Awomo-0.5 Technical Report

Awomo-WAM Team

See Contributions for the full author list.

https://github.com/Awomo-WestlakeDI/Awomo-0.5

https://huggingface.co/collections/Auwomo/awomo-05

Abstract

We introduce Awomo-0.5, a world action modeling framework that adapts pretrained visual generators to robot control. Our primary model, **Awomo-0.5I**, represents robot actions as image-like patches and jointly generates a future image and an action chunk, reusing the image transformer's input and output patch projections. We also study **Awomo-0.5V**, a video-based variant that jointly generates future video and actions through a learned action interface. Both models are pretrained on over 36 000 hours of real-robot trajectories, simulated demonstrations, public and in-house egocentric video, and UMI recordings, and our training data is organized around atomic skill families rather than end-to-end tasks. A unified action representation and source-specific validity masks enable joint training across embodiments while retaining visual supervision from data without action labels. We further introduce the Real-World Atomic Skill Benchmark on Piper, designed to assess transfer from atomic-task supervision to unseen conditions and held-out skill compositions. Evaluation spans five simulation benchmarks and this real-world benchmark. **Awomo-0.5I** achieves a 95.9% category mean on LIBERO-Plus, 64.4% overall success on RoboCasa365, and 79.7% overall success across 69 real-world evaluation entries. With Piper adaptation using only atomic-task demonstrations, it succeeds on five of six held-out compositional entries (83.3%). These results support direct reuse of image-generation interfaces for robot action learning and demonstrate skill composition in the evaluated real-world tasks.

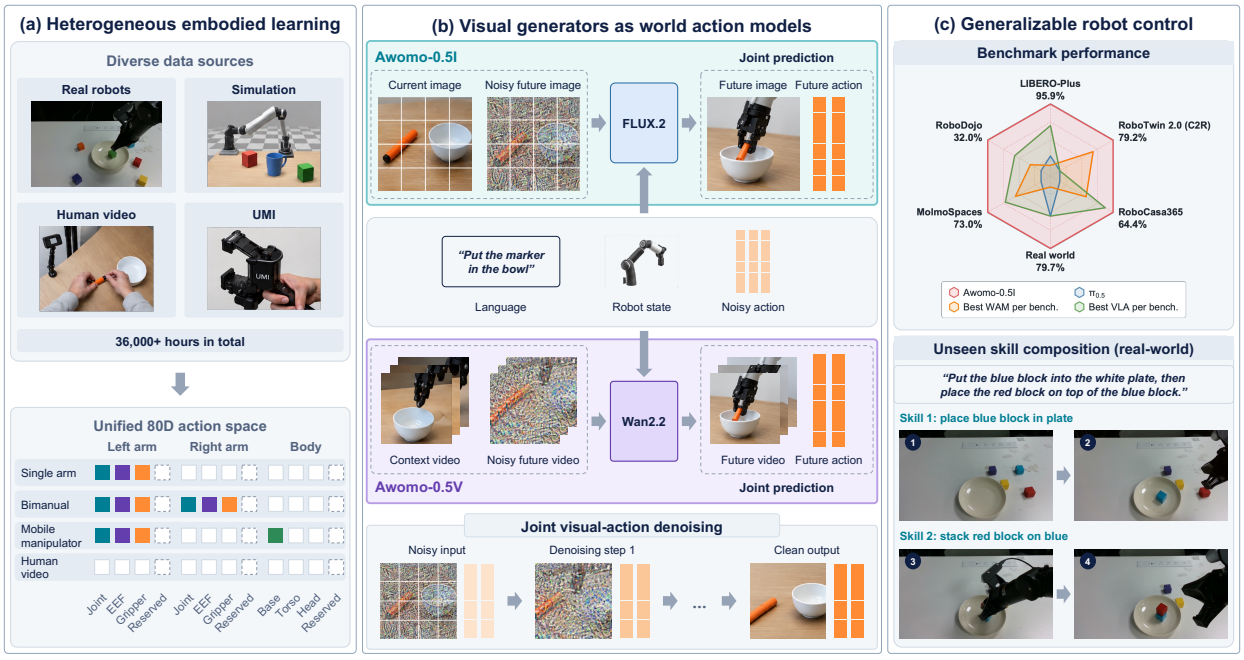


Figure 1. Overview of Awomo-0.5. We pretrain world action models on heterogeneous robot and human data in a unified action space; the resulting Awomo-0.5I performs strongly across simulation and real-world benchmarks.

Contents

1 Introduction . . . 4
2 Architecture . . . 5
2.1 Image-based action generation . . . 6
2.2 Video-based action generation . . . 6
2.3 Attention masks . . . 6
2.4 Flow-matching objective . . . 7
3 Pretraining Data . . . 8
3.1 Data Composition . . . 8
3.1.1 Real Robot Trajectories . . . 8
3.1.2 Simulated Robot Trajectories . . . 8
3.1.3 UMI Human Demonstrations . . . 10
3.1.4 Egocentric Human Videos . . . 10
3.2 Data Processing . . . 11
3.2.1 Unified Action Space . . . 11
3.2.2 Embodied Prompt . . . 12
3.2.3 Data Cleaning Pipeline . . . 12
3.3 Data Organization with Atomic Skills . . . 14
3.3.1 Design principles. . . . 14
3.3.2 Atomic skill annotation . . . 15
4 Pretraining Analysis . . . 15
4.1 Atomic Skill Analysis . . . 15
4.2 Effect of Pretraining . . . 17
4.3 Human Data Ablations . . . 18
5 Simulation Post-Training . . . 18
5.1 Simulation Results . . . 18
5.1.1 RoboDojo . . . 20
5.1.2 RoboCasa365 . . . 20
5.1.3 LIBERO-Plus . . . 20
5.1.4 MolmoSpaces . . . 21
5.1.5 RoboTwin 2.0 . . . 22
5.2 Open-loop Error and Closed-loop Success . . . 23
5.3 Action-Space Ablations . . . 25
6 Real-World Post-Training . . . 25
6.1 Atomic Skill Benchmark . . . 25
6.2 Experiment Setup . . . 27
6.3 Real-World Results . . . 27
6.3.1 Seen Conditions Evaluation . . . 27
6.3.2 Unseen Conditions Evaluation . . . 28
6.3.3 Compositional Tasks Generalization . . . 29
7 Related Work . . . 31
8 Limitations . . . 31
9 Conclusion . . . 31
10 Contributions . . . 32
A Pretraining Data Details . . . 41
A.1 UMI Action Solving . . . 41
A.2 UMI Retargeting . . . 41
A.3 Signal Thresholds . . . 41
B Simulation Post-Training Details . . . 42
B.1 RoboDojo . . . 42
B.2 RoboCasa365 . . . 42
B.3 LIBERO-Plus . . . 42
B.4 MolmoSpaces . . . 43
B.5 RoboTwin 2.0 . . . 43
B.6 Inference Configuration . . . 43

C Real-World Post-Training Details 44
C.1 Training 44
C.2 Inference 44

1 Introduction

Robot policies must follow language instructions while adapting to changes in objects, scenes, viewpoints, and task requirements. Large vision-language-action models transfer pretrained visual and semantic representations into control [1-3]. World action models draw on pretrained visual generators [4-6] to learn actions jointly with their visual consequences. Adapting these generators to robot control requires an interface for representing and predicting robot actions within the visual backbone. A central design question is how directly action generation can reuse the pretrained components used for visual generation.

We present Awomo-0.5 Fig. 1, centered on **Awomo-0.5I**, an action-as-patch world action model that directly reuses an image transformer's input and output projections for robot action generation. Each action vector is represented as an image-like patch, allowing visual and action tokens to pass through the same patch projections and transformer. Conditioned on language, the current observation, and robot state, the model jointly denoises a future image and an action chunk. We also study **Awomo-0.5V**, a video-based variant that uses learned action encoding and decoding layers within a shared video-action transformer. This variant provides an additional architecture for examining the transfer of visual generation to control.

Real-world manipulation tasks are not monolithic; they are sequences of reusable physical primitives. We therefore organize the pretraining corpus around atomic skill families rather than end-to-end task labels. Each family has an unambiguous closed-loop success criterion and covers a distinct visual and physical regime. Long-horizon episodes are decomposed into atomic segments, stored as a directed acyclic graph that supports both curriculum learning and compositional evaluation. This organization makes coverage gaps directly measurable. It also supports cross-embodiment transfer, because the same atom can be collected on a single-arm gripper, a bimanual platform, or a dexterous hand and still carry a consistent label.

Training across heterogeneous embodied data requires accommodating differences in action dimensions, coordinate conventions, and label availability. We pretrain both variants on over 36 000 hours of real-robot trajectories, simulated demonstrations, public and in-house egocentric video, and UMI recordings. To combine these sources, we use a unified action representation, embodiment-aware conditioning, and source-specific validity masks. Robot trajectories and retargeted UMI recordings supervise both visual prediction and the available action coordinates. Egocentric video without reliable action labels contributes only to the visual objective. This formulation supports joint training across sources with different levels of action supervision.

Evaluating generalization also requires distinguishing execution of learned skills from their reuse under new conditions and compound instructions. We therefore design the *Real-World Atomic Skill Benchmark* on the Piper platform. We collect 3,921 demonstrations spanning five atomic task types and construct 69 evaluation entries, divided into 44 seen atomic entries, 19 held-out atomic entries, and six compositional entries. Piper adaptation uses only demonstrations of individual task goals; compound instructions are reserved for evaluation. The compositional tasks test repeated use of a skill across objects and ordered execution of different skills. For example, placing two blocks into a plate requires repeating a placement skill, while placing one block into the plate and then stacking another on it requires the first operation to establish the conditions for the second.

We evaluate downstream adaptation on five simulation benchmarks and our real-world benchmark. The simulation suite covers robustness to perturbations, bimanual manipulation, diverse manipulation tasks, kitchen-task composition, and object interaction. **Awomo-0.5I** achieves a 95.9% category mean on LIBERO-Plus, 64.4% overall success on RoboCasa365, and 79.7% overall success on the Real-World Atomic Skill Benchmark. It completes five of the six held-out compositional entries (83.3%), while the evaluated $\pi_{0.5}$ and DreamZero baselines complete none. Across evaluations reporting both Awomo-0.5 variants, the image-based model achieves higher overall success in most settings.

This report makes three main contributions:

- We introduce two implementations for adapting pretrained visual generators to robot control: an image-based action-as-patch interface and a video-based joint video-action interface.
- We build a large, heterogeneous pretraining mixture organized by atomic capabilities and a unified, validity-masked representation that supports robot, simulation, and egocentric human data across embodiments.
- We evaluate downstream adaptation in simulation and in the real world, including a diagnostic benchmark that

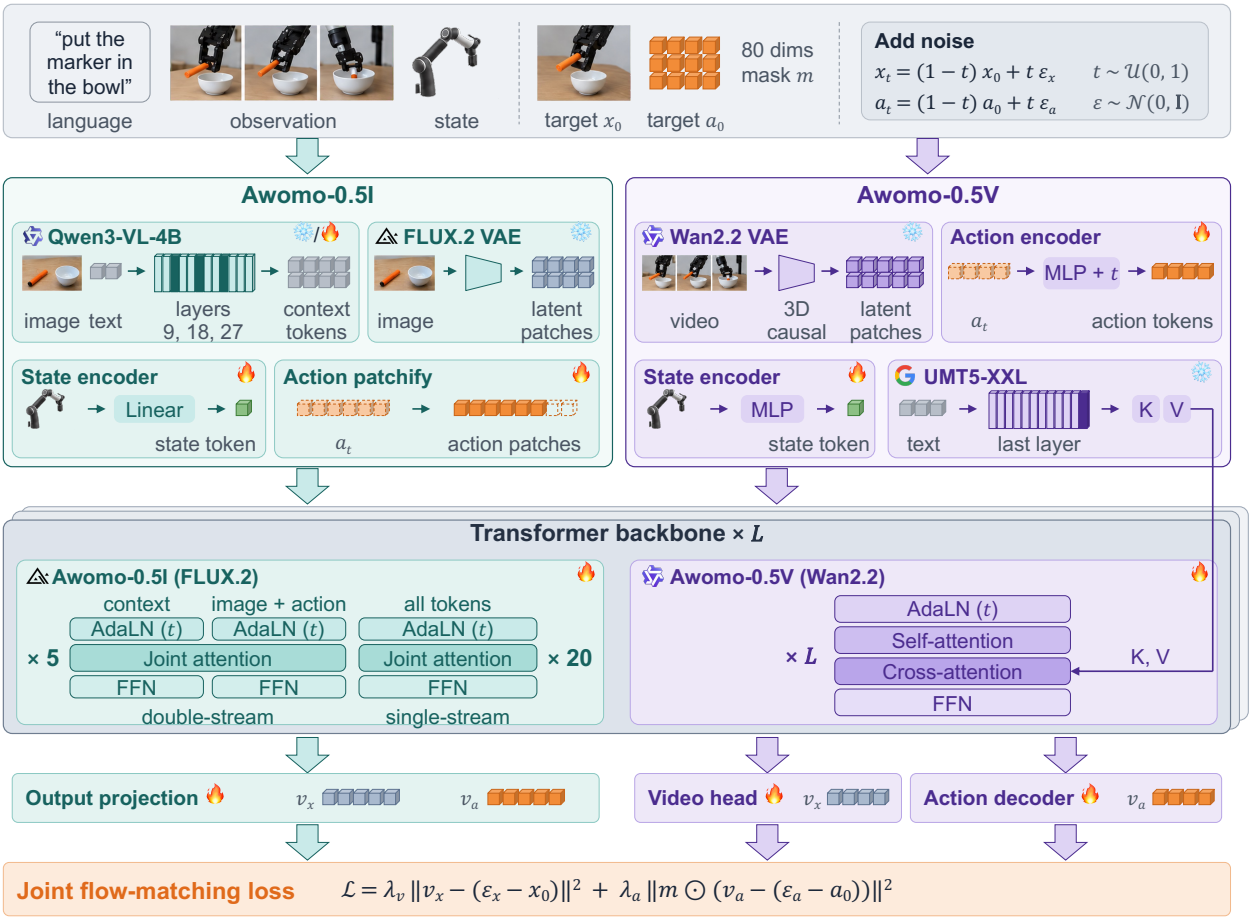


Figure 2. Training architectures of Awomo-0.5I and Awomo-0.5V.

measures transfer from an atomic task supervision to unseen conditions and multi-goal instructions.

2 Architecture

Awomo-0.5 formulates robot control as conditional flow generation over visual and action variables. The conditioning context $\mathbf{c}$ contains an embodied prompt (Sec. 3.2.2), the current camera observations, and the current robot state. The generated variables are a future visual latent $\mathbf{x}$ and an action chunk $\mathbf{a} \in \mathbb{R}^{H\times 80}$ of horizon H. Both are initialized from noise and iteratively denoised by one shared transformer; the state remains a condition and is not predicted. Multiple camera views are tiled into a single mosaic frame, so a fixed visual input interface serves single-arm, bimanual, and mobile platforms.

States and actions use the 80-dimensional unified space of Sec. 3.2.1. Each dataset scatters its native dimensions into fixed slots after per-dataset quantile normalization, and the validity mask $\mathbf{m}$ marks the slots that the embodiment provides. At inference, the generated chunk is gathered back through the same index map and de-normalized into native robot commands.

Figure 2 shows how both variants are trained. They read the same inputs: the text prompt, the camera observations, the robot state, and the targets $\mathbf{x}_0$ and $\mathbf{a}_0$, which are noised at a shared flow time t (Sec. 2.4). Each variant turns these inputs into tokens with its own encoders, processes the tokens with its pretrained transformer backbone, and maps the outputs to the visual velocity $\hat{\mathbf{v}}_x$ and the action velocity $\hat{\mathbf{v}}_a$. The variants differ in the visual prior, the text encoder, and how actions enter and leave the backbone.

2.1 Image-based action generation

Awomo-0.5I is initialized from the FLUX.2-klein-base-4B image transformer. Four encoders turn the inputs into tokens. A Qwen3-VL-4B-Instruct encoder reads the current observation followed by the text prompt; hidden states from layers 9, 18, and 27 are concatenated into 7680-dimensional context tokens, and up to 128 text positions are kept. It is frozen in pretraining and unfrozen in fine-tuning only for benchmarks with subtask goals. A linear layer maps the current state to one context token that is inserted after the valid text tokens. The frozen FLUX.2 autoencoder encodes the current frame as a clean reference image and the frame aligned with the end of the action chunk as the noisy future-image target, and both latents are flattened into 128-channel patches. Each action step, an 80-dimensional vector, is zero-padded to the same width and treated as one additional patch.

The patch sequence is therefore [reference image, noisy future image, noisy actions], and all patches share the transformer's input projection and output projection. The backbone contains 5 double-stream blocks followed by 20 single-stream blocks with hidden size 3072. In a double-stream block, the context tokens and the image and action patches keep separate timestep modulation and feed-forward weights and interact through joint attention; a single-stream block processes all tokens with one set of weights. The first 80 channels of the output at each action position give the action velocity, so the model needs no separate action expert and the pretrained image backbone processes actions directly. The future-image patches are decoded by the frozen autoencoder.

2.2 Video-based action generation

Awomo-0.5V is initialized from the Wan2.2-TI2V-5B video transformer. The frozen Wan2.2 video autoencoder encodes the observation clip into video latents, which enter through the backbone's 3D convolutional patch embedding. A learned action encoder maps noisy actions and their diffusion timestep into the transformer's token space, and an independent MLP embeds the state. The text prompt is encoded by the frozen UMT5-XXL text encoder and is not part of the token sequence.

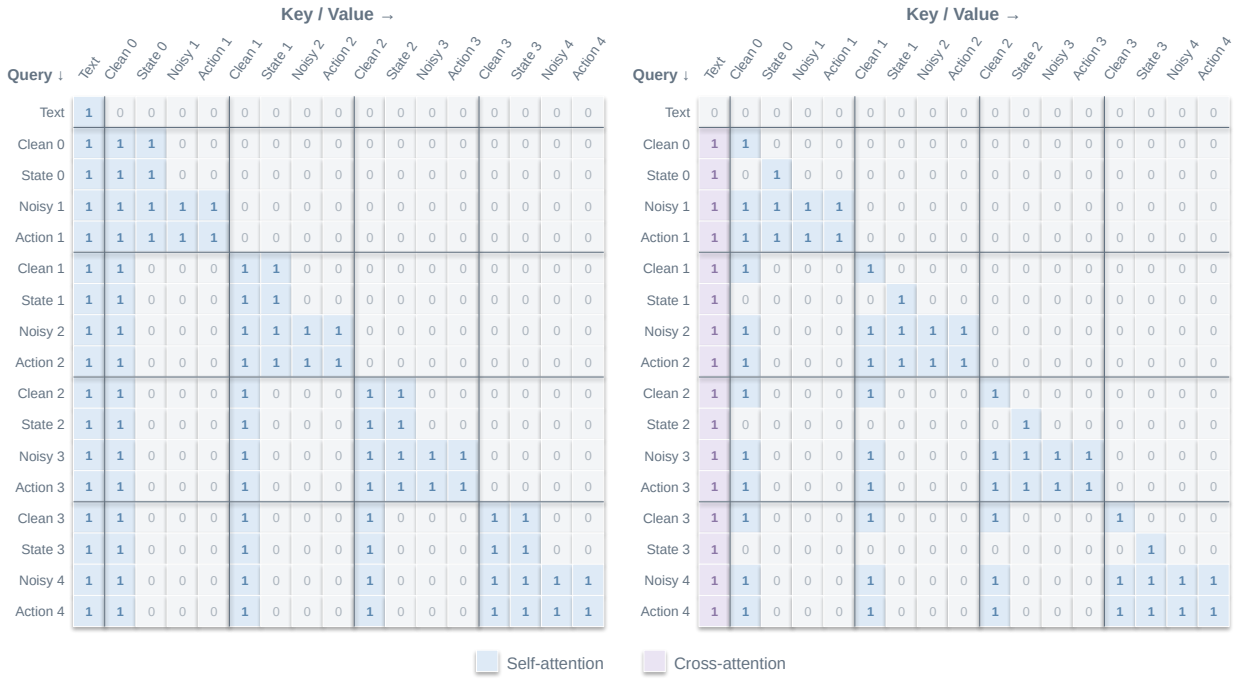
Each backbone block applies timestep modulation, self-attention over the video, state, and action tokens, cross-attention to the UMT5-XXL text features, and a feed-forward layer. A video head predicts the velocity of the video latents, and a dedicated MLP decoder maps the action tokens to the action velocity. The initial observation frame is kept clean as context and excluded from the visual loss; the model jointly denoises the following video frames and the aligned actions.

Awomo-0.5V groups time into chunks. Each chunk pairs one block of future video latents with $H = 24$ actions and the state at the start of the chunk, and pretraining uses four chunks per sample. **Awomo-0.5I** uses one chunk per sample for some datasets, conditioning on the current frame only, and several chunks per sample for others, conditioning each chunk on the clean frames up to its start (Sec. 2.3).

2.3 Attention masks

The attention masks in Fig. 3 preserve clean visual history while preventing predicted chunks from leaking future information. In **Awomo-0.5I**, the condition tokens (text, state, and clean reference image) attend only to condition tokens, and the noisy image and action tokens attend to all valid tokens. When a sample contains several chunks, the noisy image and action tokens of a chunk attend to the text, the state of that chunk, the clean images up to the chunk start, and the noisy tokens of the same chunk; noisy tokens and states of different chunks do not attend to each other. In **Awomo-0.5V**, self-attention is block-causal over chunks: clean video tokens attend to clean video up to the current block, each state token attends only to itself, and the noisy video and action tokens of a chunk attend to the clean history together with the noisy video, actions, and state of the same chunk. Text enters every non-text token through cross-attention. These masks let both variants learn from multi-chunk sequences under teacher forcing while retaining the dependency structure required at inference time.

At inference, both variables start from Gaussian noise and are integrated over a fixed number of denoising steps. The robot executes a prefix of the generated chunk and then replans from new observations; the per-benchmark settings are listed in Tab. 21.



(a) Awomo-0.5I

(b) Awomo-0.5V

Figure 3. Attention masks of Awomo-0.5I and Awomo-0.5V.

2.4 Flow-matching objective

Pretraining uses paired flow-matching objectives for visual generation and action generation. **Awomo-0.5I** predicts a future image and its aligned action chunk from the current image, instruction, and state. **Awomo-0.5V** predicts future video content together with the corresponding actions from an observation history.

Let $\mathbf{x}_0$ denote the clean visual latent of the prediction target, a future image for **Awomo-0.5I** and future video content for **Awomo-0.5V**, and let $\mathbf{a}_0 \in \mathbb{R}^{H\times 80}$ denote the aligned action chunk of horizon H in the unified action space. The conditioning context $\mathbf{c}$ contains the instruction, current visual observations, and robot state. For a flow time $t \in [0,1]$ and Gaussian noise $\boldsymbol{\epsilon}_x, \boldsymbol{\epsilon}_a \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, both variables follow the linear interpolation path

$$\mathbf{x}_t = (1-t)\,\mathbf{x}_0 + t\,\boldsymbol{\epsilon}_x, \qquad \mathbf{a}_t = (1-t)\,\mathbf{a}_0 + t\,\boldsymbol{\epsilon}_a, \tag{1}$$

whose target velocities are $\boldsymbol{\epsilon}_x - \mathbf{x}_0$ and $\boldsymbol{\epsilon}_a - \mathbf{a}_0$. The shared transformer predicts both velocities from the joint noisy input, $(\hat{\mathbf{v}}_x, \hat{\mathbf{v}}_a) = v_\theta(\mathbf{x}_t, \mathbf{a}_t, t, \mathbf{c})$, and is trained with

$$\mathcal{L}_{\text{vis}} = \mathbb{E}\Big[\big\|\hat{\mathbf{v}}_x - (\boldsymbol{\epsilon}_x - \mathbf{x}_0)\big\|_2^2\Big], \tag{2}$$

$$\mathcal{L}_{\text{act}} = \mathbb{E}\left[\frac{\delta}{H\|\mathbf{m}\|_1}\big\|\mathbf{m}\odot\big(\hat{\mathbf{v}}_a - (\boldsymbol{\epsilon}_a - \mathbf{a}_0)\big)\big\|_2^2\right], \tag{3}$$

$$\mathcal{L} = \lambda_v\,\mathcal{L}_{\text{vis}} + \lambda_a\,\mathcal{L}_{\text{act}}, \tag{4}$$

where $\mathbf{m} \in \{0,1\}^{80}$ is the validity mask of Sec. 3.2.1, broadcast over the H action steps, $\delta \in \{0,1\}$ indicates whether the sample carries reliable action labels, and λ_v and λ_a weight the visual and action terms. For robot trajectories and retargeted UMI recordings, $\delta = 1$ and both objectives are active. For egocentric sources without reliable robot commands, $\delta = 0$ removes the action loss while retaining the visual objective. Within action-labeled samples, $\mathbf{m}$ excludes unsupported embodiment dimensions, allowing a shared model to learn from platforms with different control spaces without fabricating labels. At inference, both variables start from noise at $t = 1$ and are integrated to $t = 0$ with the predicted velocities.

The objective therefore separates what each source can supervise while keeping the backbone and conditioning

interface shared. Visual-only data broadens the model's representation of objects, scenes, and manipulation dynamics; robot and retargeted data anchor that representation to executable actions.

3 Pretraining Data

3.1 Data Composition

The pretraining corpus determines the effective upper bound on how far the model can be optimized across heterogeneous embodiments and task families. Accordingly, a large-scale and heterogeneous pretraining mixture is constructed to endow the model with broad embodied perception, spatial reasoning, and action-generation capabilities. As shown in Tab. 1, the mixture is partitioned into four source categories: real robot trajectories (30 %), simulated robot trajectories (30 %), UMI human demonstrations (20 %), and pure egocentric human video (20 %). In aggregate, the corpus comprises over 36 000 hours of multi-domain interaction data.

Table 1. Composition of the Awomo-0.5 pretraining corpus.

Data type	Source	Datasets	Hours	Episodes	Weight	Action
Real robot trajectories	Public	AgiBot World Alpha [7], AgiBot World Beta [8], AgiBotWorld-2026 [9], BC-Z [10], BridgeDataV2 [11], DROID [12], Galaxea [13], RDT-1B [14], RH20T [15], RoboChallenge [16], RoboCOIN [17], RoboMINDV1/V2 [18, 19], RobotSet [20], RT-1 [21]	5.5K	336.2K	30 %	✓
	In-house	Proprietary	4.5K	105.6K		
Simulated robot trajectories	Public	InternData-A1 [22], NVIDIA Kitchen Demos [23]	2.1K	207K	30 %	✓
	In-house	Proprietary	8.5K	3.6M		
UMI human demonstrations	In-house	Proprietary	5.0K	990.9K	20 %	✓
Egocentric human videos	Public	Ego4D [24], EgoDex [25], EgoSuite-Open100K [26], EgoVerse [27], EPIC-KITCHENS [28], Xperience [29]	6.7K	351.5K	20 %	✗
	In-house	Proprietary	3.5K	1.9M		
Total			36.8K	7.5M	100 %	

3.1.1 Real Robot Trajectories

Real-world robot trajectories provide the most faithful visual observations of robots interacting with the physical world. To improve data scale, diversity, and cross-embodiment generalization, publicly available real-robot datasets are incorporated: DROID [12], AgiBot World Alpha [7], AgiBot World Beta [8], AgiBotWorld-2026 [9], RoboChallenge [16], RoboCOIN [17], RobotSet [20], Galaxea [13], RoboMINDV1/V2 [18, 19], RDT-1B [14], BridgeDataV2 [11], RH20T [15], RT-1 [21], and BC-Z [10]. Together they cover single-arm and bimanual manipulation on Franka and AgiBot platforms, among others, and amount to approximately 5521 hours of interaction. The resulting mixture delivers complementary supervision over robot morphologies, camera viewpoints, object categories, backgrounds, language instructions, and action distributions, which mitigates overfitting to any single embodiment or environment. The corpus is further supplemented by more than 4.5K hours of in-house real-robot trajectories covering a variety of manipulation tasks and platforms. Real-robot data contribute 30 % of the pretraining mixture.

3.1.2 Simulated Robot Trajectories

Beyond real-robot recordings, the mixture is complemented by simulation-generated trajectories, which reach object arrangements and task layouts that are expensive to stage physically and cover long-tail configurations that teleoperation campaigns rarely encounter by chance. Synthetic supervision reaches us through two distinct channels: public simulation datasets, and demonstrations produced by our own generation pipeline SimDataEngine. Both channels are curated and converted into the unified observation-action format before training, with each set's original task semantics preserved so that synthetic and real experience can be mixed freely.

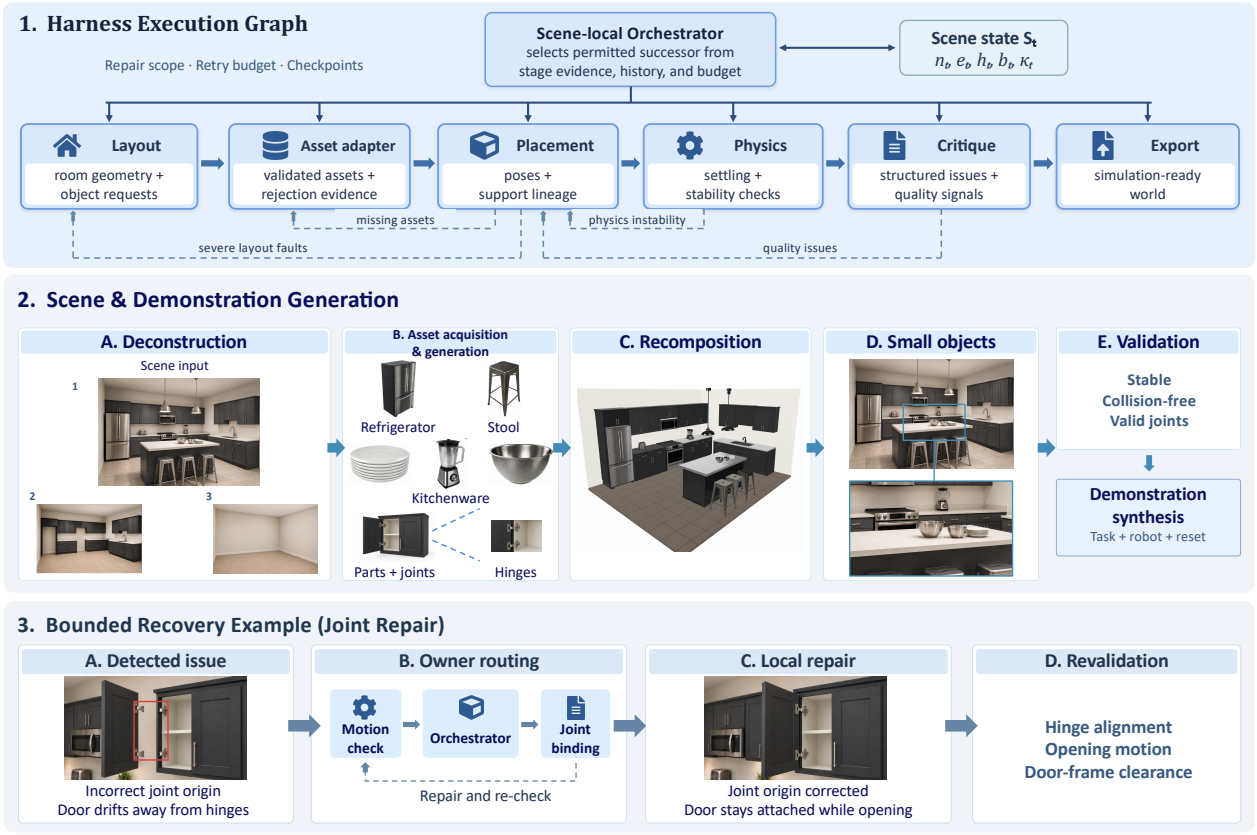


Figure 4. SimDataEngine overview. (1) The graph-native harness coordinates execution, validation, and bounded repair. (2) The illustrated image-driven route uses Unravel to reconstruct scenes using acquired or generated assets, including articulated assets generated by ISArt. PolicyForge synthesizes demonstrations from validated worlds with task, robot, and reset specifications. (3) A schematic example illustrates failure detection and owner routing, followed by joint-origin correction and motion revalidation.

Public simulation datasets. We incorporate InternData-A1 [22] and NVIDIA Kitchen Demos [23], which together contribute approximately 2145 hours. These datasets are generated by scripted or motion-planned demonstrations across a range of virtual environments. InternData-A1 covers rigid, articulated, deformable, and fluid-object manipulation, along with long-horizon tasks and multi-arm collaboration, which supplies supervision well beyond standard tabletop pick-and-place. Because these sets are produced under limited task and scene assumptions, they are not tailored to the embodiments and task families we care most about, which is what motivates the in-house channel below.

In-house simulation data. The more targeted share of synthetic supervision is produced in house, through our scalable pipeline SimDataEngine [30] (Fig. 4). SimDataEngine is an agentic, simulation-ready pipeline for world construction and demonstration synthesis. It supplies interactive, physically valid, and reproducible synthetic supervision for embodied-policy training. Generation follows a three-tier chain spanning asset, scene, and demonstration. First, a structure-aware asset service (ISArt) generates rigid and articulated objects whose parts and joints remain geometrically consistent. Scene generation proceeds through two complementary pathways: *Unravel* reconstructs editable scenes from a single image, whereas *SimForge* synthesizes environments ranging from single-room to multi-room configurations from textual descriptions; both preserve object and articulation identities. A graph-native execution harness handles construction, validation, and bounded repair, routing failures to the originating module while retaining the unaffected portion of the scene state. On the validated worlds, *PolicyForge* binds the scene, a task contract (goal, interaction, and success predicates), and a robot-embodiment record, then records synchronized multi-camera RGB imagery, proprioceptive states, and end-effector actions within NVIDIA Isaac Sim. Only episodes satisfying the task-success predicate are retained and exported in LeRobot format, supporting multiple embodiments such as Franka, the Piper dual-arm platform, and AgiBot G1/G2. Scale is expanded through digital-cousin variants and sampling across the scene-task difficulty

space. The data also serves as cross-simulator co-training material: co-training with Isaac Sim demonstrations on the MuJoCo-based LIBERO-Plus benchmark raises the overall success rate of a World Action Model from $77.2\,\%$ to $89.4\,\%$ (success on the goal and spatial task suites improves by 31.7 and 6.3 percentage points, respectively), substantiating the utility of simulation data for cross-simulator policy training. The pretraining mixture uses approximately 8.5K hours (3.6 million episodes) of SimDataEngine demonstrations.

3.1.3 UMI Human Demonstrations

The UMI human demonstrations are first-person human-manipulation demonstrations collected with DAS data-acquisition system. Acquisition employs the DAS-Ego head-mounted six-camera rig together with two DAS Gripper handheld grippers with fisheye cameras, deployed in real home, commercial, and industrial scenarios. Every camera stream operates at a resolution of at least 1600×1300 and a frame rate of $\geq 30\,\mathrm{Hz}$; the head rig provides a horizontal field of view of $270^\circ \times 150^\circ$ and simultaneously covers both hands, while each gripper integrates at least one fisheye camera with a field of view of $\geq 150^\circ$ and a magnetic encoder that logs gripper open-close distance at $\geq 50\,\mathrm{Hz}$; automotive-grade IMUs record inertial information at $200\,\mathrm{Hz}$. Head-camera pose is tracked synchronously at $\geq 30\,\mathrm{Hz}$, with global time-synchronization error $< 1\,\mathrm{ms}$ and an overall frame-drop rate $< 1\,\%$. Each 0.5-5 minute clip is collected through crowdsourcing under a strict prohibition on staged actions. Every episode synchronously contains six ego RGB streams, left and right finger RGB, high-precision 6D pose reconstruction for the ego and both fingers, IMU, gripper open-close distance, camera intrinsic and extrinsic calibration, and audio recordings, together with per-frame and per-segment validity flags and quality-inspection labels. Annotation follows a three-level structure (video level, task level, and subtask level), providing coarse semantic descriptions and fine-grained task/subtask segments. The pretraining mixture uses approximately 4963 hours of UMI recordings.

UMI Action Solving and Retargeting UMI recordings carry no robot commands, so the end-effector state and the action stream are solved from multi-camera video, 6D pose reconstructions, and raw gripper encoder readings, and are expressed in the same absolute end-effector parameterization as robot data (Apps. A.1 and A.2). The two grippers populate the two mirrored arm blocks of the unified action space, and slots the rig cannot supply are excluded by the validity mask.

3.1.4 Egocentric Human Videos

Compared with teleoperated robot trajectories, egocentric human demonstrations offer a substantially more abundant and scalable source of real-world manipulation experience. Humans continuously interact with diverse objects across open-world environments, naturally producing dexterous manipulation behaviors that span a considerably broader range of scenes, objects, and task semantics than robot teleoperation permits. Multiple studies have established that pretraining on large-scale egocentric human video endows vision-language-action models with richer manipulation priors and improves generalization to downstream robotic tasks [31-36]. Motivated by these findings, a diverse corpus of egocentric human manipulation data is integrated to provide our model with broad manipulation priors that complement the robot-trajectory data.

Public Ego datasets. The public egocentric corpus combines Ego4D [24], EgoDex [25], EgoSuite-Open100K [26], EgoVerse [27], EPIC-KITCHENS [28], and Xperience [29], which together contribute approximately 6710 hours of first-person manipulation video. Their annotations do not provide precise, robot-executable state and action labels, and human hand motion cannot be converted into end-effector commands without committing to a specific kinematic chain. Their action and proprioception channels are therefore fully masked, and the data supervises only the video stream, contributing visual dynamics and manipulation priors through the world-modeling objective. Action supervision comes instead from robot trajectories and from the UMI recordings described above.

In-house Ego videos. The in-house egocentric corpus is mined from a large pool of short first-person clips, most of which do not show a manipulating operator: many lack visible hands, show a third-person view of another person, or contain only one hand. Labeling every clip with a vision-language model is too costly at this scale, so the filter is distilled from a teacher into a lightweight student (Fig. 5). (1) Teacher labeling: Qwen3.5-27B labels a small subset of roughly 6000 clips by judging whether both hands of the same operator are clearly visible; hard cases are reviewed by humans and used to refine the teacher prompt, and the resulting labels form the student's training set. (2) Student

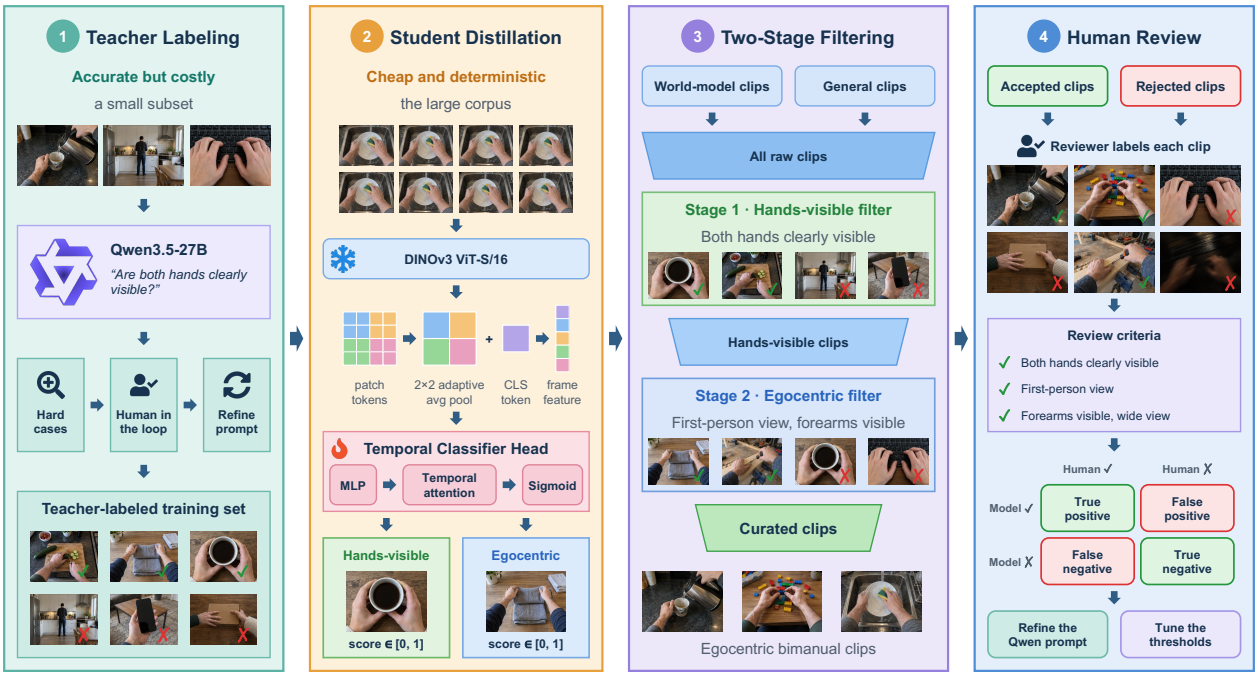


Figure 5. Egocentric clip filtering pipeline.

distillation: each clip is represented by 8 sampled frames, encoded by a frozen DINOv3 ViT-S/16 backbone into a CLS token and 2×2 adaptively pooled patch features per frame; a temporal classifier head (MLP, temporal attention, and sigmoid) aggregates the frame features into a clip-level score in $[0, 1]$. Two heads are trained, one for hand visibility and one for the egocentric view. (3) Two-stage filtering: the hands-visible head first screens all raw clips, and the egocentric head then keeps only first-person clips in which both forearms are visible within a wide field of view; both stages use a score threshold of 0.7. (4) Human review: reviewers inspect sampled accepted and rejected clips against the same criteria, and the resulting confusion counts are used to refine the teacher prompt and tune the thresholds.

3.2 Data Processing

3.2.1 Unified Action Space

All real and simulated datasets are converted into a unified observation-action format that preserves their original task semantics, enabling scalable training over heterogeneous robot experience.

Unified action space. Different datasets adopt incompatible action conventions: some provide absolute end-effector poses in Cartesian space [37], others delta end-effector commands, and yet others absolute or relative joint-space control signals. A single benchmark may pass actions through directly (for instance, RoboTwin2.0 in either a 14-D joint space or a 20-D bimanual end-effector space, and LIBERO in a 10-D end-effector space), but training one model across embodiments requires a unified action head over bodies whose native layouts differ in both dimensionality and semantics. A unified action space $\mathbf{u} \in \mathbb{R}^{80}$ with fixed slot semantics is therefore defined:

- Two mirrored 29-D arm blocks, each comprising joint angles (7), end-effector position (3), a 6D rotation (6), the gripper state (1), and 12 reserved slots. The end-effector rotation adopts a fixed row-ordered 6D representation $[R_{00}, R_{10}, R_{20}, R_{01}, R_{11}, R_{21}]$, and the gripper is encoded as $0 =$ open, $1 =$ closed. The end-effector frame is expressed in the robot coordinate system (X: forward, Y: left, Z: up).
- Three 6-D mobile/manipulator blocks: Base, Torso, and Head.
- Four additional reserved slots for embodiment-specific channels.

This yields $2 \times 29 + 3 \times 6 + 4 = 80$ dimensions. Because slot semantics are fixed, structural components shared across embodiments map to identical coordinates. Each dataset declares an index map π from its native dimensions into these

slots, with normalization applied before scattering and inverted after gathering, and incoming proprioception traverses the same map forward. The validity mask $\mathbf{m}$ marks exactly the mapped slots, so unmapped coordinates receive no gradient during training and remain on their analytic noise path at inference.

Action normalization. All action dimensions are normalized per dataset using quantile statistics. For each action dimension d in dataset k, the 1st and 99th percentiles q_k^{01} and q_k^{99} are computed over all trajectories and a linear mapping is applied, with the result clipped to $[-1, 1]$. This per-dataset quantile normalization removes scale differences across embodiments and action spaces while preserving the relative motion structure within each source.

3.2.2 Embodied Prompt

The textual component of each training example comprises three parts: embodiment identification, language instruction, and camera-view representation. Diversity across these components is preserved to enhance generalization. The three parts are illustrated by the following example:

> Embodiment identification: AgiBot G1, dual-arm, fixed base.
> Language instructions: A multi-view video shows a robot performing the task: Pick up items in the supermarket. The robot is positioned in front of the fruit stand in the supermarket environment.
> Camera view representation: The video is split into 3 views: the top view shows the primary external camera (head_camera); the bottom-left view shows the robot's wrist camera (hand_left_camera); the bottom-right view shows the robot's wrist camera (hand_right_camera).

Embodiment identification. The embodied prompt informs the model of the current control convention. Each training example is prepended with a robot-specific prompt that specifies the platform, arm configuration, and control convention in effect.

Language instructions. Language instructions come from a combination of original dataset annotations and model-generated captions provided alongside the data. All instructions pass through a quality-filtering and consistency-checking pipeline: trajectories whose language annotations are inconsistent with the observed motion are discarded, ensuring reliable language-action alignment across diverse sources.

Camera view representation. Many robot datasets provide observations from multiple camera viewpoints simultaneously, for example a head-mounted ego-centric camera together with one or two wrist-mounted cameras. To make view provenance explicit to the model, a view-representation scheme is incorporated into the prompt. This enables the backbone to form view-aware representations and lets the model attend selectively to each viewpoint when generating actions, without requiring any architectural modification.

3.2.3 Data Cleaning Pipeline

Data cleaning follows three stages, illustrated on real-robot data in Fig. 6. Visual screening also applies to the other sources, signal validation applies to real-robot, simulated, and UMI data, and language review is performed on surviving samples. The pipeline draws on Qwen-RobotManip [38] and incorporates checks for failure modes observed in our collection. Specifically, our data cleaning pipeline reduces 60K hours of raw data to 36.8K hours of validated training data.

Stage 1: Visual screening. All sources are checked for visual quality. Frames are removed if they are missing, corrupted, or undecodable; frozen, repeated, or abruptly discontinuous; uniformly black, white, or single-coloured; over- or under-exposed or affected by exposure flicker; or blurred. For egocentric recordings, intervals in which neither hand is visible are also removed. This stage alone drops $18\,\%$ of raw hours, primarily from egocentric video with poor lighting or motion blur.

Stage 2: Signal validation. Real-robot, simulated, and UMI sources carry state and action streams in addition to video, and these are validated through a chain of physical-coherence checks. The checks are applied in fixed order, and each step receives only the data that survived the previous one. Motion thresholds are calibrated per embodiment rather than globally, since a single numeric cutoff does not transfer across robots with different link lengths and control rates.

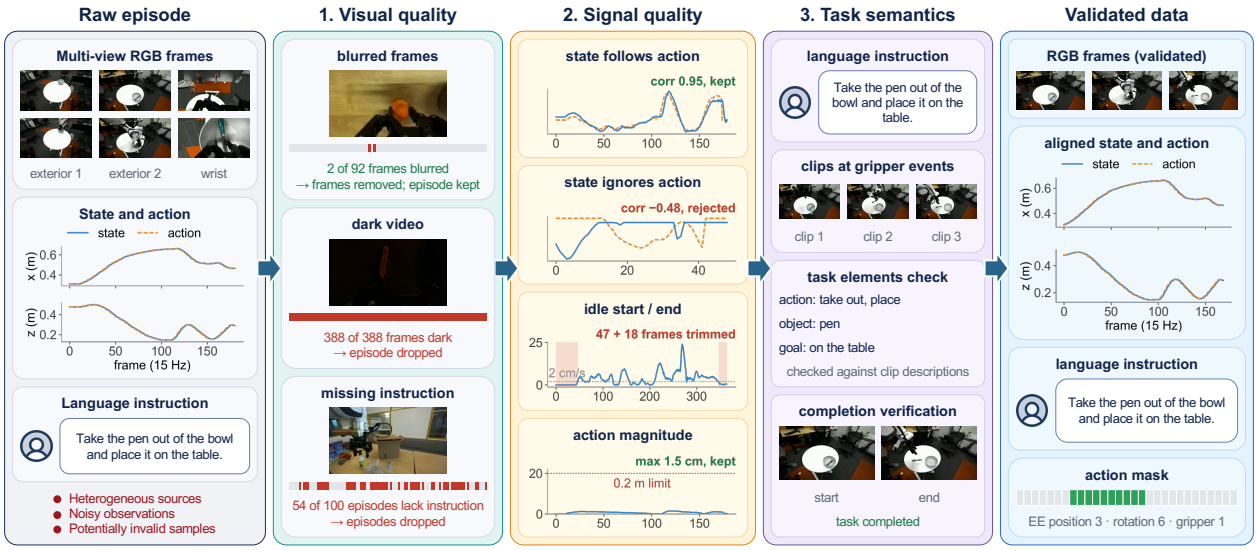


Figure 6. Real robot data cleaning pipeline.

1. **Tracking validity.** An episode is rejected when the logged end-effector state does not follow the action stream, or when video and signal streams are misaligned.
2. **Edge inactivity.** Leading and trailing stretches in which the state barely moves are trimmed. Pauses inside an episode are left alone, because excising them would concatenate frames that were never adjacent in time. Discontinuities such as jerk and spike outliers are caught separately with robust median-MAD screening.
3. **Motion corroboration.** Reported state motion is checked against the imagery. A frame that reports movement while every camera view is visually still is sensor jitter and is trimmed; conversely, an end effector that visibly moves under a frozen camera indicates a capture fault, and the episode is removed.
4. **Episode rejection.** An episode that surrenders most of its frames to the stages above, or whose video is frozen for most of its duration, is discarded.
5. **Action magnitude.** In robot data, startup and reset routines near episode boundaries emit sustained large actions, such as rapid arm retraction, that stay temporally smooth and state-consistent and so escape the earlier checks. Because such excursions skew the quantile statistics used for normalization, samples with excessive single-step end-effector translation or rotation are dropped from every source with end-effector actions.

App. A.3 lists the thresholds of each check. Signal validation removes another 11 % of the remaining hours, with the largest losses from simulation episodes that violate physics constraints and from real-robot trajectories with tracker dropout.

Stage 3: Language and semantic review. After the visual and signal stages, every surviving episode undergoes a final language check. Each instruction is screened for emptiness, malformation, and a missing action, object, or goal. Surviving instructions are then checked against the video: gripper open and close events split the episode into action-centred clips, a vision-language model describes each clip, and the clip descriptions together with sparsely sampled global frames and the original instruction decide whether the observed behaviour matches the instruction and whether the task completes. Sampled episodes from every robot dataset-embodiment pair are finally replayed and compared with the recorded multi-camera video, confirming that conventions such as gripper polarity and coordinate frames have been mapped correctly. Semantic review drops 12 % of the remaining data, chiefly egocentric video where the narrated goal does not match the observed motion.

The cumulative effect is a 1× concentration of data quality. Retained episodes have clean visuals, physically consistent signals, and semantically verified language—three properties that are jointly necessary for cross-embodiment pretraining. The 23.3K hours of rejected data are not discarded outright; they are logged with rejection tags (blur, tracking loss, semantic mismatch, *etc.*) and fed back into the collection pipeline to guide targeted re-collection.

3.3 Data Organization with Atomic Skills

We organize training data around *atomic skills* (A1–A16) rather than end-to-end tasks. This enables compositional generalization, makes coverage gaps directly measurable, and supports cross-embodiment transfer: the same "open a drawer" atom can be collected on a single-arm gripper, a bimanual platform, or a dexterous hand and still carry a consistent label. Table 2 lists the sixteen families consolidated from prior kitchen, laboratory, office, home, and mobile-manipulation taxonomies.

3.3.1 Design principles.

The sixteen families are chosen by four criteria that together ensure coverage, measurability, and reuse.

Perceptual diversity. Each family exercises a distinct visual modality. A1–A3 (Pick & Place, Door, Lid) require object–container spatial reasoning; A4–A7 (Insertion, Button, Lever, Knob) demand fine-grained contact detection; A8–A10 (Slide Rack, Pour, Wipe) involve fluid or deformable dynamics; A11–A12 (Fold, Screw) test dexterous hand–object interaction; A13–A14 (Stir, Shake) capture repetitive oscillatory motion; and A15–A16 (Regrasp, Navigation) span in-hand manipulation and whole-body motion. This spread prevents the validation set from collapsing into a single visual regime.

Action granularity. Families are aligned with the finest unit of control that still admits a binary success criterion. "Open a drawer" is an atom because it has a clear terminal condition (door angle $> 45°$), whereas "prepare breakfast" is a composite. Conversely, "grasp" and "transport" are merged into A1 because separating them would create intermediate states with no natural success predicate. The granularity is therefore task-dependent: contact-rich actions (A4, A9, A12) are kept narrow, while whole-arm motions (A1, A16) are allowed broader scope.

Closed-loop success criterion. Every family has an unambiguous physical terminal condition (object fully inserted, button depressed, container emptied, target pose reached). This makes the validation MAE a direct supervisory signal rather than a proxy: when MAE drops, the model is genuinely better at predicting the action that achieves the terminal state. Families for which no closed-loop criterion exists (e.g., "polish" without a surface-reflectance sensor) were excluded.

Composability. Families are the smallest units from which long-horizon tasks compose; L2 composites and L3 workflows are sequences of these atoms rather than new primitives. A "pick-and-place then open drawer" task is simply A1 followed by A2, each validated independently. The result is a closed set—every task in the training corpus decomposes into a subset of A1–A16. This closure property is what makes the taxonomy useful for both coverage auditing and compositional generalization.

Table 2. Pretraining atomic skill families.

ID	Family	Definition
A1	Pick & Place	Grasp, transport, and release an object
A2	Open/Close Door	Rotate or pull an articulated door to a target angle
A3	Lid	Cap a vessel or close an appliance head
A4	Insertion	Align and insert into a target opening
A5	Press Button	Make accurate contact and trigger a control
A6	Turn Lever	Flip or actuate a lever-shaped control
A7	Twist Knob	Rotate a knob to set a parameter
A8	Slide Rack	Push in or pull out a rack or tray
A9	Pour	Reorient a container to transfer contents
A10	Wipe/Clean	Clean a surface with repeated strokes
A11	Fold	Fold and shape deformable objects
A12	Screw/Fasten	Rotate a threaded part to fasten or assemble
A13	Stir	Stir continuously inside a container
A14	Shake	Move an object back and forth
A15	Regrasp/In-hand	Re-orient an object without setting it down
A16	Mobile Navigation	Navigate to a target pose in a cluttered space

3.3.2 Atomic skill annotation

Every validated episode is annotated with an atomic skill label and, where applicable, a compositional decomposition. This annotation happens *after* cleaning so that labels are applied only to physically valid trajectories.

Atomic labelling. Each episode is assigned a single primary family from A1–A16 by a two-stage labeller. First, a lightweight rule-based classifier maps the natural-language instruction to a family using verb–object patterns (e.g., "put X in Y" $\rightarrow$ A1, "open the drawer" $\rightarrow$ A2, "stir the soup" $\rightarrow$ A13). Second, a small human review panel corrects edge cases—especially instructions that use generic verbs ("manipulate the knob" could be A6 Turn Lever or A7 Twist Knob). The inter-reviewer agreement on a 1000-episode audit is $\kappa = 0.91$, so the rule-based label is accepted without review for the remaining corpus.

In addition to the primary family, every episode receives cross-cutting annotations that are used for stratified validation:

- **Scene:** kitchen, laboratory, office, or home.
- **Operating mode:** single-arm, bimanual, or mobile.
- **Object property:** rigid, articulated, deformable, or liquid.

These tags ensure that the validation set is stratified across dimensions that are orthogonal to the family label, preventing the model from overfitting to scene-specific cues.

Compositional decomposition. Long-horizon episodes are decomposed into atomic segments at gripper events (open/close transitions) and at language conjunctions ("and", "then"). A "pick-and-place then open drawer" episode is split into an A1 segment followed by an A2 segment, each inheriting the same scene and object-property tags but with distinct operating-mode annotations. The decomposition is stored as a directed acyclic graph in which nodes are atomic segments and edges encode temporal order. This graph is used for two purposes: (1) curriculum learning, where the model first trains on single-atom episodes and later on two-atom and three-atom compositions; and (2) evaluation, where held-out compositions test whether the model has learned to chain atoms rather than memorise whole-task trajectories.

Data organization. Training data is stored in a skill-indexed repository rather than a flat episode list. The top level partitions by family (A1–A16). This structure makes it trivial to audit coverage and to rebalance the training mixture when per-family validation MAE signals a bottleneck. During training, each batch samples uniformly over families, then uniformly within the selected family's scene and operating mode buckets. This two-level sampling guarantees that every family receives gradient updates while preserving the natural variation of scene and precision within each family.

4 Pretraining Analysis

This section analyzes pretraining through per-skill validation error, its effect on downstream adaptation, and ablations on human data.

4.1 Atomic Skill Analysis

We evaluate sixteen *atomic skill* families during pretraining. For each family we curate a held-out validation set disjoint from the training pools in scenes, objects, and instructions. The set covers background, object, instruction, and mixed perturbations with at least 100 episodes per condition. Every episode is labelled with its family, cross-cutting annotations (scene, operating mode, object property), and the validity mask from the cleaning pipeline.

Figure 7 shows the training loss of both variants and the per-family validation EEF MAE of **Awomo-0.5V**. In Fig. 7a, **Awomo-0.5I** reports image and action loss, and **Awomo-0.5V** reports video and action loss; raw traces are thin semi-transparent lines, and the 500-step simple moving average is drawn as thick lines, solid for action loss and dashed for image or video loss. Figure 7b draws the family mean as a thick solid line.

Training loss. Three observations can be read from the loss panels. First, the action loss in both models drops rapidly in the first $10\,000$ steps and then flattens, while the image/video loss decays more slowly and continues to decrease through step $50\,000$. This divergence is expected: the action head has fewer parameters and a direct regression target, whereas the visual backbone must reconstruct high-dimensional latent patches. The gap between action and visual loss at step $50\,000$ for Awomo-0.5I) shows that visual prediction remains the harder objective even late in training, which justifies retaining the video term in the joint loss rather than dropping it early.

Second, **Awomo-0.5I** starts with a higher image loss because its action-patch architecture processes image tokens independently, whereas **Awomo-0.5V** shares the video backbone with the diffusion prior; by step $50\,000$ both have converged to comparable levels, so the initial gap is a transient optimization effect rather than a structural limitation. The convergence is reassuring: it means the action-as-patch interface does not permanently handicap visual quality, it merely delays visual convergence by roughly 5000–$10\,000$ steps.

Third, the raw traces reveal substantial step-level noise, especially in the action loss. This noise comes from the mixture of action-supervised and vision-only batches: when a batch contains mostly egocentric video with masked actions, the action head receives weak gradient and the loss spikes. The 50-step EMA smooths this into a stable trend; we use the EMA, not the raw trace, for checkpoint selection.

Per-family MAE. The per-family validation MAE at training step s is

$$\mathrm{MAE}_c(s) \;=\; \frac{1}{|\mathcal{E}_c|\,H\,|\mathcal{M}_c|} \sum_{e\in\mathcal{E}_c} \sum_{h=1}^{H} \sum_{i\in\mathcal{M}_c} \big|\hat{a}_i^{(e,h,s)} - a_i^{(e,h)}\big|, \tag{5}$$

where $\mathcal{E}_c$ is the validation set, H the action horizon, and $\mathcal{M}_c$ the mapped end-effector slots (position, 6D rotation, and gripper) for family c. Unmapped slots are excluded by the validity mask.

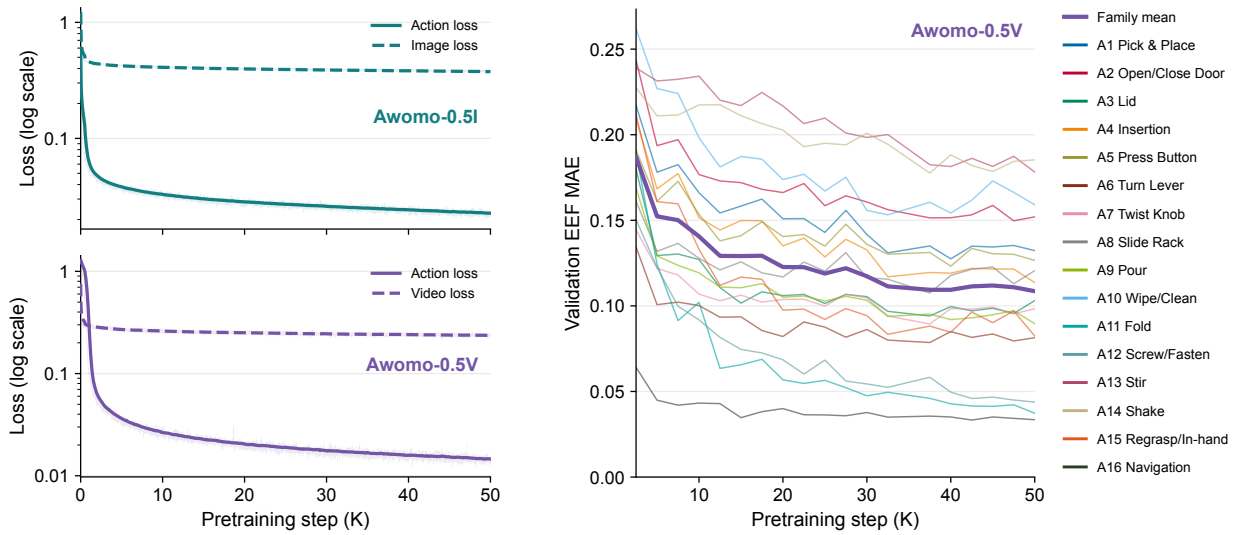
The MAE panel reveals a data-density hierarchy that mirrors training-set frequency, but with important deviations. Data-rich families—A1 Pick & Place (0.047), A5 Press Button (0.052), and A16 Navigation (0.034)—saturate below 0.06 by step $50\,000$. Their low error is not merely a volume effect: A1 and A5 also have simple kinematics (translation and a single contact), while A16 benefits from a large corpus of mobile-navigation episodes ($>1.2\,\mathrm{M}$ frames) that provide repetitive structure.

Moderate-support families (A2 Open/Close Door at 0.152, A8 Slide Rack at 0.118, A10 Wipe/Clean at 0.159) plateau in the 0.10 to 0.16 range. Their higher MAE reflects physical variability rather than scarcity: door articulations differ across embodiments (revolute vs. prismatic), rack friction coefficients vary, and wiping contact patterns are texture-dependent. This is exactly the regime where cross-embodiment pretraining helps most, because the model learns to condition on the validity mask rather than memorising embodiment-specific dynamics.

High-precision / low-data families remain the bottleneck. A4 Insertion (0.128) and A9 Pour (0.095) require sub-centimetre accuracy, so their MAE floors are intrinsically higher. A13 Stir (0.178) and A14 Shake (0.185) are the most data-scarce: together they account for less than $2\,\%$ of the training corpus, and their oscillatory trajectories are harder to compress with the action-chunk representation. A12 Screw/Fasten (0.044) is an outlier in the positive direction: despite being a precision task, it achieves low MAE because the rotation axis is largely collinear with the approach vector, reducing the effective degrees of freedom.

The family mean curve (dashed) is a convenient aggregate, but it masks the long tail: the mean at step $50\,000$ is 0.109, yet the inter-family range spans a factor of five (0.034 to 0.185). A model with low mean MAE can still fail on the high-precision subset $\{A4, A9, A12, A15\}$.

Finally, the MAE trajectories correlate with closed-loop success in a non-uniform way. On RoboCasa365 atomic-seen tasks, families with MAE below 0.08 achieve $>90\,\%$ success, while families above 0.14 drop to $60\,\%$–$75\,\%$. The relationship is roughly log-linear (Figure 10), but the slope differs by family: contact-rich families (A4, A7, A9) show a steeper slope than kinematic families (A1, A16), because small action errors at contact onset compound into large trajectory deviations. This family-dependent sensitivity is why per-family MAE is a more useful diagnostic than a single aggregate metric.



(a) Pretraining loss. **(b) Atomic skill validation EEF MAE.**

Figure 7. Pretraining loss and atomic skill validation EEF MAE. (a) Training loss of Awomo-0.5I (top) and Awomo-0.5V (bottom). (b) Validation EEF MAE of Awomo-0.5V.

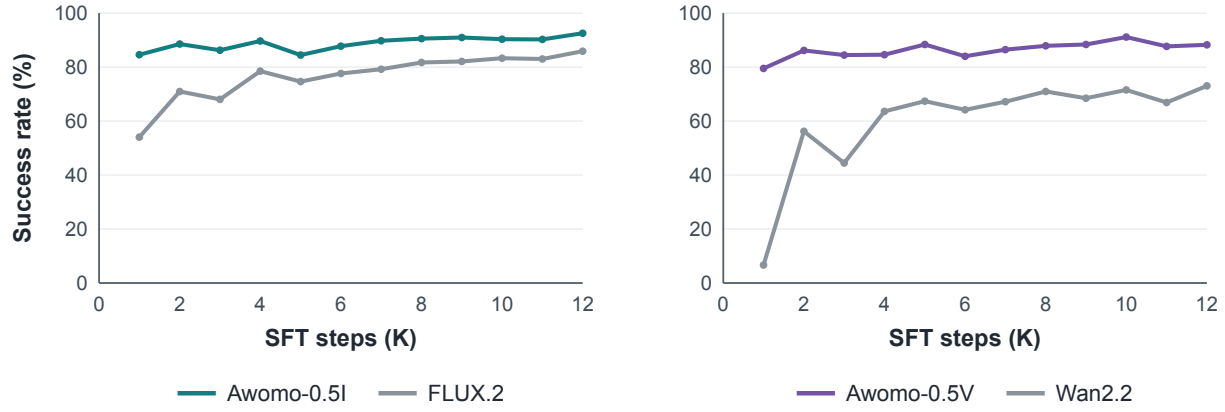


Figure 8. Effect of embodied pre-training on LIBERO-Plus.

4.2 Effect of Pretraining

Fig. 8 compares the available LIBERO-Plus (Sec. 5.1.3) fine-tuning trajectories on a 10% evaluation subset. For **Awomo-0.5V**, the run initialized from our pretrained checkpoint reaches 86.2% success after 2,000 optimizer steps, compared with 56.2% for the Wan2.2-initialized run, which normalizes the action loss over the ten valid LIBERO-Plus action dimensions and weights it twice the visual loss. At 12,000 steps, the respective success rates are 88.3% and 73.0%. Within the displayed 12,000 steps, the highest observed subset success rates are 91.1% for the pretrained run at 10,000 steps and 73.0% for the Wan2.2 run at 12,000 steps. The pretrained run therefore adapts faster and achieves higher success within these observed trajectories. For **Awomo-0.5I**, the two runs share code, downstream data, optimization recipe, and evaluation subset, and differ only in initialization: our pretrained checkpoint or the original FLUX.2-klein-base-4B weights. After 1,000 optimizer steps, the pretrained run reaches 84.6% success, compared with 54.0% for the FLUX.2-initialized run. At 12,000 steps, the last evaluated step of the FLUX.2 run, the respective success rates are 92.5% and 85.9%. Both panels display up to 12,000 optimizer steps. Because initialization is the only changed variable, this gap measures the effect of pretraining on **Awomo-0.5I** adaptation within the evaluated budget. The **Awomo-0.5V** runs differ in training configuration; their gap is descriptive and does not isolate the causal effect of pretraining.

Table 3. RoboTwin 2.0 clean success rate (%) of early Awomo-0.5V. Each checkpoint is evaluated three times.

(a) UMI pretraining

UMI pretraining	Success rate
✗	5.6
✓	**15.0**
Δ	+9.4

(b) Data mixture

UMI (h)	Ego (h)	Success rate
1,000	0	20.0
500	500	**25.2**
Δ		+5.2

4.3 Human Data Ablations

We study transfer from human data with an early version of **Awomo-0.5V** on RoboTwin 2.0, using joint visual prediction and action generation. Two separate experiments examine the benefit of an additional UMI pretraining stage and the effect of replacing part of the UMI corpus with egocentric video. Each comparison holds its downstream fine-tuning settings fixed.

UMI pretraining before robot fine-tuning. Both conditions use the same initialization and joint visual-action training objective. The baseline is fine-tuned directly on robot demonstrations; the other condition first receives 24,065 updates on UMI recordings. Both then receive five epochs of fine-tuning on the same 2,500 RoboTwin clean demonstrations, with the same optimizer, learning-rate schedule, and global batch size. We evaluate the final fine-tuning checkpoint on 50 tasks with ten episodes per task. UMI pretraining increases clean success from 5.6% to 15.0%, an improvement of 9.4 percentage points (Tab. 3a).

Replacing UMI recordings with egocentric video. At a fixed corpus size of 1,000 hours, we compare 1,000 hours of UMI with 500 hours of UMI and 500 hours of egocentric video. The mixed condition retains the same 500-hour UMI subset and replaces the other 500 hours. Both runs use the same initialization, optimizer, learning-rate schedule, global batch size of 512, and 33-frame visual windows. UMI supplies visual and action supervision, whereas egocentric video supplies visual supervision only. Each run splits its corpus into two frozen 500-hour pools, the shared UMI subset and either the remaining UMI or the egocentric video, and each batch draws equally from them. We use checkpoints after 20,000 pretraining updates and apply identical five-epoch robot fine-tuning.

The mixed corpus achieves 25.2% mean clean success, compared with 20.0% for UMI alone, a gain of 5.2 percentage points (Tab. 3b). The gain is positive in all three repeated evaluations. The number of episodes that succeed in all three evaluations also increases from 65 to 95 out of 500. Thus, in this setting, replacing half of the action-supervised UMI corpus with visual-only egocentric experience improves downstream performance. The comparison supports combining the two forms of human experience, while leaving their individual contributions and the best mixture ratio to further study.

The randomized setting shows a smaller increase, from 5.3% (26/488) for UMI alone to 6.1% (29/479) for the mixed corpus, approximately 0.7 percentage points. These scores come from one evaluation per model and exclude environment initialization failures; the clean comparison provides stronger evidence of the transfer benefit.

5 Simulation Post-Training

We study post-training of **Awomo-0.5I** and **Awomo-0.5V** on five simulation benchmarks, training a separate policy for each benchmark and evaluating it under the corresponding protocol. We report benchmark performance alongside comparisons with representative VLA and WAM models, then examine action-space choices. Training settings, evaluation episode counts, and inference configurations are listed in App. B.

5.1 Simulation Results

RoboDojo, RoboCasa365, LIBERO-Plus, MolmoSpaces, and RoboTwin 2.0 probe diverse task requirements, compositional kitchen tasks, robustness to controlled perturbations, object-interaction skills, and bimanual manipulation, respectively. Figure 9 summarizes the results across the five simulation benchmarks and the real-world benchmark,

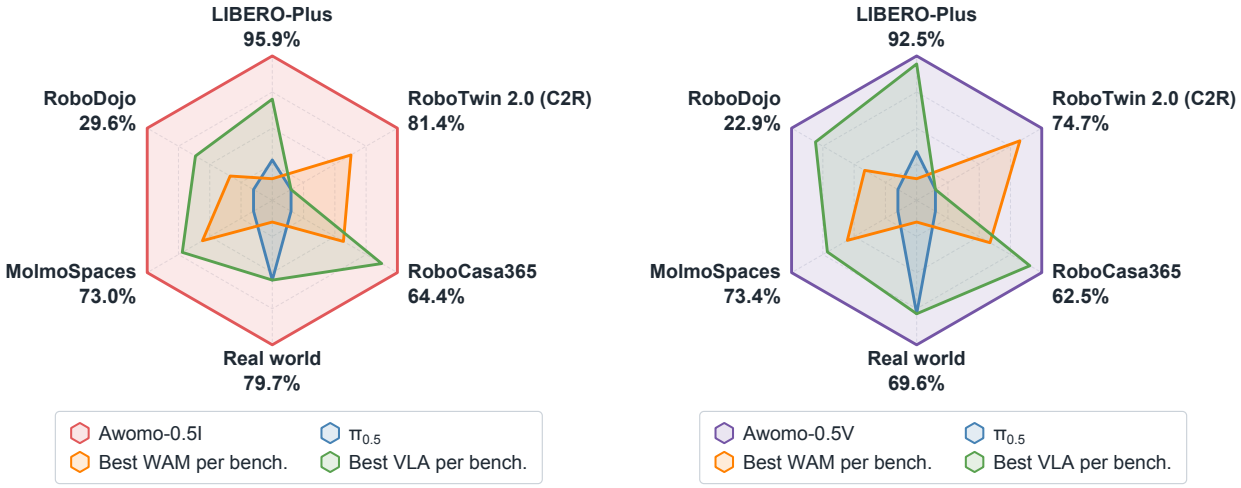


Figure 9. **Comparison across five simulation benchmarks and the real-world benchmark.**

Table 4. **RoboDojo simulation score and success rate (%), as of October 7, 2026.**

Model	Generalization	Precision	Long-Horizon	Memory	Open	Overall
VLAs						
GalaxeaVLA (G0) [39]	4.5 / 2.8	8.1 / 3.8	12.6 / 5.6	3.2 / 1.9	0.7 / 0.7	5.8 / 3.0
StarVLA [40]	3.9 / 2.3	9.9 / 4.3	14.2 / 6.5	3.3 / 2.4	0.7 / 0.6	6.4 / 3.2
Xiaomi-Robotics-0 [41]	7.4 / 5.6	8.4 / 4.6	13.5 / 6.9	5.1 / 3.7	0.2 / 0.2	6.9 / 4.2
StarVLA-OFT [42]	7.4 / 4.8	16.1 / 10.2	14.3 / 6.2	1.7 / 1.7	1.4 / 1.3	8.2 / 4.8
StarVLA-GR00T [42]	8.0 / 5.3	9.6 / 4.7	16.2 / 7.2	5.6 / 4.3	1.9 / 1.9	8.3 / 4.7
MolmoAct2 [43]	8.5 / 4.7	10.9 / 4.5	15.2 / 7.0	5.5 / 4.6	**4.8** / **4.4**	9.0 / 5.0
X-VLA [3]	10.5 / 6.8	18.3 / 12.0	16.5 / 9.8	4.8 / 3.6	0.6 / 0.5	10.1 / 6.5
VLAct [44]	9.5 / 6.3	20.6 / 15.2	20.1 / 13.7	0.7 / 0.6	2.4 / 2.3	10.7 / 7.6
StarVLA-PI_v3 [42]	11.2 / 8.1	17.8 / 12.5	18.5 / 11.0	4.6 / 4.0	2.0 / 2.0	10.8 / 7.5
InternVLA-A1.5 [45]	10.4 / 6.8	15.2 / 10.2	23.8 / 13.8	4.9 / 3.6	1.4 / 1.4	11.2 / 7.1
Pi-05 [2]	13.4 / 8.2	12.4 / 5.5	23.5 / 14.7	5.9 / 4.7	2.0 / 1.7	11.4 / 6.9
Spatial Forcing [46]	14.1 / 9.3	17.3 / 10.6	23.3 / 14.6	5.4 / 4.1	1.8 / 1.6	12.4 / 8.0
Hy-Embodied-0.5-VLA [47]	11.8 / 8.4	13.8 / 8.0	25.7 / 14.9	13.4 / 12.1	0.7 / 0.6	13.1 / 8.8
Meituan-Robotics-0 [48]	13.8 / 8.2	16.8 / 7.8	29.6 / 18.6	10.1 / 8.9	4.5 / 4.3	15.0 / 9.5
Xiaomi-Robotics-1 [49]	**23.5** / **17.0**	26.7 / 18.8	38.4 / 23.7	7.8 / 6.6	3.9 / 3.6	20.1 / 13.9
GalaxeaVLA (G0.5) [50]	18.5 / 12.8	**28.3** / **20.4**	**44.1** / **32.3**	8.6 / 7.3	1.7 / 1.6	20.2 / 14.9
DM0.5 (Dexmal) [51]	15.8 / 10.9	24.8 / 16.8	33.7 / 19.5	**47.7** / **47.4**	2.4 / 2.1	**24.9** / **19.3**
WAMs						
LDA-1B [52]	0.8 / 0.1	3.3 / 0.7	2.0 / 0.1	2.1 / 1.8	0.1 / 0.1	1.7 / 0.5
Fast-WAM [53]	2.3 / 1.1	2.0 / 0.0	9.1 / 5.2	3.5 / 3.4	0.4 / 0.4	3.5 / 2.0
AHA-WAM [54]	5.8 / 3.3	5.9 / 2.4	8.6 / 2.7	3.0 / 2.8	0.9 / 0.8	4.8 / 2.4
GigaWorld-Policy-0 [55]	5.3 / 2.9	6.2 / 1.8	15.5 / 8.9	3.5 / 2.2	0.5 / 0.5	6.2 / 3.3
X-WAM [6]	7.4 / 3.3	6.7 / 1.8	17.5 / 9.1	6.3 / 4.7	0.6 / 0.3	7.7 / 3.8
OpenWAM-α [56]	20.7 / 14.8	18.4 / 9.3	34.9 / 25.3	10.4 / 9.1	1.4 / 1.1	17.2 / 11.9
Awomo-0.5V	26.9 / 20.8	36.3 / 28.1	39.0 / 25.0	13.1 / 12.5	**28.1** / **28.1**	28.7 / 22.9
Awomo-0.5I	**30.3** / **23.8**	**39.7** / **33.5**	**40.4** / **26.8**	**42.3** / **41.2**	24.1 / 22.9	**35.3** / **29.6**

using the clean-to-random protocol for the RoboTwin axis. Each benchmark's setting, results, and comparison table follow below.

5.1.1 RoboDojo

RoboDojo [57] evaluates manipulation capabilities in simulation and the real world. Here we report the simulation benchmark, which covers generalization, precision, long-horizon execution, memory, and open-vocabulary instruction following.

Evaluation reports both task-progress score and complete-task success rate. For generalization, each metric is averaged across standard and randomized conditions. The overall score and success rate are then computed by averaging their respective values across the five capability dimensions. The reference protocol and episode counts per seed are summarized in Tab. 16 in the appendix.

Awomo-0.5I obtains an overall score of 35.3 and a success rate of 29.6% (Tab. 4). Standard and randomized generalization yield 37.70 / 30.89% and 22.92 / 16.78%, respectively, where each pair denotes score / success rate. The remaining dimensions are precision at 39.7 / 33.5%, long-horizon execution at 40.4 / 26.8%, memory at 42.3 / 41.2%, and open-vocabulary instruction following at 24.1 / 22.9%. Memory has the highest completion rate, while the success-rate drop under randomized generalization identifies robustness as a remaining challenge. The gap between long-horizon score and complete-task success also indicates that partial task progress does not consistently lead to final completion.

5.1.2 RoboCasa365

RoboCasa365 [58] provides 365 kitchen tasks across 2,500 environments. Its multi-task learning benchmark evaluates 50 target tasks comprising 18 Atomic-Seen, 16 Composite-Seen, and 16 Composite-Unseen tasks. The unseen composite split tests whether a policy can execute task compositions excluded from its downstream training data.

Evaluation uses RoboCasa version 1.0.1 and runs 30 trials for each of the 50 target tasks, yielding 1,500 episodes: 540 Atomic-Seen, 480 Composite-Seen, and 480 Composite-Unseen. Each split success rate is the fraction of successful evaluation episodes in that split. Overall success is the total number of successful episodes divided by 1,500, equivalent to weighting the three split success rates in an 18:16:16 ratio.

As shown in Tab. 5, **Awomo-0.5I** achieves 64.4% overall success (966/1,500), with 86.5% on Atomic-Seen, 58.8% on Composite-Seen, and 45.2% on Composite-Unseen. **Awomo-0.5V** reaches 62.5% (938/1,500), with respective split success rates of 85.0%, 60.4%, and 39.4%. Both models exceed Xiaomi-Robotics-1's reported overall success rate of 57.4% by 7.0 and 5.1 percentage points, respectively. **Awomo-0.5I** performs better on unseen compositions, while **Awomo-0.5V** is stronger on seen compositions. Both retain substantial success on unseen task compositions, demonstrating generalization beyond the downstream training tasks.

5.1.3 LIBERO-Plus

LIBERO-Plus [65] evaluates manipulation policies under controlled changes to camera viewpoints, robot initial states, language instructions, lighting, background textures, sensor noise, and object layouts. Its 10,030 evaluation tasks test whether a policy remains reliable when the scene and instructions differ from its training conditions.

Each evaluation covers all 10,030 task instances, with one rollout per instance; the category counts are listed in Tab. 18 in the appendix. The reported policies are selected from multiple SFT trials, so the table reports selected-checkpoint performance rather than an average over training seeds. Each category success rate is the fraction of successful rollouts in that category. The reported category mean is the arithmetic mean of the seven category success rates, giving equal weight to each perturbation type. Consequently, the 95.9% and 92.5% category means are distinct from a pooled success rate over all 10,030 rollouts, which weights categories by their episode counts.

Table 6 reports a category mean of 95.9% for **Awomo-0.5I** and 92.5% for **Awomo-0.5V**. Both models maintain success rates above 97% under background, sensor-noise, camera, and lighting perturbations. **Awomo-0.5I** reaches 94.3% on language perturbations, 91.0% on object layouts, and 89.9% on robot initial states. The corresponding **Awomo-0.5V** results are 87.9%, 83.3%, and 84.0%, identifying these three categories as its main remaining challenges.

Table 5. RoboCasa 365 success rate (%).

Model	Atomic (seen)	Composite (seen)	Composite (unseen)	Overall
VLAs				
Azero-Robotics-1 [59]	30.3	3.8	1.6	12.6
π_0 [1]	34.6	6.1	1.1	14.8
$\pi_{0.5}$ [2]	39.6	7.1	1.2	16.9
GR00T N1.6 [60]	51.1	9.4	1.7	21.9
GR00T N1.5 [60]	50.7	14.8	2.7	23.9
WorldDreamer	66.3	26.7	9.0	35.3
RLDX-1 [61]	67.6	27.9	8.5	36.0
PRTS [62]	66.3	30.3	18.8	39.6
Phasor-m7 [63]	78.7	**57.4**	31.5	56.8
Xiaomi-Robotics-1 [49]	**80.2**	57.1	**32.1**	**57.4**
WAMs				
GigaWorld-Policy 0.1 [55]	44.4	11.8	2.9	20.7
ABot-M0.5 [64]	75.6	37.7	3.3	40.3
Awomo-0.5V	85.0	**60.4**	39.4	62.5
Awomo-0.5I	**86.5**	58.8	**45.2**	**64.4**

Table 6. LIBERO-Plus success rate (%) by perturbation category.

Model	Camera	Robot	Language	Light	Background	Noise	Layout	Overall
VLAs								
GAM [66]	83.1	70.0	84.8	97.2	94.3	95.3	79.1	86.3
$\pi_{0.5}$ [2]	75.8	79.4	83.3	95.5	95.0	89.6	87.0	86.5
Multi-view VLA [67]	89.6	60.1	86.9	98.0	95.7	97.2	78.2	86.5
3DVLA [68]	75.6	77.4	88.6	97.4	97.7	85.3	86.6	86.9
ERVLA [69]	77.2	75.3	87.1	95.1	94.7	92.3	86.4	86.9
RLDX-1 [61]	75.0	**91.8**	**92.2**	97.1	95.5	80.3	80.7	87.5
Reflective VLA [70]	95.7	72.9	**92.2**	92.1	92.7	95.4	73.0	87.7
ACoT-VLA [71]	96.6	70.4	79.7	95.1	97.1	95.9	85.0	88.5
Anchor-Align VLA [72]	96.3	59.1	87.2	**99.0**	99.6	96.9	**97.4**	90.8
QuoVLA [73]	**98.4**	73.5	83.7	96.3	99.0	94.8	90.4	90.9
CAC-VLA [74]	94.8	72.5	86.6	98.9	98.6	**98.3**	87.6	91.0
Hermite-VLA$_{\text{Reg}}$ [75]	89.2	85.4	87.3	97.7	96.7	93.8	89.9	91.4
Qwen-RobotManip-Context [76]	89.9	83.9	86.5	98.6	**99.9**	97.9	87.5	**92.0**
WAMs								
Fast-WAM [53]	16.4	44.5	68.9	78.2	53.7	37.7	60.7	51.4
Cosmos-Policy [5]	75.8	63.3	81.7	96.5	88.9	92.7	82.2	83.0
ImageWAM [77]	80.8	50.3	91.4	98.1	85.5	93.8	80.5	82.9
ABot-M0.5 [64]	70.5	87.4	88.6	94.0	89.7	75.5	85.2	84.4
Being-H0.7 [78]	82.0	59.0	82.8	97.8	90.0	93.5	88.5	84.8
Awomo-0.5V	**98.7**	84.0	87.9	97.4	98.5	97.8	83.3	92.5
Awomo-0.5I	98.6	**89.9**	**94.3**	**99.5**	**98.8**	**99.1**	**91.0**	**95.9**

The two models exceed the 92.0% category mean reported for Qwen-RobotManip-Context, showing strong robustness across the evaluated perturbations.

5.1.4 MolmoSpaces

MolmoSpaces [79] provides diverse simulated environments for robot navigation and manipulation. We evaluate four manipulation task families: Open, Close, Pick, and Pick-and-Place. These families cover articulated-object interaction

Table 7. MolmoSpaces success rate (%).

Model	Pick	Pick & Place	Open	Close	Overall
VLAs					
π_0-FAST DROID [80]	22.0	10.9	11.1	38.6	20.7
LAP-VLA [81]	24.9	6.6	11.4	45.9	22.2
π_0 DROID [1]	16.2	12.5	11.0	53.1	23.2
$\pi_{0.5}$ DROID [2]	36.4	13.6	22.7	65.1	34.5
MolmoAct2 DROID [43]	48.3	23.3	11.7	71.3	38.6
PRTS-Droid [62]	53.8	27.7	12.0	76.3	42.4
WALL-OSS-0.5 [82]	73.8	46.0	16.8	78.5	53.8
PrimeR0	**77.5**	**54.6**	**27.8**	**81.3**	**60.3**
WAMs					
Psi-R2	73.1	41.5	9.0	62.2	46.4
Vana-Policy	59.7	36.1	35.1	60.8	47.9
Cosmos3-Nano-Policy [83]	65.5	35.2	32.0	79.1	53.0
Awomo-0.5V	**86.4**	58.9	**62.4**	86.0	**73.4**
Awomo-0.5I	84.5	**71.0**	49.3	**87.2**	73.0

and object transport.

For each task family, success rate is the number of successful episodes divided by the number of evaluated episodes. The combined average is the arithmetic mean of the Open, Close, Pick, and Pick-and-Place success rates, weighting the four families equally. The **Awomo-0.5V** evaluation contains 1,000 Open, 915 Close, 1,000 Pick, and 1,000 Pick-and-Place instances, each evaluated once (Tab. 19 in the appendix), totaling 3,915 episodes.

Table 7 reports a combined average of 73.4% for **Awomo-0.5V**, with 62.4% on Open, 86.0% on Close, 86.4% on Pick, and 58.9% on Pick-and-Place. **Awomo-0.5I** reaches 73.0% overall, with respective success rates of 49.3%, 87.2%, 84.5%, and 71.0%. Both models perform strongly on closing and picking, and exceed PrimeR0's reported combined average of 60.3%[1]. Their task profiles are complementary: **Awomo-0.5V** is 13.1 percentage points higher on opening, whereas **Awomo-0.5I** is 12.1 points higher on Pick-and-Place.

5.1.5 RoboTwin 2.0

RoboTwin 2.0 [100] evaluates bimanual manipulation across 50 tasks under clean and randomized scene conditions. Its domain randomization varies clutter, lighting, backgrounds, tabletop height, and language instructions, testing policy robustness to changes in the observation and task context.

We report two post-training settings: full-data uses both clean and randomized demonstrations, whereas clean-to-random uses clean demonstrations only. Both settings evaluate in clean and randomized scenes.

We compute success rates across the 50 tasks separately for clean and randomized scenes, then take the arithmetic mean of the two condition-level success rates as the reported average.

Awomo-0.5I achieves 96.0% success in clean scenes and 96.2% in randomized scenes, averaging 96.1% across the two conditions (Tab. 8). The difference between conditions is only 0.2 percentage points, indicating consistent aggregate performance under the evaluated randomization. It exceeds MotuBrain's reported average of 95.9% by 0.2 percentage points.

Under clean-to-random post-training, **Awomo-0.5I** reaches 86.8% in clean scenes and 76% in randomized scenes, with an average of 81.4% (Tab. 9). Randomized-scene success is 10.8 percentage points lower than clean-scene success in this setting. These results characterize robustness after training on clean demonstrations only.

[1] Lumos, MolmoSpaces leaderboard, https://molmospaces.allen.ai/benchmark, Jun 10, 2026.

Table 8. RoboTwin 2.0 success rate (%) under the full-data protocol: training on clean and randomized demonstrations and evaluation in clean and randomized scenes.

Model	Clean	Randomized	Average
VLAs			
π_0 [1]	65.9	58.4	62.2
X-VLA [3]	72.8	72.8	72.8
$\pi_{0.5}$ [2]	82.7	76.8	79.8
LingBot-VLA [84]	88.6	86.7	87.6
ZR-0 [85]	88.7	88.0	88.3
HoloBrain-0-QW [86]	91.9	92.3	92.1
G0.5 [50]	**93.7**	92.8	93.3
Qwen-RobotManip [76]	**93.7**	**94.0**	**93.9**
WAMs			
X-WAM [6]	89.8	90.7	90.3
LaWAM [87]	92.6	89.8	91.2
LingBot-VA [88]	92.9	91.6	92.2
Faster-WAM [89]	92.8	92.3	92.6
SelfWAM [90]	92.2	93.1	92.6
MECo-WAM [91]	93.3	92.0	92.6
AIM [92]	94.0	92.1	93.1
ImageWAM [77]	93.2	93.6	93.4
LingBot-VA 2.0 [93]	93.8	93.4	93.6
Next Forcing [94]	94.1	93.5	93.8
ABot-M0.5 [64]	94.0	94.2	94.1
MotuBrain [95]	95.8	96.1	95.9
Awomo-0.5V	92.8	91.6	92.2
Awomo-0.5I	**96.0**	**96.2**	**96.1**

Table 9. RoboTwin 2.0 success rate (%) under the clean-to-random protocol: training on clean demonstrations only and evaluation in clean and randomized scenes.

Model	Clean	Randomized	Average
VLAs			
starVLA [42]	46.5	3.2	24.8
GalaxeaVLA [39]	62.7	12.7	37.7
Xiaomi Robotics-0 [41]	62.9	18.2	40.6
EventVLA [96]	65.6	15.7	40.7
X-VLA [3]	68.0	20.9	44.5
$\pi_{0.5}$ [2]	**70.7**	**46.0**	**58.4**
WAMs			
AHA-WAM [54]	64.3	3.2	33.8
FastWAM [53]	77.8	1.9	39.9
Abot-M0 [97]	57.4	30.4	43.9
X-WAM [6]	70.0	25.8	47.9
Spatial Forcing [46]	77.2	26.7	52.0
OLA-Geo	83.0	33.6	58.3
4D-WAM [98]	81.5	41.8	61.7
GigaBrain-0.7 [99]	66.8	67.9	67.4
OLA-Sem	75.1	67.6	71.3
Awomo-0.5V	84.6	64.8	74.7
Awomo-0.5I	**86.8**	**76.0**	**81.4**

5.2 Open-loop Error and Closed-loop Success

We relate open-loop action error to closed-loop success for **Awomo-0.5I** during RoboCasa365 post-training (Sec. 5.1.2). Open-loop MAE and loss are computed on fixed reference windows of each task group every 2.5K updates from 2.5K to 60K; closed-loop SR is evaluated at 5, 10, 20, 30, 40 and 50K updates. The 60K SR in Tab. 5 comes from a separate evaluation batch and is excluded from the fits. Figure 10 plots MAE and SR against loss for each task group, and Tab. 10 lists the corresponding fits. Overall open-loop error covers Atomic-Seen and Composite-Seen windows, while overall SR covers all three task groups. On Atomic-Seen and Composite-Seen tasks, SR decreases linearly with the logarithm of loss, and MAE follows a power law of loss. On Composite-Unseen tasks, MAE and loss stay near 0.08 and 0.047 from 2.5K updates onward while SR rises from 34.4% at 5K to 42.5% at 50K, so open-loop error on unseen tasks does not follow closed-loop success.

We also measure whether the future frames predicted by each model agree with the executed rollout. For each episode, we compare the frame predicted 0.8 s ahead with the frame observed at the same time during execution, and compute two scores on the robot region: Mask IoU, the intersection over union of the predicted and executed robot masks, and Boundary F, the F-score of the two robot contours under a two-pixel tolerance. Both scores range from 0 to 100. The study covers 18 Atomic-Seen tasks with three episodes each, 54 episodes per checkpoint, and SR is computed on the same episodes. Figure 11 shows that both scores rise together with SR within the first 5K updates. After 5K updates, Mask IoU and Boundary F change by at most 3.0 points, while the Atomic-Seen SR of **Awomo-0.5I** in Fig. 10, evaluated on 540 episodes, keeps rising from 73.9% at 5K to 87.4% at 50K. Visual consistency therefore tracks early adaptation and does not separate later checkpoints.

Figure 12 shows one Atomic-Seen episode, CoffeeSetupMug from the external camera, at the first, middle and last checkpoints of each model. At step 0, the predicted arm is displaced from the executed arm, with Mask IoU of 54.7 for **Awomo-0.5V** and 21.6 for **Awomo-0.5I**. At the middle checkpoint, the predicted and executed arms nearly coincide, with Mask IoU above 86 and Boundary F above 96 for both models, and both scores change by less than 2 points at the

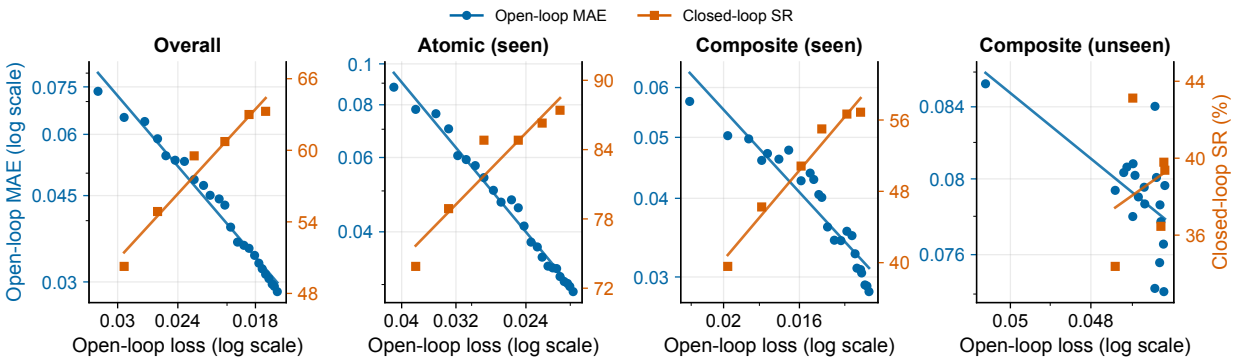


Figure 10. Open-loop loss, open-loop MAE and closed-loop SR of Awomo-0.5I on RoboCasa365.

Table 10. Fitting parameters of Fig. 10.

Task group	Fitted relation	Checkpoints	b	a	R^2	RMSE
Overall	$\log_{10}\mathrm{MAE} = a + b\log_{10}\mathrm{loss}$	24	1.48	1.11	0.98	0.018
	$\mathrm{SR} = a + b\log_{10}\mathrm{loss}$	6	−56.8	−35.6	0.96	1.0
Atomic (seen)	$\log_{10}\mathrm{MAE} = a + b\log_{10}\mathrm{loss}$	24	1.60	1.19	0.99	0.017
	$\mathrm{SR} = a + b\log_{10}\mathrm{loss}$	6	−49.6	5.1	0.90	1.5
Composite (seen)	$\log_{10}\mathrm{MAE} = a + b\log_{10}\mathrm{loss}$	24	1.35	1.04	0.90	0.028
	$\mathrm{SR} = a + b\log_{10}\mathrm{loss}$	6	−103.7	−135.9	0.97	1.1
Composite (unseen)	$\log_{10}\mathrm{MAE} = a + b\log_{10}\mathrm{loss}$	19	1.09	0.34	0.38	0.012
	$\mathrm{SR} = a + b\log_{10}\mathrm{loss}$	5	−167.5	−184.4	0.06	2.9

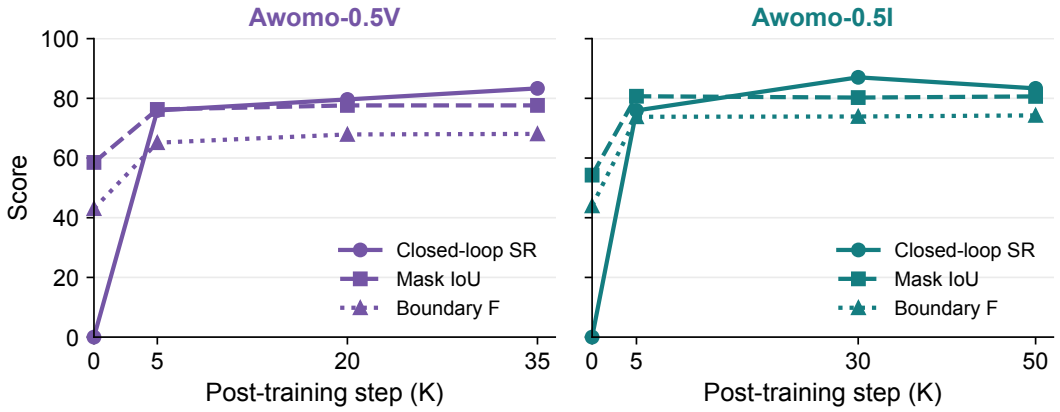


Figure 11. Visual prediction-execution consistency and closed-loop SR on RoboCasa365 Atomic-Seen tasks.

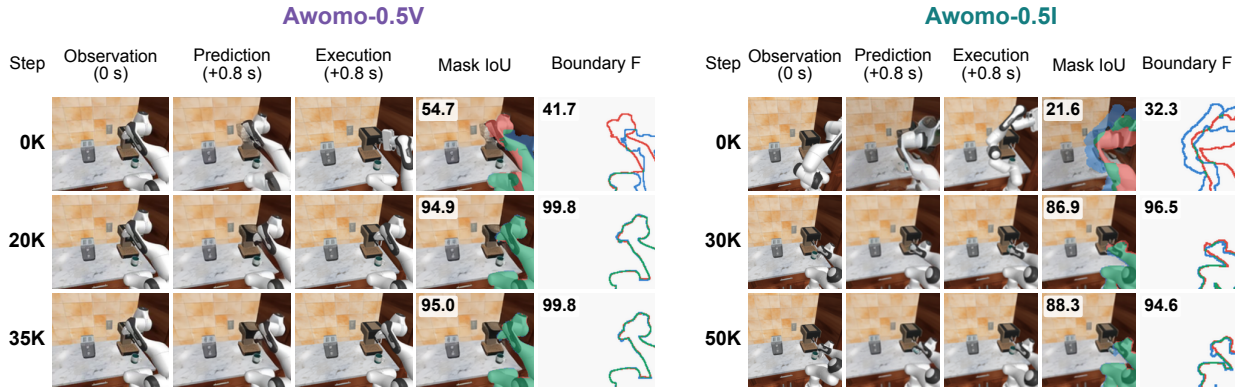


Figure 12. Predicted and executed frames on the RoboCasa365 CoffeeSetupMug task from the external camera. In the Mask IoU and Boundary F columns, red marks predicted-only robot pixels, blue marks executed-only pixels, and green marks their overlap.

Table 11. Action-space ablations.

(a) Rotation representation

Rotation	LIBERO-Plus	RoboCasa atom	Avg
3D axis-angle	88.9	79.7	84.3
6D representation	**89.0**	**80.6**	**84.8**
Δ	+0.1	+0.9	+0.5

(b) Action normalization

Normalization	LIBERO-Plus	RoboCasa atom	Avg
None	87.6	77.6	82.6
q01/q99	**89.0**	**80.6**	**84.8**
Δ	+1.4	+3.0	+2.2

last checkpoint.

5.3 Action-Space Ablations

These ablations study action-space choices at a fixed post-training budget. All comparisons use SFT step 20k, and Avg is the equal-weight mean of LIBERO-Plus and RoboCasa atom. Each table changes one action-space choice while holding the other fixed.

Rotation representation. Table 11a compares 3D axis-angle and 6D rotation with q01/q99 normalization fixed. The 6D representation is 0.5 points higher on Avg; this direction is consistent with the 6D representation used in pretraining.

Action normalization. Table 11b compares raw actions and q01/q99 clipping with 6D rotation fixed. q01/q99 improves Avg by 2.2 points, mainly through a 3.0-point gain on RoboCasa atom.

6 Real-World Post-Training

We evaluate **Awomo-0.5V** and **Awomo-0.5I** on the Piper platform, comparing them with $\pi_{0.5}$ [2] and DreamZero [4]. The experiments measure atomic manipulation under seen conditions, transfer to unseen conditions, and generalization to compositions of learned atomic skills.

6.1 Atomic Skill Benchmark

We design the Real-World Atomic Skill Benchmark to evaluate compositional generalization from atomic tasks. The design treats each individual task goal as a reusable skill unit. Seen atomic tasks assess the execution of trained skills, unseen atomic conditions assess transfer across task parameters, and compositional tasks assess whether these skills can be reused to complete multiple goals under a single instruction.

We collect 3,921 demonstrations for five atomic task types, varying block colors, spatial relations, and drawer levels within each family. Each demonstration provides supervision for an individual goal. Piper post-training uses only atomic-task demonstrations, with compositional instructions reserved for evaluation. This design tests whether learning individual operations supports repeated use of the same skill for different objects and the ordered execution of different skills. Figure 13 summarizes the training collection and evaluation design.

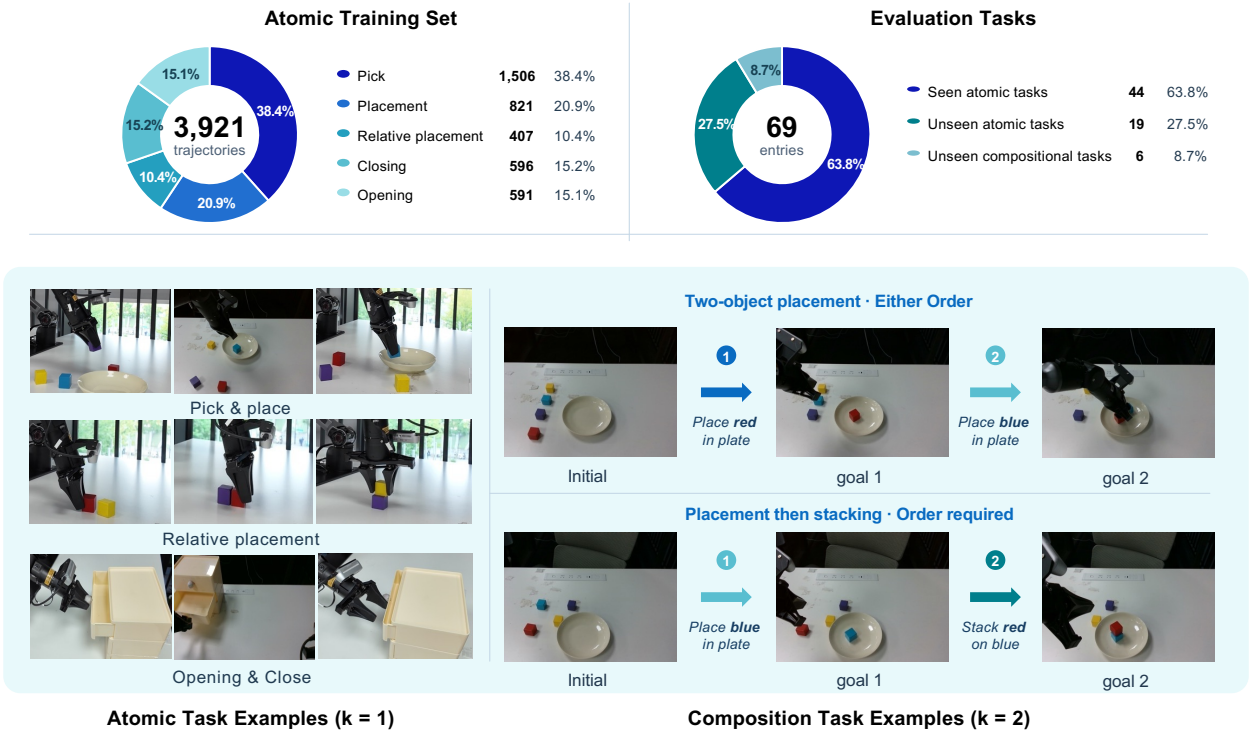


Figure 13. Real-World Atomic Skill Benchmark on Piper. Policies are trained only on single-goal demonstrations and evaluated on both atomic tasks and compositions of two trained skills. The upper panels show 3,921 atomic demonstration trajectories and 69 evaluation entries, comprising 44 seen atomic entries, 19 unseen atomic entries, and 6 compositional entries. The lower panels illustrate atomic skills and their composition. Two-object plate placement permits either order; plate placement followed by stacking requires the specified order. Composition length k counts atomic task goals, and all evaluated compositions have $k = 2$.

The training collection covers five atomic task types:

- *Pick.* The policy grasps the named object. Standalone grasping demonstrations provide supervision for object acquisition, a component shared by subsequent@ placement tasks.
- *Placement.* The policy picks up a named block and places it inside a container. Training uses blue, purple, red, and yellow blocks and a white plate. Held-out evaluation introduces green and orange blocks for white-plate placement and a black basket for placement of the four training colors.
- *Relative placement.* The policy places a target block in the instructed relation to a reference block. Training covers front, behind, right, and on-top relations, with on-top placement requiring physical stacking. Seen evaluation includes five target and reference color pairs per relation. Left placement is reserved for held-out evaluation.
- *Closing.* The policy fully closes the specified drawer. Training covers the second and third drawers from the top; first-drawer closing is reserved for held-out evaluation.
- *Opening.* The policy fully opens the specified drawer. Training covers the first and second drawers from the top; third-drawer opening is reserved for held-out evaluation. Drawer levels are numbered from top to bottom.

The benchmark contains 69 entries, each entry specifies an instruction, an object arrangement, and an initial robot pose. The 44 seen atomic entries comprise 10 single-block plate placements, 20 relative placements, 7 drawer closings, and 7 drawer openings. The 19 held-out atomic entries vary spatial relations, drawer action and level pairings, block colors, and destination containers. Together with the 6 compositional entries, they form the 25-entry unseen split. Atomic task results are aggregated by category over their color, spatial, and scene variants.

Composition length k counts the atomic tasks specified by an instruction. A single-block placement has $k = 1$, encompassing grasping, transport, and release. Four require placing two named blocks into the plate in either order, testing repeated use of the placement skill across objects. Two require plate placement followed by stacking, testing ordered integration of placement and relative-placement skills. In the latter tasks, the first operation creates the configuration required by the second: the block placed in the plate becomes the support for stacking. These two composition types assess both skill reuse across multiple objects and execution of dependent goals under a compound instruction.

Before each execution, an operator restores the object arrangement from the three-camera reference images and resets the arms to the specified starting pose. Atomic tasks have a 60-second budget, and compositional tasks have a 120-second budget. The operator ends an execution when its goal is complete. Executions that reach the time limit without completing the goal receive a failure label.

Success is scored manually as a binary outcome. Container placement requires the target block inside the specified container; relative placement requires the instructed relation; drawer manipulation requires a complete opening or closing of the target drawer. Two-object placement requires both blocks inside the plate. Ordered composition requires plate placement followed by stacking, with the completed stack inside the plate.

6.2 Experiment Setup

Models receive three RGB views from the front, left-wrist, and right-wrist cameras, together with a 14-dimensional state comprising twelve joint positions in radians and two gripper openings in meters. The workspace contains colored blocks, a white plate, a black basket, and a three-level drawer unit. Block-manipulation instructions specify the right gripper. Spatial relations follow the front-camera image: front and behind correspond to positions below and above the reference block, respectively, while left and right follow the image axes. On-top placement requires physical stacking.

We select $\pi_{0.5}$ [2] and DreamZero [4] as representative baselines for VLA models and WAMs, respectively. $\pi_{0.5}$ generates actions conditioned on visual observations and language instructions, while DreamZero jointly predicts future video and robot actions. We initialize $\pi_{0.5}$ from the official checkpoint through OpenPI, which is pretrained on more than 10,000 hours of robot data. The evaluated DreamZero baseline starts Piper adaptation from the video-pretrained Wan2.2-TI2V-5B backbone, whereas both Awomo-0.5 variants start from our pretrained checkpoints. Thus, the DreamZero baseline uses the DreamZero architecture with a Wan2.2 initialization; it is not initialized from a robot-pretrained DreamZero policy checkpoint. Both baselines are fine-tuned on the same Piper demonstration collection of 3,921 trajectories, with common atomic task definitions in the Atomic Skill Benchmark. Model-specific data conversion, optimization, and deployment settings are detailed in App. C. All policies are evaluated using the same instructions, reference scenes, execution budgets, and success criteria. These comparisons measure the performance of the complete adapted policies under a common downstream protocol; they do not isolate architecture from pretraining data and initialization.

6.3 Real-World Results

Table 12 summarizes seen conditions, unseen conditions, unseen compositional tasks, and overall performance. The unseen compositional tasks average weights the six compositional entries equally. Using the same real demonstration for fine-tuning, both Awomo-0.5 models outperform the baselines overall and on compositional generalization tasks. These results demonstrate effective adaptation of our pretrained models to real-world manipulation and their ability to reuse atomic skills to complete multi-goal instructions using only atomic-task supervision.

6.3.1 Seen Conditions Evaluation

Table 13 groups the 44 seen entries into four atomic task categories. **Awomo-0.5I** leads at 88.6%, followed by **Awomo-0.5V** at 81.8%, $\pi_{0.5}$ at 77.3%, and DreamZero at 63.6%.

Table 12. Success rate (%) on the Atomic Skill Benchmark, with success counts and evaluation totals. Composition avg. reports the mean success across the six compositional entries.

Model	Seen conditions	Unseen conditions	Unseen composition tasks avg.	Overall
Baselines				
$\pi_{0.5}$ [2]	77.3 (34/44)	**57.9 (11/19)**	0.0 (0/6)	65.2 (45/69)
DreamZero [4]	63.6 (28/44)	42.1 (8/19)	0.0 (0/6)	52.2 (36/69)
Awomo-0.5 models				
Awomo-0.5V	81.8 (36/44)	52.6 (10/19)	33.3 (2/6)	69.6 (48/69)
Awomo-0.5I	**88.6 (39/44)**	**57.9 (11/19)**	**83.3 (5/6)**	**79.7 (55/69)**

Table 13. Success rate (%) by seen atomic task category, with success counts and evaluation totals.

Model	Pick & Place to white plate	Relative placement	Close drawers 2-3	Open drawers 1-2
Baselines				
$\pi_{0.5}$ [2]	**100.0** (10/10)	50.0 (10/20)	**100.0** (7/7)	**100.0** (7/7)
DreamZero [4]	90.0 (9/10)	35.0 (7/20)	**100.0** (7/7)	71.4 (5/7)
Awomo-0.5 models				
Awomo-0.5V	**100.0** (10/10)	60.0 (12/20)	**100.0** (7/7)	**100.0** (7/7)
Awomo-0.5I	**100.0** (10/10)	**75.0** (15/20)	**100.0** (7/7)	**100.0** (7/7)

Table 14. Success rate (%) by unseen atomic task category, with success counts and evaluation totals.

Model	Left of reference	Close drawer 1	Open drawer 3	New colors to plate	To black basket
Baselines					
$\pi_{0.5}$ [2]	0.0 (0/5)	**100.0** (3/3)	0.0 (0/3)	**100.0** (4/4)	**100.0** (4/4)
DreamZero [4]	0.0 (0/5)	66.7 (2/3)	0.0 (0/3)	50.0 (2/4)	**100.0** (4/4)
Awomo-0.5 models					
Awomo-0.5V	0.0 (0/5)	**100.0** (3/3)	0.0 (0/3)	75.0 (3/4)	**100.0** (4/4)
Awomo-0.5I	0.0 (0/5)	**100.0** (3/3)	0.0 (0/3)	**100.0** (4/4)	**100.0** (4/4)

The Awomo-0.5 models' advantage over $\pi_{0.5}$ on seen conditions comes entirely from relative placement. Both Awomo-0.5 variants and $\pi_{0.5}$ achieve 100.0% on plate placement and the seen drawer tasks. Relative placement requires identifying the target and reference blocks and satisfying the instructed spatial relation. **Awomo-0.5I** and **Awomo-0.5V** improve over $\pi_{0.5}$ by 25.0 and 10.0 percentage points in this category, respectively, demonstrating more reliable execution of instructions that combine object identities with spatial constraints. **Awomo-0.5I** achieves 100.0% on front and behind placement and 80.0% on stacking, showing successful execution of both planar relations and physical block-on-block placement. Its 20.0% success rate on right placement identifies this relation as the main remaining weakness under seen conditions. Figure 14(a)-(c) illustrates successful plate placement, stacking, and drawer opening, showing the progression from the initial scene to the completed task goal.

6.3.2 Unseen Conditions Evaluation

The unseen atomic entries test five categories. Left placement uses five pairs of target and reference colors: (red, yellow), (red, blue), (blue, purple), (purple, blue), and (yellow, red). The color and container categories introduce green and orange blocks and a black basket. Container transfer uses one entry for each training color (blue, purple, red, and yellow). Table 14 aggregates the color and scene variants within each category.

The placement results reveal selective transfer of learned atomic skills. Color and container changes preserve the

goal of placing a named block inside a container. All four models achieve 100.0% on black-basket placement, and **Awomo-0.5I** maintains 100.0% on unseen colors, matching $\pi_{0.5}$. **Awomo-0.5V** achieves 75.0% on color transfer. These outcomes demonstrate successful extension of the trained placement goal to objects and destinations excluded from real adaptation.

The shared zero success rates on left placement and third-drawer opening identify a common generalization gap across the evaluated policies. Left placement requires executing a spatial relation absent from training, while third-drawer opening applies a familiar action to a level trained only for closing. All four models record successes on the corresponding seen task families, yet none completes these held-out conditions. Their failures locate the gap in extending learned operations to these specific spatial and action-level requirements. Drawer transfer further illustrates this selectivity. Both Awomo-0.5 models achieve 100.0% on first-drawer closing, although training covers that level through opening. The contrasting outcomes for first-drawer closing and third-drawer opening show asymmetric transfer between action-level pairings. Figure 14(d)-(f) illustrates successful **Awomo-0.5I** executions under changes in block color, destination container, and drawer action-level pairing.

6.3.3 Compositional Tasks Generalization

The compositional entries combine trained atomic skills under a single instruction. Four require placing two named blocks into the white plate, with either placement order permitted. Two require placing one block into the plate and then stacking a second block on it. Table 15 lists the complete prompts and scores completion of the full instruction.

Table 15. Success rate (%) on compositional instructions. Each prompt is evaluated once.

Prompt	$\pi_{0.5}$ [2]	DreamZero [4]	Awomo-0.5V	Awomo-0.5I
Two specified blocks into the white plate				
put the red and blue blocks into the white plate with the right gripper	0.0	0.0	0.0	**100.0**
put the purple and yellow blocks into the white plate with the right gripper	0.0	0.0	**100.0**	**100.0**
put the red and yellow blocks into the white plate with the right gripper	0.0	0.0	0.0	**100.0**
put the blue and purple blocks into the white plate with the right gripper	0.0	0.0	**100.0**	**100.0**
White-plate placement followed by stacking				
put the blue block into the white plate, then place the red block on top of the blue block with the right gripper	0.0	0.0	0.0	**100.0**
put the red block into the white plate, then place the yellow block on top of the red block with the right gripper	0.0	0.0	0.0	0.0
All compositional entries	0.0	0.0	33.3	**83.3**

The compositional results demonstrate reuse of atomic skills under compound instructions excluded from Piper fine-tuning. **Awomo-0.5I** and **Awomo-0.5V** achieve 83.3% and 33.3% across the six compositional entries, respectively, while both baselines achieve 0.0%. These results establish real-world transfer from training on individual task goals to executing multiple goals within a single instruction.

Two-object placement tests repeated use of the same skill for different named objects. **Awomo-0.5I** completes all evaluated object pairs, while **Awomo-0.5V** achieves 50.0%. Both Awomo-0.5 variants and $\pi_{0.5}$ achieve 100.0% on seen single-object plate placement. Their different compositional outcomes therefore distinguish the ability to extend a successful placement operation to multiple instructed objects within one execution.

Ordered placement and stacking tests the integration of two different skills with a dependency between their goals. The first operation moves the support block into the plate; the second uses that block as the stacking reference and must follow the specified order. **Awomo-0.5I** achieves 50.0% and is the only model to complete an evaluated instruction in this category. Its successful execution demonstrates coordination of container placement and relative placement under a temporal constraint. The lower success rate compared with two-object placement identifies ordered composition as a remaining challenge. Figure 14(g) and (h) illustrates these two forms of skill reuse. The two-object sequence repeats plate placement for the red and blue blocks. The ordered sequence places the blue block into the plate and then uses the resulting configuration to stack the red block on it.

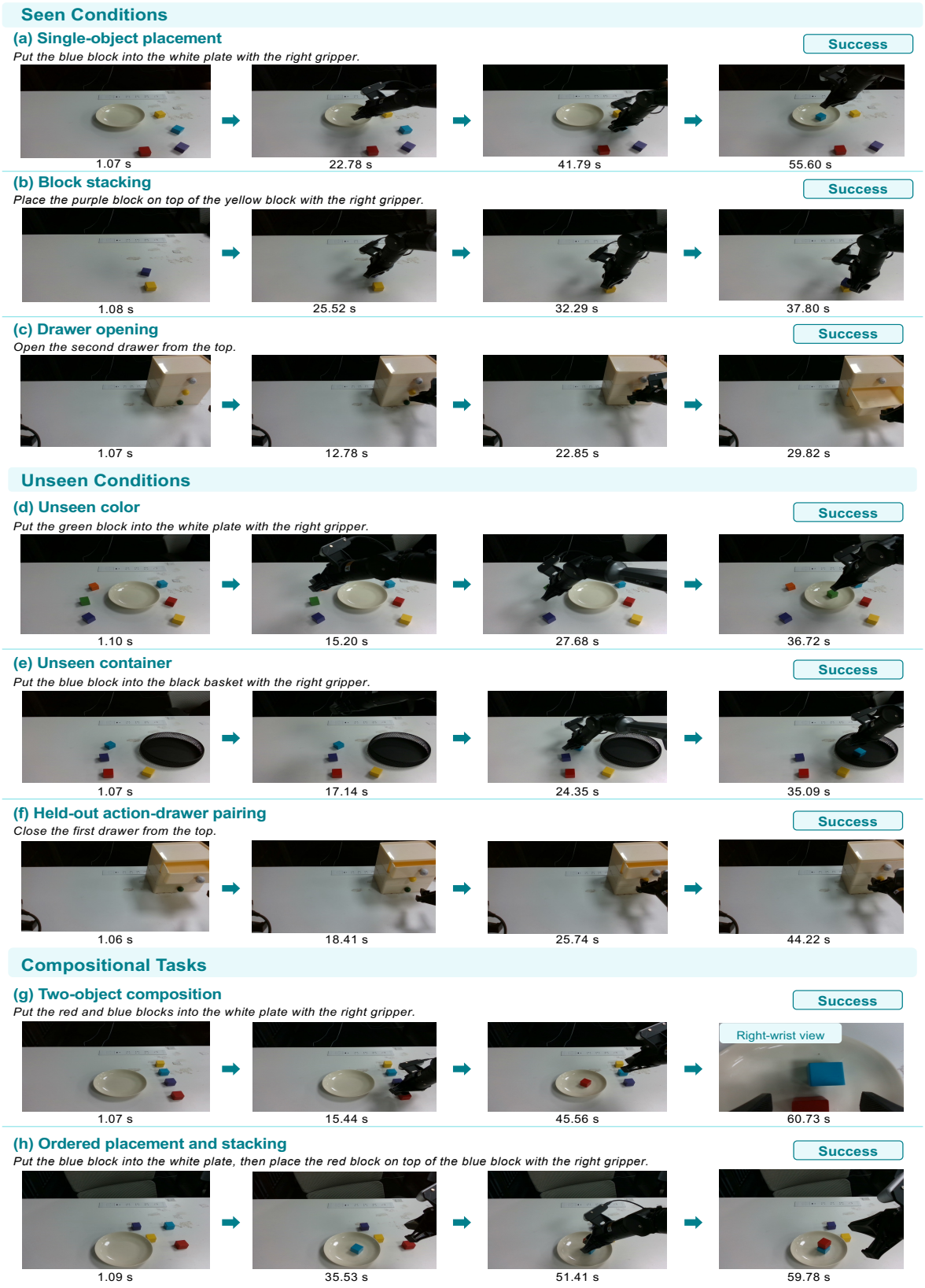


Figure 14. Representative successful Awomo-0.5I executions.

7 Related Work

Vision-language-action models. Vision-language-action models map visual observations and language instructions to robot commands, often by adapting large multimodal representations to continuous or tokenized control. π_0 and $\pi_{0.5}$ use flow-based action generation at scale, with the latter emphasizing open-world generalization [1, 2]. X-VLA studies a scalable cross-embodiment interface through soft prompts [3]. These models establish the value of large-scale semantic pretraining for action generation. Awomo-0.5 instead starts from visual-generation backbones and makes the visual future an explicit part of the control objective.

World action models. World action models jointly represent environment dynamics and robot actions. DreamZero shows that a video generator can become a zero-shot policy [4], and Cosmos Policy fine-tunes a video model for visuomotor control and planning [5]. Subsequent work explores asynchronous video-action denoising, latent dynamics, native video-action pretraining, and efficient inference [6, 53, 88, 93]. ImageWAM questions whether a single generated image can provide sufficient dynamics supervision [77]. Awomo-0.5 studies both sides of this design choice under a shared data and evaluation program: **Awomo-0.5I** integrates actions as image-like patches, while **Awomo-0.5V** retains explicit future-video generation.

Heterogeneous embodied pretraining. Large robot datasets improve diversity but differ in embodiment, observation layout, action convention, and annotation quality. Egocentric human video adds broader object and scene coverage, yet does not always provide executable robot actions. Our data pipeline addresses this mismatch with source-specific validity masks, an end-effector-centered action space, and embodiment-aware conditioning. This design complements prior efforts that scale robot experience or learn latent dynamics from egocentric video [78].

8 Limitations

Inference cost. Both variants denoise visual content together with each action chunk, using 10 to 16 denoising steps per replanning call. **Awomo-0.5V** additionally conditions on a multi-frame observation history and generates future video. On LIBERO-Plus, RoboTwin 2.0, and MolmoSpaces, **Awomo-0.5V** also applies classifier-free guidance with a scale of 5, which adds an unconditional forward pass at every denoising step (Tab. 21). This joint generation makes each control step more expensive than a policy that predicts actions alone, and it limits the replanning frequency available for reactive or high-rate control.

Generalization beyond trained skills. Skill composition works when the required atomic skills appear in the post-training data, but it extends poorly to unseen skill parameters. All evaluated policies fail the held-out left-relation and third-drawer-opening conditions, and ordered placement followed by stacking remains substantially harder than repeating a placement skill. Robustness to scene variation is also incomplete: on RoboDojo, success under randomized generalization is 14.11 percentage points lower than under standard conditions, and long-horizon tasks often make partial progress without reaching completion.

Action supervision from human data. Egocentric video contributes only to the visual objective because its actions are masked, and UMI recordings supervise only end-effector poses and a parallel-gripper aperture. Human data therefore provide no direct supervision for dexterous hands, mobile bases, or whole-body control, and the corresponding slots of the unified action space are learned from robot data alone.

9 Conclusion

Awomo-0.5 turns pretrained image and video generators into robot policies through two direct action interfaces. **Awomo-0.5I** shares image-patch projections with actions, while **Awomo-0.5V** embeds actions in a joint video-action transformer. Combined with a heterogeneous corpus of over 36 000 hours and a unified masked action space, both variants adapt to simulation and real-world manipulation tasks. The reported results favor the image-based route for LIBERO-Plus adaptation and real-world skill composition. More controlled ablations and broader deployment are required, but the present study supports visual generation as a practical backbone for scalable robot learning.

10 Contributions

Contributors are listed alphabetically by first name.

Model Training. Heng Jia, Jiahuan Zhang, Jian Zhang, Liyang Jin, Maochun Luo, Shiwen Li, Tianheng Wang, Yadong Li, Zhe Yang, Zijian Ma.

Data & Infra. Bin Zhu, Danjiao Ma, Xin Wang, Yongqiang Huang, Zhiyi Hou.

Project Leads. Jianhua Xu, Tong Zhang.

Project Sponsors. Jianhua Xu, Kaicheng Yu.

References

[1] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control, 2026. URL https://arxiv.org/abs/2410.24164.

[2] Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization, 2025. URL https://arxiv.org/abs/2504.16054.

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

[4] Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, et al. World action models are zero-shot policies. *arXiv preprint arXiv:2602.15922*, 2026.

[5] Moo Jin Kim, Yihuai Gao, Tsung-Yi Lin, Yen-Chen Lin, Yunhao Ge, Grace Lam, Percy Liang, Shuran Song, Ming-Yu Liu, Chelsea Finn, and Jinwei Gu. Cosmos policy: Fine-tuning video models for visuomotor control and planning, 2026. URL https://arxiv.org/abs/2601.16163.

[6] Jun Guo, Qiwei Li, Peiyan Li, Zilong Chen, Nan Sun, et al. Unified 4D world action modeling from video priors with asynchronous denoising. *arXiv preprint arXiv:2604.26694*, 2026. URL https://arxiv.org/abs/2604.26694.

[7] AgiBot-World-Contributors, Qingwen Bu, Jisong Cai, Li Chen, et al. Agibot world colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems. *arXiv preprint arXiv:2503.06669*, 2025. URL https://arxiv.org/abs/2503.06669.

[8] Qingwen Bu, Jisong Cai, Li Chen, et al. Agibot world colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems. *arXiv preprint arXiv:2503.06669*, 2025. URL https://arxiv.org/abs/2503.06669.

[9] AgiBot World Team. Agibot world 2026. https://huggingface.co/datasets/agibot-world/AgiBotWorld2026, 2026.

[10] Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. *arXiv preprint arXiv:2202.02005*, 2022. URL https://arxiv.org/abs/2202.02005.

[11] Homer Walke, Kevin Black, Abraham Lee, Moo Jin Kim, Max Du, Chongyi Zheng, Tony Zhao, Philippe Hansen-Estruch, Quan Vuong, Andre He, Vivek Myers, Kuan Fang, Chelsea Finn, and Sergey Levine. Bridgedata v2: A dataset for robot learning at scale. *arXiv preprint arXiv:2308.12952*, 2024. URL https://arxiv.org/abs/2308.12952.

[12] 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. URL https://arxiv.org/abs/2403.12945.

[13] Tao Jiang, Tianyuan Yuan, Yicheng Liu, Chenhao Lu, et al. Galaxea open-world dataset and g0 dual-system vla model. *arXiv preprint arXiv:2509.00576*, 2025. URL https://arxiv.org/abs/2509.00576.

[14] Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. Rdt-1b: A diffusion foundation model for bimanual manipulation. *arXiv preprint arXiv:2410.07864*, 2025. URL https://arxiv.org/abs/2410.07864.

[15] Hao-Shu Fang, Hongjie Fang, Zhenyu Tang, Jirong Liu, Chenxi Wang, Junbo Wang, Haoyi Zhu, and Cewu Lu. Rh20t: A comprehensive robotic dataset for learning diverse skills in one-shot. *arXiv preprint arXiv:2307.00595*, 2023. URL https://arxiv.org/abs/2307.00595.

[16] Adina Yakefu, Bin Xie, Chongyang Xu, et al. Robochallenge: Large-scale real-robot evaluation of embodied policies. *arXiv preprint arXiv:2510.17950*, 2025.

[17] Shihan Wu, Xuecheng Liu, Shaoxuan Xie, Pengwei Wang, et al. Robocoin: An open-sourced bimanual robotic data collection for integrated manipulation. *arXiv preprint arXiv:2511.17441*, 2025. URL https://arxiv.org/abs/2511.17441.

[18] Kun Wu, Chengkai Hou, Jiaming Liu, Zhengping Che, Xiaozhu Ju, Zhuqin Yang, Meng Li, Yinuo Zhao, Zhiyuan Xu, Guang Yang, Shichao Fan, Xinhua Wang, Fei Liao, Zhen Zhao, Guangyu Li, Zhao Jin, Lecheng Wang, Jilei Mao, Ning Liu, Pei Ren, Qiang Zhang, Yaoxu Lyu, Mengzhen Liu, Jingyang He, Yulin Luo, Zeyu Gao, Chenxuan Li, Chenyang Gu, Yankai Fu, Di Wu, Xingyu Wang, Sixiang Chen, Zhenyu Wang, Pengju An, Siyuan Qian, Shanghang Zhang, and Jian Tang. Robomind: Benchmark on multi-embodiment intelligence normative data for robot manipulation. *Robotics: Science and Systems XXI*, 2025. doi: 10.15607/RSS.2025.XXI.152. URL http://dx.doi.org/10.15607/RSS.2025.XXI.152.

[19] Chengkai Hou, Kun Wu, Jiaming Liu, Zhengping Che, et al. Robomind 2.0: A multimodal, bimanual mobile manipulation dataset for generalizable embodied intelligence. *arXiv preprint arXiv:2512.24653*, 2025. URL https://arxiv.org/abs/2512.24653.

[20] Homanga Bharadhwaj, Jay Vakil, Mohit Sharma, Abhinav Gupta, Shubham Tulsiani, and Vikash Kumar. Roboagent: Generalization and efficiency in robot manipulation via semantic augmentations and action chunking. *arXiv preprint arXiv:2309.01918*, 2023. URL https://arxiv.org/abs/2309.01918.

[21] Anthony Brohan, Yevgen Chebotar, Chelsea Finn, Karol Hausman, Alexander Herzog, Daniel Ho, Julian Ibarz, Alex Irpan, Eric Jang, Ryan Julian, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022. URL https://arxiv.org/abs/2212.06817.

[22] Yang Tian, Yuyin Yang, Yiman Xie, Zetao Cai, Xu Shi, Ning Gao, Hangxu Liu, Xuekun Jiang, Zherui Qiu, Feng Yuan, Yaping Li, Ping Wang, Junhao Cai, Jia Zeng, Hao Dong, and Jiangmiao Pang. Interndata-a1: Pioneering high-fidelity synthetic data for pre-training generalist policy. *arXiv preprint arXiv:2511.16651*, 2025. URL https://arxiv.org/abs/2511.16651.

[23] NVIDIA. Physicalai-robotics-manipulation-kitchen-demos. https://huggingface.co/datasets/nvidia/PhysicalAI-Robotics-Manipulation-Kitchen-Demos, 2025. 600 hours of human-teleoperated demonstrations across 316 kitchen manipulation tasks (55k trajectories), collected on a Franka Panda arm mounted on an Omron mobile base with three camera views... LeRobot format. No accompanying paper; cite the dataset release.

[24] 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*, page 18995-19012, 2022.

[25] Ryan Hoque, Peide Huang, David J. Yoon, Mouli Sivapurapu, and Jian Zhang. Egodex: Learning dexterous manipulation from large-scale egocentric video. 2026.

[26] Lightwheel. Egosuite-open100k: The largest fully-annotated open egocentric human dataset. https://huggingface.co/collections/LightwheelAI/egosuite-open100k, 2026.

[27] Ryan Punamiya, Simar Kareer, Zeyi Liu, Josh Citron, Ri-Zhao Qiu, Xiongyi Cai, Alexey Gavryushin, Jiaqi Chen, Davide Liconti, Lawrence Y. Zhu, et al. Egoverse: An egocentric human dataset for robot learning from around the world. *arXiv preprint arXiv:2604.07607*, 2026. URL https://arxiv.org/abs/2604.07607.

[28] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Rescaling egocentric vision: Collection, pipeline and challenges for epic-kitchens-100. *International Journal of Computer Vision*, 130:33-55, 2022.

[29] Ropedia. Xperience-10m: A large-scale egocentric multimodal dataset of human experience. https://huggingface.co/datasets/ropedia-ai/xperience-10m, 2026. Hugging Face dataset.

[30] PhysicalRSI Team, Danjiao Ma, Enhui Ma, Haohan Liu, Heng Jia, Hui Shan, Jianhua Xu, Jiahuan Zhang, Jiangdi Xu, Kaiwen Guo, Kaicheng Yu, Linwei Zhang, Liyang Jin, Maochun Luo, Pengyao Niu, Shiwen Li, Shuangyu Feng, Tong Zhang, Tianheng Wang, Xin Wang, Xiangru Huang, Yongqiang Huang, Zhaozhi Wang, and Zijian Ma. Awomo-SimDataEngine: Agentic simulation-ready world generation. *arXiv preprint arXiv:2610.02274*, 2026. URL https://arxiv.org/abs/2610.02274.

[31] Simar Kareer, Dhruv Patel, Ryan Punamiya, Pranay Mathur, Shuo Cheng, Chen Wang, Judy Hoffman, and Danfei Xu. Egomimic: Scaling imitation learning via egocentric video. page 13226-13233, 2025.

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

[33] Qixiu Li, Yu Deng, Yaobo Liang, Lin Luo, Lei Zhou, Chengtang Yao, Lingqi Zeng, Zhiyuan Feng, Huizhi Liang, Sicheng Xu, Yizhong Zhang, Xi Chen, Hao Chen, Lily Sun, Dong Chen, Jiaolong Yang, and Baining Guo. Scalable vision-language-action model pretraining for robotic manipulation with real-life human activity videos. 2026.

[34] Hao Luo, Ye Wang, Wanpeng Zhang, Sipeng Zheng, Ziheng Xi, Chaoyi Xu, Haiweng Xu, Haoqi Yuan, Chi Zhang, Yiqing Wang, Yicheng Feng, and Zongqing Lu. Being-h0.5: Scaling human-centric robot learning for cross-embodiment generalization. *arXiv preprint arXiv:2601.12993*, 2026. URL https://arxiv.org/abs/2601.12993.

[35] Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, et al. X-vla: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. *arXiv preprint arXiv:2510.10274*, 2026. URL https://arxiv.org/abs/2510.10274.

[36] Xintong Hu, Xuhong Huang, Jinyu Zhang, Yutong Yao, Yuchong Sun, Qiuyue Wang, Mingsheng Li, Sicheng Xie, Yitao Liu, Junhao Chen, Yixuan Chen, Yingming Zheng, Shuai Bai, and Tao Yu. Finevla: Fine-grained instruction alignment for steerable vision-language-action policies. *arXiv preprint arXiv:2605.27284*, 2026. URL https://arxiv.org/abs/2605.27284.

[37] Sicheng Xie, Lingchen Meng, Zijie Diao, Haidong Cao, Zhiying Du, Shuyuan Tu, Jiaqi Leng, Qiuyue Wang, Mingsheng Li, Shuai Bai, Zuxuan Wu, and Yu-Gang Jiang. Unify robot actions in camera frame. *arXiv preprint arXiv:2511.17001*, 2026. URL https://arxiv.org/abs/2511.17001.

[38] Feng Yuan, Yaping Li, Ping Wang, Junhao Cai, Jia Zeng, Hao Dong, and Jiangmiao Pang. Qwen-robotmanip

technical report: Alignment unlocks scale for robotic manipulation foundation models. *arXiv preprint arXiv:2601.18692*, 2026. URL https://arxiv.org/abs/2601.18692.

[39] Tao Jiang, Tianyuan Yuan, Yicheng Liu, Chenhao Lu, Jianning Cui, Xiao Liu, Shuiqi Cheng, Jiyang Gao, Huazhe Xu, and Hang Zhao. Galaxea open-world dataset and g0 dual-system vla model, 2025. URL https://arxiv.org/abs/2509.00576.

[40] Jinhui Ye, Ning Gao, Senqiao Yang, Jinliang Zheng, Zixuan Wang, Yuxin Chen, Pengguang Chen, Yilun Chen, Shu Liu, and Jiaya Jia. Starvla-α: Reducing complexity in vision-language-action systems, 2026. URL https://arxiv.org/abs/2604.11757.

[41] Rui Cai, Jun Guo, Xinze He, Piaopiao Jin, Jie Li, Bingxuan Lin, Futeng Liu, Wei Liu, Fei Ma, Kun Ma, Feng Qiu, Heng Qu, Yifei Su, Qiao Sun, Dong Wang, Donghao Wang, Yunhong Wang, Rujie Wu, Diyun Xiang, Yu Yang, Hangjun Ye, Yuan Zhang, and Quanyun Zhou. Xiaomi-robotics-0: An open-sourced vision-language-action model with real-time execution, 2026. URL https://arxiv.org/abs/2602.12684.

[42] StarVLA Community. Starvla: A lego-like codebase for vision-language-action model developing, 2026. URL https://arxiv.org/abs/2604.05014.

[43] Haoquan Fang, Jiafei Duan, Donovan Clay, Sam Wang, Shuo Liu, Weikai Huang, Xiang Fan, Wei-Chuan Tsai, Shirui Chen, Yi Ru Wang, Shanli Xing, Jaemin Cho, Jae Sung Park, Ainaz Eftekhar, Peter Sushko, Karen Farley, Angad Wadhwa, Cole Harrison, Winson Han, Ying-Chun Lee, Eli VanderBilt, Rose Hendrix, Suveen Ellawela, Lucas Ngoo, Joyce Chai, Zhongzheng Ren, Ali Farhadi, Dieter Fox, and Ranjay Krishna. Molmoact2: Action reasoning models for real-world deployment, 2026. URL https://arxiv.org/abs/2605.02881.

[44] Senqiao Yang, Chengyao Wang, Yuxin Chen, Zixuan Wang, Longxiang Tang, Haokun Gui, Jinhui Ye, Changsheng Lu, Xiaoyang Wu, Mingkang Zhu, Pengguang Chen, Shu Liu, Zhuotao Tian, Hengshuang Zhao, Bei Yu, and Jiaya Jia. Beyond data scaling: Representation-centric continued pre-training for vision-language-action models, 2026. URL https://arxiv.org/abs/2608.27550.

[45] Haoxiang Ma, Junhao Cai, Xiaoxu Xu, Hao Li, Yuyin Yang, Yang Tian, Jiafei Cao, Hongrui Zhu, Zherui Qiu, Zhaxizhuoma, Yuqiang Yang, Jiaqi Peng, Xueyuan Wei, Yangkun Zhu, Jiahao Jiang, Xing Gao, Hanqing Wang, Feng Yuan, Kailin Li, Xueyue Zhu, Tai Wang, Yan Ding, Jiangmiao Pang, Jia Zeng, Jingjing Zhang, Bowen Zhou, Yao Mu, Chunhua Shen, and Weinan Zhang. Internvla-a1.5: Unifying understanding, latent foresight, and action for compositional generalization, 2026. URL https://arxiv.org/abs/2607.04988.

[46] Fuhao Li, Wenxuan Song, Han Zhao, Jingbo Wang, Pengxiang Ding, Donglin Wang, Long Zeng, and Haoang Li. Spatial forcing: Implicit spatial representation alignment for vision-language-action model, 2025. URL https://arxiv.org/abs/2510.12276.

[47] He Zhang, Lingzhu Xiang, Haitao Lin, Zeyu Huang, Minghui Wang, Dingyan Zhong, Yubo Dong, Yihao Wu, Yongming Rao, Dongsheng Zhang, et al. Hy-embodied-0.5-vla: From vision-language-action models to a real-world robot learning stack, 2026. URL https://arxiv.org/abs/2606.14409.

[48] Meituan Robotics. Meituan-robotics-0: Robodojo model release. Model release, 2026. URL https://huggingface.co/wz7in/mr0.

[49] Xiaomi Robotics Team, Jun Guo, Piaopiao Jin, Jason Li, Peiyan Li, Yingyan Li, Futeng Liu, Wanli Peng, Optimus Qin, Yifei Su, Nan Sun, Qiao Sun, Runze Suo, Heyun Wang, Yunhong Wang, Rujie Wu, Caoyu Xia, Lina Zhang, Jack Zhao, Guoliang Chen, Wenlong Chen, Xinze He, Bin Li, Qing Li, Zhuorong Li, Heng Qu, Wenxuan Song, Diyun Xiang, Yifan Xie, Peiran Xu, Hangjun Ye, Wen Ye, Han Zhao, and Quanyun Zhou. Xiaomi-robotics-1: Scaling vision-language-action models with over 100k hours of real-world trajectories, 2026. URL https://arxiv.org/abs/2607.15330.

[50] Yicheng Liu, Zibin Dong, Baijun Ye, Tianyuan Yuan, Tao Jiang, et al. G0.5: One autoregressive stream for robot reasoning and action. *arXiv preprint arXiv:2608.11739*, 2026. URL https://arxiv.org/abs/2608.11739.

[51] Dexmal. Dm0.5: Next-generation vision-language-action model. Technical blog, 2026. URL https://www.dexmal.com/blog/dm0.5/index_en.html.

[52] Jiangran Lyu, Kai Liu, Xuheng Zhang, Haoran Liao, Yusen Feng, Wenxuan Zhu, Tingrui Shen, Jiayi Chen, Jiazhao Zhang, Yifei Dong, et al. Lda-1b: Scaling latent dynamics action model via universal embodied data ingestion, 2026. URL https://arxiv.org/abs/2602.12215.

[53] Tianyuan Yuan, Zibin Dong, Yicheng Liu, and Hang Zhao. Fast-wam: Do world action models need test-time future imagination?, 2026. URL https://arxiv.org/abs/2603.16666.

[54] Jisong Cai, Long Ling, Shiwei Chu, Zhongshan Liu, Jiayue Kang, Zhixuan Liang, Wenjie Xu, Yinan Mao, Weinan Zhang, Xiaokang Yang, Ru Ying, Ran Zheng, and Yao Mu. AHA-WAM: Asynchronous horizon-adaptive world-action modeling with observation-guided context routing, 2026. URL https://arxiv.org/abs/2606.09811.

[55] Angen Ye, Boyuan Wang, Chaojun Ni, Guan Huang, Guosheng Zhao, Hao Li, Hengtao Li, Jie Li, Jindi Lv, Jingyu Liu, Min Cao, Peng Li, Qiuping Deng, Wenjun Mei, Xiaofeng Wang, Xinze Chen, Xinyu Zhou, Yang Wang, Yifan Chang, Yifan Li, Yukun Zhou, Yun Ye, Zhichao Liu, and Zheng Zhu. Gigaworld-policy: An efficient action-centered world-action model, 2026. URL https://arxiv.org/abs/2603.17240.

[56] Yuran Wang, Siqiao Huang, Mingleyang Li, Chenhao Zhang, Jiaqi Liang, Weiyang Jin, Yue Chen, Xuemin Chi, Donghao Zhou, Qize Yu, Yu-Kai Wang, Yuhan Rui, Shenzhe Yao, Zhen Yuan, Zhenhao Shen, Kefei Zhu, Zijie Zhu, Ning Gao, Xiaowei Chi, Guanqi He, Shanghang Zhang, Hao Dong, Lin Shao, and Hang Zhao. Openwam: An open, modular exploration towards systematic world-action model pretraining, 2026. URL https://arxiv.org/abs/2609.07398.

[57] Tianxing Chen, Yue Chen, Zixuan Li, Junyuan Tang, Kailun Su, Haoran Lu, Weijie Wan, Baijun Chen, Songling Liu, Haowen Yan, Honghao Su, Zhiyang Dou, Kaixuan Wang, Dandan Zhang, Yunze Liu, Yan Qin, Qiwei Liang, Qiwei Wu, Zijian Lin, Wenwei Lin, Yuran Wang, Minghua He, Tianshu Wu, Ruihai Wu, Jingquan Zhou, Kai-Chong Lei, Haibao Yu, Yuanfeng Ji, Weiyang Jin, Guanyu Lin, Xiaofan Li, Qi Xiong, Renjing Xu, Zhongyu Li, Wenhao Chai, Enze Xie, Ziwei Wang, Yao Mu, Hao Dong, Wojciech Matusik, Mingyu Ding, Wenbo Ding, Ping Luo, and Masayoshi Tomizuka. RoboDojo: A unified sim-and-real benchmark for comprehensive evaluation of generalist robot manipulation policies, 2026. URL https://arxiv.org/abs/2607.04434.

[58] Soroush Nasiriany, Sepehr Nasiriany, Abhiram Maddukuri, and Yuke Zhu. RoboCasa365: A large-scale simulation framework for training and benchmarking generalist robots, 2026. URL https://arxiv.org/abs/2603.04356.

[59] yuxin101. Azero-Robotics-1. Official RoboCasa model submission, 2026. URL https://github.com/robocasa-benchmark/leaderboard/blob/main/submissions_md/Azero-Robotics-1_2026-08-17.md.

[60] NVIDIA, :, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi "Jim" Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. Gr00t n1: An open foundation model for generalist humanoid robots, 2025. URL https://arxiv.org/abs/2503.14734.

[61] Dongyoung Kim, Huiwon Jang, Myungkyu Koo, Suhyeok Jang, Taeyoung Kim, Beomjun Kim, Byungjun Yoon, Changsung Jang, Daewon Choi, Dongsu Han, Donguk Lee, Heeseung Kwon, Hojin Jeon, Jaehyun Kang, Jaekyoung Bae, Jihyuk Lee, Jimin Lee, John Won, Joonwoo Ahn, Junhyeong Park, Junyoung Sung, Kyungmin Lee, Minseong Han, Minsung Yoon, Sejune Joo, Seonil Son, Seungcheol Park, Seunggeun Cho, Seungjun Moon, Seungku Kim, Yonghoon Dong, Yongjin Cho, Youngchan Kim, Chang Hwan Kim, Dohyeon Kim, Heecheol Kim, Heewon Lee, Hensen Ahn, Hyungkyu Ryu, Hyunsoo Choi, Hyunsoo Shin, Jaeheon Jung, Jaewoo Kim, Jinwook Kim, Joochul Chang, Joonsoo Kim, Junghun Park, Jungwoo Park, Junho Cho, Junhyeok Park, Junwon Lee, Kangwook Lee, Kwanghoon Kim, Kyoungwhan Choe, Manoj Bhadu, Nayoung Oh, Sangjun Kim, Sangwoo Kim, Seunghoon Shim, Seunghyun Kim, Seungjun Lee, Seungyup Ka, Sungryol Yang, Wook Jung, Yashu Shukla, Yeonjae Lee, Yeonwoo Bae, and Jinwoo Shin. Rldx-1 technical report, 2026. URL https://arxiv.org/abs/2605.03269.

[62] Yang Zhang, Jiangyuan Zhao, Chenyou Fan, Fangzheng Yan, Tian Li, Haitong Tang, Sen Fu, Xuan'er Wu, Qizhen

Weng, Weinan Zhang, Xiu Li, Chi Zhang, Chenjia Bai, and Xuelong Li. Prts: A primitive reasoning and tasking system via contrastive representations, 2026. URL https://arxiv.org/abs/2604.27472.

[63] Phasor. Phasor-m7. Official RoboCasa model submission, 2026. URL https://github.com/robocasa-benchmark/leaderboard/blob/main/submissions_md/Phasor_m7_2026_09_15.md.

[64] Ronghan Chen, Yandan Yang, Zuojin Tang, Dongjie Huo, Tong Lin, et al. ABot-M0.5: Unified mobility-and-manipulation world action model. *arXiv preprint arXiv:2607.00678*, 2026. URL https://arxiv.org/abs/2607.00678.

[65] Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, et al. LIBERO-Plus: In-depth robustness analysis of vision-language-action models, 2025. URL https://arxiv.org/abs/2510.13626.

[66] Jisang Han, Seonghu Jeon, Jaewoo Jung, René Zurbrügg, Honggyu An, Tifanny Portela, Marco Hutter, Marc Pollefeys, Seungryong Kim, and Sunghwan Hong. Geometric action model for robot policy learning, 2026. URL https://arxiv.org/abs/2606.17046.

[67] Junjin Xiao, Dongyang Li, Yandan Yang, Shuang Zeng, Tong Lin, Xinyuan Chang, Feng Xiong, Mu Xu, Xing Wei, Zhiheng Ma, Qing Zhang, and Wei-Shi Zheng. Learning action manifold with multi-view latent priors for robotic manipulation, 2026. URL https://arxiv.org/abs/2605.11832.

[68] Zhongyu Xia, Yousen Tang, Bingqing Wei, and Yongtao Wang. 3dvla: Enhancing vision-language-action models via 3d spatial and instance understanding, 2026. URL https://arxiv.org/abs/2605.29416.

[69] Nan Sun, Yuan Zhang, Yongkun Yang, Wentao Zhao, Peiyan Li, Jun Guo, Wenxuan Song, Pengxiang Ding, Runze Suo, Yifei Su, Xin Xiao, Xinghang Li, and Huaping Liu. Revisiting embodied chain-of-thought for generalizable robot manipulation, 2026. URL https://arxiv.org/abs/2606.03784.

[70] Qing Lian, Kent Yu, and Lei Zhang. Reflective vla: In-context action consequences make vlas generalize, 2026. URL https://arxiv.org/abs/2606.25215.

[71] Linqing Zhong, Yi Liu, Yifei Wei, Ziyu Xiong, Maoqing Yao, Si Liu, and Guanghui Ren. Acot-vla: Action chain-of-thought for vision-language-action models, 2026. URL https://arxiv.org/abs/2601.11404.

[72] Dwip Dalal, Shivansh Patel, Chahit Jain, Jeonghwan Kim, Utkarsh Mishra, Yuchen Song, Alex Baratian, Hyeonjeong Ha, Heng Ji, Svetlana Lazebnik, and Unnat Jain. Generalizable vla finetuning via representation anchoring and language-action alignment, 2026. URL https://arxiv.org/abs/2607.13429.

[73] Xuan Wang, Yinan Wu, Haoran Duan, and Jungong Han. Quovla: Quotient space for vision-language-action models, 2026. URL https://arxiv.org/abs/2605.24890.

[74] Yifu Xiong, Wenhao Yu, Jiaxuan Lin, Bojun Zou, Jiahao Li, Lu Zhang, Yanyong Zhang, and Jianmin Ji. Cac-vla: Context-gated action conditioning for vision-language-action models, 2026. URL https://arxiv.org/abs/2607.04816.

[75] Qi Lv, Jianming Xing, Zhao Yang, Mingyuan Yao, Yinan Shi, Yawei Jueluo, Mike Zheng Shou, and Xiang Deng. Hermite curves as trajectory priors for vision-language-action models, 2026. URL https://arxiv.org/abs/2608.01265.

[76] Haoqi Yuan, Zhixuan Liang, Anzhe Chen, Ye Wang, Haoyang Li, et al. Qwen-RobotManip technical report: Alignment unlocks scale for robotic manipulation foundation models. *arXiv preprint arXiv:2606.17846*, 2026. URL https://arxiv.org/abs/2606.17846.

[77] Yuyang Zhang, Wenyao Zhang, Zekun Qi, He Zhang, Haitao Lin, et al. ImageWAM: Do world action models really need video generation, or just image editing? *arXiv preprint arXiv:2606.19531*, 2026. URL https://arxiv.org/abs/2606.19531.

[78] Hao Luo, Wanpeng Zhang, Yicheng Feng, Sipeng Zheng, Haiweng Xu, Chaoyi Xu, Ziheng Xi, Yuhui Fu, and Zongqing Lu. Being-h0.7: A latent world-action model from egocentric videos, 2026. URL https://arxiv.org/abs/2605.00078.

[79] Yejin Kim, Wilbert Pumacay, Omar Rayyan, Max Argus, Winson Han, et al. MolmoSpaces: A large-scale open ecosystem for robot navigation and manipulation, 2026. URL https://arxiv.org/abs/2602.11337.

[80] Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. Fast: Efficient action tokenization for vision-language-action models, 2025. URL https://arxiv.org/abs/2501.09747.

[81] Lihan Zha, Asher J. Hancock, Mingtong Zhang, Tenny Yin, Yixuan Huang, Dhruv Shah, Allen Z. Ren, and Anirudha Majumdar. Lap: Language-action pre-training enables zero-shot cross-embodiment transfer, 2026. URL https://arxiv.org/abs/2602.10556.

[82] Andy Zhai, Brae Liu, Bruno Fang, Chalse Cai, Ellie Ma, Ethan Yin, Hao Wang, Hugo Zhou, James Wang, Lights Shi, Lucy Liang, Make Wang, Qian Wang, Roy Gan, Ryan Yu, Shalfun Li, Starrick Liu, Sylas Chen, Vincent Chen, and Zach Xu. Igniting vlms toward the embodied space, 2025. URL https://arxiv.org/abs/2509.11766.

[83] NVIDIA, :, Aditi, Niket Agarwal, Arslan Ali, Jon Allen, Martin Antolini, Adeline Aubame, Alisson Azzolini, Junjie Bai, Maciej Bala, Yogesh Balaji, Josh Bapst, Aarti Basant, Mukesh Beladiya, Mohammad Qazim Bhat, Zaid Pervaiz Bhat, Dan Blick, Vanni Brighella, Han Cai, Tiffany Cai, Eric Cameracci, Jiaxin Cao, Yulong Cao, Mark Carlson, Carlos Casanova, Ting-Yun Chang, Yan Chang, Yu-Wei Chao, Prithvijit Chattopadhyay, Roshan Chaudhari, Chieh-Yun Chen, Junyu Chen, Ke Chen, Qizhi Chen, Wenkai Chen, Xiaotong Chen, Yu Chen, An-Chieh Cheng, Click Cheng, Xiu Chia, Jeana Choi, Chaeyeon Chung, Wenyan Cong, Yin Cui, Magdalena Dadela, Nalin Dadhich, Wenliang Dai, Joyjit Daw, Alperen Degirmenci, Rodrigo Vieira Del Monte, Robert Denomme, Sameer Dharur, Marco Di Lucca, Ke Ding, Wenhao Ding, Yifan Ding, Yuzhu Dong, Nicole Drumheller, Yilun Du, Aigul Dzhumamuratova, Aleksandr Efitorov, Hamid Eghbalzadeh, Naomi Eigbe, Imad El Hanafi, Hassan Eslami, Benedikt Falk, Jiaojiao Fan, Jim Fan, Amol Fasale, Sergiy Fefilatyev, Liang Feng, Francesco Ferroni, Sanja Fidler, Xiao Fu, Vikram Fugro, Prashant Gaikwad, TJ Galda, Katelyn Gao, Yihuai Gao, Wenhang Ge, Sreyan Ghosh, Arushi Goel, Vivek Goel, Akash Gokul, Rama Govindaraju, Jinwei Gu, Miguel Guerrero, Elfie Guo, Aryaman Gupta, Siddharth Gururani, Hugo Hadfield, Song Han, Ankur Handa, Zekun Hao, Mohammad Harrim, Ali Hassani, Nathan Hayes-Roth, Yufan He, Chris Helvig, Cyrus Hogg, Madison Huang, Michael Huang, Sophia Huang, Yufan Huang, Jacob Huffman, DeLesley Hutchins, Suneel Indupuru, Boris Ivanovic, Arihant Jain, Joel Jang, Ryan Ji, Yanan Jian, Dongfu Jiang, Jingyi Jin, Atharva Joshi, Nikhilesh Joshi, Pranjali Joshi, Andy Ju, Jaehun Jung, Weiwei Kang, Scott Kassekert, Jan Kautz, Ashna Khetan, Julia Kiczka, Slawek Kierat, Gwanghyun Kim, Kuno Kim, Sunny Kim, Kezhi Kong, Xin Kong, Zhifeng Kong, Tomasz Kornuta, Egor Krivov, Hui Kuang, Saurav Kumar, Chia-Wen Kuo, George Kurian, Wojciech Kutak, JF Lafleche, Himangshu Lahkar, Omar Laymoun, Jayjun Lee, Sanggil Lee, Gabriele Leone, Boyi Li, Freya Li, Jiajun Li, Jinfeng Li, Ling Li, Pengcheng Li, Shangru Li, Tingle Li, Xiaolong Li, Xuan Li, Zhaoshuo Li, Zhiqi Li, Hao Liang, Maosheng Liao, Chen-Hsuan Lin, Tsung-Yi Lin, Ming-Yu Liu, Sifei Liu, Zihan Liu, Hai Loc Lu, Xiangyu Lu, Alice Luo, Ruipu Luo, Wenjie Luo, Jiangran Lyu, Martin Ding Ma, Nic Ma, Qianli Ma, Dawid Majchrowski, Louis Marcoux, Miguel Martin, Qing Miao, Ashkan Mirzaei, Shreyas Misra, Kaichun Mo, Durra Mohsin, Hyejin Moon, Pawel Morkisz, Saeid Motiian, Kirill Motkov, Seungjun Nah, Yashraj Narang, Deepak Narayanan, Thabang Ngazimbi, Julian Ouyang, Shubham Pachori, David Page, Yatian Pang, Sehwi Park, Mahesh Patekar, Mostofa Patwary, Marco Pavone, Trung Pham, Wei Ping, Soha Pouya, Shrimai Prabhumoye, Varun Praveen, Delin Qu, Hesam Rabeti, Morteza Ramezanali, Marilyn Reeb, Xuanchi Ren, Kristen Rumley, Wojciech Rymer, Jun Saito, Yeongho Seol, John Shao, Piyush Shekdar, Tianwei Shen, Humphrey Shi, Min Shi, Stella Shi, Kevin Shih, Mohammad Shoeybi, Mateusz Sieniawski, Shuran Song, Alexander Sotelo, Amir Sotoodeh, Sunil Srinivasa, Vignesh Srinivasakumar, Bartosz Stefaniak, Rahul Heinrich Steiger, Shangkun Sun, Jiaxiang Tang, Shitao Tang, Yangyang Tang, Yue Tang, Tolou Tavakkoli, Kayley Ting, Krzysztof Tomala, Wei-Cheng Tseng, Jibin Varghese, Sergei Vasilev, Thomas Volk, Raju Wagwani, Roger Waleffe, Andrew Z. Wang, Boxiang Wang, Haoxiang Wang, Qiao Wang, Shihao Wang, Shijie Wang, Ting-Chun Wang, Yan Wang, Yu Wang, Rohit Watve, David Wehr, Fangyin Wei, Xinshuo Weng, Jay Zhangjie Wu, Kedi Wu, Hongchi Xia, Summer Xiao, Tianjun Xiao, Kevin Xie, Daguang Xu, Jiashu Xu, Mengyao Xu, Ruqing Xu, Xingqian Xu, Yao Xu, Dinghao Yang, Dong Yang, Hans Yang, Xiaodong Yang, Xuning Yang, Yichu Yang, Yurong You, Zhiding Yu, Hao Yuan, Simon Yuen, Xiaohui Zeng, Pengcuo Zeren, Cindy Zha, Haotian Zhang, Jenny Zhang, Jing Zhang, Liangkai Zhang, Paris Zhang, Shun Zhang, Xuanmeng Zhang, Zhizheng Zhang, Ann Zhao, Yilin Zhao, Yuliya Zhautouskaya, Charles Zhou, Fengzhe Zhou, Shilin Zhu, Yuke Zhu, Dima Zhylko, and Artur Zolkowski. Cosmos 3: Omnimodal world models for physical ai, 2026. URL https://arxiv.org/abs/2606.02800.

[84] Wei Wu, Fan Lu, Yunnan Wang, Shuai Yang, Shi Liu, Fangjing Wang, Qian Zhu, He Sun, Yong Wang, Shuailei Ma, Yiyu Ren, Kejia Zhang, Hui Yu, Jingmei Zhao, Shuai Zhou, Zhenqi Qiu, Houlong Xiong, Ziyu Wang, Zechen

Wang, Ran Cheng, Yong-Lu Li, Yongtao Huang, Xing Zhu, Yujun Shen, and Kecheng Zheng. A pragmatic vla foundation model, 2026. URL https://arxiv.org/abs/2601.18692.

[85] Haoyang Li, Guanlin Li, Youhe Feng, Chen Zhao, Zhuoran Wang, et al. Training vision-language-action models with dense embodied chain-of-thought supervision. *arXiv preprint arXiv:2606.30552*, 2026. URL https://arxiv.org/abs/2606.30552.

[86] Xuewu Lin, Tianwei Lin, Yun Du, Hongyu Xie, Yiwei Jin, et al. HoloBrain-0 technical report. *arXiv preprint arXiv:2602.12062*, 2026. URL https://arxiv.org/abs/2602.12062.

[87] Jialei Chen, Kai Wang, Kang Chen, Shuaihang Chen, Feng Gao, et al. LaWAM: Latent world action models for efficient dynamics-aware robot policies. *arXiv preprint arXiv:2606.15768*, 2026. URL https://arxiv.org/abs/2606.15768.

[88] Lin Li, Qihang Zhang, Yiming Luo, Shuai Yang, Ruilin Wang, et al. Causal world modeling for robot control. *arXiv preprint arXiv:2601.21998*, 2026. URL https://arxiv.org/abs/2601.21998.

[89] Weiheng Zhao, Haoyi Jiang, Xin Shi, Liu Liu, Fan Huang, et al. Faster-WAM: Efficient inference-time future conditioning for robust world action models. *arXiv preprint arXiv:2608.04404*, 2026. URL https://arxiv.org/abs/2608.04404.

[90] Bikang Pan, Fan Liu, Haotao Lu, Jingya Wang, and Ye Shi. SelfWAM: A self-grounded unified world action model for fast robot control. *arXiv preprint arXiv:2608.00725*, 2026. URL https://arxiv.org/abs/2608.00725.

[91] Jianjun Zhang, Jian Zhu, Taiyi Su, Chong Ma, Zitai Huang, et al. Learning 4D geometric priors for inference-efficient world action models. *arXiv preprint arXiv:2607.05468*, 2026. URL https://arxiv.org/abs/2607.05468.

[92] Liaoyuan Fan, Zetian Xu, Chen Cao, Wenyao Zhang, Mingqi Yuan, et al. AIM: Intent-aware unified world action modeling with spatial value maps. *arXiv preprint arXiv:2604.11135*, 2026. URL https://arxiv.org/abs/2604.11135.

[93] Qihang Zhang, Lin Li, Luyao Zhang, Shuai Yang, Yiming Luo, et al. Native video-action pretraining for generalizable robot control. *arXiv preprint arXiv:2607.08639*, 2026. URL https://arxiv.org/abs/2607.08639.

[94] Gangwei Xu, Qihang Zhang, Jiaming Zhou, Xing Zhu, Yujun Shen, et al. Next Forcing: Causal world modeling with multi-chunk prediction. *arXiv preprint arXiv:2606.11187*, 2026. URL https://arxiv.org/abs/2606.11187.

[95] MotuBrain Team, Chendong Xiang, Fan Bao, Haitian Liu, Hengkai Tan, et al. MotuBrain: An advanced world action model for robot control. *arXiv preprint arXiv:2604.27792*, 2026. URL https://arxiv.org/abs/2604.27792.

[96] Ganlin Yang, Zhangzheng Tu, Yuqiang Yang, Sitong Mao, Junyi Dong, Tianxing Chen, Jiaqi Peng, Jing Xiong, Jiafei Cao, Jifeng Dai, Wengang Zhou, Yao Mu, and Tai Wang. Eventvla: Event-driven visual evidence memory for long-horizon vision-language-action policies, 2026. URL https://arxiv.org/abs/2606.20092.

[97] Yandan Yang, Shuang Zeng, Tong Lin, Xinyuan Chang, Dekang Qi, Junjin Xiao, Haoyun Liu, Ronghan Chen, Yuzhi Chen, Dongjie Huo, Feng Xiong, Xing Wei, Zhiheng Ma, and Mu Xu. Abot-m0: Vla foundation model for robotic manipulation with action manifold learning, 2026. URL https://arxiv.org/abs/2602.11236.

[98] Lishan Yang, Wenxuan Song, Xi Wang, Pingyue Sheng, Zheng Fang, Ziyang Zhou, Junjie He, Haodong Yan, Jiayi Chen, Nan Sun, Qiao Sun, Pengwei Wang, Lingqiao Liu, Yan Wang, Yuxiang Gao, Feras Dayoub, and Haoang Li. 4d-wam: Infusing spatiotemporal awareness into world action models through trajectory fields, 2026. URL https://arxiv.org/abs/2608.08023.

[99] GigaBrain Team, Angen Ye, Axiang Sun, Can Jin, Chenxi Cheng, Chong Shi, Dengke Shang, Dingqian Zhang, Guan Huang, Guangqiang Wang, Guangqing Ding, Guo Li, Hangcong Li, Hengyu Zhong, Hongtao Lu, Jianbo Qin, Jiming Mao, Jing Zhu, Jindi Lv, Jingzhi Cui, Junjie Xie, Junyi Bao, Kai Liu, Lei Yuan, Limin Long, Lv Feng, Mingming Yu, Peng Li, Pengfei Yi, Qi Li, Qianli Zhang, Qingfang Li, Qitang Hu, Rui Zhang, Shaoyan Sun, Shibo Sun, Shiying Duan, Tenghui Chen, Tianze Liu, Weijie Ke, Wenyao Xue, Xiaofeng Wang, Xiaoyu Tian, Xinyu Liu, Xinze Chen, Yang Wang, Yankai Wang, Yejun Zeng, Yifan Li, Yifei Nie, Yilong Li, Yilong Liu,

Yongchao Feng, Yumeng Wang, Yun Ye, Zhichao Liu, Ziheng He, Zonghai Yang, and Zheng Zhu. Gigabrain-0.7: Scaling embodied foundation models to emergent capabilities with a three-system architecture, 2026. URL https://arxiv.org/abs/2608.15875.

[100] Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, et al. RoboTwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation, 2025. URL https://arxiv.org/abs/2506.18088.

A Pretraining Data Details

A.1 UMI Action Solving

UMI recordings are not robot trajectories; the end-effector state and the action stream must be solved from multi-camera video, 6D pose reconstructions, and raw gripper encoder readings. The constituent streams (camera frames at $\geq 30\,\mathrm{Hz}$, pose at $\geq 30\,\mathrm{Hz}$, and the gripper encoder at $\geq 50\,\mathrm{Hz}$) are first resampled onto a common timestamp grid, interpolating pose and encoder readings onto the visual frames only within intervals marked valid, so that tracking losses propagate as masked intervals rather than fabricated measurements. Each gripper pose is transformed into a common reference frame c using the calibrated extrinsics, and the encoder distance is mapped to a normalized closure $g_t = \mathrm{clip}\big((d_{\mathrm{open}} - d_t)/(d_{\mathrm{open}} - d_{\mathrm{closed}}),\, 0,\, 1\big)$, so that 0 denotes open and 1 denotes closed as in the unified action space, yielding the state.

A.2 UMI Retargeting

Actions follow the same absolute parameterization as robot data: each gripper contributes its end-effector position (3), 6D rotation (6), and gripper closure (1) in the reference frame. The two grippers populate the corresponding slots of the two mirrored 29-D arm blocks, so a single bimanual recording fills both arms; slots the rig cannot supply (dexterous hand, base, torso, head) are excluded by the validity mask.

A.3 Signal Thresholds

The signal validation checks of Sec. 3.2.3 use the following criteria.

1. **Tracking validity.** An episode is rejected when the logged end-effector state does not follow the action stream, judged by an amplitude ratio of at least $3\times$ together with a per-axis correlation below 0.5, or when more than $2\,\%$ of frames are misaligned between the video and signal streams.
2. **Edge inactivity.** Leading and trailing stretches in which the state barely moves are trimmed. Thresholds are set from the $p99.5$ of single-frame deltas for the embodiment in question, and a stretch counts as inactive once mean end-effector translation and geodesic rotation fall below $2\,\mathrm{cm/s}$ and $5°$/s. Pauses inside an episode are left alone, because excising them would concatenate frames that were never adjacent in time. Discontinuities such as jerk and spike outliers are caught separately with robust median-MAD screening.
3. **Motion corroboration.** Reported state motion is checked against the imagery. A frame that reports movement while every camera view is visually still is sensor jitter and is trimmed; conversely, an end effector that visibly moves under a frozen camera indicates a capture fault, and the episode is removed.
4. **Episode rejection.** An episode that surrenders more than $70\,\%$ of its frames to the stages above, or whose video is frozen for at least $90\,\%$ of its duration, is discarded.
5. **Action magnitude.** In robot data, startup and reset routines near episode boundaries emit sustained large actions, such as rapid arm retraction, that stay temporally smooth and state-consistent and so escape the earlier checks. Because such excursions skew the quantile statistics used for normalization, samples whose single-step end-effector translation exceeds $0.2\,\mathrm{m}$ or whose single-step rotation exceeds $0.5\,\mathrm{rad}$ are dropped from every source with end-effector actions.

B Simulation Post-Training Details

B.1 RoboDojo

We train the reported policy for one epoch with a global batch size of 128.

Table 16. RoboDojo simulation task and episode counts per seed.

	Generalization	Precision	Long-Horizon	Memory	Open	Total
Tasks	12	8	8	6	8	42
Episodes	600	400	400	300	400	2,100
Score weight	20%	20%	20%	20%	20%	100%

Table 16 lists the tasks and episodes of the five evaluation dimensions per seed. Generalization uses 25 standard and 25 randomized episodes per task. The paper evaluates most policies with three training seeds, totaling 6,300 episodes per policy. Overall score and success rate are computed separately as equal-weight averages across the five dimensions:

$$\text{Overall}(m) = \tfrac{1}{5}\left(m_{\text{Generalization}} + m_{\text{Precision}} + m_{\text{Long-Horizon}} + m_{\text{Memory}} + m_{\text{Open}}\right),$$

where m denotes either score or success rate.

B.2 RoboCasa365

We fine-tune **Awomo-0.5I** for 60,000 steps and **Awomo-0.5V** for 40,000 steps, both with a global batch size of 512 and initialization from their respective pretrained checkpoints. Adaptation uses Human300 demonstrations and does not use Target50 demonstrations.

Table 17. RoboCasa365 multi-task evaluation task and episode counts.

	Atomic (seen)	Composite (seen)	Composite (unseen)	Total
Tasks	18	16	16	50
Episodes	540	480	480	1,500
Score weight	36%	32%	32%	100%

Table 17 lists the tasks and episodes of the three task groups. The benchmark protocol evaluates 30 episodes per task. Overall averages task success rates across the 50-task suite:

$$\text{Overall} = \frac{18\,\text{SR}_{\text{Atomic-seen}} + 16\,\text{SR}_{\text{Composite-seen}} + 16\,\text{SR}_{\text{Composite-unseen}}}{50}.$$

B.3 LIBERO-Plus

We fine-tune both models for 10 epochs with a global batch size of 256, starting from their respective pretrained checkpoints and using the LIBERO-Plus training set together with our augmented demonstrations.

Table 18. LIBERO-Plus episode counts.

	Camera	Robot	Language	Light	Background	Noise	Layout	Total
Tasks	1,599	1,550	1,537	1,142	1,076	1,601	1,525	10,030
Episodes	1,599	1,550	1,537	1,142	1,076	1,601	1,525	10,030
Score weight	14.3%	14.3%	14.3%	14.3%	14.3%	14.3%	14.3%	100%

Table 18 lists the perturbed task instances of the seven perturbation dimensions. Each task instance is evaluated in one episode. Overall is the arithmetic mean of the seven dimension success rates, with equal weights regardless of episode counts.

B.4 MolmoSpaces

We fine-tune both models for three epochs with a global batch size of 512, starting from their respective pretrained checkpoints.

Table 19. MolmoSpaces task and episode counts for Awomo-0.5V.

	Pick-v1.1	Pick & Place-v1	Open-v1	Close-v1	Total
Tasks	1,000	1,000	1,000	915	3,915
Episodes	1,000	1,000	1,000	915	3,915
Score weight	25%	25%	25%	25%	100%

Table 19 lists the tasks and episodes of the four sub-benchmarks. Each task instance is evaluated in one episode. Overall is the macro-average of the four sub-benchmark oracle success rates, with equal weights regardless of episode counts:

$$\text{Overall} = \tfrac{1}{4}\left(\text{SR}_{\text{Pick}} + \text{SR}_{\text{Place}} + \text{SR}_{\text{Open}} + \text{SR}_{\text{Close}}\right).$$

B.5 RoboTwin 2.0

For the full-data setting, we train the reported policy for five epochs with a global batch size of 256.

Table 20. RoboTwin 2.0 task and episode counts per evaluation run.

	Clean	Randomized	Total
Tasks	50	50	50
Episodes	5,000	5,000	10,000
Score weight	50%	50%	100%

Table 20 lists the tasks and episodes per evaluation run. Both settings evaluate the same 50 tasks. The full-data protocol trains on clean and randomized demonstrations; clean-to-random trains on clean demonstrations only. For both protocols, Average is

$$\text{Average} = \tfrac{1}{2}\left(\text{SR}_{\text{Clean}} + \text{SR}_{\text{Randomized}}\right).$$

B.6 Inference Configuration

Tab. 21 summarizes the inference settings used for the reported downstream evaluations.

Table 21. Inference configurations. Control rate is the frequency of predicted actions.

Model	Benchmark	Denoising steps	Predicted actions	Replanning steps	Control rate (Hz)	CFG scale
Awomo-0.5I	LIBERO-Plus	10	16	16	20	1
	RoboTwin 2.0	10	16	16	50	1
	RoboDojo	10	16	8	25	1
	RoboCasa365	16	16	16	20	1
	MolmoSpaces	16	24	24	15	1
	Real-world	16	24	24	15	1
Awomo-0.5V	LIBERO-Plus	16	24	24	20	5
	RoboTwin 2.0	16	16	8	50	5
	RoboDojo	16	20	10	25	1
	RoboCasa365	16	24	12	15	1
	MolmoSpaces	16	24	24	15	5
	Real-world	16	24	24	15	1

C Real-World Post-Training Details

C.1 Training

The $\pi_{0.5}$ [2], DreamZero [4], and Awomo-0.5 training pipelines use the same 3,921 trajectories. The five training task families receive equal sampling weights.

***Awomo-0.5I*.** **Awomo-0.5I** is fine-tuned from the pretrained checkpoint for 46,500 optimizer updates using 16 NVIDIA H800 GPUs, a global batch size of 128, and two gradient-accumulation steps. We use AdamW with a peak learning rate of 2.83×10^{-5}, 50 warmup steps, cosine decay to 2.83×10^{-6}, weight decay of 10^{-5}, and gradient clipping at 1.0. Data augmentation is applied with probability 0.5, and EMA weights with decay 0.999 are used for evaluation.

***Awomo-0.5V* and *DreamZero*.** Both use a global batch size of 128, a 24-step action horizon, 33-frame training sequences, and data augmentation with probability 0.5. Both use 24,800 adaptation updates over 800 epochs and 16 NVIDIA H200 GPUs. The learning rate follows cosine decay from 2.83×10^{-5} to 2.83×10^{-6} after 50 warmup updates. Training image sequences are sampled at 5 Hz and robot states and actions at 15 Hz.

$\pi_{0.5}$. The $\pi_{0.5}$ adaptation runs for 18,006 updates with a per-process batch size of 24. Its learning rate follows cosine decay from 2.5×10^{-5} to 2.5×10^{-6} after 901 warmup updates. The training pipeline uses robot data sampled at 30 Hz and applies min-max normalization to states and actions.

C.2 Inference

All policies are deployed on an NVIDIA RTX 5090 host. **Awomo-0.5I**, **Awomo-0.5V**, and DreamZero use three-camera mosaics of 320×288 pixels. **Awomo-0.5I** conditions on the current mosaic, predicts a 24-step action chunk, and executes the full chunk before replanning. **Awomo-0.5V** conditions on a 25-frame observation history sampled at 5 Hz, predicts 24 actions, and executes the full chunk before replanning. $\pi_{0.5}$ receives the three current camera views separately, each resized to 224×224 pixels, together with the current robot state. It predicts a 50-step action chunk and executes the first 20 actions before acquiring new observations and replanning. Table 21 lists the denoising steps and control rates of both Awomo-0.5 models.