# ROBOWORLD100: LEARNING AND EVALUATING ROBOT WORLD MODELS FROM DIVERSE PLAY

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Action-conditioned world models promise learned simulators for robot policy evaluation and reinforcement learning, but must capture physical interactions beyond the execution patterns of individual tasks. **Task-relevant play** broadens interaction coverage beyond prescribed solutions, yet existing play datasets remain limited in task, robot platform, and observation diversity. We introduce **RoboWorld100**, a framework and benchmark integrating human-play data collection with world-model training and evaluation. It covers ten categories of manipulation and physical interaction and over 100 task–platform settings across real and simulated environments, spanning diverse robots, collection schemes, and camera configurations. Beyond visual quality, our evaluation measures **action consistency** and **task-progress consistency** to assess whether predictions faithfully reflect supplied actions and execution outcomes. Closed-loop policy evaluation and in-model reinforcement learning further examine whether learned dynamics support policy optimization and whether improvements transfer to real robots. Project website: https://roboworld100.github.io/

## 1 INTRODUCTION

Action-conditioned world models must reliably predict the consequences of robot actions to serve as simulators for policy evaluation and learning. A central obstacle is interaction coverage: demonstrations and task-driven policy rollouts concentrate on behaviors used to execute particular tasks, leaving other relevant action–outcome relationships sparsely observed (Guo et al., 2026; Yin et al., 2026). Additional failures and recoveries do not, by themselves, resolve this limitation. For example, repeated attempts by a pick-and-place policy need not systematically cover pushing the same objects. Reliable simulation calls for exploration beyond a task's usual execution patterns.

We focus on *task-relevant play*: human-guided exploration of meaningful interactions within a task environment, with freedom to vary and combine actions beyond a prescribed solution. Operators use their knowledge of objects and interaction goals to select informative behaviors without requiring a trained collection policy for each setting. Failures, recoveries, and alternative strategies arise within this broader exploration. Prior work demonstrates the value of play for robot learning and world-model adaptation (Lynch et al., 2020; Chandra et al., 2025; Yin et al., 2026); extending its coverage across embodiments, interaction requirements, and observations remains an important data-design challenge.

Evaluation presents a second obstacle: visual resemblance alone does not establish faithful action consequences. Image and video similarity can overlook errors in motion or task progression (Ge et al., 2024; Li et al., 2025a). Recent evaluations consequently examine physical reliability and functional utility (Shang et al., 2026a; Jiang et al., 2026a). Across diverse interaction environments, evaluation should also localize model weaknesses and test whether predictive accuracy supports policy assessment and learning.

We introduce **RoboWorld100** (Figure 1), a dataset and evaluation framework spanning 104 task–platform settings, ten interaction categories, nine collection platforms, and six camera-view types. It is the first task-relevant human-play dataset that combines real and simulated interactions to support systematic adaptation and evaluation of action-conditioned world models across diverse embodiments, interaction categories, and camera configurations. We organize this coverage into *micro-worlds*, each defined by its embodiment, observation–action interface, and scene and object config-



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

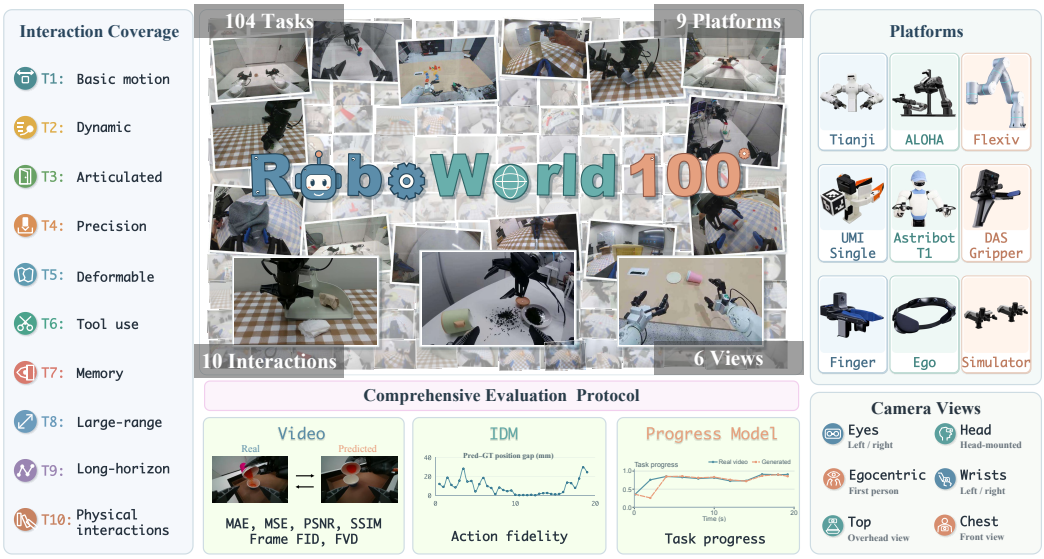



Figure 1: **RoboWorld100 Overview.** A framework integrating task-relevant human-play collection with robot world-model training and evaluation, assessing visual quality, action consistency, and task-progress consistency.

urations. Their requirements range from basic manipulation to precision, deformation, memory, and long-horizon interaction. With reliable general-purpose simulation still an open goal, these bounded environments provide a measurable route forward: adapt pretrained models, identify their limits, and progressively address broader and more demanding interactions.

Our evaluation combines visual quality with action consistency and task-progress consistency, and tracks their evolution during continuous rollouts to assess long-horizon stability. The interaction taxonomy supports tracing prediction errors to specific settings. Experiments show that task-relevant play improves prediction quality and reveal model weaknesses that visual metrics alone can miss. Closed-loop policy evaluation and in-model reinforcement learning further support the use of learned simulators for policy assessment and improvement, with gains transferring to real robots in the tested settings.

Our contributions are summarized as follow:

- A task-relevant human-play dataset covering 104 task–platform settings, organized by interaction requirements to support world-model adaptation across diverse embodiments and observations.
- A paired evaluation protocol that measures visual quality, action consistency, and task-progress consistency, and tracks their evolution during continuous rollouts to assess long-horizon stability.
- Empirical evidence for the value of play under equal episode budgets, together with evaluations of learned simulators for policy assessment and reinforcement learning with real-robot transfer.

## 2 Related Work

**Action-Conditioned World Models** Action-conditioned world models are forward dynamics models that predict future states from observation histories and control inputs (Ai et al., 2025). Prior work has learned the consequences of actions in visual-feature and pixel spaces to support planning and control (Ebert et al., 2018; Zhou et al., 2024a; Assran et al., 2025). As data volumes and model performance have increased, their applications have expanded from prediction and planning to policy evaluation (GigaWorld Team et al., 2026; Li et al., 2025b; Tseng et al., 2025; Jeon et al., 2026), the generation of policy training data (Guo et al., 2026), and reinforcement learning of policies within world models to reduce the high cost of real-robot interaction (Jiang et al., 2026b; Zhu et al., 2026; Sharma et al., 2026; Qiu et al., 2026). However, action-conditioned world models

Table 1: Comparison of robot manipulation datasets.

| Dataset | Play | Embodiments | Tasks | Trajectories | Domain | Collection Method |
|---|---|---|---|---|---|---|
| DROID (Khazatsky et al., 2024) | × | 1 | 86 | 76k | Real | Human Teleoperation |
| BridgeData V2 (Walke et al., 2023) | × | 1 | N/A | 60.096k | Real | Human+auto |
| RoboMIND (Wu et al., 2024) | × | 4 | 479 | 112k | Real+Sim | Human Teleoperation |
| RoboCOIN (Wu et al., 2025) | × | 15 | 421 | 180k+ | Real | Human Teleoperation |
| AGIBOT WORLD 2026 | ✓ | 1 | 10[†] | N/A | Real | Human Teleoperation |
| CMU PlayFusion (Chen et al., 2023) | ✓ | 1 | N/A | 0.576k | Real | Human Teleoperation |
| DiWA (Chandra et al., 2025) | ✓ | 1 | 3 | 1.496k | Real | Human Teleoperation |
| PlayWorld (Yin et al., 2026) | ✓ | 1 | N/A | 6.868k[*] | Real | Autonomous |
| **RoboWorld100 (Ours)** | ✓ | **9** | **104** | **13k** | **Real+Sim** | **Human collection** |

[†]The AGIBOT row covers the Rich Interaction subset: 10 distinct released task IDs; its total trajectory count remains unverified. DiWA includes 3 released task labels and 1,496 episodes (1,252 train + 244 validation). [*]PlayWorld reports 6,868 episodes in the public preview (6,182 train + 686 validation). CMU PlayFusion uses the public TFDS release (576 episodes).

trained on data with narrow coverage are prone to out-of-distribution prediction errors. These errors may be exploited during policy optimization, preventing improvements within the model from translating into real-world gains (Yin et al., 2026). Supplementing the data with failure cases and contact interactions has been shown to be an important means of correcting such prediction biases (Guo et al., 2026; Yin et al., 2026). RoboWorld100 supports the adaptation of existing models to diverse interaction environments and examines both their predictive consistency and their utility for policy optimization.

**Evaluation Frameworks for World Models** Some video prediction evaluations primarily focus on the visual quality of generated observations and their similarity to real data (Yan et al., 2021; Voleti et al., 2022; Gao et al., 2022). Recently, evaluation has expanded to dimensions such as motion consistency, physical commonsense, and task semantics (Huang et al., 2024; Bansal et al., 2025; Li et al., 2025a; Yue et al., 2025). Studies also assess the functional utility of world models through planning, policy evaluation, and learning within the model (Qin et al., 2024; Zhang et al., 2026; Fan et al., 2026; Shang et al., 2026a;b), or further examine action following, physical reliability, and behavioral executability (Yang et al., 2026; Chen et al., 2026; Jiang et al., 2026a; Liu et al., 2026b). Building on this broader view of evaluation, RoboWorld100 compares generated and reference executions through visual quality, action consistency, and task-progress consistency, and connects these measurements to downstream simulator applications.

**Enhancing World Model Performance with Play Data** Play broadens the data distribution through unstructured exploration and has been used to improve policy performance (Lynch et al., 2020; Lynch & Sermanet, 2020; Cui et al., 2022; Chen et al., 2023; Rosete-Beas et al., 2023; Mees et al., 2022) and train world models (Sekar et al., 2020; Nematollahi et al., 2025; Chandra et al., 2025). However, existing work on play data remains limited in collection scope and scalability. Autonomous data collection relies on existing policies and their corresponding execution interfaces, with interaction coverage constrained by the capabilities of the data collection policy (Zhou et al., 2024b; Yin et al., 2026). Studies using human play for world models have primarily been validated on specific robots and in particular manipulation environments, with a focus on skill learning and policy adaptation (Nematollahi et al., 2025; Chandra et al., 2025). In contrast, we establish a diverse human play data collection framework, broadening the coverage of complex physical interactions to provide a more substantial data foundation for improving the dynamics prediction accuracy of world models. Table 1 compares the collection scope of RoboWorld100 with that of existing robot manipulation datasets.

# 3 METHOD

## 3.1 DATA COLLECTION, PROCESSING, AND TRAINING

**Data collection.** We design micro-worlds in real and simulated environments around ten interaction categories (Appendix A), selecting objects and scene configurations that expose the corre-

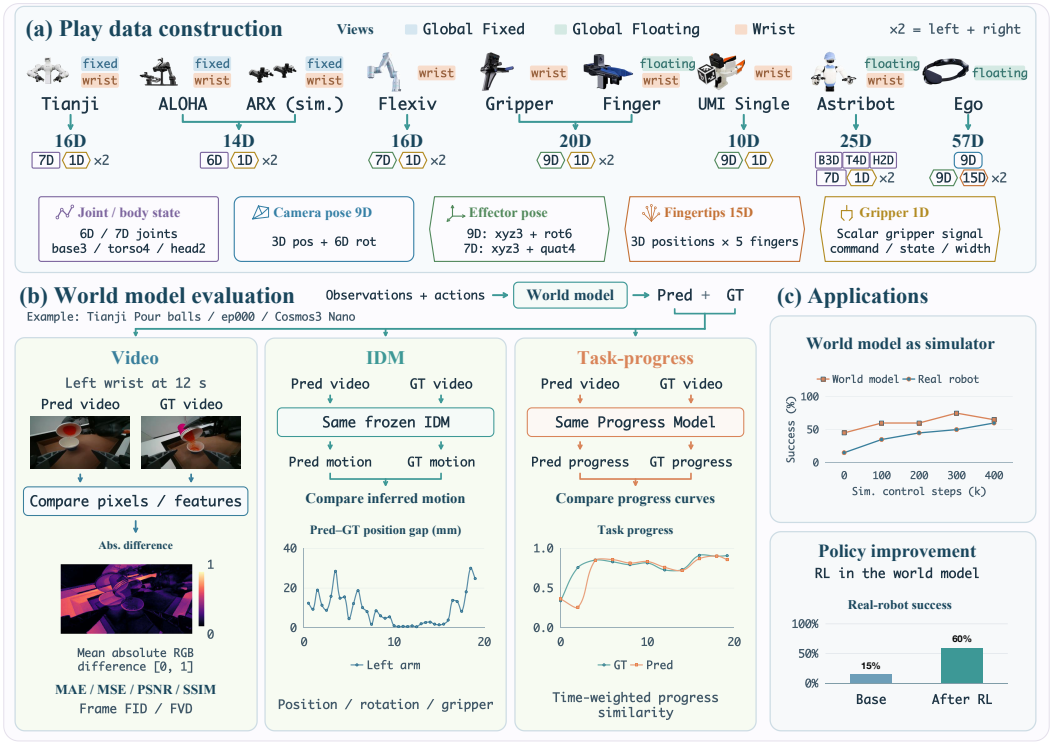



Figure 2: **RoboWorld100 pipeline.** (a) Collection platforms, camera views, and action representations. (b) Action-conditioned generation and evaluation. (c) Learned-simulator applications for policy evaluation and reinforcement learning.

sponding interaction challenges. Operators collect task-relevant play through teleoperation, handheld devices, or direct hand interaction. Task objectives guide exploration while leaving operators free to repeat and combine actions. We record approximately 20-second training episodes, retaining successful manipulations, failures, ineffective actions, and recovery with their subsequent effects. Evaluation includes successful and failed executions, with a separate long-rollout subset of approximately one-minute episodes.

**Data processing and model training.** Source-specific adapters align observations with recorded or reconstructed motion to form synchronized video–action sequences while preserving platform-specific action semantics (Appendix B). We filter invalid segments, partition data by source episode, and estimate normalization statistics from training data only. Model-specific interfaces then arrange single-view or multi-view observations and encode actions as numerical sequences or control videos, supporting training across tasks and collection platforms. We fine-tune pretrained action-conditioned world models to predict future frames from initial observations and action sequences, following each model's native conditioning interface and retaining its original generative objective (Appendix C).

### 3.2 Evaluation Protocol

An adapted world model should reproduce both the motion caused by an action and its effect on the ongoing interaction. We evaluate paired generated and reference videos from the same test episode under matched observation histories, recorded actions, timestamps, and camera views. Visual quality, action consistency, and task-progress consistency provide complementary measurements, reported separately rather than combined into a single score. We further assess long-horizon stability by tracking how these metrics evolve during continuous rollouts. Aggregation details appear in Appendix D; Section 3.3 evaluates downstream policy success using task-completion labels.

#### 3.2.1 VIDEO GENERATION QUALITY

We assess visual fidelity through pixel-level reconstruction (MAE, MSE, and PSNR), structural similarity (SSIM (Wang et al., 2004)), and feature-distribution agreement. The latter is measured by Frame FID (Heusel et al., 2017) and FVD (R3D-18) (Unterthiner et al., 2018; Tran et al., 2018), computed over frame and clip features within each task, respectively. These metrics quantify visual agreement with real executions, complementing the action- and task-level evaluations below.

#### 3.2.2 ACTION CONSISTENCY

Action consistency measures whether generated videos depict the motion observed when the supplied actions are executed in the reference environment. Since camera configurations and action representations vary across platforms, we train platform-specific inverse dynamics models (IDMs) on collected videos and motion labels. Within each platform, all evaluated world models share a frozen IDM and the same visual inputs and sampling protocol. The IDM independently estimates motion and endpoint states from matched generated and reference video windows, and we compare the two predictions.

For robot and tool platforms, we report position error (Euclidean distance in mm), rotation error (geodesic angle in degrees), and gripper error (absolute difference normalized by the IDM training range). These components are averaged separately over valid windows and effectors within each trajectory. For Ego recordings, we separately report camera and wrist pose errors and fingertip position errors. The resulting discrepancies measure motion consistency under the IDM without establishing the correctness of contact or object dynamics. IDM inputs, motion representations, and metric definitions are detailed in Appendix D.3.

#### 3.2.3 TASK-PROGRESS CONSISTENCY

Task-progress consistency assesses whether generated videos reproduce the advancement, stagnation, and regression observed in real execution. A fixed progress model, Robometer (Liang et al., 2026), independently estimates real and generated progress, $p_j, \hat{p}_j \in [0, 1]$, using the same task description and camera view and only video frames observed up to timestamp $t_j$.

We compare the curves at matched physical timestamps $t_0 < \cdots < t_K$ without temporal warping. Let $e_j = |p_j - \hat{p}_j|$ denote their absolute discrepancy. We convert the time-averaged discrepancy into a similarity score using trapezoidal integration:

$$S_{\text{prog}} = 100 \left[ 1 - \frac{1}{t_K - t_0} \sum_{j=0}^{K-1} \frac{e_j + e_{j+1}}{2} (t_{j+1} - t_j) \right]. \tag{1}$$

Time weighting accounts for unequal sampling intervals. Higher scores in $[0, 100]$ indicate closer agreement with estimated real-video progress, rather than task success. Sampling and integration details appear in Appendix D.4.

#### 3.2.4 LONG-HORIZON STABILITY

To measure how prediction quality changes during continuous rollout, we divide each long trajectory into three equal-duration stages. For metric $m$, let $q_m^{(k)}$ denote its score in stage $k$. We define temporal degradation as

$$D_m = \sigma_m \big( q_m^{(3)} - q_m^{(1)} \big), \qquad \sigma_m = \begin{cases} -1, & m \in \{\text{PSNR}, \text{SSIM}, S_{\text{prog}}\}, \\ 1, & \text{error metrics.} \end{cases} \tag{2}$$

Positive values indicate deterioration. We retain all three stage scores alongside $D_m$, since small changes alone do not imply accurate predictions. Generation continues across stage boundaries, and progress estimates retain their original causal history. Video fidelity, local motion discrepancy, and progress consistency are analyzed separately; implementation and uncertainty estimation appear in Appendix D.5.

### 3.3 APPLICATIONS

A world model adapted to a collection of micro-worlds can serve as a closed-loop simulator: a policy acts on the current observations, the model predicts the resulting observations, and these predictions condition the next decision. We distinguish two uses: estimating the performance of a fixed policy and providing interactions to improve it. These applications test whether predictive consistency translates into useful decision-making feedback.

#### 3.3.1 POLICY SUCCESS-RATE ESTIMATION

An ideal world-model simulator should faithfully capture the dynamical consequences of actions, with policy execution closely matching that in the real world. A high-fidelity world model can therefore accurately estimate policy performance, reducing the substantial resource costs of real-robot evaluation.

#### 3.3.2 IN-MODEL REINFORCEMENT LEARNING AND TRANSFER

When a world model achieves sufficient physical fidelity, a key application is to serve as a simulator for reinforcement learning (Jiang et al., 2026b; Zhu et al., 2026; Jiang et al., 2026c; Yin et al., 2026). We therefore use SAC (Haarnoja et al., 2018) with DSRL (Wagenmaker et al., 2025) to evaluate the effectiveness of our world model. With the base policy $\pi_{0.5}$ (Intelligence et al., 2025) frozen, we optimize a latent-noise policy:

$$\mathcal{L}_\pi(\phi) = \mathbb{E}_{x\sim\mathcal{D},\, z\sim\pi_\phi(\cdot|x)}\left[\alpha \log \pi_\phi(z \mid x) - \min_{1\le j\le N_Q} Q_{\omega_j}(x, z) + \beta\mathcal{L}_{\mathrm{KL}}(x)\right], \tag{3}$$

$$\mathcal{L}_{\mathrm{KL}}(x) = D_{\mathrm{KL}}(\pi_\phi(\cdot \mid x) \,\|\, \pi_{\mathrm{ref}}(\cdot \mid x))\,. \tag{4}$$

Here, $\pi_\phi$ is the latent-noise policy. The reference policy $\pi_{\mathrm{ref}}$ is a frozen copy of $\pi_\phi$ at its near-zero-noise initialization. The KL regularizer limits excessive deviation of the learned noise distribution from its frozen reference.

The $N_Q$ critics $Q_{\omega_j}$ approximate the action-value function $Q^{\pi_\phi}$ of the latent-noise policy in the world model. We define the value function and reward as

$$Q^{\pi_\phi}(x_t, z_t) = \mathbb{E}_{\pi_\phi}\left[\sum_{m=0}^{T-t-1} \gamma^{Km} r_{t+m} \,\middle|\, x_t, z_t\right], \qquad r_t = -1 + r_t^{\mathrm{progress}} + r_t^{\mathrm{dynamics}}, \tag{5}$$

where $\gamma$ is the per-action discount factor and $T$ is the terminal decision step. The constant $-1$ penalizes elapsed time at each decision step, encouraging faster task completion. The progress term $r_t^{\mathrm{progress}}$ is the task-completion score predicted by the general-purpose reward model Robometer (Liang et al., 2026), while $r_t^{\mathrm{dynamics}}$ penalizes motion to maintain the executability of the optimized policy on the real robot, including command-correction, velocity, and acceleration penalties detailed in Appendix F.2. We terminate training trajectories when Robometer's predicted success probability exceeds a task-specific threshold, avoiding learning from subsequent idle states.

## 4 EXPERIMENTS

### 4.1 WORLD-MODEL PREDICTION ACROSS DIVERSE INTERACTIONS

Our evaluation spans all ten interaction categories and all nine collection platforms in RoboWorld100. We compare Cosmos Nano and Cosmos Edge (Agarwal et al., 2026) with GigaWorld using shared training episodes and held-out trajectories within each platform.

Table 2 summarizes the overall evaluation results, highlighting how our multidimensional protocol reveals differences that a single metric would miss. Selecting models by PSNR, SSIM, or gripper error would favor Cosmos Edge over Cosmos Nano, yet Cosmos Nano has smaller IDM position and rotation discrepancies and better task-progress agreement. One possible explanation is that localized

interaction errors can have limited influence on image-level similarity while affecting task progression; gripper error likewise captures only one component of the action. Joint evaluation therefore distinguishes visual resemblance, local motion agreement, and task advancement, providing a more complete assessment of the predicted execution.

Table 2: **World-model prediction on the Evaluation split.**

| Model | Video quality | | | | | | Action consistency | | | Progress |
|---|---|---|---|---|---|---|---|---|---|---|
| | MAE ↓ | MSE ↓ | PSNR ↑ (dB) | SSIM ↑ | Frame FID ↓ | FVD ↓ | Pos. ↓ (mm) | Rot. ↓ (°) | Grip. ↓ (norm.) | Score ↑ (0–100) |
| Nano | **0.0590** | 0.0149 | 18.72 | 0.6531 | **19.12** | **9.67** | **11.07** | **2.07** | 0.0189 | **91.85** |
| Edge | 0.0592 | **0.0147** | **18.86** | **0.6543** | 23.56 | 12.93 | 11.43 | 2.09 | **0.0174** | 91.26 |
| GigaWorld† | 0.0794 | 0.0217 | 17.10 | 0.5848 | 34.90 | 18.48 | 16.32 | 2.88 | 0.0239 | 89.23 |

*Coverage.* Categories receive equal weight. IDM aggregates exclude Ego because its camera and hand motion errors are not directly comparable to robot/tool motion errors. Video/progress include Ego only for Cosmos Nano and Cosmos Edge, as the official GigaWorld† release lacks an adapter for Ego camera and hand motion. FVD uses R3D-18.

Table 3 reports results by platform. On Flexiv, GigaWorld has the highest PSNR but the largest position error and lowest progress similarity. On ALOHA and Tianji, Cosmos Edge leads in PSNR and FVD, while Cosmos Nano better matches motion and progress. Model rankings therefore vary across both platforms and evaluation dimensions.

### 4.2 Long-Horizon Stability

We apply the three-stage protocol in Section 3.2.4 to five continuous Long rollout trajectories per task on the shared tasks. Generation proceeds without restarts; we report early, middle, and late scores with whole-trajectory bootstrap intervals, together with the change from early to late.

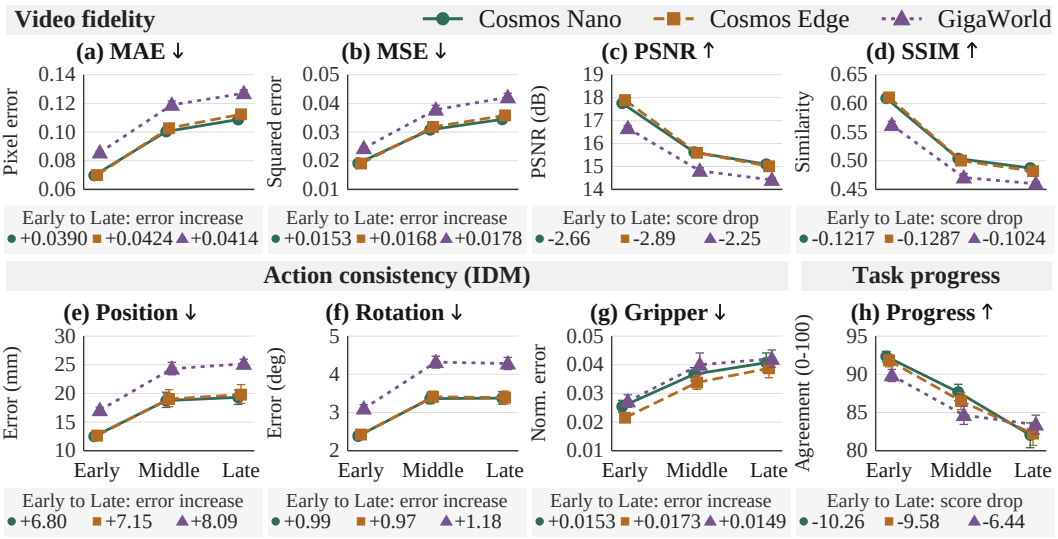



Figure 3: Three-stage Long rollout performance on shared tasks. Error bars: 95% trajectory-bootstrap intervals. Bottom strips show Early-to-Late error increases (+) or score drops (−) in the displayed units. MAE/MSE use $[0, 1]$ pixels; gripper errors use training-range normalization.

All models lose visual and progress consistency as motion error grows (Figure 3). GigaWorld has the smallest PSNR drop but the lowest PSNR throughout, so slower degradation does not imply better predictions. Its highest late-stage progress mean also coexists with the largest position error, showing that task-level stability can mask motion discrepancies.

Table 3: **Evaluation results by collection platform.** Tasks are equally weighted.

**(a) Platform-wise evaluation**

| Platform | | | ALOHA | Tianji | Flexiv | Astribot | Finger | Gripper | UMI | Sim. | Ego |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Video quality** | PSNR ↑ (dB) | Cosmos Nano | 18.08 | 21.66 | 17.50 | 16.77 | **17.20** | **18.36** | 22.66 | 19.87 | **20.13** |
| | | Cosmos Edge | **18.59** | **22.47** | 17.98 | **18.28** | 16.99 | 18.01 | **23.19** | **20.80** | 20.07 |
| | | GigaWorld | 17.42 | 21.04 | **18.36** | 16.98 | 16.43 | 15.90 | 20.13 | 19.01 | – |
| | SSIM ↑ (×100) | Cosmos Nano | 72.50 | 82.98 | 68.82 | 62.28 | **55.00** | **57.62** | 74.73 | 81.09 | **75.33** |
| | | Cosmos Edge | **73.65** | **84.16** | 70.35 | **66.28** | 53.88 | 56.56 | **76.67** | **82.69** | 75.28 |
| | | GigaWorld | 70.44 | 81.53 | **71.18** | 63.02 | 52.35 | 48.52 | 63.90 | 77.60 | – |
| | FVD ↓ | Cosmos Nano | 12.04 | 13.01 | **8.06** | 8.36 | **12.43** | **7.95** | **5.07** | 10.67 | **10.54** |
| | | Cosmos Edge | **10.87** | **11.49** | 8.48 | **8.08** | 19.35 | 10.95 | 5.88 | **9.27** | 21.75 |
| | | GigaWorld | 22.76 | 16.06 | 10.99 | 15.63 | 20.57 | 18.90 | 8.58 | 19.46 | – |
| **Action consistency** | Position ↓ (mm) | Cosmos Nano | **18.14** | **10.05** | 16.97 | 17.35 | **6.75** | **9.48** | **8.15** | **14.70** | See (b) |
| | | Cosmos Edge | 18.49 | 10.14 | **16.66** | **14.67** | 7.19 | 10.35 | 8.41 | 14.86 | |
| | | GigaWorld | 24.28 | 13.01 | 17.45 | 16.27 | 11.53 | 16.40 | 13.53 | 18.16 | |
| | Rotation ↓ (°) | Cosmos Nano | **4.92** | **2.64** | 2.72 | 2.84 | **1.71** | **1.10** | **1.16** | **4.65** | |
| | | Cosmos Edge | 5.04 | 2.72 | **2.67** | **1.91** | 1.78 | 1.21 | **1.16** | 4.81 | |
| | | GigaWorld | 6.15 | 3.49 | 2.78 | 2.07 | 2.60 | 2.05 | 1.84 | 5.60 | |
| | Gripper ↓ (×100) | Cosmos Nano | 3.92 | 2.53 | 3.40 | 1.25 | **1.84** | 1.26 | **1.82** | **1.87** | |
| | | Cosmos Edge | 2.23 | **2.35** | **3.30** | **1.24** | 1.88 | **1.18** | 1.85 | 1.88 | |
| | | GigaWorld | **2.07** | 2.46 | 3.95 | 1.34 | 2.91 | 1.70 | 3.06 | 2.17 | |
| **Task progress** | Similarity ↑ (0–100) | Cosmos Nano | **90.89** | **92.04** | **81.46** | 90.89 | **92.52** | **94.27** | **90.44** | 94.03 | **90.39** |
| | | Cosmos Edge | 90.38 | 91.06 | 80.53 | **91.04** | 92.22 | 93.55 | 89.92 | **94.40** | 89.67 |
| | | GigaWorld | 84.75 | 89.06 | 78.57 | 89.36 | 88.11 | 91.83 | 87.61 | 93.62 | – |

**(b) Ego motion consistency (auxiliary)**

| World model | Camera | | Left hand | | | Right hand | | |
|---|---|---|---|---|---|---|---|---|
| | Position ↓ (mm) | Rotation ↓ (°) | Wrist pos. ↓ (mm) | Wrist rot. ↓ (°) | Fingertip ↓ (mm) | Wrist pos. ↓ (mm) | Wrist rot. ↓ (°) | Fingertip ↓ (mm) |
| Cosmos Nano | **30.92** | **2.00** | **40.96** | **8.21** | **10.42** | **43.34** | **7.96** | **9.20** |
| Cosmos Edge | 34.53 | 2.17 | 46.80 | 9.27 | 12.54 | 48.72 | 9.23 | 10.85 |

### 4.3 Effect of Play Data

We compare Cosmos Edge trained on two data sources: *Task-centered data*, collected to complete prescribed tasks, and RoboWorld100 training data (*Play*), collected through task-relevant exploration. For each task and data source, we use 100 training episodes, five fine-tuning epochs, and the same 20 Evaluation trajectories. RoboWorld100 training data improve task-averaged visual quality, action consistency, and task-progress consistency (Table 4), supporting the value of broader interaction experience for world-model learning.

### 4.4 Policy Success-Rate Estimation

We compare the same policy on the real robot and in two learned simulators: one trained on RoboWorld100 training data (*Ours*) and the other on task-centered data collected separately for comparison (*Task-centered*). For each of four Tianji tasks, we fine-tune a separate $\pi_{0.5}$ policy on 80 teleoperated expert demonstrations and run ten trials per task in each environment. Table 5 reports their success rates. Using RoboWorld100 training data reduces the mean absolute gap to real-robot success from 40 to 15 percentage points relative to the task-centered baseline, supporting more faithful policy-performance estimation.

Table 4: Effect of play (Cosmos Edge).

| Metric | Task-centered | RoboWorld100 |
|---|---|---|
| PSNR (dB) ↑ | 20.00 | **21.49** |
| SSIM ↑ | 0.8089 | **0.8389** |
| FVD ↓ | 25.55 | **13.50** |
| Position (mm) ↓ | 14.39 | **11.88** |
| Rotation (°) ↓ | 3.84 | **3.04** |
| Gripper (norm.) ↓ | 0.0751 | **0.0295** |
| Progress (0–100) ↑ | 85.79 | **89.73** |

Evaluation split; equal task weights; FVD: R3D-18.

Table 5: Fixed-policy success-rate estimation.

| Task | Success rate | | | ΔSR (pp) ↓ | |
|---|---|---|---|---|---|
| | Real | Task-centered | Ours | Task-centered | Ours |
| Pull a tissue | 7/10 | 10/10 | 8/10 | 30 | **10** |
| Roll a dice | 5/10 | 9/10 | 7/10 | 40 | **20** |
| Pour balls | 2/10 | 10/10 | 5/10 | 80 | **30** |
| Wipe | 8/10 | 9/10 | 8/10 | 10 | **0** |
| Average | 55% | 95% | 70% | 40 | **15** |

Training data: Task-centered / RoboWorld100 (Ours). ΔSR: absolute gap to real-robot success (pp).

### 4.5 In-Model Reinforcement Learning and Real-Robot Transfer

Table 6: Real-robot success.

| Task | Base | RoboWorld100 RL |
|---|---|---|
| Pour balls | 3/20 | **12/20** |
| Clean tea residue | 2/20 | **6/20** |
| Roll a dice | 11/20 | **20/20** |
| **Average** | 26.667% | **63.333%** |

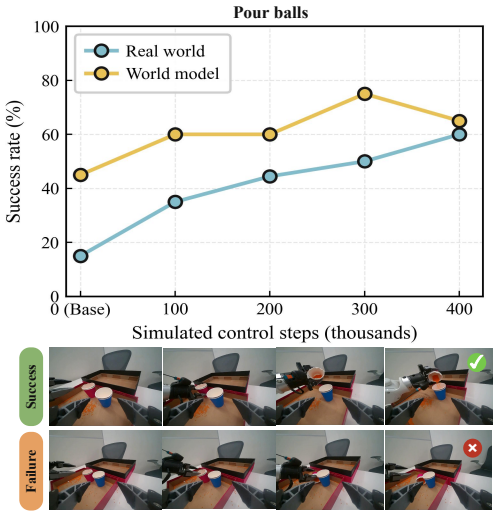



Figure 4: In-model RL on the pouring task: success rates and example executions.

We evaluate in-model RL on three Tianji tasks. Each policy uses 80 demonstrations, disjoint from world-model training, for SFT and RL initial states. Without expanding the training domain, optimized policies improve real-robot performance under randomized initial positions and background perturbations (Table 6). The largest gain is 45 points; robustness improves on the grasping task *Roll a dice*, while the longer-horizon *Clean tea residue* task gains 20 points. Figure 4 shows improvement during policy optimization.On the pouring task, estimated in-model success exceeds real-robot success, consistent with a remaining sim-to-real gap. Nevertheless, the observed transfer gains support the simulator's utility for policy optimization.

## 5 Conclusion

Extending RoboWorld100 toward broader learned simulation requires progress in both physical evaluation and interaction coverage. Our current metrics do not directly assess the correctness of contact and deformation, and the vision-language models we tested, including ChatGPT, were not reliable judges of these effects. Human-play coverage also depends on operator experience, scene design, and collection costs, leaving rare interactions and states encountered by learned policies potentially underrepresented. More targeted physical evaluation and collection guided by model errors are natural next steps.

Within these limits, RoboWorld100 provides a dataset and evaluation framework for adapting robot world models to diverse micro-worlds through task-relevant human play. Human knowledge guides exploration beyond prescribed task solutions across ten interaction categories and 104 task–platform settings. Paired evaluation measures visual quality, action consistency, and task-progress consistency while tracking long-horizon stability. Experiments support the value of play for prediction and policy evaluation, with in-model reinforcement learning improving real-robot success in the tested settings. These micro-worlds provide a measurable route toward reliable learned simulation.

## AI USE STATEMENT

In this work, we use generative AI tools for polishing our manuscript, including language refinement and grammar checking of our original draft. We do not use AI tools for other research tasks requiring disclosure. We review all AI-assisted content and take full responsibility for the final paper.

## REFERENCES

Niket Agarwal, Arslan Ali, Jon Allen, Martin Antolini, Adeline Aubame, Alisson Azzolini, Junjie Bai, Maciej Bala, Yogesh Balaji, Josh Bapst, et al. Cosmos 3: omnimodal world models for physical AI. *arXiv preprint arXiv:2606.02800*, 2026.

Bo Ai, Stephen Tian, Haochen Shi, Yixuan Wang, Tobias Pfaff, Cheston Tan, Henrik I Christensen, Hao Su, Jiajun Wu, and Yunzhu Li. A review of learning-based dynamics models for robotic manipulation. *Science Robotics*, 10(106):eadt1497, 2025.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-JEPA 2: self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.

Hritik Bansal, Zongyu Lin, Tianyi Xie, Zeshun Zong, Michal Yarom, Yonatan Bitton, Chenfanfu Jiang, Yizhou Sun, Kai-Wei Chang, and Aditya Grover. VideoPhy: evaluating physical commonsense for video generation. In *International Conference on Learning Representations*, volume 2025, pp. 102075–102121, 2025.

Akshay L Chandra, Iman Nematollahi, Chenguang Huang, Tim Welschehold, Wolfram Burgard, and Abhinav Valada. DiWA: diffusion policy adaptation with world models. *arXiv preprint arXiv:2508.03645*, 2025.

Lili Chen, Shikhar Bahl, and Deepak Pathak. PlayFusion: skill acquisition via diffusion from language-annotated play. In *Conference on Robot Learning*, pp. 2012–2029. PMLR, 2023.

Sixiang Chen, Jiaming Liu, Jixian Wu, Yichen Guo, Tinghao Wang, Siyuan Qian, Hao Chen, Jiajun Cao, Jian Tang, and Shanghang Zhang. Do robotic world models really follow actions? Diagnosing and aligning action-conditioned generation for policy learning. *arXiv preprint arXiv:2608.24885*, 2026.

Zichen Jeff Cui, Yibin Wang, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. From play to policy: conditional behavior generation from uncurated robot data. *arXiv preprint arXiv:2210.10047*, 2022.

Frederik Ebert, Chelsea Finn, Sudeep Dasari, Annie Xie, Alex Lee, and Sergey Levine. Visual foresight: model-based deep reinforcement learning for vision-based robotic control. *arXiv preprint arXiv:1812.00568*, 2018.

Chun-Kai Fan, Xiaowei Chi, Xiaozhu Ju, Hao Li, Yong Bao, Yu-Kai Wang, Lizhang Chen, Zhiyuan Jiang, Kuangzhi Ge, Ying Li, et al. Wow, wo, val! A comprehensive embodied world model evaluation Turing test. *arXiv preprint arXiv:2601.04137*, 2026.

Zhangyang Gao, Cheng Tan, Lirong Wu, and Stan Z Li. SimVP: simpler yet better video prediction. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3160–3170. IEEE, 2022.

Songwei Ge, Aniruddha Mahapatra, Gaurav Parmar, Jun-Yan Zhu, and Jia-Bin Huang. On the content bias in Fréchet video distance. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7277–7288. IEEE, 2024.

GigaWorld Team, Angyuan Ma, Boyuan Wang, Bohan Li, Chaojun Ni, Guo Li, Guan Huang, Guosheng Zhao, Hao Li, Hengtao Li, et al. GigaWorld-1: a roadmap to build world models for robot policy evaluation. *arXiv preprint arXiv:2607.02642*, 2026.

Yanjiang Guo, Tony Lee, Lucy Xiaoyang Shi, Jianyu Chen, Percy Liang, and Chelsea Finn. VLAW: iterative co-improvement of vision-language-action policy and world model. *arXiv preprint arXiv:2602.12063*, 2026.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, pp. 1861–1870. PMLR, 2018.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. *Advances in Neural Information Processing Systems*, 30, 2017.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, et al. VBench: comprehensive benchmark suite for video generative models. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21807–21818. IEEE, 2024.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_{0.5}$: a Vision-Language-Action model with Open-World Generalization. *arXiv preprint arXiv:2504.16054*, 2025.

Byeongguk Jeon, Seonghyeon Ye, JaeHyeok Doo, Sungdong Kim, Minjoon Seo, Hyungmok Son, and Kimin Lee. RoboWorld: fast and reliable neural simulators for generalist robot policy evaluation. *arXiv preprint arXiv:2607.01060*, 2026.

Feng Jiang, Yang Chen, Kyle Xu, Yuchen Liu, Haifeng Wang, Zhenhao Shen, Jasper Lu, Shengze Huang, Yuanfei Wang, Chen Xie, et al. RoboWM-Bench: a benchmark for evaluating world models in robotic manipulation. *arXiv preprint arXiv:2604.19092*, 2026a.

Zhennan Jiang, Kai Liu, Yuxing Qin, Shuai Tian, Yupeng Zheng, Mingcai Zhou, Zhengtao Zhang, Chao Yu, Haoran Li, and Dongbin Zhao. World4RL: diffusion world models for policy refinement with reinforcement learning for robotic manipulation. *IEEE Robotics and Automation Letters*, 11 (10):12240–12247, 2026b. doi: 10.1109/LRA.2026.3728345.

Zhennan Jiang, Shangqing Zhou, Yutong Jiang, Zefang Huang, Mingjie Wei, Yuhui Chen, Tianxing Zhou, Zhen Guo, Hao Lin, Quanlu Zhang, et al. WoVR: world models as reliable simulators for post-training VLA policies with RL. *arXiv preprint arXiv:2602.13977*, 2026c.

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.

Dacheng Li, Yunhao Fang, Yukang Chen, Shuo Yang, Shiyi Cao, Justin Wong, Michael Luo, Xiaolong Wang, Hongxu Yin, Joseph Gonzalez, et al. WorldModelBench: judging video generation models as world models. *Advances in Neural Information Processing Systems*, 38, 2025a.

Yaxuan Li, Yichen Zhu, Junjie Wen, Chaomin Shen, and Yi Xu. WorldEval: world model as real-world robot policies evaluator. *arXiv preprint arXiv:2505.19017*, 2025b.

Anthony Liang, Yigit Korkmaz, Jiahui Zhang, Minyoung Hwang, Abrar Anwar, Sidhant Kaushik, Aditya Shah, Alex S Huang, Luke Zettlemoyer, Dieter Fox, et al. Robometer: scaling general-purpose robotic reward models via trajectory comparisons. *arXiv preprint arXiv:2603.02115*, 2026.

Yumeng Liu, Xiao-Xiao Long, Marc Habermann, Xuanze Yang, Cheng Lin, Yuan Liu, Yuexin Ma, and Ligang Liu. HGGT: robust and flexible 3D hand mesh reconstruction from uncalibrated images. *arXiv preprint arXiv:2603.23997*, 2026a.

Zeyu Liu, Zhangzhe Zhu, Yang Zhang, Chenyou Fan, Chenjia Bai, and Xuelong Li. KineBench: benchmarking embodied world models via IDM-free kinematic grounding. *arXiv preprint arXiv:2607.19876*, 2026b.

Corey Lynch and Pierre Sermanet. Language conditioned imitation learning over unstructured data. *arXiv preprint arXiv:2005.07648*, 2020.

Corey Lynch, Mohi Khansari, Ted Xiao, Vikash Kumar, Jonathan Tompson, Sergey Levine, and Pierre Sermanet. Learning latent plans from play. In *Conference on Robot Learning*, pp. 1113–1132. PMLR, 2020.

Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. CALVIN: a benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters*, 7(3):7327–7334, 2022.

Iman Nematollahi, Branton DeMoss, Akshay L Chandra, Nick Hawes, Wolfram Burgard, and Ingmar Posner. LUMOS: language-conditioned imitation learning with world models. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8219–8225. IEEE, 2025.

Yiran Qin, Zhelun Shi, Jiwen Yu, Xijun Wang, Enshen Zhou, Lijun Li, Zhenfei Yin, Xihui Liu, Lu Sheng, Jing Shao, et al. WorldSimBench: towards video generation models as world simulators. *arXiv preprint arXiv:2410.18072*, 2024.

Boxiang Qiu, Liliang Chen, Yue Liao, Nan Wang, Lintao Wang, Jiayi Luo, Wenzhi Zhao, Shengcong Chen, Di Chen, Ye Li, et al. GE-Sim 2.0: a roadmap towards comprehensive closed-loop video world simulators for robotic manipulation. *arXiv preprint arXiv:2605.27491*, 2026.

Javier Romero, Dimitrios Tzionas, and Michael J Black. Embodied hands: modeling and capturing hands and bodies together. *arXiv preprint arXiv:2201.02610*, 2022.

Erick Rosete-Beas, Oier Mees, Gabriel Kalweit, Joschka Boedecker, and Wolfram Burgard. Latent plans for task-agnostic offline reinforcement learning. In *Conference on Robot Learning*, pp. 1838–1849. PMLR, 2023.

Ramanan Sekar, Oleh Rybkin, Kostas Daniilidis, Pieter Abbeel, Danijar Hafner, and Deepak Pathak. Planning to explore via self-supervised world models. In *International Conference on Machine Learning*, pp. 8583–8592. PMLR, 2020.

Yu Shang, Zhuohang Li, Yiding Ma, Weikang Su, Xin Jin, Ziyou Wang, Lei Jin, Xin Zhang, Yinzhou Tang, Haisheng Su, et al. WorldArena: a unified benchmark for evaluating perception and functional utility of embodied world models. *AI Open*, 7:208–226, 2026a.

Yu Shang, Yinzhou Tang, Yiding Ma, Zhuohang Li, Lei Jin, Weikang Su, Xin Jin, Zhaolu Wang, Ziyou Wang, Xin Zhang, et al. WorldArena 2.0: extending embodied world model benchmarking on modality, functionality and platform. *arXiv preprint arXiv:2605.17912*, 2026b.

Ansh Kumar Sharma, Yixiang Sun, Ninghao Lu, Yunzhe Zhang, Jiarao Liu, and Sherry Yang. World-Gymnast: training robots with reinforcement learning in a world model. *arXiv preprint arXiv:2602.02454*, 2026.

Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6450–6459. IEEE, 2018.

Wei-Cheng Tseng, Jinwei Gu, Qinsheng Zhang, Hanzi Mao, Ming-Yu Liu, Florian Shkurti, and Lin Yen-Chen. Scalable policy evaluation with video world models. *arXiv preprint arXiv:2511.11520*, 2025.

Thomas Unterthiner, Sjoerd Van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. Towards accurate generative models of video: a new metric & challenges. *arXiv preprint arXiv:1812.01717*, 2018.

Vladyslav Usenko, Nikolaus Demmel, and Daniel Cremers. The double sphere camera model. In *2018 International Conference on 3D Vision (3DV)*, pp. 552–560. IEEE, 2018.

Vikram Voleti, Alexia Jolicoeur-Martineau, and Chris Pal. MCVD: masked conditional video diffusion for prediction, generation, and interpolation. *Advances in Neural Information Processing Systems*, 35:23371–23385, 2022.

Andrew Wagenmaker, Mitsuhiko Nakamoto, Yunchu Zhang, Seohong Park, Waleed Yagoub, Anusha Nagabandi, Abhishek Gupta, and Sergey Levine. Steering your diffusion policy with latent space reinforcement learning. *arXiv preprint arXiv:2506.15799*, 2025.

Homer Rich Walke, Kevin Black, Tony Z Zhao, Quan Vuong, Chongyi Zheng, Philippe Hansen-Estruch, Andre Wang He, Vivek Myers, Moo Jin Kim, Max Du, et al. BridgeData V2: a dataset for robot learning at scale. In *Conference on Robot Learning*, pp. 1723–1736. PMLR, 2023.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

Kun Wu, Chengkai Hou, Jiaming Liu, Zhengping Che, Xiaozhu Ju, Zhuqin Yang, Meng Li, Yinuo Zhao, Zhiyuan Xu, Guang Yang, et al. RoboMIND: benchmark on multi-embodiment intelligence normative data for robot manipulation. *arXiv preprint arXiv:2412.13877*, 2024.

Shihan Wu, Xuecheng Liu, Shaoxuan Xie, Pengwei Wang, Xinghang Li, Bowen Yang, Zhe Li, Kai Zhu, Hongyu Wu, Yiheng Liu, et al. RoboCOIN: an open-sourced bimanual robotic data collection for integrated manipulation. *arXiv preprint arXiv:2511.17441*, 2025.

Wilson Yan, Yunzhi Zhang, Pieter Abbeel, and Aravind Srinivas. VideoGPT: video generation using VQ-VAE and transformers. *arXiv preprint arXiv:2104.10157*, 2021.

Tianzhuo Yang, Zihan Shen, Zirui Mi, Zhaoyi Zhang, Jiayi Zhou, Jiaming Ji, Juntao Dai, Jiawei Chen, Boyuan Chen, and Yaodong Yang. MiraBench: evaluating action-conditioned reliability in robotic world models. *arXiv preprint arXiv:2605.29360*, 2026.

Tenny Yin, Zhiting Mei, Zhonghe Zheng, Miyu Yamane, David Wang, Jade Sceats, Samuel M Bateman, Lihan Zha, Apurva Badithela, Ola Shorinwa, et al. PlayWorld: learning robot world models from autonomous play. *arXiv preprint arXiv:2603.09030*, 2026.

Hu Yue, Siyuan Huang, Yue Liao, Shengcong Chen, Pengfei Zhou, Liliang Chen, Maoqing Yao, and Guanghui Ren. EWMBench: evaluating scene, motion, and semantic quality in embodied world models. *arXiv preprint arXiv:2505.09694*, 2025.

Jiahan Zhang, Muqing Jiang, Nanru Dai, Taiming Lu, Arda Uzunoglu, Shunchi Zhang, Yana Wei, Jiahao Wang, Vishal Patel, Paul Liang, et al. World-in-World: world models in a closed-loop world. In *International Conference on Learning Representations*, volume 2026, pp. 55660–55699, 2026.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: world models on pre-trained visual features enable zero-shot planning. *arXiv preprint arXiv:2411.04983*, 2024a.

Zhiyuan Zhou, Pranav Atreya, Abraham Lee, Homer Walke, Oier Mees, and Sergey Levine. Autonomous improvement of instruction following skills via foundation models. *arXiv preprint arXiv:2407.20635*, 2024b.

Fangqi Zhu, Zhengyang Yan, Zicong Hong, Quanxin Shou, Xiao Ma, and Song Guo. WMPO: world model-based policy optimization for vision-language-action models. In *International Conference on Learning Representations*, volume 2026, pp. 62486–62502, 2026.

# APPENDIX CONTENTS

**A Dataset Catalog and Coverage** **15**
A.1 Task inventory and annotation status . . . . . 15
A.2 Task coverage and trajectory splits . . . . . 18
A.3 Examples across interaction categories . . . . . 19

**B Data Processing and Action Representations** **21**
B.1 Episode assembly and temporal alignment . . . . . 21
B.2 Platform-specific video and action adapters . . . . . 21
B.3 Our Ego hand-motion reconstruction pipeline . . . . . 22
B.4 Validity filtering and model-ready export . . . . . 24

**C World-Model Adaptation and Training** **24**
C.1 Model interfaces and optimization . . . . . 24
C.2 Platform-specific multi-task training . . . . . 25
C.3 Matched-episode play-data comparison . . . . . 25

**D Evaluation Implementation and Validation** **26**
D.1 Paired inputs, coverage, and aggregation . . . . . 26
D.2 Visual metrics and reproducible settings . . . . . 26
D.3 Inverse Dynamics Evaluation Details . . . . . 27
D.4 Robometer Task-Progress Evaluation Details . . . . . 28
D.5 Three-stage long-horizon stability . . . . . 29

**E Additional World-Model Results and Ablations** **30**
E.1 World-model prediction results . . . . . 30
E.2 Long-horizon stability results . . . . . 33
E.3 Effect of play data . . . . . 34

**F Policy Evaluation and Real-Robot Transfer** **35**
F.1 Policy success-rate estimation . . . . . 35
F.2 In-model reinforcement learning and transfer . . . . . 36

# A Dataset Catalog and Coverage

This appendix records task identities, collection platforms, and dataset coverage. Processing and action definitions appear in Appendix B; model adaptation and compute in Appendix C; evaluator implementation in Appendix D; additional prediction results in Appendix E; and policy evaluation and real-robot transfer in Appendix F.

## A.1 Task inventory and annotation status

Table 7 lists 104 task–platform settings, grouped by the ten interaction categories in Table 8.

Primary categories reflect the main interaction requirement; a task may involve several interaction types. Goal descriptions remain fixed across successful, failed, and partial executions. The accompanying task-catalog file retains source-directory mappings and available split information.

Table 7: **RoboWorld100 task catalog.** The 104 task–platform settings are grouped by primary interaction category. Original task IDs link descriptions to collection platforms.

| ID | Task description | Platform |
|---|---|---|
| **01** | **Basic manipulation** | 22 tasks |
| 009 | Turn the die so that the side with three dots faces up. | Tianji |
| 012 | Flip over the UNO card in the middle. | Flexiv |
| 022 | Pick out the green toy from the pile. | Astribot |
| 023 | Place the paper cup on the plate. | Astribot |
| 024 | Cover the small ball on the table with the paper cup. | Astribot |
| 031 | Place the phone on the phone stand. | ALOHA |
| 033 | Pick up the trash from the table and place it in the dustpan. | ALOHA |
| 034 | Flip the paper cup over. | ALOHA |
| 044 | Pick up the small ball with one gripper and pass it to the other gripper. | Finger |
| 048 | Use the gripper to transfer the small balls from one container to the other. | Finger |
| 056 | Separate the nested paper cups. | Finger |
| 060 | Stand the fallen yogurt bottle upright on the table. | Finger |
| 071 | Put the pen in the pencil case. | Gripper |
| 073 | Put the lid on the pot. | Gripper |
| 090 | Push the T-shaped block to the center of the table with the gripper. | Gripper |
| 092 | Rotate the T-shaped block on the table by 180 degrees. | Gripper |
| 094 | Pick up the small balls from the table and place them in the container. | Gripper |
| 095 | Stack the two plates. | Gripper |
| 097 | Put the three balls into the paper cup. | UMI Single |
| 099 | Place the paper cup on the plate. | UMI Single |
| 100 | Put the pen into the pen holder. | UMI Single |
| 102 | Show both sides of the power bank to the camera, then return it green-side up. | Simulation |

*Continued on next page*

**Table 7 RoboWorld100 task catalog** *continued*

| ID | Task description | Platform |
|---|---|---|
| **02** | **Dynamic interaction** | 7 tasks |
| 001 | Pour the particles from the red cup to the blue cup. | Tianji |
| 003 | Pour the small beads from the cup into the funnel. | Tianji |
| 047 | Pour the small balls from the toy bowl into the toy pot. | Finger |
| 077 | Use the plate to stop the rolling ball. | Gripper |
| 081 | Strike one ball with the gripper so it hits the ball in front and sends that ball into the dustpan. | Gripper |
| 093 | Catch the ball rolling down the ramp with the paper cup. | Gripper |
| 098 | Push the ball with the gripper to the black marker. | UMI Single |
| **03** | **Articulated objects** | 10 tasks |
| 005 | Remove the clip from the documents. | Tianji |
| 032 | Secure the documents with the clipboard clip. | ALOHA |
| 038 | Attach the small clip to the paper. | Ego |
| 049 | Open the cabinet door with the gripper. | Finger |
| 050 | Unfold the phone stand. | Finger |
| 053 | Turn the knob. | Finger |
| 055 | Unzip the suitcase. | Finger |
| 061 | Align the toy screw and nut, then screw them together. | Finger |
| 064 | Open the drawer and put the bowl inside. | Finger |
| 083 | Open the notebook on the table. | Gripper |
| **04** | **High-precision tasks** | 25 tasks |
| 010 | Draw a line connecting the two circles on the whiteboard with the marker. | Tianji |
| 014 | Plug the Ethernet cable into the docking station. | Flexiv |
| 015 | Stack the blocks on the table. | Flexiv |
| 016 | Place both Tower of Hanoi disks onto the center peg. | Flexiv |
| 017 | Arrange the three cylinders from smallest to largest, from left to right. | Flexiv |
| 018 | Insert the coin into the coin bank. | Flexiv |
| 019 | Place the egg in a slot in the egg tray. | Flexiv |
| 020 | Insert the straw into the milk tea cup. | Flexiv |
| 025 | Pass the chopstick through the ring on the table. | Astribot |
| 027 | Place two blocks side by side, then lay the third across them to form a bridge. | Astribot |
| 030 | Join the two blocks together. | Astribot |
| 040 | Unlock the cabinet with the key and open the door. | Ego |
| 043 | Insert the card into the card holder. | Ego |
| 045 | Nest one paper cup inside the other. | Finger |
| 046 | Draw a triangle on the board with the whiteboard marker. | Finger |
| 051 | Point to the patterns on the whiteboard with the marker. | Finger |
| 054 | Put the cap on the marker. | Finger |
| 057 | Insert the card reader into the computer's USB port. | Finger |

*Continued on next page*

**Table 7 RoboWorld100 task catalog** *continued*

| ID | Task description | Platform |
|---|---|---|
| **04** | **High-precision tasks** *continued* | 25 tasks |
| 062 | Arrange the cylinders on the table in size order. | Finger |
| 069 | Place the rope in the groove. | Gripper |
| 074 | Insert each cylinder into the slot that matches its size. | Gripper |
| 075 | Put the cap on the USB card reader. | Gripper |
| 079 | Place two paper cups side by side, then stand the third on top of them. | Gripper |
| 085 | Plug the charger into the power strip. | Gripper |
| 088 | Place one wooden block on top of the other two. | Gripper |
| **05** | **Deformable objects** | 11 tasks |
| 004 | Straighten the rope on the table. | Tianji |
| 007 | Pull out one tissue. | Tianji |
| 008 | Push the sponge into the paper cup. | Tianji |
| 026 | Unfold the folded towel on the table. | Astribot |
| 029 | Push the plush toy into the paper cup. | Astribot |
| 035 | Wrap the rope around the post on the table. | ALOHA |
| 036 | Cover the toys on the table with the towel. | ALOHA |
| 070 | Fold the towel. | Gripper |
| 072 | Flatten the towel. | Gripper |
| 076 | Tie a loose knot in the thick rope. | Gripper |
| 080 | Put the towel inside the small cabinet. | Gripper |
| **06** | **Tool use** | 9 tasks |
| 002 | Sweep the tea leaves from the table into the dustpan with the small broom. | Tianji |
| 006 | Wipe the board clean with the eraser. | Tianji |
| 013 | Collect the scattered paper clips from the table with the magnet. | Flexiv |
| 021 | Press the stamp onto the ink pad, then stamp the white paper once. | Flexiv |
| 028 | Use the cardboard to gather the paper clips on the table into a pile. | Astribot |
| 037 | Use the spoon to transfer the small balls from the pot into the bowl. | ALOHA |
| 041 | Cut the paper strip along the black line with the scissors. | Ego |
| 052 | Use the tongs to transfer the small balls from one container to the other. | Finger |
| 086 | Sweep the small balls into the dustpan with the small shovel. | Gripper |
| **07** | **Memory & occlusion** | 4 tasks |
| 058 | Inspect the cards, turn them face down, then reveal the matching pair. | Finger |
| 063 | Put the cup in the drawer, close it, then reopen it and retrieve the cup. | Finger |
| 078 | Hide the ball under a cup, shuffle the three cups, then reveal the ball. | Gripper |
| 089 | Inspect the colored blocks, cover them, then uncover them in the specified color order. | Gripper |
| **08** | **Large workspace** | 2 tasks |
| 039 | Move the box from the table to the front of the cabinet across the room. | Ego |
| 087 | Move the small box from the floor onto the table. | Gripper |

*Continued on next page*

**Table 7 RoboWorld100 task catalog** *continued*

| ID | Task description | Platform |
|---|---|---|
| **09** | **Long-horizon tasks** | 10 tasks |
| 011 | Pour the tea leaves from the paper cup into the teapot, then put the lid on. | Flexiv |
| 042 | Open the document pouch and put the documents inside. | Ego |
| 059 | Open the small case and put the items inside. | Finger |
| 065 | Move the stack of papers to the other table, then pick up the top sheet. | Finger |
| 066 | Close the notebook, put the pens in the pen holder, then place the cup on the plate. | Finger |
| 082 | Put the objects from the table inside the small cabinet. | Gripper |
| 084 | Fold the towel and put it inside the small cabinet. | Gripper |
| 101 | Place the croissant on the plate, with the plate and cup in their marked areas. | Simulation |
| 103 | Move the keyboard to the marked area and restore the surrounding obstacles. | Simulation |
| 104 | Place the yellow middle book in the marked area and restack the remaining books. | Simulation |
| **10** | **Physics-driven interaction** | 4 tasks |
| 067 | Pull the sheet of paper out from under the objects. | Finger |
| 068 | Knock down all the dominoes. | Finger |
| 091 | Push the bottle to move it. | Gripper |
| 096 | Push the cylinder with the gripper so it rolls up the ramp. | Gripper |

### A.2 Task coverage and trajectory splits

Table 8 reports the number of task–platform settings in each interaction category.

Table 8: Task-catalog coverage by primary interaction category.

| Primary category | Settings | Primary category | Settings |
|---|---|---|---|
| Basic manipulation | 22 | Tool use | 9 |
| Dynamic interaction | 7 | Memory & occlusion | 4 |
| Articulated objects | 10 | Large workspace | 2 |
| High-precision tasks | 25 | Long-horizon tasks | 10 |
| Deformable objects | 11 | Physics-driven interaction | 4 |
| Total task–platform settings | | | 104 |

Following temporal alignment and quality filtering (Appendix B), trajectories are grouped by task and collection platform. Splits are assigned at the level of the original collection episode. All camera views, resampled sequences, and temporal crops derived from an episode inherit the same assignment, preventing overlap between training and test data. Trajectory counts are computed from unique source episodes.

The standard experimental protocol specifies 100 training, 20 evaluation, and five long-rollout trajectories per setting. The evaluation split measures video prediction under recorded actions, while the long-rollout split assesses state consistency and error accumulation over extended autoregressive predictions. Model-selection data are held out from the training sources, and normalization statistics are estimated using only the training split. Evaluation and long-rollout trajectories are held fixed across compared models.

### A.3 EXAMPLES ACROSS INTERACTION CATEGORIES

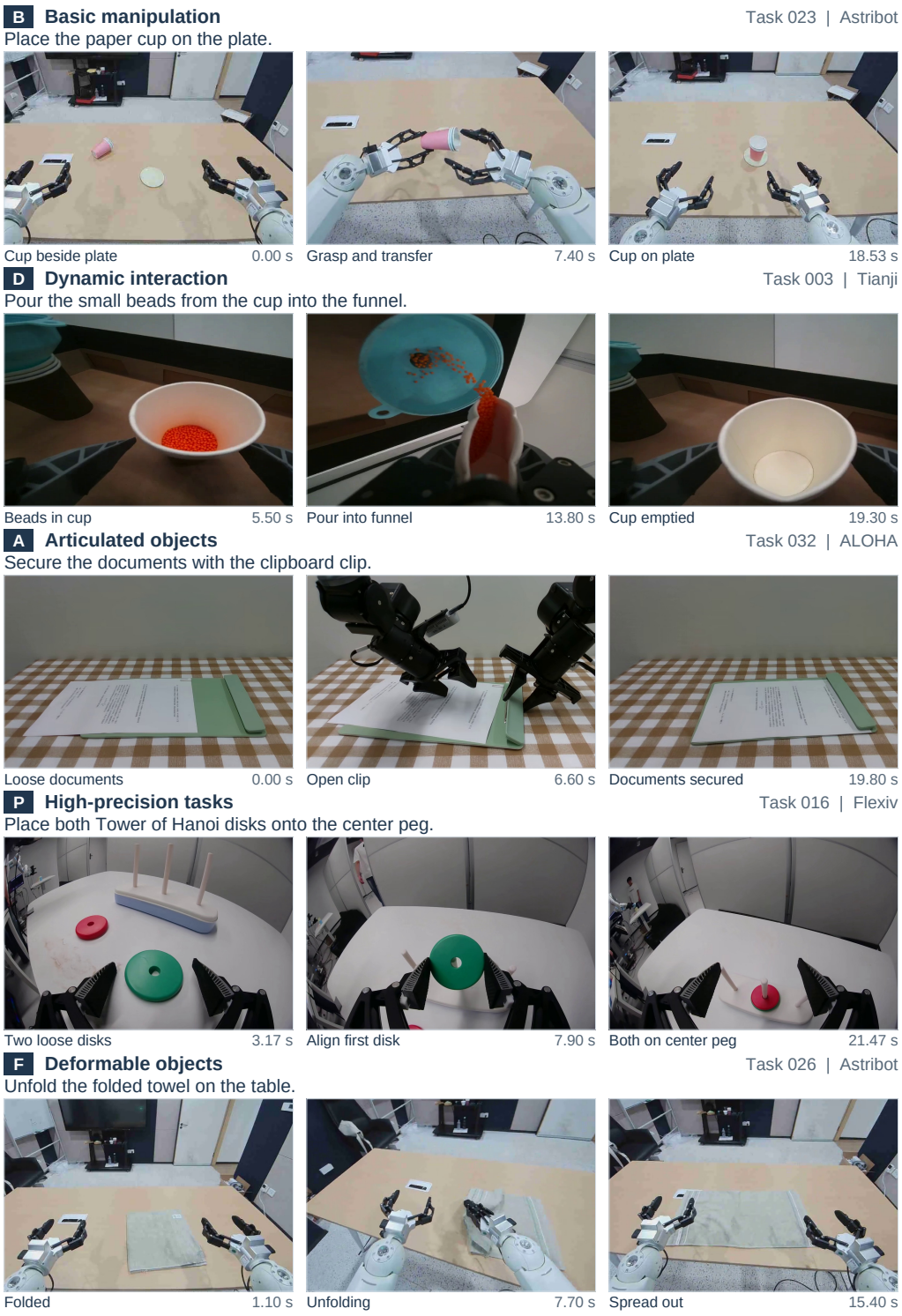



Figure 5: **Collected trajectories across interaction categories (1/2).** Three chronological frames per category show object states and interaction stages. Each row uses one trajectory, one camera, and a fixed crop; timestamps indicate elapsed time. Task IDs refer to Table 7.




Figure 6: **Collected trajectories across interaction categories (2/2).** Examples of tool use, memory and occlusion, large-workspace manipulation, long-horizon tasks, and physics-driven interaction. Frames follow the same convention as Figure 5.

# B Data Processing and Action Representations

This appendix describes how collection records are converted into synchronized video–action sequences, including the platform adapters and our Ego hand-motion reconstruction pipeline. Collection scenarios and task coverage are described in Section 3.1 and Appendix A; model training and IDM evaluation are detailed in Appendices C and D.3, respectively.

The pipeline standardizes episode organization and the preparation of model inputs through source-specific adapters. Each adapter specifies its action dimensions, units, coordinate conventions, and temporal correspondence according to the source representation. The resulting sequences share a sample structure while retaining distinct physical meanings across platforms.

## B.1 Episode assembly and temporal alignment

We process each continuous recording as a source episode. Camera streams, robot commands or states, sensor timestamps, and calibration records are decoded before constructing model inputs. For sources that require motion estimation, pose reconstruction precedes action conversion. Figure 7 illustrates these stages on an Ego recording.

Video and motion records are matched on a common timeline within their shared time range. Camera order is fixed for each adapter, and the same spatial processing is applied to every frame. For $T$ observations, the model input contains $T - 1$ transition conditions. Recorded commands at time $t$, measured states at time $t + 1$, and relative motion from $t$ to $t + 1$ are aligned according to their respective definitions, rather than being treated as interchangeable labels.

Valid segments and temporal windows retain their source episode and timestamps. All views and derived segments of an episode inherit the same training or test assignment (Appendix A.2). Recording quality is assessed separately from task outcome, so unsuccessful interactions remain usable when their observations and motion records are valid.

## B.2 Platform-specific video and action adapters

Table 9 summarizes the video and action interfaces used by our reference data adapters. The dimensions refer to world-model conditioning before any model-specific padding. Native records are retained alongside the converted representation.

Table 9: **Platform-specific observations and action representations.** Camera-view types follow Figure 2; dimensions refer to numerical conditions before model-specific encoding or padding.

| Platform | Camera views | Action conditioning | Dim. |
|---|---|---|---|
| ALOHA | Fixed global; wrist | Next measured state: 6 joints and a gripper scalar per arm. | 14 |
| Tianji | Fixed global; wrist | Recorded command: 7 joints and a gripper target per arm. | 16 |
| ARX (sim.) | Fixed global; wrist | Next state: 6 measured joints and the applied gripper target per arm. | 14 |
| Flexiv | Wrist | TCP target relative to the current pose: translation (3), quaternion (4), and gripper target (1) per arm. | 16 |
| Astribot | Moving global; wrist | Next body state: base (3), torso (4), head (2), and 7 joints plus a gripper scalar per arm. | 25 |
| Gripper | Wrist | Relative tool pose (9) and calibrated gripper scalar (1) per side. | 20 |
| Finger | Moving global; wrist | Relative pose (9) and recorded gripper scalar (1) per side. | 20 |
| UMI Single | Wrist | Relative tool pose (9) and normalized opening width (1). | 10 |
| Ego | Moving global | Relative camera pose (9), plus relative wrist pose (9) and endpoint fingertip coordinates (15) per hand. | 57 |

View selection and spatial layout depend on the model interface. A 9D relative pose comprises 3D translation and 6D rotation in the starting frame. Next states and relative motions correspond to $t + 1$ and $t \to t + 1$, respectively; Ego fingertip coordinates use the endpoint wrist frame.

**Robot records.** ALOHA records a front view and two wrist views. Its 14D condition uses the measured state at $t + 1$, ordered as six joints and one gripper value for each arm. The benchmark adapter places a $640 \times 360$ front image above two $320 \times 180$ wrist images, then adds two reflection-padded rows at the top and bottom to produce a $544 \times 640$ input (height $\times$ width).

Astribot records four views: the left and right eye cameras and one wrist camera on each arm. Its 25D condition uses the measured state at $t + 1$, retaining the chassis, torso, both arms and grippers, and head coordinates. For the reported benchmark, the input adapter selects the left-eye and both wrist views and uses the $544 \times 640$ layout described above.

Tianji uses four views, arranged with the two eye cameras above the two wrist cameras. Its 16D condition selects zero-based indices $[0, \ldots, 6, 28, 7, \ldots, 13, 29]$ from the recorded 30D command at $t$, preserving the left-arm/gripper and right-arm/gripper order at 10 Hz. The benchmark adapter resizes each $960 \times 540$ view to $320 \times 180$, forms a $2 \times 2$ mosaic, removes its bottom 12 rows, and adds two reflection-padded rows at the top and bottom, yielding $352 \times 640$ inputs.

Flexiv uses both wrist views and expresses each recorded absolute TCP target in the corresponding current TCP frame. Each arm contributes translation (3), a unit quaternion in $wxyz$ order (4), and the unchanged gripper target (1), giving 16 dimensions. The visual adapter removes eight rows from both the top and bottom of each $1600 \times 1296$ image, resizes it to $320 \times 256$, and concatenates the two views into $256 \times 640$ video.

ARX (sim.) records a head view and two wrist views. Its 14D condition uses the next recorded state: six measured joint positions and the applied, normalized gripper target per arm. Since source observations are recorded after action execution, the adapter pairs observations at $t$ and $t + 1$ with the state recorded at $t + 1$.

**Handheld collection.** Gripper uses two wrist views; Finger uses two head views and two wrist views. The adapters match images, device-provided poses, and gripper signals to a 15 Hz timeline by nearest timestamps; Finger additionally enforces a 40 ms synchronization tolerance. For each side, consecutive poses define relative translation and rotation in the starting frame, with rotation encoded by the first two matrix columns. Appending the gripper value at the transition start yields ten dimensions per side, concatenated left then right. Gripper maps encoder values to $[0, 1]$ using open/closed calibration, while Finger retains the raw scalar before training-set normalization.

UMI Single supplies one wrist view and timestamp-aligned tool trajectories and gripper widths from the upstream video–IMU reconstruction and calibration pipeline. Its native seven-dimensional state comprises position, a rotation vector, and opening width. Our model adapter selects matched observations and poses at 15 Hz, converts consecutive tool poses to 3D relative translation and 6D rotation, and appends the opening width at the transition start. For the benchmark, widths are normalized to $[0, 1]$ using the range observed in training episodes. The resulting 10D conditions are paired with the corresponding wrist-camera frames.

### B.3 Our Ego hand-motion reconstruction pipeline

Ego collection records direct human interaction without robot joint commands. We implement a pipeline that combines the device's calibrated cameras and VIO with hand detection, HGGT (Liu et al., 2026a) reconstruction, and MANO (Romero et al., 2022) hand geometry. Our implementation provides synchronization, camera-aware crop construction, geometric alignment, validity handling, and the conversion to world-model actions. The reconstruction uses all six available cameras, while the prediction model uses the main head view.

**Synchronization and hand detection.** We match the six image streams and VIO poses by timestamp, using a 1 ms tolerance and consuming each source observation at most once. The synchronized timeline is sampled at 15 Hz. A separate detector/tracker state is maintained for each camera, with left and right hands identified independently and a detection-confidence threshold of 0.3. Only short, bounded detection gaps of at most two sampled frames are interpolated when the surrounding boxes have compatible positions and scales.

**Multi-view reconstruction.** For each hand, at least two valid views are required. Using the calibrated Double Sphere camera model (Usenko et al., 2018), we transform the detected regions into

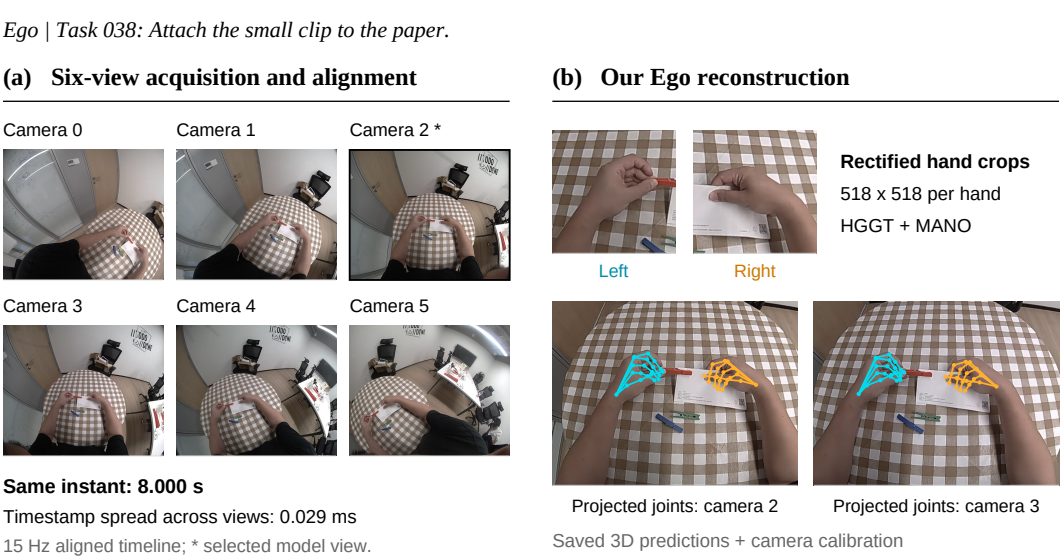



Figure 7: **From an Ego recording to video–action inputs.** All panels use the same task-038 episode. (a) Six synchronized source views at 8.000 s; camera 2 supplies the model video. (b) Calibrated hand crops and reconstructed joints projected into two views. (c) Camera and wrist translation per transition, with excluded frames shaded. (d) Consecutive observations and their paired actions, above the full 57-channel action sequence. Cyan and orange identify the left and right hands. Image crops in (b,d) aid visibility; the heatmap uses per-channel $z$-scores only for display. Joint overlays are saved reconstruction outputs, not pose ground truth.

$518 \times 518$ virtual pinhole crops with a padding factor of 1.2. HGGT predicts hand pose and shape from these crops, and MANO produces the articulated hand mesh and 21 joints. We aggregate the shape parameters per hand across reliable frames and hold that shape fixed over the episode, while allowing articulation and global pose to change.

The reconstructed hand is then placed in the VIO world frame. Crop-space joint predictions are mapped back to calibrated camera rays. We initialize candidate global poses from view-pair triangulation and refine the rigid hand pose with a robust reprojection objective, keeping the MANO scale and articulation fixed during this alignment. Views with reprojection residuals above 20 pixels in crop coordinates are rejected; a fit requires at least two supporting views. If this fit fails, the im-

plementation falls back to fitting global translation against the detected bounding boxes. Figure 7(b) shows the calibrated hand crops and reconstructed joints projected into two original camera views.

**Coordinate frames and the 57D representation.** Let $T_t^c$, $T_t^r$, and $T_t^l$ map the ego-camera, right-wrist, and left-wrist frames into the VIO world frame. The ego-camera frame is the midpoint of cameras 2 and 3: translations are averaged and rotations use the shortest-geodesic midpoint on $\mathrm{SO}(3)$. Wrist origins are MANO joint 0, with orientations given by the MANO root rotation.

For $x \in \{c, r, l\}$, we form

$$\Delta T_t^x = (T_t^x)^{-1} T_{t+1}^x, \qquad \mathbf{d}_t^x = \left[\Delta \mathbf{p}_t^x, \mathrm{rot6d}(\Delta R_t^x)\right] \in \mathbb{R}^9, \tag{6}$$

where $\mathrm{rot6d}$ concatenates the first and second rotation-matrix columns. Thus translation is expressed in the starting camera or wrist frame. For each hand $h \in \{r, l\}$, fingertip $k$ is expressed in the endpoint wrist frame as

$$\mathbf{q}_{t+1,k}^h = (R_{t+1}^h)^\top \left(\mathbf{f}_{t+1,k}^{h,\mathrm{world}} - \mathbf{p}_{t+1}^h\right), \qquad k \in \{\text{thumb, index, middle, ring, little}\}. \tag{7}$$

Stacking the five fingertip positions into $\mathbf{q}_{t+1}^h \in \mathbb{R}^{15}$, we obtain

$$\mathbf{a}_t^{\mathrm{Ego}} = \left[\mathbf{d}_t^c, \mathbf{d}_t^r, \mathbf{q}_{t+1}^r, \mathbf{d}_t^l, \mathbf{q}_{t+1}^l\right] \in \mathbb{R}^{57}. \tag{8}$$

The three 9D motion blocks and two 15D fingertip blocks contribute $9+9+15+9+15 = 57$ dimensions. Relative motion captures where the camera and hands move; endpoint fingertip coordinates capture hand configuration. Positions use metres, and 6D rotation entries are dimensionless.

### B.4 Validity filtering and model-ready export

Ego retains synchronization flags, detection provenance, hand-validity masks, supporting views, and reprojection residuals. An exported frame requires synchronized images and VIO, valid estimates for both hands, and finite pose and joint values. Consecutive retained frames form an action only when their timestamp gap is at most 100 ms; larger gaps split the sequence into separate valid segments. The native six-view export is retained, and camera 2 supplies the main-view video used by the Ego adapter.

Converted episodes store video, transition conditions, timestamps, and source metadata. Adapter-specific checks cover decoded-frame counts, action dimensions, and temporal correspondence. Flexiv uses recorded action-validity flags, Finger bounds synchronization error, and Ego applies the reconstruction checks above. Training windows are formed from the converted sequences using the selected view layout and action representation. Normalization statistics are computed from the training split and reused for evaluation; source timestamps and available validity information are retained for tracing the selected segments.

## C World-Model Adaptation and Training

### C.1 Model interfaces and optimization

We adapt Cosmos Nano, Cosmos Edge, and GigaWorld to the observations and action representations in Appendix B.2. Both Cosmos models receive numerical actions, normalized per task using the training-set 1st and 99th percentiles, clipped to $[-1, 1]$, and padded to 64 channels with a mask retaining the original dimension. GigaWorld instead renders platform-specific motion into control videos, using control statistics estimated from the same selected training episodes. These interfaces retain each model's native generative objective.

For the multi-task comparison, each training window contains 33 consecutive frames and 32 transition conditions, with frame and window strides of one. The first image, task description, and subsequent actions condition prediction of the remaining frames. For GigaWorld, the first image initializes short-term latent history; the middle- and long-term history slots are zero in these training windows. Future reference images supply prediction targets, not history conditioning.

Table 10 gives the settings recovered from the saved runtime configurations and training code. Cosmos updates its generation branches, time embeddings, visual projections, and action modules, with

LoRA disabled. GigaWorld updates LoRA weights and the input and short-, middle-, and long-history patch projections. The parameter counts refer to optimized tensors, excluding frozen weights and exponential moving averages; they were checked against all 26 final checkpoints.

Table 10: Optimization settings for the multi-task comparison. Trainable counts are numbers of parameters, not numbers of tensors. Both Cosmos variants share the optimization settings below.

| **Setting** | **Cosmos Nano** | **Cosmos Edge** | **GigaWorld** |
|---|---|---|---|
| Training epochs | 2 for Finger, Gripper, and UMI Single; 5 for all other platforms | | |
| Initialization | Cosmos3-Nano | Cosmos3-Edge | Giga-World-1, Stage-1 Nano + scene adapter |
| Trainable parameters | 6,982,401,216 | 1,423,379,648 | 209,590,272 |
| Optimizer | FusedAdam (AdamW mode) | | AdamW |
| Adam $\beta_1, \beta_2$ | $(0.9, 0.99)$ | | $(0.9, 0.999)$ |
| Weight decay | 0.05 | | $10^{-4}$ |
| Configured base LR | $10^{-4}$ | | $3 \times 10^{-5}$ |
| LR schedule | Linear multiplier $0.4 \rightarrow 0$; no warmup | | Constant |
| Action-module LR | $5\times$ base LR before scheduling | | Not applicable |
| Gradient clipping | Norm 1.0 | | Norm 1.0 |
| Compute precision | BF16; FP32 master weights | | BF16 autocast; FP32 trainable weights |

The Cosmos schedule makes the initial LR $4 \times 10^{-5}$ for the selected generation modules and $2 \times 10^{-4}$ for the action input/output projections and modality embedding. Its decay horizon equals the platform's total update budget.

These runs start from the corresponding released pretrained weights with a fresh optimizer. GigaWorld additionally loads the released Stage-1 scene LoRA and partial transformer weights, rather than a checkpoint already adapted to a RoboWorld100 task. The selected checkpoint is always the final checkpoint at the prescribed training budget; neither Evaluation nor Long rollout scores select it. The source record `appendix/data/training_provenance.json` lists the 26 checkpoint locations, export checksums, configuration hashes, and training-job identifiers. The imported pretrained packages did not retain upstream revision identifiers; the record therefore identifies the available package paths and saved initialization receipts without assigning an unverified revision.

### C.2 Platform-specific multi-task training

We adapt each model separately to each collection platform, pooling play episodes from the selected tasks on that platform. The data adapters in Appendix B.2 form synchronized video–action sequences while preserving the platform's camera configuration and action semantics. Training clips are sampled within individual source episodes and valid continuous segments, including those retained by the Ego reconstruction pipeline.

Models compared on the same platform use the same selected training episodes and optimizer-update count, with platform-specific epoch budgets listed in Table 10. Each run uses eight NVIDIA RTX PRO 6000 GPUs, except for UMI Single, which uses eight NVIDIA GeForce RTX 5090 GPUs. The held-out evaluation protocol is described in Appendix D.1.

### C.3 Matched-episode play-data comparison

Task-centered collection focuses on completing the prescribed task, whereas RoboWorld100 training data (*Play*) are collected through task-relevant exploration beyond its usual completion sequence. For the comparison in Section 4.3, we initialize a separate Cosmos Edge model for each task and data source from the same pretrained weights. Each source supplies 100 training episodes, with five epochs, training seed 20260920, a global batch of eight, and 32-transition windows with frame and window strides of one. Normalization is estimated independently from each training set. Because episode lengths differ, this experiment matches episode counts and epochs rather than optimizer updates or total frame exposure.

Evaluation follows the paired protocol in Appendix D.1. Both data conditions use 20 denoising steps, guidance 1, and inference seed 42. Results receive equal task weights and are reported in Appendix E.3.

# D Evaluation Implementation and Validation

## D.1 Paired inputs, coverage, and aggregation

We evaluate generated and reference videos from the same held-out source episode under matched recorded actions, task descriptions, timestamps, and camera layouts. Generation starts from the initial real observation; subsequent visual history comes from generated frames. Compared predictions use the same frozen evaluators and preprocessing. Each exported pair contains lossless 8-bit RGB arrays with identical dimensions. The evaluation records bind these arrays to the source episode, model checkpoint, and evaluator configuration through file and frame checksums. Across world models evaluated on the same task, source identifiers, frame counts, frame rates, image shapes, and reference-frame hashes agree.

The evaluation spans ten interaction categories and nine collection platforms. Cosmos Nano and Cosmos Edge cover all selected tasks; GigaWorld covers the non-Ego subset because its official interface lacks an adapter for Ego camera and hand motion. Each task contributes 20 Evaluation and five Long rollout trajectories. Table 2 reports each model's Evaluation coverage, while Table 11 compares all three models on identical task support. Position, rotation, and gripper aggregates use the same non-Ego task set for all models. Ego camera, wrist, and fingertip errors are reported separately in Table 15.

For MAE, MSE, PSNR, SSIM, action consistency, and progress consistency, we first obtain one score per trajectory, then average trajectories equally within each task. Overall world-model comparisons average task means within interaction categories, followed by equal category averaging. Platform summaries and the play-data comparison instead average tasks equally within their respective task sets. Frame FID and FVD are computed from pooled samples within each task and split before applying the same task or category weights. Frame and clip sampling is matched across compared conditions. Evaluation and Long rollout remain separate throughout aggregation.

For per-trajectory metrics, 95% confidence intervals use 10,000 paired bootstrap resamples with NumPy's `default_rng(42)`. Within each fixed task, trajectory indices are sampled with replacement and shared across compared models or data conditions; the aggregation weights specified above remain unchanged. Differences are computed within each resample. These exploratory intervals describe trajectory-sampling uncertainty for the evaluated tasks and checkpoints, rather than variation across training seeds or a wider task population, and are not adjusted for multiple comparisons. Platform-specific training is described in Appendix C.2.

## D.2 Visual metrics and reproducible settings

**Pixel and structural metrics.** MAE and MSE are averaged over all pixels, RGB channels, and frames of each trajectory after scaling RGB values to $[0, 1]$. PSNR uses the trajectory-level MSE and a unit data range, equivalently a range of 255 on the original 8-bit arrays; it is not an average of frame-level PSNR values. Exactly zero MSE gives infinite PSNR. SSIM uses `scikit-image`'s structural-similarity implementation with a $7 \times 7$ uniform window, sample covariance normalization, $K_1 = 0.01$, $K_2 = 0.03$, and data range 255. It is evaluated separately over RGB channels and averaged over channels and frames. The complete model canvas and initial conditioning frame are retained, with no additional cropping or temporal resampling.

**Frame FID.** Frame FID (Heusel et al., 2017) uses 2,048-dimensional InceptionV3 pool features from `torch-fidelity`, with its standard image resizing and normalization. Every frame contributes, using batches of 32. TorchMetrics accumulates first and second feature moments in double precision and computes the Fréchet distance between the reference and generated frame distributions within each task and split.

**FVD (R3D-18).** We extract 512-dimensional features from Kinetics-400-pretrained R3D-18 with its classification layer removed. Clips contain 16 consecutive frames at the native video frame rate and start every eight frames; a final window ending at the last frame is appended when needed. All eligible clips are used, without random subsampling. Clips never cross episode or reconstruction-segment boundaries. Segments shorter than 16 frames are excluded from FVD without padding, and a task-level estimate requires at least two clips per video source.

For feature extraction, RGB values are scaled to $[0, 1]$, spatially resized to $112 \times 112$ using trilinear interpolation with `align_corners=False`, and normalized with the Kinetics weight-specific channel means and standard deviations. The temporal length remains 16 and the batch size is eight. The Fréchet distance uses double-precision features, unbiased sample covariances, and SciPy's matrix square root. A $10^{-6}$ diagonal offset is used only if the covariance-product square root is nonfinite. We name this metric FVD (R3D-18) because values obtained with a different video backbone are not directly comparable.

### D.3 Inverse Dynamics Evaluation Details

#### D.3.1 Platform-specific IDM training

Each platform-specific inverse dynamics model (IDM) is trained on paired visual observations and corresponding motion labels. Supervision comes from robot or device records for physical collection, environment states for simulation, and VIO and hand reconstruction for Ego. The deterministic regression models use a frozen SigLIP2 visual encoder with trainable projection, fusion, and motion-prediction heads. Finger and Gripper instead use GR00T-based sequence predictors supervised on short action sequences.

After training, each evaluator is frozen and applied independently to generated and reference videos. All world models evaluated on a platform share the same evaluator, preprocessing, motion representation, and temporal windows.

#### D.3.2 Visual inputs and motion labels

Given images at the start and end of a temporal window, the IDM estimates the intervening motion and gripper or hand configuration. Neither supplied world-model actions nor recorded motion labels are inputs to this scoring pass.

ALOHA and Astribot use one front/left-eye view and two wrist views per endpoint; Tianji uses both eye and both wrist views. Flexiv and Gripper use two wrist views, Finger uses two head and two wrist views, and UMI Single uses its single wrist view. Ego uses the full image and detected left/right hand crops at two distinct endpoints, repeated according to the E0 input layout. The adapter removes only documented model padding before applying the evaluator's camera extraction and preprocessing, identically for generated and reference frames.

For platform $d$, the paired predictions and their component discrepancies are

$$\begin{aligned} \hat{z}^{v}_{t,\Delta} &= f_{\phi_d}\left(O^{v}_{t}, O^{v}_{t+\Delta}\right), \qquad v \in \{\mathrm{real}, \mathrm{gen}\}, \\ e^{(c)}_{t,\Delta} &= d_c\left(\hat{z}^{\mathrm{gen}}_{t,\Delta}, \hat{z}^{\mathrm{real}}_{t,\Delta}\right). \end{aligned} \tag{9}$$

Here, $O^{v}_{t}$ denotes the selected camera observations, and $\hat{z}^{v}_{t,\Delta}$ contains the inferred motion and endpoint state for temporal gap $\Delta$. The observation combination and temporal gap follow the device-specific input and sampling protocol. The superscript $\mathrm{real}$ denotes the ground-truth video. The component distance $d_c$ compares the corresponding position, rotation, or gripper predictions, as defined below. Both predictions use matched windows of the same execution.

For direct pose predictors, $\hat{z}$ contains relative translation, 6D rotation, and an endpoint gripper value for each effector. Finger and Gripper predict 12 relative actions, which are composed into an endpoint pose before comparison. Their stochastic inference uses the same deterministic per-episode and per-batch seed for the generated and reference inputs, with a fixed batch size of eight. Both video sources are therefore compared using the same endpoint representation, independently of the world model's native control dimensions.

Robot pose labels are obtained either directly from recorded states or reconstructed from joint states using fixed forward kinematics. Depending on the device version, gripper labels are derived from either state feedback or control commands. For Ego data, camera and hand-motion labels are obtained from VIO and hand reconstruction, together with validity masks from the source estimators. Normalization statistics are computed exclusively from the IDM training data, and predictions are converted back to the corresponding physical units or predefined scales before evaluation. Labels from different sources are not assumed to have equal measurement accuracy.

For simulation, the 25 Hz source timeline is sampled at 15 Hz using the nearest recorded timestamps, with an eight-frame endpoint gap and four-frame stride. Both endpoints contain head, left-wrist, and right-wrist views. The 20D output concatenates each arm's relative link-6 pose and endpoint gripper state; gripper discrepancies use the frozen training-range normalization. The same 50 trajectories per model are scored as in the video and progress evaluations, with matching source pixels verified at the sampled GT endpoints.

#### D.3.3 Action-consistency metrics

For each matched temporal window, let $\hat{\mathbf{p}}^{\,v}$, $\hat{R}^{\,v}$, and $\hat{g}^{\,v}$ denote the position, rotation, and gripper components of the IDM output for video source $v \in \{\mathrm{real}, \mathrm{gen}\}$. We separately measure their discrepancies, with lower values indicating greater action consistency:

$$
\begin{aligned}
e_{\mathrm{pos}} &= \left\| \hat{\mathbf{p}}^{\,\mathrm{gen}} - \hat{\mathbf{p}}^{\,\mathrm{real}} \right\|_2 , \\
e_{\mathrm{rot}} &= \arccos\left( \mathrm{clip}\left[ \frac{\mathrm{Tr}\left( (\hat{R}^{\,\mathrm{gen}})^{\top} \hat{R}^{\,\mathrm{real}} \right) - 1}{2}, -1, 1 \right] \right) , \\
e_{\mathrm{grip}} &= \left| \hat{g}^{\,\mathrm{gen}} - \hat{g}^{\,\mathrm{real}} \right| .
\end{aligned}
\tag{10}
$$

Both predictions describe motion over the same temporal window in the same reference coordinate system. Position discrepancies are reported in millimeters, while rotation discrepancies are converted from the $\mathrm{SO}(3)$ geodesic angle to degrees. When rotation is predicted using a 6D representation, both predictions are first decoded into valid rotation matrices before comparison.

The native gripper discrepancy in Equation 10 is divided by the corresponding effector's frozen training range, $q_{99} - q_{01}$, before aggregation. The reported normalized value is dimensionless and is not an open/close classification error rate. For Ego, camera and wrist pose discrepancies are computed separately; fingertip error is the mean Euclidean distance over the five predicted fingertip positions in the endpoint wrist frame, reported separately for each hand.

#### D.3.4 Temporal sampling and aggregation

We sample complete windows within each valid source segment using fixed platform-specific gaps and strides, identically for generated and reference videos. Component discrepancies are averaged over corresponding effectors and valid windows to obtain one score per trajectory; the task and category weights then follow Appendix D.1. Thus longer trajectories do not receive greater weight simply because they contain more windows. Ego components remain separate from the robot/tool aggregates.

### D.4 Robometer Task-Progress Evaluation Details

#### D.4.1 Evaluation objective and inputs

Robometer (Liang et al., 2026) measures whether generated videos reproduce the advancement, stagnation, and regression estimated from the corresponding real execution. The same fixed evaluator scores both videos independently under the paired-input protocol in Appendix D.1. The task description remains unchanged across successful, failed, and partially completed executions. The resulting score measures progress consistency, not task success; the evaluator's separate success probability is stored independently and is not evidence of real-world policy success.

We use Robometer-4B with the same prompt template for real and generated videos. Both retain the full input canvas, including the world model's camera layout for multi-view inputs.

#### D.4.2 HISTORY SAMPLING AND PROGRESS ESTIMATION

Progress is evaluated every two seconds, with both the beginning and end of each continuous trajectory explicitly included. At each endpoint frame $i$, the history contains $\min(8, i+1)$ uniformly spaced indices from 0 to $i$, rounded to the nearest frame. Robometer's prediction for the final sampled frame is taken as the current progress estimate, using no frames after that timestamp.

At timestamps $t_0, \ldots, t_K$, the real and generated progress estimates are

$$p_j = R_\psi\left(V^{\text{real}}_{\le t_j}, g\right), \qquad \hat{p}_j = R_\psi\left(V^{\text{gen}}_{\le t_j}, g\right). \tag{11}$$

The evaluator $R_\psi$ includes both the history-sampling procedure and endpoint progress estimation described above, conditioned on the task description $g$. Progress values lie in $[0, 1]$.

The two progress curves are compared at corresponding physical timestamps without dynamic time warping. This preserves discrepancies caused by generated executions progressing too quickly, too slowly, or becoming stalled.

#### D.4.3 TIME-WEIGHTED ERROR AND SIMILARITY

For trajectories with positive duration, Equation 1 applies trapezoidal integration to the pointwise discrepancies $e_j = |p_j - \hat{p}_j|$. Each interval contributes its duration multiplied by the mean discrepancy at its two endpoints. Dividing the accumulated error by $t_K - t_0$ gives the time-averaged error $E_{\text{prog}}$, which is converted to similarity as $S_{\text{prog}} = 100(1 - E_{\text{prog}})$.

Time weighting ensures that intervals contribute according to their actual duration, preventing a final interval shorter than two seconds from receiving the same weight as a full sampling interval. Lower $E_{\text{prog}}$ and higher $S_{\text{prog}}$ indicate greater agreement between real and generated progress trajectories, with $S_{\text{prog}} \in [0, 100]$.

If a source recording contains reconstruction gaps, each valid continuous segment is scored with its own causal history. Segment errors are weighted by their elapsed duration to obtain one source-recording score; no interval is integrated across a gap. The resulting scores follow the aggregation weights in Appendix D.1; uninterrupted trajectories are used for the long-horizon comparison in Appendix D.5.

### D.5 THREE-STAGE LONG-HORIZON STABILITY

**Temporal partition and prediction continuity.** For a continuous trajectory with $N$ frames at frame rate $f$, we use elapsed timestamps $t_i = i/f$ and duration $T = (N-1)/f$. The three intervals are $[0, T/3)$, $[T/3, 2T/3)$, and $[2T/3, T]$. Each frame belongs to exactly one interval. All stages come from the original autoregressive prediction; the partition introduces no additional reference frames, generation resets, or changes to the supplied actions. The initial conditioning frame remains included, consistently with the whole-trajectory protocol. Stage position is relative to each trajectory's duration, not a shared absolute timestamp.

**Video fidelity.** We recompute MAE, MSE, PSNR, and SSIM within each stage using the inputs and metric settings in Appendix D.2. PSNR is computed from the MSE over the entire stage, rather than by averaging frame-level PSNR. Streaming reductions use double precision and are checked against the saved whole-trajectory scores. Frame FID and FVD remain distributional measures in the whole-trajectory tables; they are not treated as per-trajectory stability scores from three short clips.

**Motion consistency.** The original IDM predictions and valid windows are reused. A window is assigned to the stage containing its physical midpoint, retaining the evaluator's original gap and stride. Windows may straddle an artificial third boundary, but never a source discontinuity. Each stage averages valid-window component errors using the original effector aggregation. Simulation's 15 Hz evaluator indices are mapped through the saved native-frame indices to the 25 Hz video timeline.

Position, rotation, and gripper discrepancies retain their respective units; their degradation values are not pooled into one motion score.

**Progress consistency.** Let $e(t)$ be the piecewise-linear interpolation of the saved absolute progress discrepancies $e_j = |p_j - \hat{p}_j|$. The stage score is

$$S^{(k)}_{\text{prog}} = 100 \left[ 1 - \frac{3}{T} \int_{(k-1)T/3}^{kT/3} e(t)\, dt \right], \qquad k \in \{1, 2, 3\}. \tag{12}$$

We insert boundary values by linear interpolation of $e_j$ and integrate using the trapezoidal rule. This preserves the original whole-trajectory score exactly as the mean of the three stage scores. Each saved progress estimate uses the prefix from the original continuous trajectory start; progress is neither re-estimated from a cropped third nor renormalized within a stage.

**Aggregation and uncertainty.** We compute $D_m$ for each source trajectory in the uninterrupted Long rollout subset shared by all three models. Aggregation and paired bootstrapping follow Appendix D.1, with each trajectory resampled together with all its stages; stages, frames, and windows are not independent replicates. Overall results use equal category weights, while platform summaries average tasks equally within the platform.

**Interpretation.** A positive $D_m$ indicates poorer late-stage agreement; a negative value indicates improvement. Both stage scores and degradation must be considered: uniformly poor predictions may have little degradation, and task stages may differ in difficulty. The middle-stage score exposes nonmonotonic changes that the endpoint difference alone may conceal. The metric characterizes temporal changes under the fixed evaluators, rather than isolating accumulated state drift or directly measuring physical correctness. In particular, IDM degradation tracks changes in local motion disagreement, not integrated pose error.

# E Additional World-Model Results and Ablations

## E.1 World-model prediction results

Table 11 complements the main results in Table 2 with Evaluation and whole-trajectory Long rollout scores on the task set shared by all three models. All metric families use identical task coverage and equal category weights, following Appendix D.1.

Table 11: World-model prediction on shared tasks. Evaluation and Long rollout results use the same non-Ego task set for all models. Frame FID and FVD are computed within each task before aggregation.

| | Video quality | | | | | | IDM discrepancy | | | Progress |
|---|---|---|---|---|---|---|---|---|---|---|
| Model | MAE ↓ | MSE ↓ | PSNR ↑ (dB) | SSIM ↑ | Frame FID ↓ | FVD ↓ (R3D-18) | Position (mm) ↓ | Rotation (°) ↓ | Gripper (norm.) ↓ | Progress (0–100) ↑ |
| *Evaluation* | | | | | | | | | | |
| Nano | **0.0624** | **0.0161** | 18.39 | **0.6342** | **19.06** | **9.63** | **11.07** | **2.07** | 0.0189 | **92.27** |
| Edge | 0.0629 | **0.0161** | **18.51** | 0.6341 | 23.18 | 12.28 | 11.43 | 2.09 | **0.0174** | 91.78 |
| GigaWorld | 0.0794 | 0.0217 | 17.10 | 0.5848 | 34.90 | 18.48 | 16.32 | 2.88 | 0.0239 | 89.23 |
| *Long rollout* | | | | | | | | | | |
| Nano | **0.0930** | **0.0281** | **15.97** | **0.5333** | **35.59** | **21.82** | **16.86** | **3.04** | 0.0343 | **87.35** |
| Edge | 0.0950 | 0.0289 | 15.96 | 0.5308 | 41.31 | 26.72 | 17.15 | 3.08 | **0.0314** | 86.84 |
| GigaWorld | 0.1106 | 0.0348 | 15.16 | 0.4976 | 52.83 | 33.99 | 22.18 | 3.90 | 0.0364 | 85.98 |

All ten categories receive equal weight. Gripper is normalized by the IDM training range. Bold marks the best mean at the displayed precision, not statistical significance.

**Progress-score uncertainty.** Table 12 reports paired progress differences using the bootstrap protocol in Appendix D.1. Nano–Edge includes Ego; comparisons with GigaWorld use the shared non-Ego task set. The Evaluation interval supports higher progress consistency for Nano than Edge,

whereas the Long rollout intervals do not establish a reliable Nano–Edge or Edge–GigaWorld ordering.

Table 12: Paired progress-similarity differences (first model minus second) on the tasks shared by each pair. Brackets give exploratory 95% intervals from within-task trajectory resampling with equal category weights.

| Comparison | Evaluation | Long rollout |
|---|---|---|
| Nano–Edge | $0.59\ [0.18, 1.03]$ | $0.63\ [-0.30, 1.49]$ |
| Nano–GigaWorld | $3.04\ [2.45, 3.63]$ | $1.37\ [0.29, 2.44]$ |
| Edge–GigaWorld | $2.55\ [1.98, 3.12]$ | $0.87\ [-0.11, 1.85]$ |

**Platform-level results.** Tables 13 and 14 report all metrics with equal task weights within each platform. Since platforms contain different tasks, these summaries describe conditional performance rather than isolating platform effects. On ALOHA Evaluation, Edge has higher PSNR than Nano, while Nano has higher progress similarity. On Flexiv Long rollout, GigaWorld leads in progress and all three IDM components despite having the lowest overall progress mean on the shared task set. Edge has the lowest Long rollout position discrepancy on ALOHA and Tianji. These differences support retaining visual, motion, and progress measurements separately.

Ego camera, wrist, and fingertip discrepancies are reported separately in Table 15, since their motion representations differ from robot or tool motion.

Table 13: Platform-level Evaluation results. Tasks receive equal weight within each platform; each contributes 20 trajectories. Bold marks the best mean at the displayed precision. Dashes denote Ego components reported in Table 15. GigaWorld was not evaluated on Ego.

| Model | MAE ↓ | MSE ↓ | PSNR ↑ (dB) | SSIM ↑ | Frame FID ↓ | FVD ↓ (R3D-18) | Position (mm) ↓ | Rotation (°) ↓ | Gripper (norm.) ↓ | Progress (0–100) ↑ |
|---|---|---|---|---|---|---|---|---|---|---|
| *ALOHA* | | | | | | | | | | |
| Nano | 0.0614 | 0.0160 | 18.08 | 0.7250 | 20.78 | 12.04 | **18.14** | **4.92** | 0.0392 | **90.89** |
| Edge | **0.0574** | **0.0143** | **18.59** | **0.7365** | **20.74** | **10.87** | 18.49 | 5.04 | 0.0223 | 90.38 |
| GigaWorld | 0.0713 | 0.0184 | 17.42 | 0.7044 | 30.45 | 22.76 | 24.28 | 6.15 | **0.0207** | 84.75 |
| *Tianji* | | | | | | | | | | |
| Nano | 0.0316 | 0.0071 | 21.66 | 0.8298 | **28.97** | 13.01 | **10.05** | **2.64** | 0.0253 | **92.04** |
| Edge | **0.0285** | **0.0059** | **22.47** | **0.8416** | 30.79 | **11.49** | 10.14 | 2.72 | **0.0235** | 91.06 |
| GigaWorld | 0.0376 | 0.0080 | 21.04 | 0.8153 | 42.09 | 16.06 | 13.01 | 3.49 | 0.0246 | 89.06 |
| *Flexiv* | | | | | | | | | | |
| Nano | 0.0664 | 0.0184 | 17.50 | 0.6882 | **19.56** | **8.06** | 16.97 | 2.72 | 0.0340 | **81.46** |
| Edge | **0.0615** | 0.0165 | 17.98 | 0.7035 | 20.88 | 8.48 | **16.66** | **2.67** | **0.0330** | 80.53 |
| GigaWorld | 0.0642 | **0.0148** | **18.36** | **0.7118** | 22.85 | 10.99 | 17.45 | 2.78 | 0.0395 | 78.57 |
| *Astribot* | | | | | | | | | | |
| Nano | 0.0732 | 0.0214 | 16.77 | 0.6228 | 19.00 | 8.36 | 17.35 | 2.84 | 0.0125 | 90.89 |
| Edge | **0.0581** | **0.0153** | **18.28** | **0.6628** | **18.84** | **8.08** | **14.67** | **1.91** | **0.0124** | **91.04** |
| GigaWorld | 0.0761 | 0.0202 | 16.98 | 0.6302 | 37.27 | 15.63 | 16.27 | 2.07 | 0.0134 | 89.36 |
| *Finger* | | | | | | | | | | |
| Nano | **0.0752** | **0.0196** | **17.20** | **0.5500** | **24.15** | **12.43** | **6.75** | **1.71** | **0.0184** | **92.52** |
| Edge | 0.0793 | 0.0207 | 16.99 | 0.5388 | 32.29 | 19.35 | 7.19 | 1.78 | 0.0188 | 92.22 |
| GigaWorld | 0.0866 | 0.0234 | 16.43 | 0.5235 | 44.05 | 20.57 | 11.53 | 2.60 | 0.0291 | 88.11 |
| *Gripper* | | | | | | | | | | |
| Nano | **0.0648** | **0.0156** | **18.36** | **0.5762** | **15.08** | **7.95** | **9.48** | **1.10** | 0.0126 | **94.27** |
| Edge | 0.0687 | 0.0169 | 18.01 | 0.5656 | 19.81 | 10.95 | 10.35 | 1.21 | **0.0118** | 93.55 |
| GigaWorld | 0.0950 | 0.0268 | 15.90 | 0.4852 | 31.21 | 18.90 | 16.40 | 2.05 | 0.0170 | 91.83 |

Continued on next page

Table 13 (continued)

| Model | MAE ↓ | MSE ↓ | PSNR ↑ (dB) | SSIM ↑ | Frame FID ↓ | FVD ↓ (R3D-18) | Position (mm) ↓ | Rotation (°) ↓ | Gripper (norm.) ↓ | Progress (0–100) ↑ |
|---|---|---|---|---|---|---|---|---|---|---|
| *UMI* | | | | | | | | | | |
| Nano | 0.0344 | 0.0056 | 22.66 | 0.7473 | **9.25** | **5.07** | **8.15** | **1.16** | **0.0182** | **90.44** |
| Edge | **0.0327** | **0.0051** | **23.19** | **0.7667** | 9.56 | 5.88 | 8.41 | **1.16** | 0.0185 | 89.92 |
| GigaWorld | 0.0509 | 0.0098 | 20.13 | 0.6390 | 20.00 | 8.58 | 13.53 | 1.84 | 0.0306 | 87.61 |
| *Simulation* | | | | | | | | | | |
| Nano | 0.0362 | 0.0108 | 19.87 | 0.8109 | **17.72** | 10.67 | **14.70** | **4.65** | **0.0187** | 94.03 |
| Edge | **0.0315** | **0.0086** | **20.80** | **0.8269** | 18.39 | **9.27** | 14.86 | 4.81 | 0.0188 | **94.40** |
| GigaWorld | 0.0452 | 0.0129 | 19.01 | 0.7760 | 31.75 | 19.46 | 18.16 | 5.60 | 0.0217 | 93.62 |
| *Ego* | | | | | | | | | | |
| Nano | **0.0430** | **0.0100** | **20.13** | **0.7533** | **20.72** | **10.54** | – | – | – | **90.39** |
| Edge | 0.0448 | 0.0101 | 20.07 | 0.7528 | 30.75 | 21.75 | – | – | – | 89.67 |

Table 14: Platform-level Long rollout results. Tasks receive equal weight within each platform; each contributes 5 trajectories. Bold marks the best mean at the displayed precision. Dashes denote Ego components reported in Table 15. GigaWorld was not evaluated on Ego.

| Model | MAE ↓ | MSE ↓ | PSNR ↑ (dB) | SSIM ↑ | Frame FID ↓ | FVD ↓ (R3D-18) | Position (mm) ↓ | Rotation (°) ↓ | Gripper (norm.) ↓ | Progress (0–100) ↑ |
|---|---|---|---|---|---|---|---|---|---|---|
| *ALOHA* | | | | | | | | | | |
| Nano | 0.0687 | 0.0180 | 17.48 | 0.6997 | **29.47** | 20.03 | 20.10 | **4.89** | 0.0382 | **88.24** |
| Edge | **0.0637** | **0.0162** | **17.97** | **0.7113** | 32.88 | **17.10** | **19.62** | 4.91 | 0.0248 | 88.18 |
| GigaWorld | 0.0765 | 0.0199 | 17.03 | 0.6845 | 40.27 | 28.05 | 23.82 | 5.94 | **0.0203** | 85.20 |
| *Tianji* | | | | | | | | | | |
| Nano | 0.0514 | 0.0133 | 18.92 | 0.7401 | **51.46** | 23.51 | 13.67 | **3.33** | 0.0289 | **88.92** |
| Edge | **0.0477** | **0.0119** | **19.56** | **0.7498** | 52.03 | **21.83** | **13.50** | 3.41 | **0.0285** | 86.19 |
| GigaWorld | 0.0545 | 0.0130 | 18.94 | 0.7442 | 63.05 | 24.69 | 16.66 | 4.22 | 0.0325 | 86.45 |
| *Flexiv* | | | | | | | | | | |
| Nano | 0.0939 | 0.0291 | 15.41 | 0.6172 | 39.87 | **17.37** | 19.75 | 2.97 | 0.0630 | 69.91 |
| Edge | 0.0865 | 0.0263 | 15.91 | 0.6376 | 38.60 | 18.16 | 18.77 | 2.94 | 0.0646 | 71.94 |
| GigaWorld | **0.0829** | **0.0208** | **16.92** | **0.6600** | **35.92** | 20.03 | **18.09** | **2.87** | **0.0575** | **73.25** |
| *Astribot* | | | | | | | | | | |
| Nano | 0.0902 | 0.0280 | 15.63 | 0.5807 | **33.14** | 21.73 | 19.84 | 3.55 | **0.0216** | 86.16 |
| Edge | **0.0785** | **0.0229** | **16.57** | **0.6032** | 34.27 | **19.61** | **15.24** | **2.17** | 0.0243 | **86.71** |
| GigaWorld | 0.0891 | 0.0249 | 16.06 | 0.5941 | 48.27 | 26.79 | 16.35 | 2.29 | 0.0219 | 86.07 |
| *Finger* | | | | | | | | | | |
| Nano | **0.1151** | **0.0360** | **14.54** | **0.4314** | **50.76** | **31.33** | **12.94** | **3.39** | 0.0417 | **89.76** |
| Edge | 0.1247 | 0.0397 | 14.12 | 0.4084 | 66.84 | 47.46 | 13.27 | 3.52 | **0.0407** | 84.69 |
| GigaWorld | 0.1380 | 0.0452 | 13.51 | 0.3975 | 80.76 | 56.08 | 18.76 | 4.23 | 0.0493 | 83.06 |
| *Gripper* | | | | | | | | | | |
| Nano | **0.1002** | **0.0295** | **15.60** | **0.4545** | **25.73** | **17.57** | **15.86** | **1.76** | 0.0262 | 87.60 |
| Edge | 0.1059 | 0.0319 | 15.19 | 0.4460 | 31.05 | 22.74 | 17.12 | 1.89 | **0.0227** | **88.69** |
| GigaWorld | 0.1270 | 0.0409 | 14.11 | 0.3895 | 41.58 | 29.54 | 23.62 | 2.87 | 0.0278 | 88.55 |
| *UMI* | | | | | | | | | | |
| Nano | **0.0568** | 0.0122 | 19.17 | 0.5923 | **19.64** | **9.60** | 12.44 | 1.75 | 0.0285 | **85.82** |
| Edge | 0.0572 | **0.0121** | **19.19** | **0.6008** | 20.11 | 10.44 | **12.39** | **1.73** | **0.0260** | 84.77 |
| GigaWorld | 0.0772 | 0.0181 | 17.52 | 0.5208 | 36.92 | 15.54 | 17.35 | 2.41 | 0.0440 | 83.07 |

Continued on next page

Table 14 (continued)

| Model | MAE ↓ | MSE ↓ | PSNR ↑ (dB) | SSIM ↑ | Frame FID ↓ | FVD ↓ (R3D-18) | Position (mm) ↓ | Rotation (°) ↓ | Gripper (norm.) ↓ | Progress (0–100) ↑ |
|---|---|---|---|---|---|---|---|---|---|---|
| *Simulation* | | | | | | | | | | |
| Nano | 0.0506 | 0.0170 | 17.90 | 0.7768 | 39.74 | 29.04 | **20.96** | **7.07** | 0.0516 | 84.44 |
| Edge | **0.0444** | **0.0141** | **18.71** | **0.7917** | **36.81** | **25.48** | 20.99 | 7.47 | 0.0413 | 85.77 |
| GigaWorld | 0.0549 | 0.0175 | 17.73 | 0.7524 | 44.81 | 31.27 | 23.34 | 7.94 | **0.0372** | **91.62** |
| *Ego* | | | | | | | | | | |
| Nano | **0.0604** | **0.0149** | **18.44** | **0.6905** | **41.45** | **20.64** | – | – | – | **88.58** |
| Edge | 0.0686 | 0.0172 | 17.83 | 0.6717 | 57.14 | 37.39 | – | – | – | 87.12 |

Table 15: Ego motion discrepancies over tasks 038 and 039, with equal task weights. Camera, wrist, and fingertip components are reported separately; all are errors (lower is better). Bold marks the lowest mean.

| | Evaluation | | Long rollout | |
|---|---|---|---|---|
| Component | Nano | Edge | Nano | Edge |
| Camera position (mm) | **30.92** | 34.53 | **33.66** | 36.81 |
| Camera rotation (°) | **2.00** | 2.17 | **2.20** | 2.33 |
| Left wrist position (mm) | **40.96** | 46.80 | **40.87** | 46.33 |
| Left wrist rotation (°) | **8.21** | 9.27 | **6.37** | 7.46 |
| Right wrist position (mm) | **43.34** | 48.72 | **43.04** | 48.06 |
| Right wrist rotation (°) | **7.96** | 9.23 | **7.79** | 8.54 |
| Left fingertip (mm) | **10.42** | 12.54 | **11.36** | 13.24 |
| Right fingertip (mm) | **9.20** | 10.85 | **9.48** | 10.63 |

### E.2 Long-horizon stability results

Table 16 reports all trajectory-level stability components on the shared task set. Table 17 gives the corresponding platform summaries. Stage scores retain their original units; all degradation values use the sign convention in Equation 2.

Table 16: Three-stage Long rollout results. $D_m$ is signed degradation with its paired trajectory-bootstrap 95% interval.

| Metric | Model | Early | Middle | Late | $D_m$ [95% CI] |
|---|---|---|---|---|---|
| MAE | Nano | 0.0698 | 0.1004 | 0.1088 | $+0.0390\ [0.0366, 0.0414]$ |
| | Edge | 0.0700 | 0.1027 | 0.1124 | $+0.0424\ [0.0404, 0.0446]$ |
| | GigaWorld | 0.0857 | 0.1190 | 0.1271 | $+0.0414\ [0.0390, 0.0437]$ |
| MSE | Nano | 0.0191 | 0.0308 | 0.0344 | $+0.0153\ [0.0140, 0.0165]$ |
| | Edge | 0.0190 | 0.0318 | 0.0358 | $+0.0168\ [0.0157, 0.0180]$ |
| | GigaWorld | 0.0244 | 0.0378 | 0.0421 | $+0.0178\ [0.0166, 0.0189]$ |
| PSNR (dB) | Nano | 17.76 | 15.62 | 15.09 | $+2.66\ [2.49, 2.83]$ |
| | Edge | 17.89 | 15.60 | 15.01 | $+2.89\ [2.74, 3.03]$ |
| | GigaWorld | 16.67 | 14.81 | 14.41 | $+2.25\ [2.14, 2.37]$ |
| SSIM | Nano | 0.6089 | 0.5038 | 0.4873 | $+0.1217\ [0.1159, 0.1277]$ |
| | Edge | 0.6104 | 0.5002 | 0.4818 | $+0.1287\ [0.1232, 0.1340]$ |
| | GigaWorld | 0.5623 | 0.4707 | 0.4598 | $+0.1024\ [0.0939, 0.1105]$ |
| Position (mm) | Nano | 12.51 | 18.75 | 19.31 | $+6.80\ [5.64, 8.09]$ |
| | Edge | 12.64 | 19.02 | 19.79 | $+7.15\ [5.67, 8.91]$ |
| | GigaWorld | 17.06 | 24.31 | 25.16 | $+8.09\ [7.19, 8.96]$ |

Continued on next page

Table 16 (continued)

| Metric | Model | Early | Middle | Late | $D_m$ [95% CI] |
|---|---|---|---|---|---|
| Rotation (°) | Nano | 2.38 | 3.36 | 3.38 | $+0.99$ $[0.83, 1.16]$ |
| | Edge | 2.42 | 3.41 | 3.39 | $+0.97$ $[0.78, 1.15]$ |
| | GigaWorld | 3.10 | 4.32 | 4.28 | $+1.18$ $[1.01, 1.35]$ |
| Gripper (norm.) | Nano | 0.0254 | 0.0367 | 0.0407 | $+0.0153$ $[0.0117, 0.0191]$ |
| | Edge | 0.0215 | 0.0338 | 0.0388 | $+0.0173$ $[0.0136, 0.0210]$ |
| | GigaWorld | 0.0272 | 0.0400 | 0.0421 | $+0.0149$ $[0.0117, 0.0181]$ |
| Progress (points) | Nano | 92.33 | 87.66 | 82.06 | $+10.26$ $[8.51, 12.10]$ |
| | Edge | 91.81 | 86.49 | 82.23 | $+9.58$ $[7.76, 11.35]$ |
| | GigaWorld | 89.86 | 84.65 | 83.42 | $+6.44$ $[5.12, 7.75]$ |

Table 17: Temporal degradation by collection platform. Tasks receive equal weight within each platform. Positive values indicate deterioration.

| Platform | Model | PSNR drop (dB) | Position increase (mm) | Progress drop (points) |
|---|---|---|---|---|
| ALOHA | Nano | +0.36 | +0.53 | +5.70 |
| | Edge | +0.92 | +0.54 | +3.12 |
| | GigaWorld | +0.29 | -1.30 | +2.60 |
| Tianji | Nano | +3.02 | +5.21 | +7.33 |
| | Edge | +2.91 | +4.47 | +10.11 |
| | GigaWorld | +2.48 | +4.97 | +2.64 |
| Flexiv | Nano | +2.48 | +0.78 | +20.73 |
| | Edge | +2.60 | +0.25 | +20.29 |
| | GigaWorld | +1.55 | -1.34 | +12.45 |
| Astribot | Nano | +0.43 | -0.48 | +9.20 |
| | Edge | +1.28 | +0.15 | +7.16 |
| | GigaWorld | +1.15 | +0.12 | +5.86 |
| Finger | Nano | +2.43 | +6.47 | +6.31 |
| | Edge | +3.04 | +7.54 | +13.06 |
| | GigaWorld | +2.88 | +8.35 | +8.75 |
| Gripper | Nano | +3.22 | +8.28 | +14.16 |
| | Edge | +3.11 | +8.91 | +10.81 |
| | GigaWorld | +2.12 | +9.13 | +8.72 |
| UMI | Nano | +3.44 | +5.65 | +6.63 |
| | Edge | +4.03 | +4.51 | +3.36 |
| | GigaWorld | +1.62 | +3.10 | +3.68 |
| Simulation | Nano | +3.31 | +7.86 | +15.36 |
| | Edge | +3.61 | +7.03 | +11.05 |
| | GigaWorld | +3.36 | +5.54 | +6.02 |

The Nano–Edge difference in position degradation is $-0.35$ mm (95% interval $[-1.07, 0.30]$), and their progress-drop difference is $0.69$ points ($[-1.36, 2.81]$). These intervals do not establish a reliable ordering between the two models. GigaWorld loses fewer progress points than either model, but this does not imply better local motion prediction: its position errors are higher in all three stages.

### E.3 Effect of play data

Tables 18 and 19 compare Cosmos Edge trained on task-centered data and RoboWorld100 training data (*Play*), following Appendices C.3 and D. Results use equal task weights, with Evaluation and Long rollout reported separately.

Play improves all reported visual metrics and aggregate position and progress consistency on both splits. The paired progress-difference intervals are positive for both splits, although gains vary across tasks and several task-level progress intervals include zero.

Table 18: Aggregate motion and progress comparison with Cosmos Edge, using equal task weights. $\Delta S$ is the Play minus Task-centered progress score; brackets give its paired 95% bootstrap interval. Bold marks the better mean.

| | Position (mm) ↓ | | Progress (0–100) ↑ | | |
|---|---|---|---|---|---|
| Split | Task-centered | Play | Task-centered | Play | $\Delta S$ [95% CI] |
| Evaluation | 14.39 | **11.88** | 85.79 | **89.73** | 3.94 [2.73, 5.14] |
| Long rollout | 18.55 | **15.73** | 83.24 | **86.87** | 3.63 [1.87, 5.42] |

Table 19: Complete visual metrics for the task-centered and play data comparison with Cosmos Edge, using equal task weights. Bold marks the better mean.

| Data | MAE ↓ | MSE ↓ | PSNR ↑ (dB) | SSIM ↑ | Frame FID ↓ | FVD ↓ (R3D-18) |
|---|---|---|---|---|---|---|
| *Evaluation* | | | | | | |
| Task-centered | 0.0466 | 0.0135 | 20.00 | 0.8089 | 42.11 | 25.55 |
| Play | **0.0355** | **0.0091** | **21.49** | **0.8389** | **30.43** | **13.50** |
| *Long rollout* | | | | | | |
| Task-centered | 0.0657 | 0.0206 | 17.48 | 0.7513 | 63.60 | 47.88 |
| Play | **0.0529** | **0.0147** | **18.83** | **0.7695** | **47.43** | **26.31** |

# F Policy Evaluation and Real-Robot Transfer

Sections 4.4 and 4.5 report the completed four-task fixed-policy comparison and three-task real-robot RL evaluation. The controls below describe requirements for extending simulator comparisons and calibration across policy checkpoints; prospective controls should be distinguished from the completed results. The RL formulation and training settings follow in Section F.2.

## F.1 Policy success-rate estimation

**Controlled simulator comparison.** The Task-centered simulator is trained on task-centered data collected separately for comparison; Ours uses RoboWorld100 training data collected through task-relevant play. These conditions reflect collection protocols, not success/failure partitions of RoboWorld100. For controlled extensions, the simulators should share the world-model backbone, pretrained initialization, observation/action interface, and declared training-budget rule. Selected episode counts, frame exposure, and optimization updates should be reported for both conditions.

**Closed-loop trials.** For each task, fixed policies or checkpoints with different ability levels will run in both learned simulators and on the real robot. Trials share the task objective, policy checkpoint, observation/action interface, control frequency, and time limit, with matched initial-condition distributions. Subsequent actions are produced from each environment's own observations. The protocol evaluates closed-loop policy performance rather than replaying one recorded action sequence. Test initializations and source episodes are kept separate from policy training, world-model training, and evaluator selection; policy-only holdout is insufficient for this purpose.

**Completion labels and success rates.** Each task has an explicit completion criterion, applied consistently to simulated and real executions. Manual labels or independently validated task-specific outcome checks determine binary outcomes. Progress similarity does not directly supply these labels. For environment $E$, success rate $\hat{s}_E$ is the number of successful trials divided by the total number of trials. Table 5 reports ten trials per task and environment for the real robot and the two

learned simulators. For simulator $M$, signed bias is $100(\hat{s}_M - \hat{s}_{\text{real}})$ and absolute error is its magnitude, both in percentage points. Errors are computed for each task–policy pair before averaging, so overestimation and underestimation cannot cancel in the absolute-error summary. The reported comparison uses one policy per task and equal task weights.

**Uncertainty and policy ranking.** The reported success counts and rate gaps are point estimates based on ten trials per task and environment. A broader calibration study should include binomial confidence intervals and multiple policy checkpoints. Matched initializations support paired estimates of success-rate differences; otherwise trial sets should be treated as independent. Within-task policy-ranking correlations require multiple checkpoints and nonconstant success estimates; the present one-policy-per-task comparison does not establish such rankings.

### F.2 In-model reinforcement learning and transfer

#### F.2.1 Formulations

In Table 20, $x_t$ denotes the input to the noise policy, $o_t$ the visual observation, $s_t$ the state feedback, $h_t$ the visual history, and $\ell$ the task instruction. The frozen action generator $G_{\theta_0}$ repeats a single noise vector over a horizon $H$ and executes the first $K$ actions. The joint index set $\mathcal{J}$ excludes grippers. The control interval is $\delta$; $\Delta_j$, $s_j^v$, and $s_j^a$ are task-specific command limits and motion scales. Command and velocity history are retained across action chunks. The replay buffer is $\mathcal{D}$, $\mathcal{I}$ is a randomly sampled subset of target critics, and sg denotes stop-gradient. The terminal flag $d_t$ excludes time-limit truncation.

Table 20: Formulations for latent-noise optimization and reward computation.

| Component | Formulation |
|---|---|
| Latent-noise policy | $u_t \sim \mathcal{N}(\mu_\phi(x_t), \operatorname{diag}\sigma_\phi^2(x_t)), \quad z_t = b\tanh u_t$ |
| Action generation | $A_t^{\text{raw}} = [G_{\theta_0}(o_t, s_t, \ell; \mathbf{1}_H z_t^\top)]_{1:K}$ |
| Gaussian KL | $\mathcal{L}_{\text{KL}}(x) = \frac{1}{2}\sum_{i=1}^{d_z}\left[\frac{\sigma_{\phi,i}^2 + (\mu_{\phi,i} - \mu_{\text{ref},i})^2}{\sigma_{\text{ref},i}^2} - 1 + 2\log\frac{\sigma_{\text{ref},i}}{\sigma_{\phi,i}}\right]$ |
| Progress reward | $r_t^{\text{progress}} = R_{\psi_0}^{\text{progress}}(h_{t+1}, \ell)$ |
| Dynamics penalty | $r_t^{\text{dynamics}} = -\lambda_c C_t^c - \lambda_v C_t^v - \lambda_a C_t^a$ |
| Command projection | $a_{t,k,j} = \text{clip}\big(a_{t,k,j}^{\text{raw}}, \max\{l_j, a_{t,k-1,j} - \Delta_j\}, \min\{u_j, a_{t,k-1,j} + \Delta_j\}\big)$ |
| Projection correction cost | $C_t^c = \frac{1}{K\lvert\mathcal{J}\rvert}\sum_{k=1}^{K}\sum_{j\in\mathcal{J}}\left(\frac{a_{t,k,j}^{\text{raw}} - a_{t,k,j}}{\Delta_j}\right)^2$ |
| Command velocity and acceleration | $v_{t,k,j} = \frac{a_{t,k,j} - a_{t,k-1,j}}{\delta}, \quad \dot{v}_{t,k,j} = \frac{v_{t,k,j} - v_{t,k-1,j}}{\delta}$ |
| Normalized motion | $\tilde{v}_{t,k,j} = v_{t,k,j}/s_j^v, \quad \tilde{a}_{t,k,j} = \dot{v}_{t,k,j}/s_j^a$ |
| Squared excess | $E(x) = (\max\{\lvert x\rvert - 1, 0\})^2$ |
| Huber penalty | $H_1(x) = \frac{1}{2}\min(\lvert x\rvert, 1)^2 + \max(\lvert x\rvert - 1, 0)$ |
| Velocity cost | $C_t^v = \sum_{k=1}^{K}\sum_{j\in\mathcal{J}}[\tilde{v}_{t,k,j}^2 + E(\tilde{v}_{t,k,j})]$ |

*Table 20 continued*

| Component | Formulation |
|---|---|
| Acceleration cost | $C_t^a = \sum_{k=1}^{K} \sum_{j \in \mathcal{J}} [H_1(\tilde{a}_{t,k,j}) + E(\tilde{a}_{t,k,j})]$ |
| Critic target | $y_t = r_t + \gamma^K (1 - d_t) \frac{1}{\lvert\mathcal{I}\rvert} \sum_{j \in \mathcal{I}} Q_{\bar{\omega}_j}(x_{t+1}, z_{t+1}),$ |
| | $z_{t+1} \sim \pi_\phi(\cdot \mid x_{t+1})$ |
| Critic loss | $\mathcal{L}_Q = \mathbb{E}_{\mathcal{D}} \left[ \frac{1}{N_Q} \sum_{j=1}^{N_Q} (Q_{\omega_j}(x_t, z_t) - \mathrm{sg}(y_t))^2 \right]$ |
| Temperature loss | $\mathcal{L}_\alpha(\xi) = -\mathbb{E}_{x \sim \mathcal{D},\, z \sim \pi_\phi(\cdot \mid x)}$ |
| | $\left[ \xi \, \mathrm{sg}(\log \pi_\phi(z \mid x) + \mathcal{H}_{\mathrm{tar}}) \right], \qquad \xi = \log \alpha$ |
| Target-critic update | $\bar{\omega}_j \leftarrow (1 - \tau)\bar{\omega}_j + \tau \omega_j$ |

### F.2.2 Training Settings

Table 21: Shared training settings across the three tasks. Decimal hyperparameters are reported to three decimal places.

| Setting | Value |
|---|---|
| Noise dimension $d_z$ | 32 |
| Noise bound $b$ | 1 |
| Action generation horizon $H$ | 50 |
| Executed action length $K$ | 5 |
| Actor hidden dimensions | [512, 512] |
| Critic hidden dimensions | [768, 512, 256] |
| Initial mean-head gain | 0.010 |
| Initial pre-tanh standard deviation | 0.100 |
| KL coefficient $\beta$ | 0.010 |
| Number of critics $N_Q$ | 5 |
| Target-critic subset size | 2 |
| Target entropy $\mathcal{H}_{\mathrm{tar}}$ | $-16$ |
| Actor / critic learning rate | $10^{-5}$ / $10^{-5}$ |
| Temperature learning rate | $3 \times 10^{-4}$ |
| Actor / critic weight decay | $10^{-6}$ / $10^{-6}$ |
| Batch size | 128 |
| discount $\gamma$ | 0.990 |
| Critic updates per actor update | 1 |
| UTD | 1 |
| Target smoothing coefficient $\tau$ | 0.005 |
| Warmup steps | 25,000 |
| Max training steps | 500,000 |
| Max episode steps | 240 |
| Progress-reward weight | 1 |
| Correction coefficient $\lambda_c$ | 0.010 |
| Velocity-cost coefficient $\lambda_v$ | $1.974 \times 10^{-3}$ |
| Acceleration-cost coefficient $\lambda_a$ | $4.798 \times 10^{-3}$ |