxi Fan. Dreamgen: Unlocking generalization in robot learning through video world models. In *9th Annual Conference on Robot Learning*, 2025. URL https://openreview.net/forum?id=3CnxNqmklv.

Yang Jin, Zhicheng Sun, Ningyuan Li, Kun Xu, Hao Jiang, Nan Zhuang, Quzhe Huang, Yang Song, Yadong Mu, and Zhouchen Lin. Pyramidal flow matching for efficient video generative modeling. *arXiv preprint arXiv:2410.05954*, 2024.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Simar Kareer, Karl Pertsch, James Darpinian, Judy Hoffman, Danfei Xu, Sergey Levine, Chelsea Finn, and Suraj Nair. Emergence of human to robot transfer in vision-language-action models. *arXiv preprint arXiv:2512.22414*, 2025.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

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

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

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.

Po-Chen Ko, Jiayuan Mao, Yilun Du, Shao-Hua Sun, and Joshua B. Tenenbaum. Learning to act from actionless videos through dense correspondences. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Mhb5fpA1T0.

Nishanth Kumar, William Shen, Fabio Ramos, Dieter Fox, Tomás Lozano-Pérez, Leslie Pack Kaelbling, and Caelan Reed Garrett. Open-world task and motion planning via vision-language model inferred constraints. *arXiv preprint arXiv:2411.08253*, 2024.

Jason Lee, Jiafei Duan, Haoquan Fang, Yuquan Deng, Shuo Liu, Boyang Li, Bohan Fang, Jieyu Zhang, Yi Ru Wang, Sangho Lee, et al. Molmoact: Action reasoning models that can reason in space. *arXiv preprint arXiv:2508.07917*, 2025.

Shuang Li, Yihuai Gao, Dorsa Sadigh, and Shuran Song. Unified video action model. *arXiv preprint arXiv:2503.00200*, 2025a.

Yi Li, Yuquan Deng, Jesse Zhang, Joel Jang, Marius Memmel, Raymond Yu, Caelan Reed Garrett, Fabio Ramos, Dieter Fox, Anqi Li, et al. Hamster: Hierarchical action models for open-world robot manipulation. *arXiv preprint arXiv:2502.05485*, 2025b.

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 *8th Annual Conference on Robot Learning*, 2024. URL https://openreview.net/forum?id=InT87E5sr4.

Junbang Liang, Pavel Tokmakov, Ruoshi Liu, Sruthi Sudhakar, Paarth Shah, Rares Ambrus, and Carl Vondrick. Video generators are robot policies. *arXiv preprint arXiv:2508.00795*, 2025.

Yue Liao, Pengfei Zhou, Siyuan Huang, Donglin Yang, Shengcong Chen, Yuxin Jiang, Yue Hu, Jingbin Cai, Si Liu, Jianlan Luo, et al. Genie envisioner: A unified world foundation platform for robotic manipulation. *arXiv preprint arXiv:2508.05635*, 2025.

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

Feng Liu, Shiwei Zhang, Xiaofeng Wang, Yujie Wei, Haonan Qiu, Yuzhong Zhao, Yingya Zhang, Qixiang Ye, and Fang Wan. Timestep embedding tells: It's time to cache for video diffusion model. *arXiv preprint arXiv:2411.19108*, 2024.

Hao Liu, Matei Zaharia, and Pieter Abbeel. Ring attention with blockwise transformers for near-infinite context. *arXiv preprint arXiv:2310.01889*, 2023. URL https://arxiv.org/abs/2310.01889.

Jiacheng Liu, Chang Zou, Yuanhuiyi Lyu, Junjie Chen, and Linfeng Zhang. From reusing to forecasting: Accelerating diffusion models with taylorseers. *arXiv preprint arXiv:2503.06923*, 2025.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

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

NVIDIA Corporation. Nvidia model-optimizer, 2024. URL https://github.com/NVIDIA/Model-Optimizer.

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

Physical Intelligence. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

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.

Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. Progprompt: Generating situated robot task plans using large language models. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

Generalist AI Team. Gen-0: Embodied foundation models that scale with physical interaction. *Generalist AI Blog*, 2025. https://generalistai.com/blog/preview-uqlxvb-bb.html.

Team Wan. Wan: Open and advanced large-scale video generative models. 2025.

Hansi Teng, Hongyu Jia, Lei Sun, Lingzhi Li, Maolin Li, Mingqiu Tang, Shuai Han, Tianning Zhang, WQ Zhang, Weifeng Luo, et al. Magi-1: Autoregressive video generation at scale. *arXiv preprint arXiv:2505.13211*, 2025.

Homer Walke, Kevin Black, Abraham Lee, Moo Jin Kim, Max Du, Chongyi Zheng, Tony Zhao, Philippe Hansen-Estruch, Quan Vuong, Andre He, Vivek Myers, Kuan Fang, Chelsea Finn, and Sergey Levine. Bridgedata v2: A dataset for robot learning at scale. In *Conference on Robot Learning (CoRL)*, 2023.

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

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 *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=NxoFmGgWC9.

Jianwei Yang, Reuben Tan, Qianhui Wu, Ruijie Zheng, Baolin Peng, Yongyuan Liang, Yu Gu, Mu Cai, Seonghyeon Ye, Joel Jang, Yuquan Deng, Lars Liden, and Jianfeng Gao. Magma: A foundation model for multimodal AI agents, 2025.

Sherry Yang, Yilun Du, Seyed Kamyar Seyed Ghasemipour, Jonathan Tompson, Leslie Pack Kaelbling, Dale Schuurmans, and Pieter Abbeel. Learning interactive real-world simulators. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=sFyTZEqmUY.

Seonghyeon Ye, Joel Jang, Byeongguk Jeon, Se June 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 *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=VYOe2eBQeh.

Tianwei Yin, Qiang Zhang, Richard Zhang, William T Freeman, Fredo Durand, Eli Shechtman, and Xun Huang. From slow bidirectional to fast autoregressive video diffusion models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 22963–22974, 2025.

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. *arXiv preprint arXiv:2503.22020*, 2025.

Wenliang Zhao, Lujia Bai, Yongming Rao, Jie Zhou, and Jiwen Lu. UniPC: A unified predictor-corrector framework for fast sampling of diffusion models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=hrkmlPhp1u.

Ruijie Zheng, Yongyuan Liang, Shuaiyi Huang, Jianfeng Gao, Hal Daumé III, Andrey Kolobov, Furong Huang, and Jianwei Yang. TraceVLA: Visual trace prompting enhances spatial-temporal awareness for generalist robotic policies. In *The Thirteenth International Conference on Learning Representations*, 2025a.

Ruijie Zheng, Jing Wang, Scott Reed, Johan Bjorck, Yu Fang, Fengyuan Hu, Joel Jang, Kaushil Kundalia, Zongyu Lin, Loic Magne, et al. Flare: Robot learning with implicit world modeling. *arXiv preprint arXiv:2505.15659*, 2025b.

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 manipulations in cross-task generalization. *arXiv preprint arXiv:2505.15660*, 2025.

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.

Chuning Zhu, Raymond Yu, Siyuan Feng, Benjamin Burchfiel, Paarth Shah, and Abhishek Gupta. Unified world models: Coupling video and action diffusion for pretraining on large robotic datasets. *arXiv preprint arXiv:2504.02792*, 2025.

A APPENDIX

A.1 ADDITIONAL ABLATIONS

Few-shot Embodiment Adaptation. We post-trained the DREAMZERO-AgiBot policy on a new bimanual manipulator (YAM robot) using only 11 unique tasks (∼30 minutes) of play data. As illustrated in Figure 8, despite limited data and diversity, the post-trained policy retains strong language following ability, even generalizing to novel objects unseen during training, including pumpkins, teddy bears, pens, cup noodles, and paper bags. Preliminary experiments reveal strong video-action alignment even with minimal data, demonstrating promising cross-embodiment transfer. Consistent with findings from AgiBot embodiment, the primary failure modes come from video prediction error. We hypothesize that incorporating more diverse tasks during post-training will improve video prediction in future iterations.

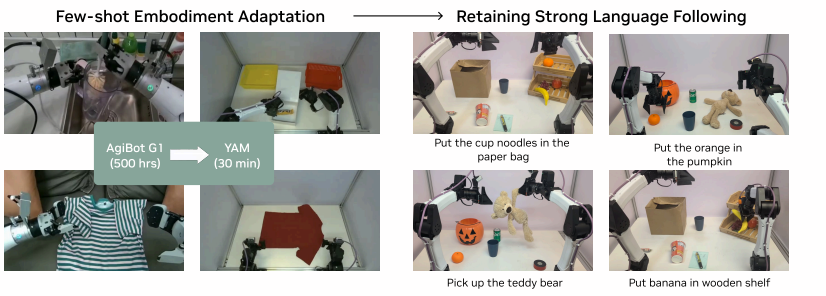


Figure 8: **Few-shot Embodiment Adaptation.** We explore few-shot embodiment adaptation by post-training on 30 minutes of new embodiment data and evaluating on novel objects unseen during training.

Table 3: DREAMZERO**-Flash Evaluation.** Task progress on table bussing with varying denoising steps. DREAMZERO-Flash recovers most 4-step performance using only 1 step.

Method	Steps	Task Progress	Speed	× Speedup
DREAMZERO	4	83%	350ms	1.00
DREAMZERO	1	52%	150ms	2.33
DREAMZERO-Flash	1	74%	150ms	2.33

DreamZero-Flash Evaluation. We evaluate whether DREAMZERO-Flash can maintain task performance under aggressive single-step denoising. As shown in Table 3, reducing DREAMZERO from 4 steps to 1 step drops task progress substantially (83% → 52%) on the table bussing task. By decoupling the video and action timesteps, DREAMZERO-Flash recovers most of this performance at single-step inference (74%), only 9% below the 4-step baseline while being ∼2× faster. This demonstrates that decoupled noise scheduling enables effective speed–accuracy trade-offs for real-time deployment.

A.2 SAMPLES OF POLICY ROLLOUT

Fig 9 and 10 show samples of diverse rollouts of DREAMZERO, including unseen tasks, where the robots are executing tasks that were totally unseen during training such as "Fist bump the human" or "Pass spatula from right arm to left arm".

A.3 MODEL AND TRAINING DETAILS

We visualize the attention mask for training and inference in Figure 11. For DREAMZERO, we set each chunk as $K = 2$ latent frames. From preliminary results, we have observed that $K = 2$ outperforms $K = 1$ empirically. We set the number of chunks $M = 4$ by default. If the trajectory length is shorter than $M = 4$, M can be smaller than 4. For Agibot training data, the video is

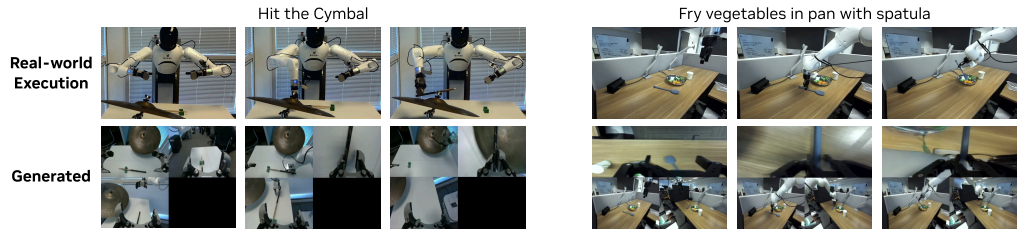


Figure 9: **Joint Video and Action Prediction**. DREAMZERO jointly generates both video and action modality. We observe that the predicted action is closely aligned with the generated video. Note that the illustrated examples are totally unseen tasks.

Figure 10: **Free-form Evaluation.** DREAMZERO executes diverse tasks from natural language instructions, including unseen tasks that belong to object manipulation, tool use, and human-robot interaction.

sampled at 5FPS ratio and action is sampled at 30Hz. We use the action horizon of $H = 48$. Therefore, the video and action span 1.6 seconds per chunk. For DROID training, the video is sampled at 5FPS ratio and action is sampled at 15Hz. We use the action horizon of $H = 24$. Therefore, same as Agibot, the video and action span 1.6 seconds per chunk. The maximum context length is 8 latent frames (4x2), which is equivalent to 33 raw frames, spanning 6.6 seconds. We leave increasing the visual context for WAMs as future work.

A.4 BIDIRECTIONAL VS. AUTOREGRESSIVE WAMS

In Figure 12 we compare the modality alignment between bidirectional and autorgressive WAMs. Given a sampled timestamp point from a trajectory, bidirectional WAMs can apply video subsampling to maintain language following capability. From preliminary experiments, we observed that not applying video subsampling and instead setting the video span same as the action span lead to significant degradation on language following capability. Although video subsampling mitigates this issue, the language and video can be still largely misaligned if the sampling point is close to the end of the trajectory. Also, video subsampling naturally distorts native FPS, which can potentially lead to misalignment video and action. On the other hand, autoregressive WAMs mitigates the misalignment issue by utilizing video context. We conjecture that by leveraging the positional embedding information, autoregressive WAMs can implictly map the video chunk and part of the language that corresponds to the video during training, mitigating misalignment between video and language. Also, since language following capability can be improved with video and language alignment, autoregressive WAMs can use a fixed FPS, which aligns the video and action modality.

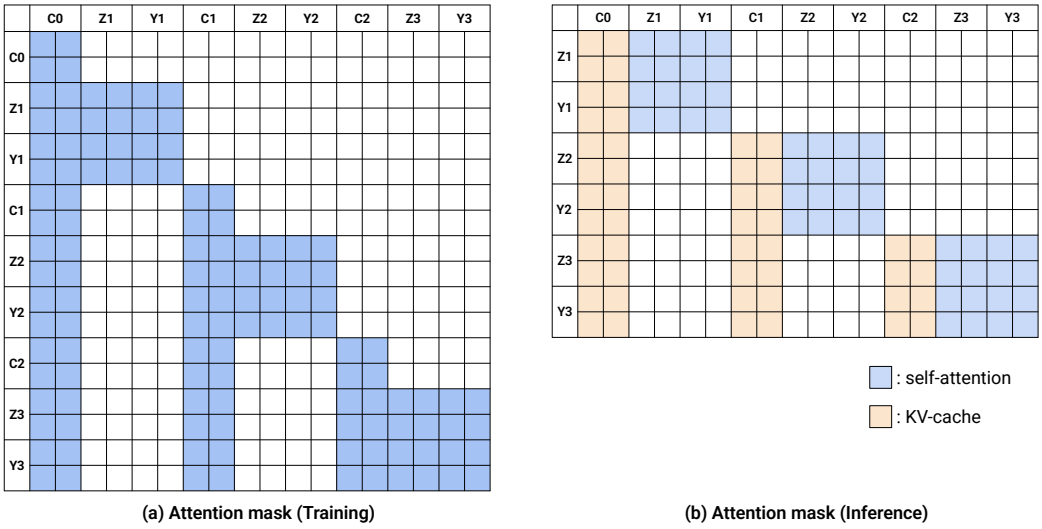


Figure 11: **Attention startegy of** DREAMZERO**.** (a) Self-Attention mask for DreamZero training. Given conditioning frames (C0, C1, C2), we train the model to predict the velocities of next frames (Z1, Z2, Z3) and actions (Y1, Y2, Y3). (b) During inference, we compute the KV-cache of conditional frames and concatenate them to predict the action and frames.

A.5 REAL-TIME EXECUTION DETAILS

This appendix provides additional details on the system-, implementation-, and model-level optimizations introduced in Section 3.2.

Diffusion-based WAMs inherit powerful generalization from video foundation models, but their iterative denoising process creates a fundamental tension with reactive robotic control. We address three questions: (1) What prevents diffusion WAMs from being reactive policies? (2) What execution structure resolves this? (3) What constraints does that structure impose on the model and system?

A.5.1 THE REACTIVITY GAP

Reactive policies must respond to environmental changes within tens of milliseconds. A naive implementation of DREAMZERO on a single GPU requires approximately 6.2 seconds per action chunk due to three bottlenecks: (1) iterative denoising across 16 diffusion steps required for smooth actions, (2) the computational cost of a 14B parameter DiT backbone, and (3) sequential execution that blocks robot motion during inference. This latency makes closed-loop control infeasible.

A.5.2 ASYNCHRONOUS CLOSED-LOOP EXECUTION

Our first step towards resolving this is through asynchronous execution that decouples inference from action execution. Rather than waiting for each inference to complete, the motion controller continuously executes the most recent action chunk while inference runs concurrently on the latest observation. This structure transforms the latency constraint from "inference must complete before the robot moves" to "inference must complete before the current action chunk expires." In our experiments, we deploy policies at an action horizon of 48 steps at 30Hz control frequency (1.6 seconds per chunk). Hence, we target inference latency below approximately 200ms to ensure sufficient overlap for smooth, reactive control.

A.5.3 SYSTEM-LEVEL OPTIMIZATIONS

Given the asynchronous execution structure, we optimize inference throughput through parallelism and caching.

Algorithm 1 DREAMZERO Training (Flow Matching)

Input: Dataset $\mathcal{D}$, Text condition $\mathbf{c}$
Hyperparams: Number of chunks M
Model: $\mathbf{u}_\theta$ (Joint Video-Action DiT)
while not converged **do**
  Sample trajectory $\tau \sim \mathcal{D}$
  Encode video to clean latents $\mathbf{z}_1^{1:M}$, normalize actions $\mathbf{a}_1^{1:M}$
  Split τ into M chunks
  for $k = 1, \dots, M$ **do**
    Define clean context $\mathcal{C}_k \leftarrow \bigcup_{j=1}^{k-1} \{\mathbf{z}_1^j, \mathbf{a}_1^j\}$
    Sample timestep $t_k \sim \mathcal{U}(0, 1)$
    if Flash Mode **then**
      $t_{vid} \sim \text{Beta}(3, 1)$, $t_{act} \sim \mathcal{U}(0, 1)$
    else
      $t_{vid} \leftarrow t_k,\ t_{act} \leftarrow t_k$
    end if
    Sample noise $\mathbf{z}_0^k, \mathbf{a}_0^k \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
    Interpolate (Eq. 2):
    $\mathbf{z}_t^k \leftarrow t_{vid}\mathbf{z}_1^k + (1 - t_{vid})\mathbf{z}_0^k$
    $\mathbf{a}_t^k \leftarrow t_{act}\mathbf{a}_1^k + (1 - t_{act})\mathbf{a}_0^k$
    Predict velocity:
    $\mathbf{v}_{pred} \leftarrow \mathbf{u}_\theta([\mathbf{z}_t^k, \mathbf{a}_t^k]; \mathcal{C}_k, \mathbf{c}, \mathbf{q}_k, t_k)$
    Target vel. $\mathbf{v}_t^k := [\mathbf{z}_1^k, \mathbf{a}_1^k] - [\mathbf{z}_0^k, \mathbf{a}_0^k]$
    Loss $\mathcal{L} \leftarrow ||\mathbf{v}_{pred} - \mathbf{v}_t^k||^2$
    Update $\theta \leftarrow \theta - \eta \nabla \mathcal{L}$
  end for
end while

Algorithm 2 DREAMZERO Inference (Closed-Loop Control)

Input: Instruction $\mathbf{c}$, Initial Image $\mathbf{o}_{\text{init}}$, State $\mathbf{q}_{\text{init}}$
Hyperparams: Steps N, Cache Thresh ϵ
Init: $\mathcal{KV} \leftarrow \emptyset$, $\mathbf{v}_{\text{prev}} \leftarrow \emptyset$, $\mathbf{q}_{\text{curr}} \leftarrow \mathbf{q}_{\text{init}}$
// 1. Prefill Cache (Context Phase, $t = 0$)
$\mathbf{z}_{\text{init}} \leftarrow \text{VAE}(\mathbf{o}_{\text{init}})$
// Pass clean video, no action/state
$(\cdot, \cdot, \mathcal{KV}) \leftarrow \mathbf{u}_\theta([\mathbf{z}_{\text{init}}, \emptyset]; \mathcal{KV}, \mathbf{c}, \emptyset, t = 0, \text{update=True})$
// 2. Autoregressive Loop
while task not done **do**
  Sample $\mathbf{x}_0 = [\mathbf{z}_0, \mathbf{a}_0] \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
  for $i = 0 \dots N - 1$ **do**
    $t_i, t_{i+1} \leftarrow \text{Scheduler}(i, N)$
    if $\mathbf{v}_{\text{prev}} \neq \emptyset$ **and** $\text{CosSim}(\mathbf{v}_{\text{prev}}, \mathbf{v}_{\text{last}}) > \epsilon$ **then**
      $\mathbf{v}_i \leftarrow \mathbf{v}_{\text{prev}}$
    else
      $(\mathbf{v}_i^{\text{vid}}, \mathbf{v}_i^{\text{act}}, \cdot) \leftarrow \mathbf{u}_\theta(\mathbf{x}_{t_i}; \mathcal{KV}, \mathbf{c}, \mathbf{q}_{\text{curr}}, t_i, \text{update=False})$
      $\mathbf{v}_i \leftarrow [\mathbf{v}_i^{\text{vid}}, \mathbf{v}_i^{\text{act}}]$
      $\mathbf{v}_{\text{prev}} \leftarrow \mathbf{v}_i$
    end if
    $\mathbf{x}_{t_{i+1}} \leftarrow \mathbf{x}_{t_i} + \text{Step}(\mathbf{v}_i, t_i, t_{i+1})$
  end for
  // 3. Execution & Cache Update
  $\hat{\mathbf{a}} \leftarrow \text{Filter}(\mathbf{x}_1^{\text{action}})$
  Async Execute $\hat{\mathbf{a}}$ on Robot
  Real observation $\mathbf{o}_{\text{real}}, \mathbf{q}_{\text{real}}$
  $\mathbf{q}_{\text{curr}} \leftarrow \mathbf{q}_{\text{real}}$, $\mathbf{z}_{\text{real}} \leftarrow \text{VAE}(\mathbf{o}_{\text{real}})$
  $(\cdot, \cdot, \mathcal{KV}) \leftarrow \mathbf{u}_\theta([\mathbf{z}_{\text{real}}, \emptyset]; \mathcal{KV}, \mathbf{c}, \emptyset, t = 0, \text{update=True})$
end while

2-D Parallelism. To address the computation bottleneck inherent to DiT, we employed two parallelism techniques: (1) Classifier-Free Guidance (CFG) parallelism, and (2) Context parallelism. Standard CFG requires two distinct model evaluations—the conditional and unconditional forward passes—which are typically executed sequentially. We parallelized these operations by distributing the conditioned and null-conditioned score estimations across two independent GPUs. This effectively reduced the latency per diffusion step by nearly half, with no impact on overall model quality.

We further improve performance by parallelizing token processing across four GPUs using context parallelism. Several approaches address the communication overheads introduced by attention under context parallelism: (a) DeepSpeed-Ulysses (Jacobs et al., 2023) shards attention heads across devices and relies on all-to-all communication to exchange QKV tensors and attention outputs; (b) Llama3 (Grattafiori et al., 2024) shards query tokens across GPUs and uses all-gather operations to replicate key and value tensors; (c) Ring Attention (Liu et al., 2023) overlaps computation and communication by partitioning attention into phases interleaved with point-to-point exchanges.

In our implementation, we adopt the all-gather strategy for key and value tensors to perform self-attention across devices. For cross-attention, key and value tensors are replicated and cached, incurring no communication overhead. Applying context parallelism across four GPUs reduces inference latency by 47% without observable degradation in video or action quality.

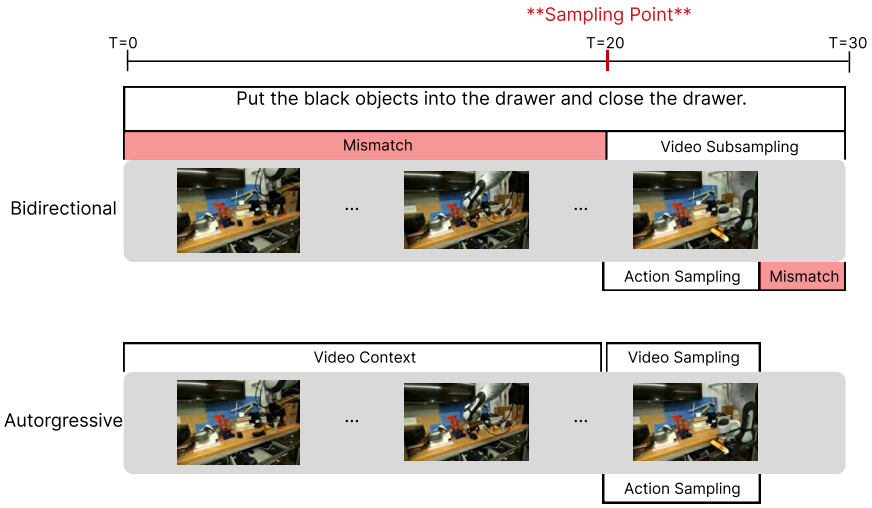


Figure 12: We illustrate the difference between bidirectional and autoregressive WAMs. Unlike bidirectional WAMs which suffer from video-language and video-action alignment, autoregressive WAMs mitigate this misalignment by using video context.

DiT Caching. The iterative nature of DiT inference imposes a significant computational bottleneck for real-time applications. Previous efforts have focused on training-free caching techniques that exploit temporal redundancies across diffusion steps. TeaCache (Liu et al., 2024) leverages the relative L_1 difference between timestep-embedding modulated inputs as a heuristic to estimate output variance and skip redundant computations, while TaylorSeer (Liu et al., 2025) employs higher-order Taylor expansions to extrapolate future latent states from historical derivatives. In DreamZero, we implement a caching mechanism that exploits the directional consistency of velocity vectors learned during flow matching.

During inference, the model tracks cosine similarity between successive velocity predictions. When this metric exceeds a predefined threshold (τ), the model bypasses the DiT forward pass for a window of a few steps by reusing the cached velocity vector. This adaptive scheduling concentrates computational resources on critical trajectory updates, reducing the average number of DiT steps from 16 to 4 with minimal degradation to predicted video and action fidelity.

Asynchronous Execution. Sequential execution of action blocks introduces stalls while waiting for upstream models to produce the next chunk, making the robot more open-loop and less reactive to real-time state changes, especially with large chunk sizes. We use an asynchronous execution mechanism that decouples model inference from action execution, allowing both stages to run concurrently. The motion controller always executes the most recent action scheduled for the current timestamp, while the inference module always uses the latest observation.

A.5.4 Implementation-level Optimizations

Torch Compile and CUDA Graphs. Inference is predominantly CPU-bound due to kernel launch overheads and Python execution. We employ torch.compile with CUDA Graphs (mode="reduce-overhead") and enforce full graph capture (fullgraph=True) to eliminate graph breaks. In addition to reducing CPU overhead, torch.compile decreases memory bandwidth requirements through operator fusion. We apply compilation to five model components: the diffusion transformer, scheduler, text encoder, image encoder, and VAE. We enforce static shapes (dynamic=False), which results in multiple recompilations during the first inference trajectory due to the evolving KV cache shape. From the second trajectory onward, inference proceeds without recompilation. To enable error-free compilation, we refactor the model to follow a functional programming paradigm: the KV cache is passed explicitly as input and returned as output of the compiled function.

Post-Training Quantization. We implement a mixed-precision strategy using the NVIDIA Model Optimizer (NVIDIA Corporation, 2024) on Blackwell (SM100) architecture. We quantize model weights and activations to NVFP4 (E2M1) while maintaining sensitive QKV projections and Softmax operations in FP8 (E4M3). To preserve numerical stability, we employ FP16 accumulation for non-linear operations including LayerNorm and RoPE. This configuration improves latency with negligible impact on generated video and action quality.

Kernel-Level Enhancements. We use the cuDNN backend for dot-product attention via PyTorch Scaled Dot-Product Attention (SDPA), requiring PyTorch version $\geq$2.9. Earlier versions of the Transformer Engine library may also access these efficient cuDNN kernels.

Scheduler Optimizations. The initial Flow UniPC scheduler (Zhao et al., 2023) implementation required CPU execution for several operations, causing frequent CPU–GPU synchronization and GPU stalls. We migrated these operations to GPU, eliminating unnecessary CPU overhead.

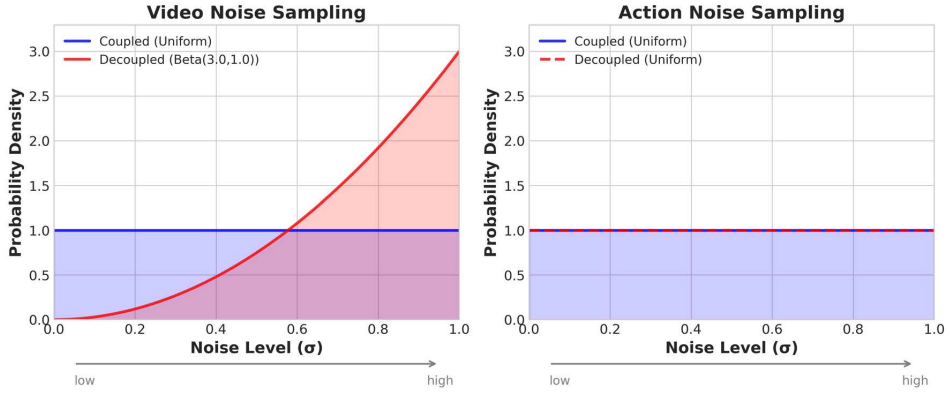


Figure 13: **Decoupled Noise Schedules.** DREAMZERO uses coupled noise for video and action (both uniform). DREAMZERO-Flash biases video toward high-noise states via a Beta distribution while keeping action noise uniform, training the model to predict clean actions from noisy visual context.

A.5.5 MODEL-LEVEL OPTIMIZATIONS — DREAMZERO-FLASH

In the standard DREAMZERO formulation, video and action modalities share the same denoising timestep t_k:

$$t_k^{\text{video}} = t_k^{\text{action}} = t_k, \quad t_k \sim \mathcal{U}(0,1) \tag{4}$$

DREAMZERO-Flash decouples these schedules by biasing video timesteps toward lower values (higher noise) while keeping action timesteps uniform:

$$t_k^{\text{video}} = 1 - \eta, \quad \eta \sim \text{Beta}(\alpha,\beta), \quad t_k^{\text{action}} \sim \mathcal{U}(0,1) \tag{5}$$

where $\alpha > \beta$ (e.g., $\alpha = 3, \beta = 1$). Since $\text{Beta}(\alpha,\beta)$ with $\alpha > \beta$ concentrates mass near $\eta \approx 1$, the transformed variable $t_k^{\text{video}} = 1 - \eta$ is biased toward 0, corresponding to high-noise video states. The noisy samples become:

$$\mathbf{z}_t^k = t_k^{\text{video}}\mathbf{z}_1^k + (1 - t_k^{\text{video}})\mathbf{z}_0^k, \quad \mathbf{a}_t^k = t_k^{\text{action}}\mathbf{a}_1^k + (1 - t_k^{\text{action}})\mathbf{a}_0^k \tag{6}$$

For $\text{Beta}(3,1)$, $\mathbb{E}[\eta] = 0.75$, yielding $\mathbb{E}[t_k^{\text{video}}] = 0.25$ compared to 0.5 in the coupled setting. This exposes the model during training to configurations where actions must be predicted from predominantly noisy visual context (Fig. 13), aligning training with rapid-action-denoising inference where actions denoise from noise level $1 \to 0$ in one step while video remains partially noisy.

Action Chunk Smoothing. Generated action chunks may contain high-frequency noise from denoising. We apply filtering to ensure stable real-world behavior: first upsampling the action chunk to $2\times$ resolution via cubic interpolation, then applying a Savitzky-Golay filter (window size 21, polynomial order 3) to suppress noise while preserving trajectory shape, and finally downsampling to original resolution.

A.5.6 SUMMARY

Table 4 summarizes cumulative speedups. System and implementation optimizations yield $\sim$10$\times$ speedup on H100 and $\sim$17$\times$ on GB200; adding DREAMZERO-Flash achieves 41.5$\times$ on GB200, reducing latency from 6.2s to 150ms. With the exception of DiT caching and quantization, all optimizations are mathematically equivalent to baseline and show no measurable performance degradation.

Optimization	**H100**	**GB200**
Baseline	1$\times$	1.1$\times$
System-level		
+ CFG Parallelism	1.9$\times$	1.8$\times$
+ DiT Caching	5.5$\times$	5.4$\times$
Implementation-level		
+ Torch Compile + CUDA Graphs	8.9$\times$	10.9$\times$
+ Kernel & Scheduler Opts.	9.6$\times$	14.8$\times$
+ Quantization (NVFP4)	—	16.6$\times$
Model-level		
+ DREAMZERO-Flash	—	38$\times$

Table 4: **Cumulative inference speedups.** Each row includes all optimizations above it. Entries marked "—" indicate features not applicable to that hardware.

A.6 EXPERIMENTAL SETUP DETAILS

We validate our hypotheses on two robot embodiments: the AgiBot G1 mobile bimanual manipulator and the Franka single-arm robot. We apply separate pretraining for each embodiment, leaving cross-embodiment transfer for future work. The experimental setup for AgiBot G1 is illustrated in Figure 14.

We compare against two state-of-the-art VLAs: GR00T N1.6 (Bjorck et al., 2025) and $\pi_{0.5}$ (Physical Intelligence, 2025). For each baseline, we evaluate two initialization strategies: (1) *from-scratch*, using pretrained VLM weights without prior robot data training, and (2) *from-pretrained*, using official checkpoints pretrained on thousands of hours of cross-embodiment robot data. Both variants are then trained on identical data as DREAMZERO: $\sim$500 hours of teleoperation data we collected for AgiBot G1, and DROID (Khazatsky et al., 2024) for Franka. We keep the compute budget comparable across all methods by matching total batch size and gradient steps.

A.6.1 PRETRAINING

Data. Our data collection philosophy differs from that of existing Vision-Language-Action (VLA) models. While recent works have shown that VLAs can learn effective policies from moderate-sized datasets, these approaches typically still rely on structured, task-focused demonstrations to ensure consistent behavior. We hypothesize that learning to only predict actions without encoding the knowledge about future world states makes it challenging to leverage highly heterogeneous, non-repetitive data effectively, as the model must implicitly infer dynamics from noisy state-action pairs. In contrast, we hypothesize DREAMZERO's world modeling objective enables effective learning from diverse demonstrations, allowing us to prioritize breadth and utility over repetition during data collection.

Using AgiBot G1, we collect approximately 500 hours of teleoperation data across 22 unique environments—including homes, restaurants, supermarkets, coffee shops, and offices—prioritizing task diversity and real-world utility over task-specific repetition. As shown in Figure 15, each episode averages around 4 minutes and encompasses approximately 42 subtasks—significantly longer-horizon than typical robotic manipulation datasets (Khazatsky et al., 2024; Walke et al., 2023). The skill distribution reflects real-world deployment requirements: navigation enables movement between

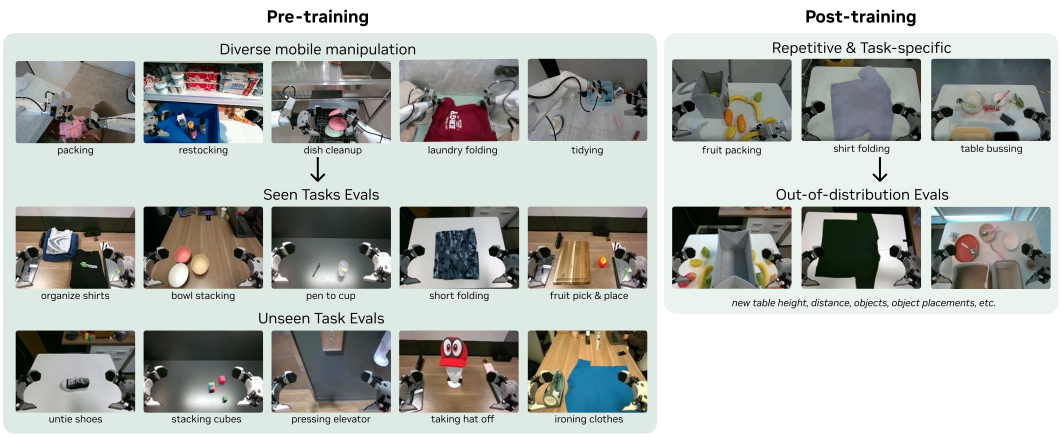
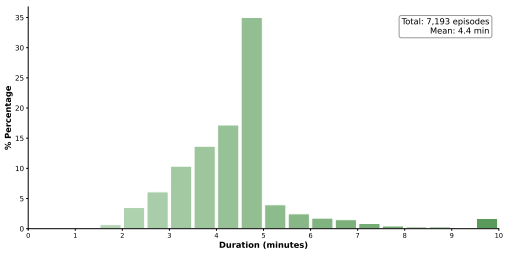
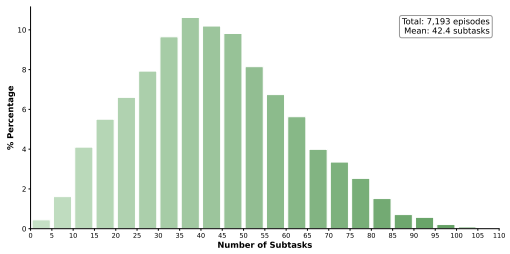


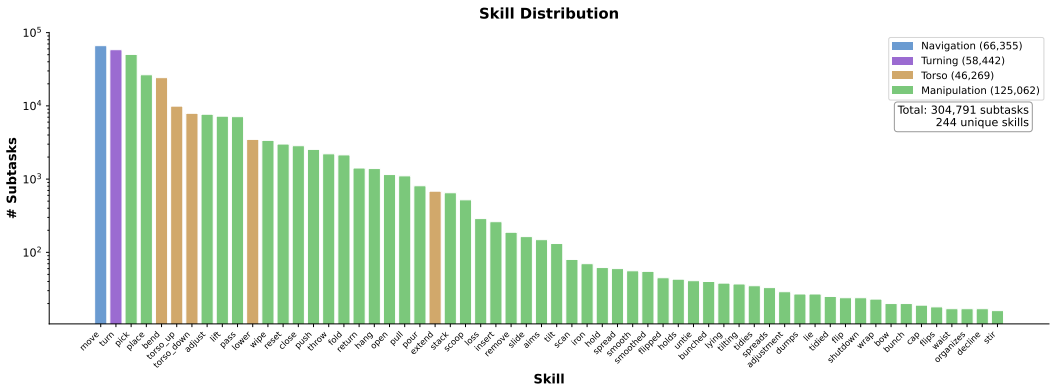
Figure 14: **Evaluation Set-up.** We are first-citizens of generalization evals, where the default setting is *unseen* environment and objects for both pre-training and post-training evaluations.



(a) Episode Duration Distribution



(b) Subtask Count Distribution per Episode



(c) Skill Distribution

Figure 15: **Distribution statistics** for the AgiBot pretraining corpus: episode durations, subtask density, and skill coverage across 7.2K episodes (∼500 hours)

.

workspaces, while torso adjustments allow interaction with objects at varying heights (shelves, cabinets). Additional details on the data collection pipeline are provided in Appendix A.7.

We also validate DreamZero on the Franka single-arm robot using DROID (Khazatsky et al., 2024), one of the most heterogeneous publicly available robotic datasets, achieving 1st place on the RoboArena Leaderboard (Atreya et al., 2025)—a public distributed benchmark for evaluating DROID-trained policies. This demonstrates the effectiveness of WAMs on diverse, open-source

data and enables reproducibility prior to the release of our in-house AgiBot dataset. We open-source the checkpoint and inference code to run some DROID-sim evals in PolaRiS (Jain et al., 2025).

Training. We use Wan2.1-I2V-14B-480P (Team Wan, 2025), a 14B image-to-video diffusion model, as the backbone for DREAMZERO. We train for 100K gradient steps with a global batch size of 128 for AgiBot and 75K gradient steps with a global batch size of 128 for DROID datasets. We update all DiT blocks, the state encoder, action encoder, and action decoder, while freezing the text encoder, image encoder, and VAE. For both datasets, we filter out idle actions and use relative joint positions as the default action representation. We also have some ablation results (Section 5.2) where we initialize from Wan2.1-I2V-5B-480P to see the effect of model size (5B vs. 14B).

Evaluation Protocol. We evaluate models out of the box after pretraining. Our default evaluation setting is *unseen environments, unseen objects*—because our pretraining and post-training data were collected in a different geographic location from our evaluation sites, every benchmark inherently tests out-of-distribution generalization rather than interpolation within the training distribution. We evaluate on two categories: *seen* and *unseen* tasks. We define the granularity of a task as a combination of the motion required for the task and the object type. For example, if the pretrained data contains folding a red-colored shirt and evaluate the model to fold a black-colored shirt with a different size, it is considered as a *seen* task. On the other hand, if we evaluate the model to fold socks, it is considered as an unseen task because the motion required to fold socks is different from folding a shirt.

AgiBot Evaluation Protocol. For *seen* tasks, we select 10 tasks from the pretraining distribution, including pick-and-place variants, stacking, wiping, and folding; we run 8 rollouts per task across 4 robots, each in different environments and different objects (80 rollouts total per checkpoint). We divide 10 seen tasks into three categories: PnP-Easy (Pick and place fruit, Wipe the mess, Take out fruit from bag), PnP-Hard (Pick and place fork/spoon, put the pen in pen holder, put the cup on the coaster, stack bowls/cups in a row), and Contact-Rich Manipulation (fold shirts, fold shorts, stack clothes). For *unseen* tasks, we evaluate 10 tasks absent from training—such as ironing, painting, pulling carts, cube stacking, removing a hat from a mannequin, and untying shoe laces—with 8 rollouts per task across 4 robots (80 rollouts total per checkpoint).

We provide the evaluation setup for both seen and unseen tasks in Tables 5 and 6 for Agibot setup. Each row shows the initial frame and instruction for 4 robots. We conduct 2 rollouts per robot for each task by varying the objects and locations.

DROID Evaluation Protocol. We evaluate on 20 seen tasks and 20 *unseen tasks* (verbs and objects absent from DROID), performing 2 rollouts per task, for a total of 80 evaluation rollouts across 40 tasks for each checkpoint. We compare DREAMZERO against the publicly released $\pi_{0.5}$-DROID and an internally trained GR00T N1.6-DROID checkpoint. Object positions are fixed across checkpoints to ensure fairness. Each rollout is scored from 0 to 1.0 based on partial task completion. We provide the evaluation setup for both seen and unseen tasks in Tables 7a and 7b for DROID.

A.6.2 POST-TRAINING

Beyond pre-training, we evaluate whether WAMs improve fine-tuning performance on task-specific data using the AgiBot robot.

Data. We collect post-training data on three downstream tasks with varying levels of distribution diversity:

- *Shirt folding* (33 hrs): Fold a flattened t-shirt through 5 sequential stages. We randomize initial shirt position across 2 shirt types. **Lowest diversity.**
- *Fruit packing* (12 hrs): Pack 10 fruits from a table into a bag. We randomize fruit combinations and positions of fruits and bag. **Medium diversity.**
- *Table bussing* (40 hrs): Clear 5 pieces of trash into a trash bin and 5 pieces of dishware (dish, bowl, fork, and spoon) into a dish bin. We randomize object types, combinations, and positions. **Highest diversity.**

Training. We post-train for 50K gradient steps per task. As in pretraining, we update all parameters except the text encoder, image encoder, and VAE.

Table 5: Seen Tasks Evaluation Setup

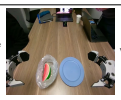
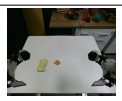
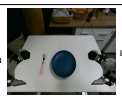
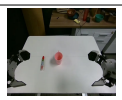
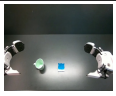
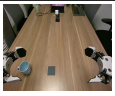
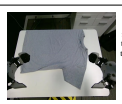
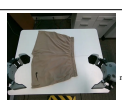
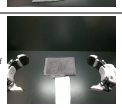
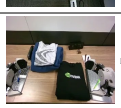
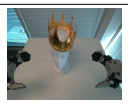
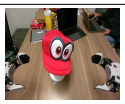
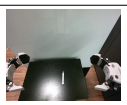
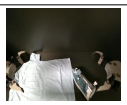
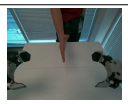
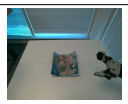
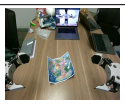
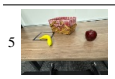
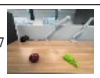
Category	#	Task	Image	Instruction	Image	Instruction	Image	Instruction	Image	Instruction
PnP Easy	1	PnP Fruit		The left arm picks up the Banana on the table and places it in the Blue Plate.		The left arm picks up the lime on the table and places it on the light green plate.		The left arm picks up the peach on the table and places it on the baking pan.		The left arm picks up the green pear on the table and places it on the blue checkered bowl.
	2	Taking out Fruit		The left arm picks up the Mango from the plastic bag and places it into the Wooden Basket		The left arm picks up the yellow pear from the plastic bag and places it onto the blue tray.		The left arm picks up the purple grapes from the plastic bag and places it into the brown basket.		The left arm picks up the watermelon from the plastic bag and places it onto the Blue Plate.
	3	Wipe the Mess		The left arm uses a sponge to wipe the coffee spill off the table.		The left arm used a black cloth to wipe the white powder off the table.		The left arm uses a napkin to wipe the creamer spill off the table.		The left arm used a paper towel to wipe the water off the table.
PnP Hard	4	PnP Fork/Spoon		The left arm picks up the Pink Fork from the table and places it onto the blue plate.		The left arm picks up the orange fork from the table and places it into the glass cup		The left arm picks up the light blue fork from the table and places it onto the orange plate		The left arm picks up the green fork from the table and places it into the blue plate.
	5	Put Pen in Holder		The left arm picks up the Red Marker pen from the table and placed it into the pen holder.		The left arm picked up the black marker from the table and placed it into the pen holder.		The left arm picked up the white marker pen from the table and placed it into the pen holder.		The left arm picked up the mechanical pencil from the table and placed it into the pen holder.
	6	Put Cup on Coaster		The left arm picks up the clear cup from the table and places it on the grey coaster.		The left arm picks up the plastic cup from the table and places it on the blue coaster.		The left arm picks up the plastic cup from the table and places it on the white coaster.		The left arm picks up the pink cup from the table and places it on the gray coaster.
	7	Stack Bowls/Cups		The robot reaches to grip the green bowl, moves it to the middle wooden bowl, and releases it to stack. It then reaches to grip the white bowl, moves it to the same location, and releases it onto the stack.		The robot reaches its left arm to grip the blue plate, moves it to the middle light green plate, and releases it to stack. It then reaches its right arm to grip the red plate, moves it over the middle stack of plates, and releases it to finish the task.		The robot reaches its left arm to grip the paper bowl, moves it over the middle white bowl, and releases it to stack. It then reaches its left arm to grip the blue checkered bowl, moves it over the stack, and releases it to finish the task.		The robot reaches its left arm to grip the pink bowl, moves it over the middle beige bowl, and releases it to stack. It then reaches its left arm to grip the white bowl, moves it over the stack, and releases it onto the stack.
Contact Rich	8	Folding Shirts		Both arms grip the bottom of the light grey short sleeve and fold it toward the middle. They then pull the short sleeve across the table to the edge. Next, both arms grasp the top of the shirt and fold it down to the middle. Finally, the right arm grips the collar and folds it down to complete the task.		Both arms fold the bottom of the green short sleeve to the middle. They then pull the shirt toward the edge of the table. Next, both arms fold the top of the shirt down to the middle. Finally, the right arm grasps the collar and folds it down to complete the task.		Both arms fold the bottom of the logo short sleeve to the middle. They then pull the shirt toward the edge of the table. Next, both arms fold the top of the shirt down to the middle. Finally, the right arm grasps the collar and folds it down to complete the task.		Both arms fold the bottom of the gray short sleeve to the middle. They then pull the shirt toward the edge of the table. Next, both arms fold the top of the shirt down to the middle. Finally, the left arm grasps the collar and folds it down to finish.
	9	Folding Shorts		Both arms fold the bottom of the tan shorts toward the middle. Then, the right arm folds the shorts in half from right to left to complete the fold.		Both arms fold the bottom of the grey shorts toward the middle. Then, the right arm folds the shorts in half from right to left to complete the fold.		Both arms fold the bottom of the white shorts toward the middle. Then, the right arm folds the shorts in half from right to left to complete the task.		Both arms fold the bottom of the green shorts toward the middle. Then, the right arm folds the shorts in half from right to left to complete the fold.
	10	Stacking Clothes		Both arms pick up the black shirt and place it on the stack of clothes.		Both arms pick up the white shirt and place it on the gray towel.		Both arms pick up the black hoodie and place it on the stack of clothes.		Both arms pick up the dark gray shirt and place it on the stack of clothes.

Table 6: Unseen Tasks Evaluation Setup

#	Task	Image	Instruction	Image	Instruction	Image	Instruction	Image	Instruction
1	**Untie Shoelaces**		The robot coordinates both arms to simultaneously grasp the two loops of the shoelace. It then moves both arms outward in synchronized, opposing directions until the shoelace is fully untied.		The robot reaches its left arm to grasp the blue box and hold it steady. It then reaches its right arm toward the blue ribbon and moves it to untie the knot.		The robot coordinates both arms to simultaneously grasp the two loops of the knot of the box. It then moves both arms outward in synchronized, opposing directions until the knot of the box is fully untied.		The robot coordinates both arms to simultaneously grasp the two loops of the knot of the package. It then moves both arms outward in synchronized, opposing directions until the knot of the package is fully untied.
2	**Remove /Put Hat**		The robot reaches its right arm to gasp the crown and lifts it to remove it from the mannequin's head.		The robot reaches its left arm to grasp the hat and lifts it to remove it from the mannequin's head.		The robot reaches its left arm to grasp the hat and lifts it to remove it from the mannequin's head.		The robot reaches its left arm to grasp the hat and lifts it to remove it from the mannequin's head.
3	**Draw Circle**		The robot reaches its left arm to pick up the red marker and the left arm moves the marker to draw a circle on the book.		The robot reaches its right arm to pick up the marker and the right arm draws a circle on the whiteboard with the marker.		The robot reaches its left arm to pick up the black marker and the left arm moves the marker to draw a circle on the paper		The robot reaches its right arm to pick up the marker and the right arm moves the marker to draw a line on the whiteboard.
4	**Take out Straw**		The left arm holds the cup on the table. Then the right arm pulls the straw out of the cup.		The left arm holds the cup on the table. Then the right arm pulls the straw out of the cup.		The left arm holds the cup on the table. Then the right arm pulls the straw out of the cup.		The right arm holds the cup on the table.Then the left arm pulls the straw out of the cup.
5	**Cube Stacking**		The robot reaches its right arm to pick up the green cube, moves it over the red cube, and releases it to stack. It then reaches its right arm to pick up the yellow cube, moves it over the stack, and releases it onto the green cube to finish the task.		The robot reaches its right arm to pick up the red cube, moves it over the colorful cube, and releases it to stack. It then reaches its right arm to pick up the green cube, moves it over the red cube, and releases it to finish the tower.		The robot reaches its right arm to pick up the white cube, moves it over the green cube, and releases it to stack. It then reaches its left arm to pick up the blue cube, moves it over the stack, and releases it onto the white cube to finish the task.		The robot reaches its right arm to pick up the red cube, moves it over the blue cube, and releases it to begin the stack. It then reaches its left arm to pick up the orange cube, moves it over the stack, and releases it onto the red cube to complete the three-tier structure.
6	**Painting**		The left arm grabs the brush. Then left arm paints with the brush on the notebook		The right arm grabs the brush. Then right arm paints with the brush on the paper		The left arm grabs the brush. Then left arm paints with the brush on the notebook		The right arm grabs the brush. Then right arm paints with the brush on the notebook
7	**Ironing**		The robot reaches its left arm to grasp the iron and moves it across the shorts to iron it.		The robot reaches its right arm to grasp the iron and moves it across the shirt to iron it.		The robot reaches its left arm to grasp the iron and moves it across the shirt to iron it.		The robot reaches its left arm to grasp the iron and moves it across the shirt to iron it.
8	**Shake Hands**		The right arm of the robot grasp the human hand to shake hands. It then initiates a rhythmic up-and-down motion to perform the handshake.		The left arm shakes the hand of the human up and down		The right arm shakes the hand of the human up and down		The left arm of the robot grasp the human hand to shake hands. It then initiates a rhythmic up-and-down motion to perform the handshake.
9	**Folding (Map)**		The left arm grabs the left side of the map. The right arm folds the right side of the map. The left arm folds the left side of the map.		The left arm grabs the left side of the map. The right arm folds the right side of the map. The left arm folds the left side of the map.		The right arm grabs the right side of the map. The left arm folds the left side of the map. The right arm folds the right side of the map.		The right arm grabs the right side of the map.The left arm folds the left side of the map. The right arm folds the right side of the map.
10	**Pulling Cart**		The robot reaches its right arm to grasp the cart and pulls it.		The robot reaches its left arm to grasp the cart and pulls it.		The robot reaches its right arm to grasp the cart and pulls it.		The robot reaches its left arm to grasp the cart and pulls it forward.

#	Image	Instruction	#	Image	Instruction
1		Move the cup forward then put the marker inside the cup	11		Move the bowl on the left to the right side of the table.
2		Put the marker in the blue box	12		Pick up the pencil and put it on the bowl
3		Remove the pair of gloves from the open drawer and put it on the table	13		Pick the marker up from the table and put it in the bowl
4		Put the marker on table	14		Place the bowl next to the marker
5		Pick up the apple and put it in the basket	15		Remove a lemon from the bowl
6		Put the towel on the white cup	16		Move the grapes to the left
7		Put the towel in the pan	17		Move the green grapes backwards
8		Put the hat on the table	18		Put the bread inside the toaster
9		Put the pair of scissors into the drawer	19		Push the lever on the bread toaster downwards
10		Put the towel in the basket	20		Pick up the cup from the bowl and put it in the other cups

(a) Seen tasks on DROID

#	Image	Instruction	#	Image	Instruction
1		**Orient** the mug so the handle is to the right	11		**Hook** the hat onto the tripod
2		**Fan** the burger	12		**Pinch** the binder clip to release the papers
3		**Slice** the bread with the knife	13		**Withdraw** the bread from the toaster and place on the plate
4		**Type** 'hi' on the keyboard	14		**Cinch** the drawstring of the bag
5		**Extricate** the straw from the cup	15		**Dispense** the mustard onto the bread
6		**Reveal** the object under the cup	16		**Bake** the croissant in the oven
7		**Match** the objects to their corresponding bowl	17		**Fry** the vegetables in the pan with the spatula
8		**Maneuver** the blocks through the matching hole	18		**Depress** the lever on the toaster
9		**Affix** the magnet to the tray	19		**Elevate** the yellow block to the highest platform
10		**Combine** the nuts and batteries	20		**Weave** the wire through the holes of the box

(b) Tasks with unseen verbs on DROID

Evaluation Protocol. We measure average task progress across 10 rollouts per task, with a 120-second time limit. Task progress is defined as: (1) folding stages completed out of 5 for shirt folding, (2) fruits successfully packed out of 10 for fruit packing and (3) items cleared for table bussing. Following Barreiros et al. (2025), we apply an image overlay to the initial scene to reduce variance.

A.7 AgiBot Diverse Data Collection Strategy

Our data collection philosophy prioritizes *diversity over repetition*. Unlike conventional approaches that collect hundreds of demonstrations per task in controlled lab settings, we collect data across 22 real-world environments spanning homes, restaurants, supermarkets, coffee shops, offices, warehouses, laboratories, and hotels (Figure 16).

A.7.1 Daily Collection Workflow

Each day, teleoperators receive a printed task sheet listing available tasks for their assigned area (e.g., kitchen area, living room area, checkout counter). For each episode, they select three tasks from this sheet and execute them consecutively. Each task typically requires 1–2 minutes, resulting in approximately 5-minute episodes.

At the end of each day, teleoperators log the frequency count for each task. Once a task reaches 50 episodes, it is deprecated and removed from the task sheet. Teleoperators are incentivized to propose new tasks, which they inevitably must do as existing tasks become deprecated. This mechanism continuously expands the task distribution throughout data collection, yielding a long-tail of diverse behaviors.

Because we prioritize utility over repetition, our tasks are naturally more coarse-grained than typical robot learning datasets. Examples include organizing items, ground garbage cleaning, shopping

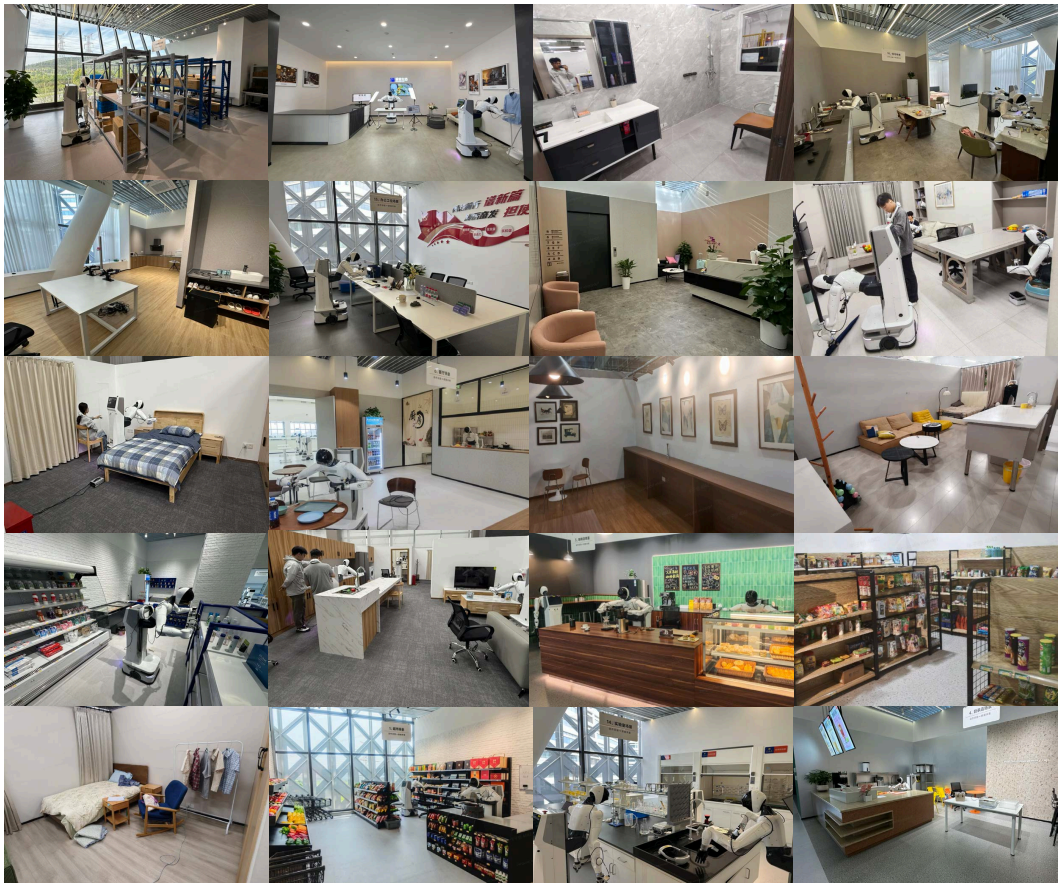


Figure 16: **Data Collection Environments.** We collect teleoperation data across 22 diverse real-world environments, including offices, laboratories, restaurants, supermarkets, coffee shops, warehouses, homes, hotels, and retail stores. This diversity enables DREAMZERO to generalize to unseen environments without task-specific fine-tuning.

basket return, toy box tidying, table tidying, and clothes hanging—but the full set grows organically as the collection progresses.

A.7.2 MULTI-TASK EPISODE STRUCTURE

The three-task episode structure serves two purposes: it maximizes diversity within each episode and encourages the model to learn smooth task transitions. For example, a single episode might involve (1) clearing dishes from a table, (2) wiping the table surface, and (3) organizing condiments. This design yields an average of 42 subtasks per episode (Figure 15), significantly more than typical single-task datasets.

This combination of environment diversity, task deprecation with forced expansion, and multi-task episodes produces a heterogeneous dataset that differs substantially from conventional robot learning corpora. Rather than learning narrow task-specific policies, DREAMZERO learns generalizable skills that transfer across environments and tasks.

A.8 FAILURE CASE ANALYSIS

In Figure 17, we illustrate the generated video by DREAMZERO and execution rollout for both AgiBot and DROID. Overall, the robot execution follows the visual plan generated on the video modality side. For AgiBot video generated by DREAMZERO, the robot picks up the marker with left arm and passes the marker to the right arm. Consistent with the generated video, for the execution

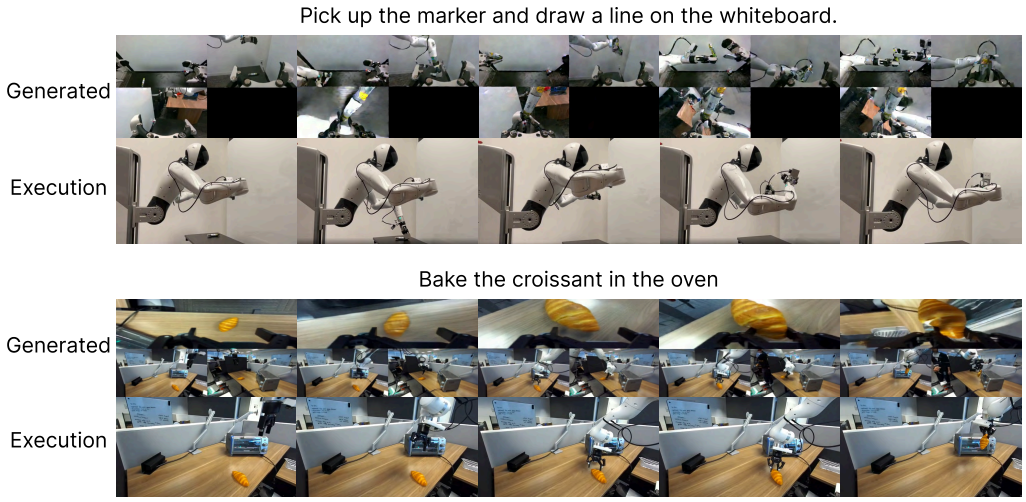


Figure 17: **Illustration of generated and executed pair.** We illustrate the generated video and action execution pair.

rollout, the robot picks up the top part of the marker, but instead of drawing a line on the whiteboard, the left arm passes the marker to the right arm. For DROID video generated by DREAMZERO, the robot picks up the bread instead of opening the oven first. Aligned with the generated video, for the execution rollout, the robot picks up the bread first instead of opening the oven, leading to the rollout being stuck after the robot has reached to the oven with bread held. This implies that improving the language following capability of WAMs could potentially lead to better action execution.

A.9 RELATED WORK

A.9.1 VISION LANGUAGE ACTION MODELS

Utilizing Foundation Models for Robotics. Developing foundation models (Bommasani et al., 2021) for physical artificial intelligence has emerged as a significant research frontier. One line of work involves using existing, pre-trained foundation models as "black-box" reasoners to handle high-level task planning. These works usually involve *modular* systems, where the foundation models generate sequences of instructions, visual traces, or affordances that are subsequently executed by specialized, low-level robotic policies or controllers (Brohan et al., 2023b; Huang et al.; Singh et al., 2023; Driess et al., 2023; Huang et al., 2023; Kumar et al., 2024). While this modularity simplifies complex planning and enables stronger generalization (Li et al., 2025b; Lee et al., 2025) and efficiency (Dreczkowski et al., 2025), it is contingent upon having a pre-existing library of low-level skills and a robust interface to bridge the gap between abstract reasoning and physical execution. Additionally, these decoupled systems face the risk of compounding errors across modules.

VLAs. On the other hand, end-to-end models such as Vision-Language-Action models (VLAs) (Brohan et al., 2022; 2023a; Kim et al., 2024a; Zheng et al., 2025a; Ye et al., 2025; Yang et al., 2025; Black et al., 2024a; Bjorck et al., 2025; Physical Intelligence, 2025; Gemini Robotics Team, 2025; Bu et al., 2025), have gained popularity by moving away from a rigid hierarchy of planning and control, combining language-conditioned semantics and low-level robot actions within the same model. VLAs are often initialized from large vision-language (VLM) models pre-trained on web-scale datasets. While pushing the frontier on visual-semantic knowledge transfer, these models are pre-trained on *static* image-text datasets, which limits their ability to inherit spatiotemporal priors required to transfer knowledge to new physical skills.

Generalization using VLAs. Generalization in VLAs has been mostly demonstrated on object and semantic level (Brohan et al., 2023a; Gao et al., 2025) while generalization to completely new skills and environments has remained limited. In particular, existing work utilizing VLAs achieves environment generalization by collecting human teleoperation data across hundreds of diverse envi-

ronments for specific tasks (Physical Intelligence, 2025). Furthermore, while current VLAs attempt to achieve task generalization by covering a large library of language-conditioned motion primitives (Gemini Robotics Team, 2025), this approach is fundamentally constrained by the impracticality of capturing the vast amount of possible physical interactions and motions with a fixed set of episode-level language-conditioned tasks. In contrast, video-based world models learn from every consecutive frame pair in the data, while also leveraging large-scale video pretraining to understand physical dynamics.

A.9.2 Video Model-based Robot Policies

Video Generation in Robotics. Prior works show that video generation models can be used to synthesize robot trajectories and extract executable actions at test-time through various approaches: inverse-dynamics models (Du et al., 2023; Zhou et al., 2024), optical flow as dense correspondence (Ko et al., 2024), or trajectory prediction as high-level planning (Yang et al., 2024; Du et al., 2024). Other works generate human videos—either with 3D tracking (Liang et al., 2024) or for novel scenes and motions (Bharadhwaj et al., 2024; Chen et al., 2025)—and train policies using point tracking objectives. Most recently, (Jang et al., 2025; Luo et al., 2025) demonstrated that video generation models can produce synthetic robot data for unseen behaviors in novel environments, leveraging the strong generalization capabilities of these models.

Joint Video and Action Generation. Another line of work couples video and action generation for end-to-end learning. These methods demonstrate that incorporating a world modeling objective alongside action prediction improves multi-task performance, sample efficiency, and generalization to novel scenes and objects. Previous work (Wu et al., 2024; Cheang et al., 2024; Li et al., 2025a; Zhu et al., 2025; Zhao et al., 2025; Zheng et al., 2025b; Won et al., 2025) learns to do joint world modeling and action prediction from scratch or from VLAs, while more recent work (Kim et al., 2026; Liao et al., 2025; Hu et al., 2024; Liang et al., 2025; Pai et al., 2025) leverages pretrained video diffusion models to inherit rich visual dynamics priors. We refer to these models collectively as *World Action Models (WAMs)* since they leverage world modeling capability (predicting the future state) for action prediction. In contrast to prior WAMs, DreamZero systematically explores data diversity and scale to expose the full generalization potential of WAMs, adopts an autoregressive architecture better suited for long-horizon world–action modeling, and achieves state-of-the-art generalization across both novel tasks and environments.

Why WAMs. WAMs built upon video diffusion backbones inherit rich spatiotemporal priors from web-scale data, capturing the best of both paradigms: the seamless gradient flow of end-to-end VLAs and dense world modeling supervision for planning. Central to this approach is learning the joint distribution of video and action—DreamZero simultaneously learns both modalities, with video prediction serving as an implicit visual planner that guides action generation. This formulation not only means that improving robotic capabilities reduces to improving video generation, but also enables two capabilities that elude current VLAs: zero-shot generalization to novel tasks and effective learning from heterogeneous robot data.