# DREAM4ACT: A SHARED VISUAL ACTION INTERFACE FOR MULTI-EMBODIMENT VIDEO–ACTION MODELING

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

## ABSTRACT

Video generation models encode rich spatiotemporal priors about scene dynamics. To serve robot control, a general video-action model must reason bidirectionally: predicting future observations under given actions, and inferring actions from observed or desired outcomes. Existing approaches fall short—either predicting joint-space vectors through dedicated action heads that vary across embodiments, or visualizing only end-effector quantities that discard full-arm articulation and still require embodiment-specific inverse-kinematics mappings. We present **Dream4ACT**, a video-action world model that renders target joint configurations as four **action views** via URDF-based forward kinematics from prescribed virtual cameras independent of physical observation cameras. These visual sequences preserve embodiment-specific geometry while sharing a video-modeling pipeline with RGB observations. A diffusion transformer jointly processes multi-view physical-camera observations and four action-view streams, conditioned on semantic features from a frozen vision-language model followed by a trainable adapter. Masked flow matching supports forward dynamics, inverse dynamics, and joint observation-action generation by varying which future sequences are corrupted. For execution, training-free multiview matching against URDF-based renderings recovers joint targets without a learned embodiment-specific decoder. Dream4ACT achieves 88.98% success rate on RoboTwin 2.0, and 65.66 score on TriWorldBench. Ablation studies further confirm the benefit of action-view conditioning for high-quality video generation.

## 1 INTRODUCTION

Video–action world models offer a promising foundation for embodied intelligence by connecting visual prediction with robot control. A general model should support two complementary directions: predicting future observations under candidate controls and inferring controls consistent with observed or desired future observations. Large-scale video pretraining provides spatiotemporal priors (Wan et al., 2025), motivating growing interest in adapting video models for robot prediction and control (Cheang et al., 2024; Li et al., 2026b).

Recent works use video models either to provide visual plans or predictive features for a separate controller (Du et al., 2023; Hu et al., 2024), or to jointly predict observations and actions, often through modality-specific heads (Li et al., 2025; 2026b). Joint-space vectors lack explicit spatial structure and vary across embodiments. Shared policies support heterogeneous action spaces (Team et al., 2024; Liu et al., 2025), while latent-frame encodings reuse video generation for action prediction (Kim et al., 2026). Visual interfaces further represent end-effector quantities as images (Li et al., 2026c; Zhen et al., 2026), but do not explicitly encode the full arm configuration. Executing these end-effector commands still requires an embodiment-specific mapping to joint targets, potentially with singular solutions or no solutions.

Our key insight is to represent target articulated configurations as visual content modeled alongside RGB observations. For joint-controlled manipulation, we seek a shared visual interface that retains arm and gripper configuration information, supports joint training across embodiments, and permits recovery of executable joint targets.



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.

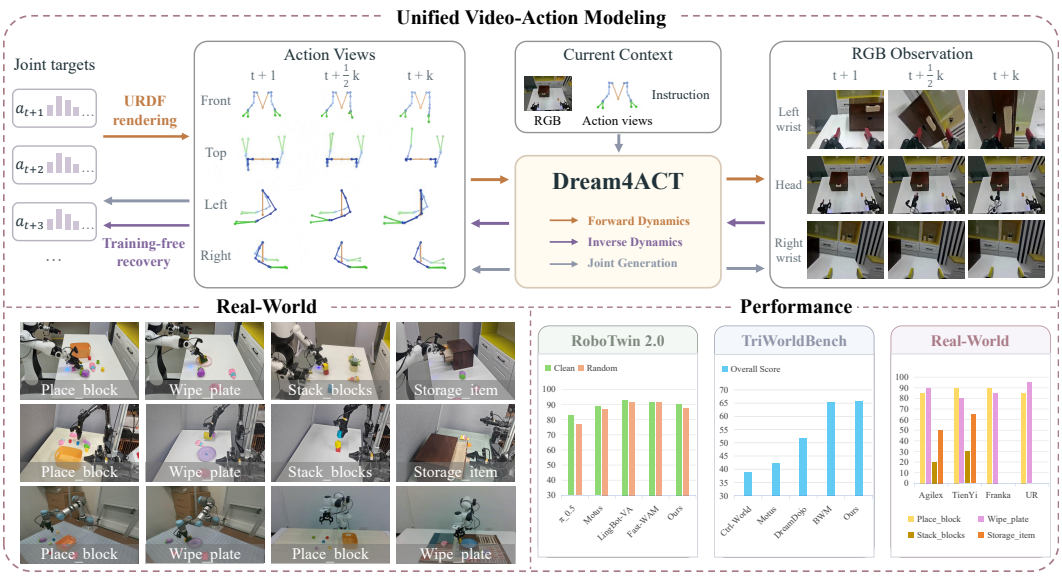



Figure 1: **Overview of Dream4ACT.** *Top:* A shared action-view interface supports forward dynamics, inverse dynamics, and joint generation, with training-free recovery of joint targets. *Bottom left:* Examples of multi embodiments real-world task execution. *Bottom right:* Success rates and scores show strong performance in both robotic manipulation and action-conditioned video generation.

We instantiate this interface as *action views*: multiview images of target joint configurations obtained through URDF-based forward kinematics and rendering. Unlike observation-aligned robot renderings (Chen et al., 2026; Gu et al., 2026), our fixed virtual cameras are independent of physical observation cameras, removing the need for physical-camera extrinsic calibration in action-view construction. Saved camera presets remain fixed throughout training and inference, with consistent view roles across embodiments. A fixed tensor shape enables shared tokenization and prediction across embodiments, while the rendered articulated geometry preserves the information needed for configuration matching. Joint targets are then recovered by training-free, URDF-constrained optimization, without a learned embodiment-specific action decoder.

Building on this representation, **Dream4ACT** jointly models physical-camera observations and four action-view streams with a shared video autoencoder and diffusion transformer, conditioned on instruction-grounded scene features. Masked flow matching selects future conditioning and prediction streams, enabling forward dynamics, inverse dynamics, and joint generation with shared weights across jointly trained embodiments. Our model achieves 88.98% average success on RoboTwin 2.0 and a TriWorldBench score of 65.66 using a separately trained checkpoint; closed-loop evaluations cover five simulated embodiments and four real-world platforms. A separately trained ablation compares action-view rendering with camera-aligned skeleton conditioning on the same recorded joint-state sequences. Together, these evaluations demonstrate the interface's strong capabilities both as a generated representation recoverable into executable commands and as a conditioning signal for future RGB prediction.

In summary, our contributions are as follows:

- We introduce *action views*, a fixed-shape multiview representation of target joint configurations that allows heterogeneous joint spaces to share a common video-modeling interface while retaining embodiment-specific geometry.
- We develop a training-free, URDF-constrained multiview recovery to map predicted action views to joint targets without a learned embodiment-specific action decoder.
- Extensive experiments on simulation and real-world demonstrate that our model has strong performance on robotic manipulation and cross-view observation generation, with controlled ablation study validating the superior capability of generative visual quality.

## 2 Related Work

**Video–Action World Models.** Unified video–action models differ in how they couple modalities and select conditioning modes. UVA (Li et al., 2025) uses separate diffusion decoders with masked inputs, whereas UWM (Zhu et al., 2025) couples video and action diffusion with independent noise levels. The conditioning schedule and action representation are separate design choices: masking specifies which modalities are generated, whereas the representation determines how their contents encode control. Expert-based models coordinate modality-specific computation (Bi et al., 2026; Li et al., 2026b), while action-conditioned predictors such as Ctrl-World (Guo et al., 2026) generate multiview rollouts from numerical controls. Recent world action models further explore task generalization and embodiment adaptation (Ye et al., 2026), efficient inference (Yuan et al., 2026), and semantic or 3D alignment (Li et al., 2026d; Yang et al., 2026). Joint prediction and flexible conditioning are thus prior capabilities, distinct from the choice of action representation.

**Action Representations.** Standardized vector interfaces support cross-embodiment policies (Team et al., 2024; Liu et al., 2025), whereas rendered representations additionally expose spatial robot geometry. Latent actions (Bruce et al., 2024; Wei et al., 2026) and visual-motion representations (Ko et al., 2024; Bi et al., 2026) offer alternatives to explicit joint vectors, but their conversion to executable commands depends on the chosen decoder or controller. For example, Hydra-0 (Li et al., 2026a) uses a trained action head, while Masked Visual Actions (Alzayer et al., 2026) uses a learned inverse dynamics model. Visual interfaces differ in both the quantities they encode and their recovery mechanisms. Cosmos Policy (Kim et al., 2026) encodes actions as latent frames without a separate action-generation architecture. Sharing the video backbone therefore does not necessarily imply an explicit geometric encoding of the robot. SpatialVAM (Li et al., 2026c) geometrically recovers end-effector positions from virtual-view heatmaps but learns rotation and gripper decoding. Action Images (Zhen et al., 2026) geometrically recovers end-effector pose and gripper commands from multiview images and supports joint generation, action-conditioned prediction, and action labeling. Separately, BridgeV2W (Chen et al., 2026) and GeniWorld (Gu et al., 2026) condition video prediction on URDF-rendered motion aligned with observation viewpoints, requiring the corresponding camera geometry. Dream4ACT instead renders full target joint configurations in separate action-view streams and approximately recovers joint targets through URDF-constrained multiview matching. Its prescribed virtual cameras remove physical-camera extrinsic calibration from action-view construction and recovery, while the URDF supplies embodiment-specific geometry without a learned per-robot decoder head.

## 3 Method

**Problem formulation.** Our goal is to build a general video-action model. At time step $t$, the conditioning context is $c_t = (I_t, S_t, U, l)$, where $I_t$ comprises RGB images from head and wrist cameras, $S_t = (q_t, g_t)$ contains joint positions $q_t$ and gripper states $g_t$, $U$ is the URDF of the specific embodiment, and $l$ is the task instruction. Let $A_{t+1:t+k}$ denotes the target joint configurations and gripper states over a prediction horizon $k$. We model $p_\theta(\cdot)$ in three modes, distinguished by which future sequences are observed or generated:

$$\text{Forward Dynamics:} \quad p_\theta(I_{t+1:t+k} \mid c_t, A_{t+1:t+k}), \tag{1}$$

$$\text{Inverse Dynamics:} \quad p_\theta(A_{t+1:t+k} \mid c_t, I_{t+1:t+k}), \tag{2}$$

$$\text{Joint Generation:} \quad p_\theta(I_{t+1:t+k}, A_{t+1:t+k} \mid c_t). \tag{3}$$

In practice, the action variables are represented by action views; predicted views are converted back to joint configurations through the recovery procedure.

**Overview.** Dream4ACT uses a shared visual action interface (Figure 2). URDF rendering converts current and target joint configurations into four fixed virtual-camera action views (Section 3.1), while a trainable adapter over a frozen VLM extracts instruction-grounded conditions (Section 3.2). A shared video VAE separately encodes RGB and action-view sequences into latent tokens (Section 3.3). Conditioned on the adapter outputs, a diffusion transformer jointly models these tokens under masked flow matching, supporting forward dynamics, inverse dynamics, and joint generation (Section 3.4). Finally, training-free URDF-constrained Gauss–Newton matching recovers joint targets from generated action views (Section 3.5).

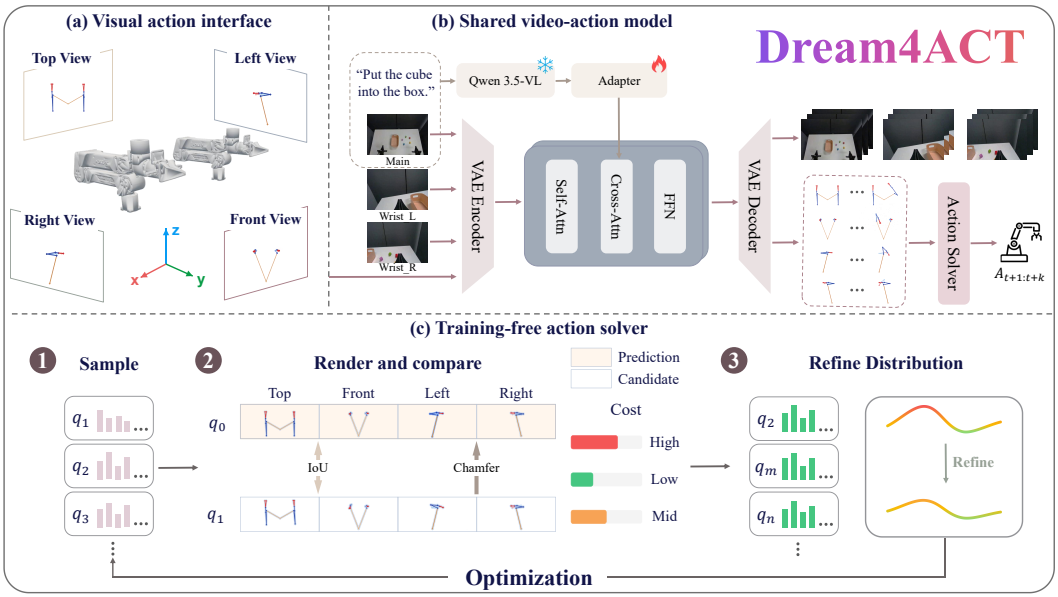



Figure 2: **Architecture of Dream4ACT.** *(a)* URDF-based rendering produces four action views. *(b)* A shared video VAE and diffusion transformer model RGB observations and action views with VLM-derived semantic conditioning, here we show the joint generation mode. *(c)* Training-free multiview action solver recovers joint targets from predicted views.

### 3.1 Visual Action Representation

We represent target states in image space to obtain a fixed-size action representation across embodiments with different dynamics and joint dimensions. Let $S_{t:t+k}$ denote the current and target configurations. For URDF $U$, action view $i$ at step $t'$ is

$$x_{t'}^{i} = R(\mathrm{FK}(S_{t'}; U); K_i, T_i), \quad i \in V, \quad t' = t, \ldots, t+k. \tag{4}$$

Here $R(\cdot)$ renders the articulated geometry, $\mathrm{FK}(\cdot)$ denotes forward kinematics and $K_i, T_i$ specify the intrinsics and extrinsics of virtual camera $i$. We use $|V| = 4$ cameras whose coordinate frames and rendering conventions are shared across embodiments. The resulting action views have a fixed shape independent of joint dimensionality while retaining embodiment-specific geometry. They therefore provide heterogeneous joint spaces with a common visual interface for shared tokenization and video modeling. More details about action rendering are provided in Appendix A.2.

### 3.2 Instruction-Grounded Semantic Adapter

We ground the task instruction $l$ in the current scene using a frozen pretrained vision-language model (VLM), Qwen3.5-VL 9B (Team, 2026). It jointly processes $l$ and the current head-camera frame $I_t^{\mathrm{head}}$. For a VLM with $L$ layers, the last four hidden layers are partitioned into visual and language tokens:

$$h^{(\ell)} = \left[h_{\mathrm{vis}}^{(\ell)}; h_{\mathrm{lang}}^{(\ell)}\right], \quad \ell \in \mathcal{L}_4 = \{L-3, L-2, L-1, L\}. \tag{5}$$

Then we concatenate each token group along the feature dimension across layers and project it to the adapter dimension $d$:

$$H_{\mathrm{vis}} = P_{\mathrm{vis}}\left(\mathrm{Concat}_{\ell \in \mathcal{L}_4} h_{\mathrm{vis}}^{(\ell)}\right), \quad H_{\mathrm{lang}} = P_{\mathrm{lang}}\left(\mathrm{Concat}_{\ell \in \mathcal{L}_4} h_{\mathrm{lang}}^{(\ell)}\right), \tag{6}$$

$P_{\mathrm{vis}}(\cdot)$ and $P_{\mathrm{lang}}(\cdot)$ denote visual and language projection layers separately. Then the adapter uses four learnable queries $Q \in \mathbb{R}^{4\times d}$ to compress the visual features:

$$\bar{Q} = \mathrm{CrossAttn}\left(Q, H_{\mathrm{vis}}, H_{\mathrm{vis}}\right). \tag{7}$$

The result is concatenated with the language tokens along the sequence dimension and refined by joint self-attention:

$$F_{\text{sem}} = \text{SelfAttn}\left([\bar{Q}; H_{\text{lang}}]\right) \in \mathbb{R}^{N_c \times d}, \quad N_c = 4 + N_{\text{lang}}. \tag{8}$$

Here, $N_{\text{lang}}$ is the language-token count. Adapter's structure is presented in Figure 4, it is trained jointly with the generative backbone.

### 3.3 Multi-Sequence Latent Tokenization

Over the window $t : t + k$, the model processes multi physical RGB observations $I_{t:t+k}$ and four action views $x_{t:t+k}$. We index the multi sequences by $s \in \mathcal{S} = \mathcal{S}_{\text{rgb}} \cup V$ and denote each by $y^s_{t:t+k}$, where $y^s = I^s$ for $s \in \mathcal{S}_{\text{rgb}}$ and $y^s = x^s$ for $s \in V$. A shared Wan2.2 VAE encoder (Wan et al., 2025) $\mathcal{E}$ is applied to each sequence separately,

$$z_s = \mathcal{E}(y^s_{t:t+k}), \quad s \in \mathcal{S}, \tag{9}$$

The latent grid is indexed by temporal positions $f$ and spatial positions $p$. Here $f = 0$ is the current frame latent and $f \geq 1$ denotes future latents. Before joint processing, each latent feature receives a modality and view embedding,

$$\tilde{z}_s = z_s + e^{\text{mod}}_{\mu(s)} + e^{\text{view}}_s, \quad s \in \mathcal{S}, \tag{10}$$

where $\mu(s) \in \{\text{rgb}, \text{action}\}$ selects one of two modality embeddings and $e^{\text{view}}_s$ identifies one of multi sequence slots. These embeddings are shared across embodiments. For rotary position embeddings (RoPE) (Su et al., 2024), the token at latent frame $f$ and spatial position $p$ receives

$$\phi(s, f, p) = (f,\ p + \delta_s), \tag{11}$$

where $\delta_s$ places sequence $s$ in a distinct spatial region. This shares the temporal coordinate while preventing spatial-position collisions across sequences. The annotated grids are then flattened and concatenated as $z = [\tilde{z}_s]_{s \in \mathcal{S}}$ to build the input of diffusion transformer.

### 3.4 Joint Diffusion Transformer

We choose a diffusion transformer (DiT) (Peebles & Xie, 2023) as our generative backbone. Each block applies joint self-attention across all input sequences, then makes cross-attention to the semantic condition $F_{\text{sem}}$ in Equation 8, and finally a position-wise feed-forward network (FFN).

Let $L$ denotes the sequence length of $s$, we extend conditional flow matching (Lipman et al., 2022; Liu et al., 2022) with modality-specific noise levels and a binary mask $m \in \{0, 1\}^{|\mathcal{S}| \times L}$. Sequence $s$ uses $\tau_s = \tau_{\text{rgb}}$ for $s \in \mathcal{S}_{\text{rgb}}$ and $\tau_s = \tau_{\text{act}}$ for $s \in V$. Clean and corrupted latents are represented jointly as

$$\tilde{z}^{\tau,m}_{s,f} = (1 - m_{s,f}) z_{s,f} + m_{s,f}\left[\tau_s z_{s,f} + (1 - \tau_s)\epsilon_{s,f}\right], \quad \epsilon_{s,f} \sim \mathcal{N}(0, \mathbf{I}), \tag{12}$$

where $m_{s,f} = 0$ preserves the data and $m_{s,f} = 1$ follows the linear path from noise at $\tau_s = 0$ to data at $\tau_s = 1$. The velocity field $v_\theta$ is trained only on corrupted positions:

$$\mathcal{L} = \mathbb{E}_{z,\epsilon,\boldsymbol{\tau},m}\left[\sum_{s \in \mathcal{S}} \sum_{f=0}^{L-1} m_{s,f} \left\| [v_\theta(\tilde{z}^{\boldsymbol{\tau},m}; \boldsymbol{\tau}, m, F_{\text{sem}})]_{s,f} - (z_{s,f} - \epsilon_{s,f}) \right\|_2^2\right], \tag{13}$$

where $\boldsymbol{\tau} = (\tau_s)_{s \in \mathcal{S}}$ collects the sequence-wise noise levels, $[\cdot]_{s,f}$ selects the output for sequence $s$ at latent frame $f$. The mask also selects the operating mode while keeping the current RGB and action-view latents at $f = 0$ clean:

$$m_{s,f} = \mathbb{1}[s \in M \wedge f \geq 1], \quad M \subseteq \mathcal{S}. \tag{14}$$

One of the three choices of $M$ in Table 1 is sampled for each training example.

Table 1: Mode-specific generation choices. Frame $f = 0$ always remains clean.

| **Mode** | $M$ | **RGB streams** ($f \geq 1$) | **Action views** ($f \geq 1$) |
|---|---|---|---|
| Forward dynamics | $\mathcal{S}_{\text{rgb}}$ | noised | clean |
| Inverse dynamics | $V$ | clean | noised |
| Joint generation | $\mathcal{S}$ | noised | noised |

Table 2: Algorithm for training-free action recovery.

**Input:** Predicted action views $\{\hat{x}^i_{t'}\}_{i\in V,\, t'=t+1:t+k}$; embodiment-specific URDF $U$; virtual cameras parameters $\{K_i, T_i\}_{i\in V}$; current state $S_t$; foreground threshold $\eta$, dilation radius r, and update stride $\lambda$.
**Output:** Recovered predicted trajectory $\hat{A}_{t+1:t+k}$.

$\hat{A}_t \leftarrow S_t$.
**for** $t' = t+1, \ldots, t+k$ **do**
  $\tilde{x}^i_{t'} \leftarrow \mathrm{dil}\big(\mathbb{1}\big[\hat{x}^i_{t'} > \eta\big]; r\big)$ for every $i \in V$.
  Sample candidate and render corresponding action-views using Equation 4.
  Calculate score $E_{t'}$ using Equation 16.
  $\hat{A}_{t'} \leftarrow \mathrm{Optimization}(E_{t'}, \hat{A}_{t'-1}, U)$. *warm start*
**end for**
**return** $\hat{A}_{t+1:t+k}$.

### 3.5 Training-Free Action Recovery

The IDM and JGM predict action views, which must be converted to joint configurations for execution. Instead of an embodiment-specific decoder, we match each prediction against URDF-based renderings of candidate configurations across all four virtual cameras, reducing single-view ambiguity. For camera $i$ and step $t'$, we binarize and dilate the prediction as

$$\tilde{x}^i_{t'} = \mathrm{dil}\big(\mathbb{1}\big[\hat{x}^i_{t'} > \eta\big]; r\big), \tag{15}$$

where $\eta$ is the foreground threshold and $\mathrm{dil}(\cdot)$ denotes the dilation operation around the foreground action view with the dilation radius $r$. For candidate $A \in \mathcal{Q}(U)$, where $\mathcal{Q}(U)$ encodes the URDF joint limits, is rendered as $x^i(A) = R(\mathrm{FK}(A; U); K_i, T_i)$ and scored by

$$E_{t'}(A) = \sum_{i\in V} \Big[1 - \mathrm{IoU}\big(x^i(A),\, \tilde{x}^i_{t'}\big)\Big] + \lambda \sum_{i\in V} d^{\rightarrow}\big(\tilde{x}^i_{t'},\, x^i(A)\big), \tag{16}$$

where $\lambda \geq 0$, IoU operation measures silhouette overlap and

$$d^{\rightarrow}(\Omega, \Omega') = \frac{1}{|\Omega|} \sum_{p\in\Omega} \min_{p'\in\Omega'} \|p - p'\|_2 \tag{17}$$

is the one-sided Chamfer distance between foreground pixels. The IoU term rewards overlap, while the Chamfer term distinguishes disjoint silhouettes. Table 2 summarizes the training-free action recovery procedure. More details about the mechanism are provided in Appendix A.3.

## 4 Experiments

We evaluate our model along three complementary axes: closed-loop manipulation, nulti embodiments capability and action-conditioned multiview prediction.

### 4.1 Simulation

**Benchmarks.** We evaluate closed-loop manipulation capability on RoboTwin 2.0 (Chen et al., 2025), a simulation benchmark comprising 50 bimanual manipulation tasks. We collect 2,500 clean and 25,000 randomized demonstrations, corresponding to 50 and 500 demonstrations per task, respectively. Randomization varies backgrounds, tabletop objects, and lighting.

Table 3: Success rates (%) on RoboTwin2.0 over 50 clean and 50 randomized tasks. Table 10 presents a comparison of success rates for each individual task.

| **Method** | Clean | Randomized | **Avg.** |
|---|---|---|---|
| $\pi_{0.5}$ (Intelligence et al., 2025) | 82.74 | 76.76 | 79.80 |
| Motus (Bi et al., 2026) | 88.66 | 87.02 | 87.80 |
| LingBot-VA (Li et al., 2026b) | 92.90 | 91.50 | **92.20** |
| Fast-WAM (Yuan et al., 2026) | 91.88 | 91.78 | 91.80 |
| **Ours** | 90.50 | 87.46 | 88.98 |

Table 4: **TriWorldBench results**. Dream4ACT is evaluated on the official 500-episode test set and reports six key indicators.

| Model | TVC | TA | P3D | MQ | TC | VQ | Overall |
|---|---|---|---|---|---|---|---|
| Ctrl-World (Guo et al., 2026) | 57.42 | 43.72 | 34.67 | 29.28 | 46.17 | 16.64 | 38.98 |
| Motus (Bi et al., 2026) | 66.70 | 49.69 | 34.60 | 24.63 | 26.56 | 16.26 | 42.35 |
| Genie Envisioner (Liao et al., 2026) | 62.39 | 33.17 | 54.00 | 20.17 | 32.18 | 17.46 | 40.73 |
| DreamDojo (Gao et al., 2026) | 69.63 | 56.24 | 43.84 | 27.96 | 60.84 | 21.02 | 51.72 |
| BWM (Team et al., 2026) | **81.87** | **86.05** | 60.40 | 41.29 | 62.81 | **31.42** | 65.54 |
| **Ours** | 81.63 | 84.22 | **61.30** | **41.66** | **64.88** | **31.42** | **65.66** |

To evaluate action-conditioned world modeling, we use TriWorldBench (Liu et al., 2026), which assesses generated videos from synchronized head and wrist cameras. We reuse the same checkpoint without benchmark-specific fine-tuning and switch to forward dynamics mode. Given the evaluation action trajectories, we render the corresponding action-view sequences for conditioning. Following the official submission protocol, we generate corresponding head and wrist videos per episode, with frame counts matching the supplied trajectory lengths, for official evaluation.

**Baselines.** For manipulation, we compare against $\pi_{0.5}$ (Intelligence et al., 2025), Motus (Bi et al., 2026), LingBot-VA (Li et al., 2026b), and Fast-WAM (Yuan et al., 2026), covering vision–language–action policies, unified video–action models, and world action models. We report publicly available success rates for these methods. For action-conditioned visual prediction on TriWorldBench, we include Ctrl-World (Guo et al., 2026), Motus (Bi et al., 2026), Genie Envisioner (Liao et al., 2026), DreamDojo (Gao et al., 2026), and BWM (Team et al., 2026) as strong baselines.

**Metrics.** For manipulation, we follow RoboTwin 2.0 official task-success criteria and evaluate each of the 50 tasks over 100 trials per configuration. For TriWorldBench, we report its six aggregate dimensions: tri-view consistency (TVC), task alignment (TA), physical and 3D coherence (P3D), motion quality (MQ), temporal consistency (TC), and visual quality (VQ), together with the official overall TWB-Score. Higher scores are better for all these reported metrics. TVC measures agreement across observation views, whereas TA assesses task alignment; these video scores complement the executable-task success rates.

**Results.** Table 3 shows that our model achieves success rates of 90.50% and 87.46% under clean and randomized settings, respectively, averaging 88.98%. Compared with publicly reported results, Dream4ACT exceeds $\pi_{0.5}$ by 7.76 and 10.7 percentage points and slightly higher than Motus, while LingBot-VA and Fast-WAM achieve higher scores. These results demonstrate effective closed-loop manipulation through generated action views and training-free recovery. The full 50-task evaluation uses a separately trained checkpoint. The multi-embodiment evaluation uses a checkpoint jointly trained across five embodiments, with its parameters fixed across all five robots.

In forward dynamics mode, Dream4ACT achieves a TWB-Score of 65.66, comparable to BWM's 65.54 (Table 4). Among the listed methods, our method obtains the highest reported P3D, MQ, and TC scores and ties BWM on VQ. Its TVC and TA scores are 81.63 and 84.22, respectively, compared with BWM's 81.87 and 86.05, showing competitive cross-view consistency and task alignment. Together with the manipulation results, they support the dual role of action views as conditioning inputs for multiview prediction and as generated representations recoverable into executable commands.

Table 5: Success rates (%) and end-effector recovery errors of the same jointly trained checkpoint across five embodiments on 31 shared RoboTwin 2.0 tasks. We use 25 trials per task and setting, Avg combines Clean and Randomized results. Recovery errors use GT action views with five held-out trajectories per task and 40 future frames per window, aggregated with equal task weighting.

| **Robot** | **Clean** | **Random** | **Position (mm)** $\downarrow$ | **Rotation** ($^\circ$) $\downarrow$ | **Avg** $\uparrow$ |
|---|---|---|---|---|---|
| Aloha-Agilex | 81.94 | 82.45 | 1.39 | 1.57 | 82.19 |
| Piper | 81.55 | 82.32 | 1.66 | 1.71 | 81.94 |
| ARX-X5 | 82.19 | 80.26 | 1.22 | 1.95 | 81.23 |
| Franka-Panda | 64.52 | 62.45 | 4.11 | 9.45 | 63.48 |
| UR5-Xsg | 31.87 | 30.06 | 11.85 | 21.39 | 30.97 |

**Multi-embodiment evaluation.** We evaluate the jointly trained checkpoint across five embodiments: Aloha-Agilex, ARX-X5, Franka-Panda, Piper, and UR5-Xsg. To keep task composition consistent, we use the intersection of tasks with available demonstrations for all five robots, yielding 31 tasks from the official 50-task suite. This subset defines the multi-embodiment training component and its evaluation task set, separate from the full 50-task evaluation above. During training, we use 50 clean and 500 randomized episodes for each embodiment–task pair. For test, we run 25 trials per task per robot under each clean and randomized setting, totaling 775 trials per setting, and average success rates equally across tasks. Model parameters remain unchanged across all five embodiments, with the corresponding URDF used for action-view rendering and recovery.

Table 5 reports the results. Aloha-Agilex, ARX-X5, and Piper achieve average success rates above 81%, while Franka-Panda and UR5-Xsg achieve 63.48% and 30.97%, respectively. These results demonstrate that a single jointly trained checkpoint supports executable control across distinct kinematic structures through a common visual action interface, without learned embodiment-specific action heads. More analysis is provided in Appendix A.5.

### 4.2 Real-World Experiments

**Settings.** We evaluate a single jointly trained checkpoint on two bimanual platforms, Aloha-Agilex and TienYi2.5 Pro, and two single-arm platforms, Franka Research 3 and UR5e. All four platforms are evaluated on `place_block` and `wipe_plate`; the bimanual platforms are additionally evaluated on `stack_blocks` and `storage_item`. We collect 60 demonstrations per embodiment–task pair with randomized object placements, totaling 720 demonstrations across 12 pairs. Demonstrations use the nominal background for each setup, without background variation or additional randomized distractor objects. Bimanual platforms use one head camera and two wrist cameras, whereas single-arm platforms use one wrist camera and one external camera. Model parameters remain unchanged across platforms, with each robot's URDF used for rendering and action recovery.

Table 6: Real-world success rates (%) using the same jointly trained checkpoint across four robot platforms. Each embodiment–task pair is evaluated over 20 trials under scene variations, including randomized background and randomized distractor objects. **Common** averages the first two tasks; **All** averages all tasks evaluated on each platform. "–" denotes a task not evaluated on that platform.

| **Robot** | Place_block | Wipe_plate | Stack_blocks | Storage_item | **Common** | **All** |
|---|---|---|---|---|---|---|
| Aloha-Agilex | 85.0 | 90.0 | 20.0 | 50.0 | 87.5 | 61.3 |
| TienYi2.5 Pro | 90.0 | 80.0 | 30.0 | 65.0 | 85.0 | 66.3 |
| Franka | 90.0 | 85.0 | – | – | 87.5 | – |
| UR5e | 85.0 | 95.0 | – | – | 90.0 | – |

**Results.** On the two common tasks, Aloha-Agilex, TienYi2.5 Pro, Franka Research 3, and UR5e achieve mean success rates of 87.5%, 85.0%, 87.5%, and 90.0%, respectively (Table 6). The extended bimanual evaluations additionally cover stacking and storage, yielding four-task averages of 61.3% and 66.3% for Aloha-Agilex and TienYi2.5 Pro. These results demonstrate that the same checkpoint supports real-world manipulation across single-arm and bimanual platforms with differ-

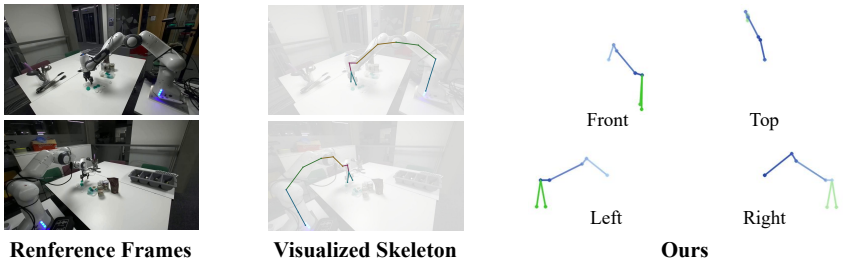



Figure 3: **Comparison of conditioning representations.** The same recorded joint configurations are represented as camera-aligned skeleton projections or action-view renderings from four prescribed virtual cameras.

ent observation configurations, using a common visual action interface and training-free recovery without learned embodiment-specific action heads. More details are provided in Appendix A.4.

### 4.3 Ablation Study

We compare camera-aligned skeleton conditioning with our fixed-view robot rendering for RGB prediction on DROID (Khazatsky et al., 2024) dataset. Both representations are constructed from the same recorded joint-state sequences. Following the camera-refinement procedure used in PointWorld (Huang et al., 2026), the baseline projects the Franka skeleton into the observation views using refined physical-camera calibration. Our representation instead renders the articulated robot from four prescribed virtual cameras, without requiring physical-camera extrinsics for its construction. Figure 3 provides a visual comparison.

We train two separate variants on the same 1,000 trajectories uniformly sampled from DROID sources excluding CLVR and RAD. Both variants are evaluated using the 10k checkpoint. These ablation models are trained from scratch. Evaluation uses 50 trajectories uniformly sampled from the held-out CLVR/RAD pool. Both variants predict 41-frame RGB sequences conditioned on recorded joint states. We compute PSNR, SSIM, and LPIPS against temporally aligned ground-truth physical-camera observations and average the scores over the test set.

Table 7: RGB prediction fidelity on 50 held-out DROID trajectories from CLVR and RAD. Both variants condition on recorded joint states and predict 41-frame sequences.

| **Method** | **DROID** | | |
|---|---|---|---|
| | PSNR ↑ | SSIM ↑ | LPIPS ↓ |
| Skeleton rendering | 23.19 | 0.899 | 0.105 |
| Action-view rendering (Ours) | **24.33** | **0.906** | **0.093** |

Table 7 shows that our action-view rendering improves PSNR from 23.19 to 24.33 dB and SSIM from 0.899 to 0.906, while reducing LPIPS from 0.105 to 0.093. The improvements across all three metrics support action-view rendering as an effective conditioning representation for RGB prediction, without requiring alignment of the conditioning views to the physical cameras. This complements the control experiments, which evaluate generated action views as representations recoverable into executable commands.

## 5 Conclusion

We presented Dream4ACT, a video–action world model for joint-controlled manipulation that represents target joint configurations as fixed-shape action views. Masked flow matching supports forward dynamics, inverse dynamics, and joint generation, while training-free URDF-constrained matching recovers joint targets for execution. Experiments on RoboTwin2.0 and real robots show that the same weights support closed-loop control across multiple jointly trained embodiments and our forward dynamics provides high quality task-aligned multi-view generation.

### AI USE STATEMENT

In this work, we used generative AI tools to polish the manuscript's language, review the logical consistency of the method description and interpretation of experimental results, and assist with optimization of the implementation code. We did not use these tools to originate the core research ideas or create manuscript illustrations. This disclosure concerns external AI assistance; outputs of the models studied in this work are presented as experimental results. The authors reviewed and tested the AI-assisted implementation code. We also checked all references and comparisons with prior work for accuracy and proper attribution, including checks for fabricated citations and plagiarism. We take responsibility for the final manuscript, implementation, and reported results, including content produced with generative AI assistance.

### REPRODUCIBILITY STATEMENT

To facilitate reproducibility, Section 3 specifies the model architecture and training objective, and Section 4 describes the datasets and evaluation protocols. We also provide more implementation details in Appendix A.1, while Appendices A.2 details action-view rendering, including virtual-camera presets and appearance conventions, and A.3 describes the training-free action recovery procedure. Appendix A.4 further documents the real-world tasks, hardware setups, and inference configuration. Appendix A.5 specifies the different success rates in the multi-embodiments experiment as shown in Table 11. We further discuss the limitations to provide a more comprehensive overview of our method in Appendix A.6. We plan to publicly release the implementation and model checkpoints to support independent reproduction and further research.

## REFERENCES

Hadi Alzayer, Wenlong Huang, Haonan Chen, Christopher Luey, Lvmin Zhang, Maneesh Agrawala, Gordon Wetzstein, Li Fei-Fei, Yilun Du, Jiajun Wu, et al. Masked visual actions for unified world modeling. *arXiv preprint arXiv:2607.19343*, 2026.

Hongzhe Bi, Hengkai Tan, Shenghao Xie, Zeyuan Wang, Shuhe Huang, Haitian Liu, Ruowen Zhao, Yao Feng, Chendong Xiang, Yinze Rong, et al. Motus: A unified latent action world model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 35101–35113, 2026.

Jake Bruce, Michael D Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, et al. Genie: Generative interactive environments. In *Forty-first international conference on machine learning*, 2024.

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

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, et al. Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025.

Yixiang Chen, Peiyan Li, Jiabing Yang, Keji He, Xiangnan Wu, Yuan Xu, Kai Wang, Jing Liu, Nianfeng Liu, Yan Huang, et al. Bridgev2w: Bridging video generation models to embodied world models via embodiment masks. *arXiv preprint arXiv:2602.03793*, 2026.

Yilun Du, Sherry Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Josh Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. *Advances in neural information processing systems*, 36:9156–9172, 2023.

Shenyuan Gao, William Liang, Kaiyuan Zheng, Ayaan Malik, Seonghyeon Ye, Sihyun Yu, Wei-Cheng Tseng, Yuzhu Dong, Kaichun Mo, Chen-Hsuan Lin, et al. Dreamdojo: A generalist robot world model from large-scale human videos. *arXiv preprint arXiv:2602.06949*, 2026.

Chenghao Gu, Hanyang Yu, Jingbo Zhang, Haitao Lin, Wenyao Zhang, Jinghe Wang, Hanglei Jin, Shuzhao Xie, Jingyan Jiang, and Zhi Wang. Geniworld: A generalizable interactive world model for robotic manipulation via visual actions. *arXiv preprint arXiv:2608.06332*, 2026.

Yanjiang Guo, Lucy Shi, Jianyu Chen, and Chelsea Finn. Ctrl-world: A controllable generative world model for robot manipulation. In *International Conference on Learning Representations*, volume 2026, pp. 6121–6138, 2026.

Yucheng Hu, Yanjiang Guo, Pengchao Wang, Xiaoyu Chen, Yen-Jen Wang, Jianke Zhang, Koushil Sreenath, Chaochao Lu, and Jianyu Chen. Video prediction policy: A generalist robot policy with predictive visual representations. *arXiv preprint arXiv:2412.14803*, 2024.

Wenlong Huang, Yu-Wei Chao, Arsalan Mousavian, Ming-Yu Liu, Dieter Fox, Kaichun Mo, and Li Fei-Fei. Pointworld: Scaling 3d world models for in-the-wild robotic manipulation. *arXiv preprint arXiv:2601.03782*, 2026.

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.

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

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

Po-Chen Ko, Jiayuan Mao, Yilun Du, Shao-Hua Sun, and Joshua B Tenenbaum. Learning to act from actionless videos through dense correspondences. In *International Conference on Learning Representations*, volume 2024, pp. 40938–40958, 2024.

Hongyu Li, Bowen Wen, Xinghao Zhu, Yixuan Wang, Yilun Du, Yunzhu Li, George Konidaris, Stan Birchfield, Soha Pouya, Chenran Li, et al. Hydra-0: Action flow for generalist world modeling and control. *arXiv preprint arXiv:2608.18077*, 2026a.

Lin Li, Qihang Zhang, Yiming Luo, Shuai Yang, Ruilin Wang, Fei Han, Mingrui Yu, Zelin Gao, Nan Xue, Xing Zhu, et al. Causal world modeling for robot control. *arXiv preprint arXiv:2601.21998*, 2026b.

Peiyan Li, Yixiang Chen, Yuan Xu, Jiabing Yang, Xiangnan Wu, Jun Guo, Nan Sun, Long Qian, Xinghang Li, Xin Xiao, Jing Liu, Nianfeng Liu, Tao Kong, Yan Huang, Liang Wang, and Tieniu Tan. Spatialvam:spatial-aware multi-view video diffusion as a data-efficient robot policy, 2026c. URL https://arxiv.org/abs/2604.03181.

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

Shuang Li, Yihuai Gao, Dorsa Sadigh, and Shuran Song. Unified video action model, 2025. URL https://arxiv.org/abs/2503.00200.

Yue Liao, Pengfei Zhou, Siyuan Huang, Donglin Yang, Shengcong Chen, Yuxin Jiang, Yue Hu, Si Liu, Jianlan Luo, Liliang Chen, et al. Genie envisioner: A unified world foundation platform for robotic manipulation. In *International Conference on Learning Representations*, volume 2026, pp. 88446–88463, 2026.

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

Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation. In *International Conference on Learning Representations*, volume 2025, pp. 29982–30009, 2025.

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

Xuanyi Liu, Haofeng Wang, Ruiqi Li, Danni Yu, Rui Wan, Ruixu Zhang, Siyu Tao, Xue Yang, Shaofeng Zhang, Zicheng Zhang, Jiaqi Zhang, and Siwei Ma. Triworldbench: A tri-view consistency perspective on embodied world models. *arXiv preprint arXiv:2609.26314*, 2026.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4172–4182. IEEE, 2023.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

BWM Team et al. Bwm: A low-cost high-fidelity world simulator for robot learning. *arXiv preprint arXiv:2607.29302*, 2026.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

Qwen Team. Qwen3. 5-omni technical report. *arXiv preprint arXiv:2604.15804*, 2026.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Yufan Wei, Kun Zhou, Lingjun Mao, Zijun Zhang, Ziming Xu, Ziqiao Xi, Shuang Liang, Ruobing Han, Yuchen Yan, Xinyue Wang, et al. Causally debiased latent action model for embodied action conditioned world models. *arXiv preprint arXiv:2607.09185*, 2026.

Lishan Yang, Wenxuan Song, Xi Wang, Pingyue Sheng, Zheng Fang, Ziyang Zhou, Junjie He, Haodong Yan, Jiayi Chen, Nan Sun, et al. 4d-wam: Infusing spatiotemporal awareness into world action models through trajectory fields. *arXiv preprint arXiv:2608.08023*, 2026.

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

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

Haoyu Zhen, Zixian Gao, Qiao Sun, Yilin Zhao, Yuncong Yang, Yilun Du, Pengsheng Guo, Tsun-Hsuan Wang, Yi-Ling Qiao, and Chuang Gan. Action images: End-to-end policy learning via multiview video generation. *arXiv preprint arXiv:2604.06168*, 2026.

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

# A APPENDIX

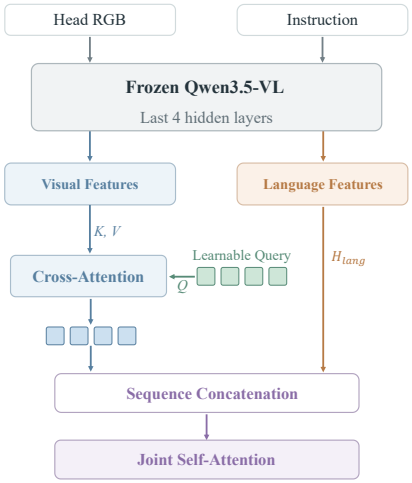



Figure 4: Trainable adapter modulates main view and instruction information.

## A.1 IMPLEMENTATION DETAILS.

**Video encoding and decoding.** We use the Wan2.2VAE (Wan et al., 2025) encoder $\mathcal{E}$ and decoder $\mathcal{D}$, shared across RGB and action-view sequences. Each sequence is encoded separately, with a spatial downsampling factor of 16 and a temporal downsampling factor of 4. A $T$-frame sequence has the latent shape

$$y^s \in \mathbb{R}^{3\times T\times H\times W} \quad \xrightarrow{\mathcal{E}} \quad z_s \in \mathbb{R}^{48\times L\times (H/16)\times (W/16)}, \qquad L = 1 + \frac{T-1}{4}, \tag{18}$$

for $T = 4n + 1$ and spatial dimensions divisible by 16. The initial frame is encoded separately. At the sequence resolution of $320 \times 224$ pixels (width $\times$ height), each latent grid has spatial size $14 \times 20$. For the single-frame-conditioned window in Section 3.3, $T = k+1$: the simulation horizon $k = 40$ and real-world horizon $k = 80$. Generated latents are decoded sequence-wise as $\hat{y}^s = \mathcal{D}(\hat{z}_s)$ before RGB evaluation or action recovery. The $640 \times 480$ main-view input to the semantic branch is separate from these lower-resolution video sequences.

**Training-mode sampling.** Table 8 summarizes the optimization settings, resolutions, horizons, and sampling probabilities. For each training example, we first select forward dynamics, inverse dynamics, or joint generation with probabilities 0.2, 0.2, and 0.6. These are operating modes of one shared model, not separately trained networks. Following Table 1, the target stream set is $M = \mathcal{S}_{\text{rgb}}$, $M = V$, or $M = \mathcal{S}$, respectively; streams outside $M$ remain clean conditions. We then sample image-to-video (I2V) or video-to-video (V2V) conditioning with probabilities 0.7 and 0.3.

**Temporal-prefix conditioning.** Here, I2V and V2V specify the clean temporal prefix of the streams selected for generation. I2V preserves only the initial frame, whereas V2V preserves a longer prefix and predicts the remaining segment. The V2V prefix covers one fifth of a simulation training clip and one quarter of a real-world training clip. Writing $L_c$ for the number of clean prefix latents, the training mask generalizes Equation 14 to

$$m_{s,f} = \mathbb{1}[s \in M \wedge f \geq L_c], \qquad 0 \leq f < L, \tag{19}$$

with $L_c = 1$ for I2V and $L_c > 1$ for V2V. The loss remains restricted to corrupted positions as in Equation 13. This temporal-prefix choice does not remove mode-specific future conditioning: forward dynamics still conditions on action views, and inverse dynamics on RGB sequences. V2V is used only during training; all reported inference uses I2V, recovering the single-frame mask in the main text.

Table 8: Training hyperparameters.

| Hyperparameter | Value |
|---|---|
| Training Resource | $16 \times$ Nvidia B200 |
| Batch Size | 16 |
| Optimizer | AdamW |
| Weight Decay | $1 \times 10^{-2}$ |
| Learning Rate | $1 \times 10^{-5}$ |
| Learning Rate Scheduler | Cosine Scheduler |
| Learning Rate Warmup Steps | 1000 |
| Simulation Action Chunk | 40 |
| Real-World Action Chunk | 80 |
| Main view Resolution | $640 \times 480$ |
| RGB Sequence Resolution | $320 \times 224$ |
| Action-View Sequence Resolution | $320 \times 224$ |
| Forward Dynamics Sample Probability | 0.2 |
| Inverse Dynamics Sample Probability | 0.2 |
| joint Generation Sample Probability | 0.6 |
| Image-to-Viedo Sample Probability | 0.7 |
| Video-to-Video Sample Probability | 0.3 |

### A.2 RENDERING.

**URDF-based rendering.** Forward kinematics maps $S_t$ and $A_{t+1:t+k}$ to current and future action views. We render the URDF-connected `base_link`, arm-joint nodes, and two gripper nodes at $(W, H) = (320, 224)$ pixels. Base-to-arm links are yellow, arm nodes blue, and gripper colors vary from green (open) to red (closed). Camera-space depth modulates color intensity while preserving these roles.

**Task-level camera rig.** For each dataset–embodiment–task training group, we will save one `task_rig.json`, shared by its episodes. Four virtual cameras have the ordered roles *front*, *top*, *left*, and *right*; no physical RGB calibration is used. Extrinsics are fixed first: each camera looks at the mean arm-root position over the group's initial frames, at distance `dist`.

**Intrinsic selection.** For view $i$, let $\mathcal{P}_i$ contain camera-space joint points from all valid frames of all training episodes in the group, discarding points with $Z \le 10^{-4}$. With occupancy $o$ (default $0.95$), the union determines

$$h_{w,i} = \max_{(X,Y,Z)\in\mathcal{P}_i} |X/Z|, \qquad h_{h,i} = \max_{(X,Y,Z)\in\mathcal{P}_i} |Y/Z|,$$
$$f_i = \min\left(\frac{oW}{2h_{w,i}}, \frac{oH}{2h_{h,i}}\right).$$

Each view has its own $f_i$, with square pixels ($f_{x,i} = f_{y,i} = f_i$), zero skew, and fixed principal point $(W/2, H/2)$. Thus $(u, v) = (f_i X/Z + W/2, f_i Y/Z + H/2)$. Maxima are taken over points, frames, and episodes, without averaging. The occupancy bound applies along both image dimensions about the optical axis; for asymmetric trajectories, the farther side determines the focal length, leaving more margin on the opposite side.

If $\mathcal{P}_i$ is empty or either extent is near zero, we use $f_i = (H/2)/\tan(\phi/2)$ with vertical field of view $\phi = 45°$. Changing `dist` changes camera positions; focal lengths are recomputed to maintain the occupancy bound.

**Consistent rendering at inference.** Saved intrinsics and extrinsics remain fixed across frames and episodes and are reused for conditioning views and action recovery. Test trajectories do not determine the preset. View roles, ordering, resolution, and appearance rules are shared across embodiments; numerical camera parameters are group-specific.

### A.3 Action Recovery Mechanism.

**Overview.** Recovery combines frame-wise estimation, window-level refinement, and gripper decoding (Table 2). At local frame $j = 0, \ldots, k$, corresponding to time $t + j$, $u_j = (q_j, g_j)$ contains arm configurations and provisional gripper variables. We anchor $u_0 = S_t$ without using future ground-truth states. The URDF, virtual-camera preset, and rendering conventions match the training action views.

**Stage A: frame-wise estimation.** The experimental GPU path refines 32 initializations per frame with batched Levenberg–Marquardt (LM). Neighboring-frame solutions supply additional initializations in Jacobi-style rounds. Anchor frames are solved at stride $s_{\text{anchor}} = 3$; intermediate configurations are interpolated and batch-refined, and high-cost segments are revisited. An independent forward chain propagates estimates from the observed frame without accepting candidate-pool seeds. Disagreement among restart solutions with similar costs provides the *spread* signal for Stage B.

**Gauss–Newton refinement.** The `ridge` option extracts medial-axis information from predicted RGB action views. Multiview image-field residuals define the local least-squares problem, separately from the rendered score in Equation 16. With residual vector $r_j$, Jacobian $J_j = \partial r_j / \partial u_j$, and residual weights $W_j$, the damped normal equations are

$$\big(J_j^\top W_j J_j + \mu M\big)\Delta u_j = -J_j^\top W_j r_j, \tag{20}$$

where $M$ is a positive-definite damping metric. Setting $\mu = 0$ gives Gauss–Newton; positive damping stabilizes the LM step. Coarse-to-fine field scales guide image alignment.

**Stage B: window-level estimation.** Given frame-wise estimates $\hat{u}_j$, we fit a cubic B-spline trajectory $z_j = (BC)_{j,:}^\top$ with knots every four frames:

$$\min_C \sum_{j=0}^{k} (z_j - \hat{u}_j)^\top \widetilde{H}_j (z_j - \hat{u}_j) + \|D^{(2)} BC\Lambda^{1/2}\|_F^2, \tag{21}$$

Here $B$ is the temporal basis, $C$ contains spline coefficients, and $D^{(2)}$ is the second-difference operator. Diagonal data weights $\widetilde{H}_j$ use the diagonal of $J_j^\top W_j J_j$, attenuated for large Stage A spread. Diagonal $\Lambda$ controls coordinate-wise smoothness, with stronger regularization for each arm's sixth joint in the supplied bimanual configuration (`lam_joint6`=5). The fit balances image evidence against temporal regularization, which is part of estimation.

With `n_outer`=2, the first window fit uses Stage A estimates. A single-start refinement at finer image-field scales updates the estimates and curvature weights for the second fit, retaining Stage A spread. After each fit, configurations are clipped to joint limits and frame 0 is restored to $S_t$; frame 0 is also restored after refinement.

**Gripper decoding.** After Stage B, the renderer's red–green fingertip encoding replaces provisional gripper estimates with color-decoded values. A three-frame median filter is applied, and only missing runs of at most three frames are interpolated.

**Rendered-score evaluation.** After continuous color decoding, Equation 16 is evaluated once per future frame for reporting. IoU measures silhouette overlap; the one-sided Chamfer term measures predicted-to-rendered foreground distance, including disjoint silhouettes. These assess image-space agreement, not joint-space accuracy, and do not drive LM updates.

### A.4 Real-World Experiment Details.

**Task design.** The four tasks span target selection, tool-mediated contact, bimanual coordination, and sequential object interaction. As in Section 4.2, all four platforms are evaluated on the first two tasks, and only the bimanual platforms are evaluated on the remaining two:

1. **Place_block**. A box and several small blocks are placed on the table, with randomized block positions. The robot must select and grasp the specific blue block, place it inside the box, and then return to its initial pose.

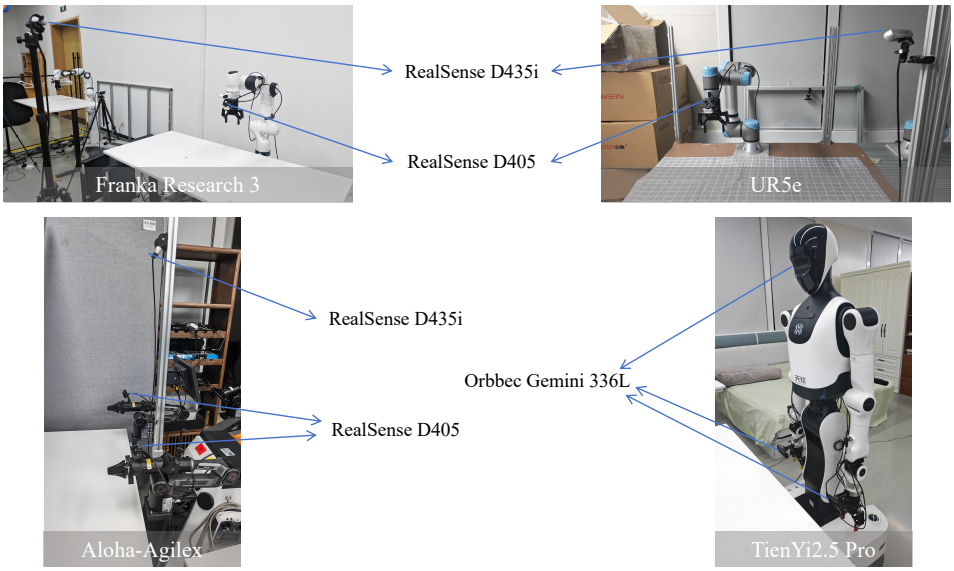



Figure 5: Real-world robots hardware configuration. Franka and UR embodiment use one front camera and one wrist camera, while Aloha-Agilex and TienYi use one head camera and two wrist cameras.

2. **Wipe_plate**. A plate containing dirty water is placed on the table, with a sponge initially positioned to its right. The robot must grasp the sponge, wipe the plate clean, and place the sponge on the opposite side of the plate.
3. **Stack_blocks**. A yellow block is placed on the left side of the table, and the red and blue blocks are placed on the right, with positions randomized within these regions. The robot must coordinate both arms to stack three blocks, ordered red, yellow, and blue from bottom to top.
4. **Storage_item**. A drawer and a block are placed on the table. The left arm first opens the drawer; the right arm then grasps the block and places it inside; finally, the left arm closes the drawer.

**Settings.** Figure 5 illustrates the physical setups. Franka Research 3 and UR5e each use one external RealSense D435i camera, one wrist-mounted RealSense D405 camera, and a Robotiq gripper. Aloha-Agilex uses a head-mounted RealSense D435i camera, two wrist-mounted RealSense D405 cameras, and two Robotiq grippers. TienYi2.5 Pro uses Orbbec Gemini 336L cameras for all three viewpoints (one head and two wrists), together with two Robotiq grippers. The two or three RGB sequences are encoded separately and processed by sequence-specific RoPE offsets (Section 3.3). This input formulation supports joint training across embodiments with different numbers of observation views using a single shared model.

**Inference configuration.** All four platforms use the same jointly trained checkpoint, with their corresponding URDFs and saved virtual-camera presets used for rendering and recovery. During inference, our model predicts an 80-step action chunk ($k = 80$), encoding target joint configurations and gripper states as action views for recovery into executable commands (Section 3.5). In our experimental setup with an NVIDIA A800 GPU, inference and action recovery take approximately 6.9 s and 2.7 s per chunk, respectively. We execute each predicted action chunk in full. After execution, we obtain new observations and use them to plan the next action chunk. Figure 6 shows each embodiment-task execution progress.

### A.5 Multi-Embodiment Experiment Analysis.

**Evaluation protocol.** We evaluate action recovery using five held-out trajectories per task on the 31 shared RoboTwin 2.0 tasks. The solver recovers joint configurations from ground-truth (GT) action views, and forward kinematics gives the corresponding end-effector poses. Let $(\hat{\mathbf{p}}, \hat{\mathbf{R}})$ and

Table 9: Action recovery errors and task success across five embodiments on 31 shared RoboTwin 2.0 tasks. Avg. success averages Clean and Randomized evaluations, with 25 trials per task, per embodiment, and per setting.

| Embodiment | Position (mm) ↓ | Rotation (°) ↓ | Avg. success (%) ↑ |
|---|---|---|---|
| Aloha-Agilex | 1.39 | 1.57 | 82.19 |
| Piper | 1.66 | 1.71 | 81.94 |
| ARX-X5 | 1.22 | 1.95 | 81.23 |
| Franka-Panda | 4.11 | 9.45 | 63.48 |
| UR5-Xsg | 11.85 | 21.39 | 30.97 |

$(\mathbf{p}, \mathbf{R})$ denote the recovered and GT poses, with positions expressed in millimeters. We compute

$$e_{\text{pos}} = \|\hat{\mathbf{p}} - \mathbf{p}\|_2, \tag{22}$$

$$e_{\text{rot}} = \frac{180}{\pi} \arccos\left(\text{clip}\left(\frac{\text{tr}(\mathbf{R}^\top \hat{\mathbf{R}}) - 1}{2}, -1, 1\right)\right). \tag{23}$$

Position and rotation errors are reported in millimeters and degrees, respectively. Rotation error measures the relative end-effector rotation angle. Each recovery window contains one known initial frame and 40 future frames. We take the larger error across arms at each frame, average over each complete trajectory, and then equally average over trajectories and tasks. Overlapping frames are counted only once, and the initial frame is excluded from scoring.

**Results and analysis.** Table 9 reports GT-view recovery errors alongside closed-loop task success. The lower success rates of Franka-Panda and UR5-Xsg should also be interpreted in the context of their simulated bimanual configurations, assembled from two single-arm robots rather than designed as integrated dual-arm systems. Limitations in this integration may affect inter-arm coordination and joint-target tracking, introducing execution effects not captured by the URDF-based kinematic model used for recovery. Consequently, even accurately recovered joint targets may not be realized precisely by the simulated system, affecting grasping and placement. The lower demonstration-collection success rates observed for these configurations are consistent with such execution challenges. Together with the measured recovery errors, these configuration-level factors may contribute to their lower closed-loop success rates.

### A.6 Limitations and Future Work.

The demonstrated scope of Dream4ACT is bounded by its training data and embodiment coverage. Our evaluation includes five simulated embodiments and four real-world platforms; broader task coverage and transfer to unseen embodiments remain to be validated. Recovery requires an embodiment-specific URDF and rendering conventions consistent with those used during training (Appendices A.2 and A.3). Kinematic mismatch can affect physical execution, while joint-recovery accuracy also depends on generated-view fidelity and geometric observability. Explicit multiview generation and numerical action recovery incur substantial latency: in our real-world setup, they take approximately 6.9 s and 2.7 s, respectively, per 80-step chunk (Appendix A.4). The current implementation therefore supports chunk-based execution rather than real-time, per-step replanning. Expanding demonstration and embodiment coverage, improving robustness to geometric mismatch, and accelerating generation and recovery are promising directions for extending this shared action interface while retaining recovery without learned embodiment-specific decoders.

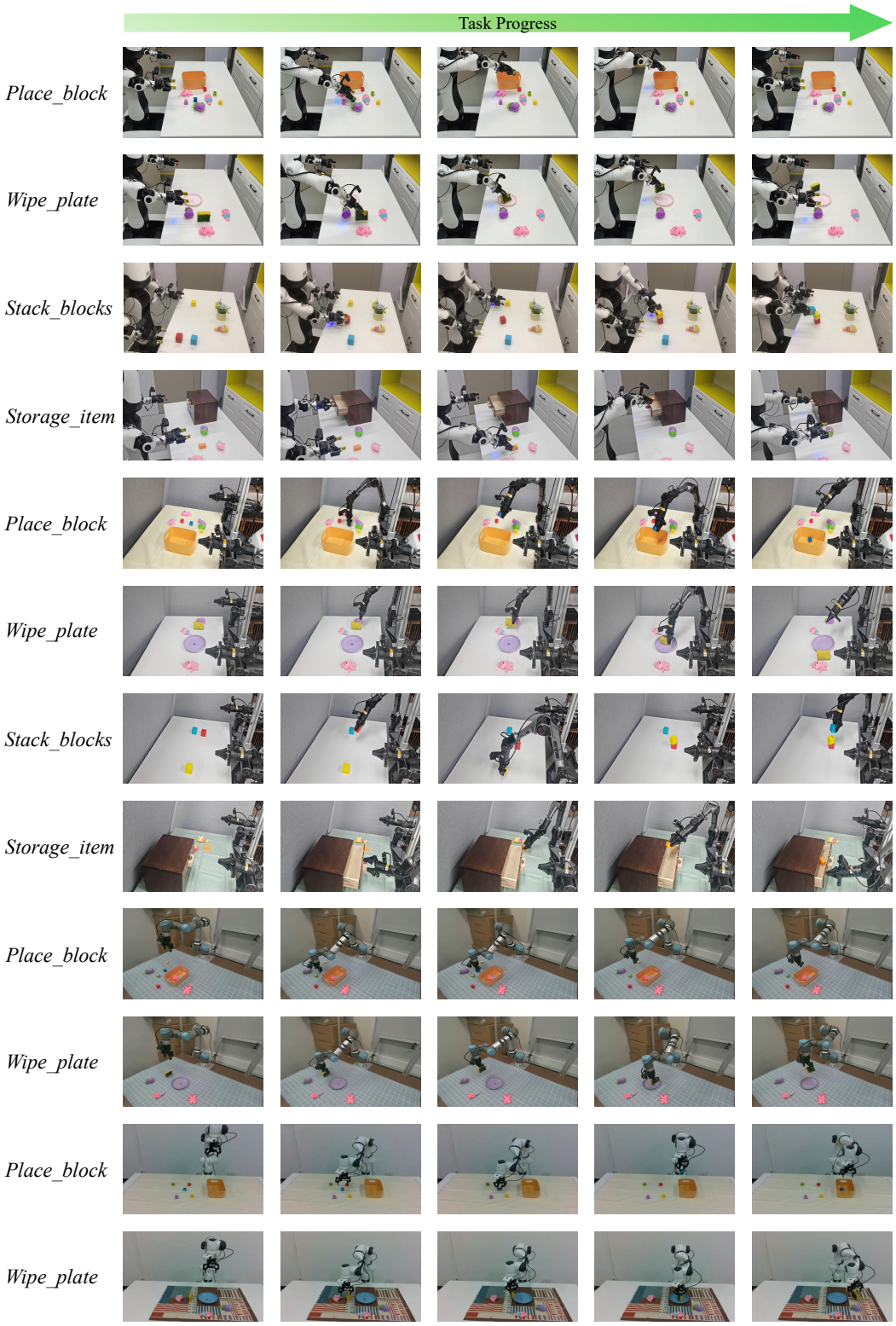



Figure 6: Execution progress of real-world manipulation tasks. Each row shows representative temporal snapshots for one task, illustrating how the robot complete place_block, wipe_plate, stack_blocks, and storage_item.

Table 10: Per-task success rates (%) on the 50-task RoboTwin 2.0 benchmark under clean and randomized (Rand.) settings. Baseline scores are taken from publicly reported results. Bold marks the highest score for each task and setting.

| **Simulation Task** | $\pi_{0.5}$ | | Motus | | LingBot-VA | | Fast-WAM | | **Ours** | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Clean | Rand. | Clean | Rand. | Clean | Rand. | Clean | Rand. | Clean | Rand. |
| *Adjust Bottle* | **100%** | 99% | 89% | 93% | 90% | 94% | **100%** | **100%** | 91% | 97% |
| *Beat Block Hammer* | 96% | 93% | 95% | 88% | 96% | **98%** | 99% | 97% | **100%** | 96% |
| *Blocks Ranking RGB* | 92% | 85% | 99% | 97% | 99% | 98% | **100%** | **100%** | 96% | 90% |
| *Blocks Ranking Size* | 49% | 26% | 75% | 63% | **94%** | 96% | **94%** | **98%** | 62% | 58% |
| *Click Alarmclock* | 98% | 89% | **100%** | **100%** | 99% | **100%** | **100%** | **100%** | **100%** | **100%** |
| *Click Bell* | 99% | 66% | **100%** | **100%** | **100%** | **100%** | **100%** | **100%** | **100%** | **100%** |
| *Dump Bin Bigbin* | 92% | **97%** | 95% | 91% | 89% | 96% | **97%** | 96% | 90% | 93% |
| *Grab Roller* | **100%** | **100%** | **100%** | **100%** | **100%** | **100%** | **100%** | **100%** | **100%** | **100%** |
| *Handover Block* | 66% | 57% | 86% | 73% | **99%** | 78% | 95% | 81% | 97% | **86%** |
| *Handover Mic* | 98% | 97% | 78% | 63% | 94% | 96% | **99%** | **100%** | 81% | 92% |
| *Hanging Mug* | 18% | 17% | 38% | 38% | 40% | 28% | **58%** | **62%** | 47% | 36% |
| *Lift Pot* | 96% | 85% | 96% | 99% | **100%** | 99% | **100%** | **100%** | 99% | 97% |
| *Move Can Pot* | 51% | 55% | 34% | 74% | **94%** | **97%** | 90% | 88% | 88% | 91% |
| *Move Pillbottle Pad* | 84% | 61% | 93% | 96% | 99% | **99%** | **100%** | **99%** | 99% | 97% |
| *Move Playingcard Away* | 96% | 84% | **100%** | 96% | **100%** | 99% | **100%** | **100%** | 99% | **100%** |
| *Move Stapler Pad* | 56% | 42% | 83% | **85%** | **91%** | 79% | 77% | 64% | 88% | 76% |
| *Open Laptop* | 90% | 96% | 95% | 91% | 92% | 94% | **98%** | **100%** | 82% | 78% |
| *Open Microwave* | 34% | 77% | **95%** | **91%** | 82% | 86% | 62% | 45% | 47% | 48% |
| *Pick Diverse Bottles* | 81% | 71% | **90%** | **91%** | 89% | 82% | 80% | 85% | 89% | 87% |
| *Pick Dual Bottles* | 93% | 63% | 96% | 90% | **100%** | **99%** | **100%** | 96% | 99% | 98% |
| *Place A2B Left* | 87% | 82% | 88% | 79% | **97%** | **93%** | 95% | **93%** | 96% | 88% |
| *Place A2B Right* | 87% | 84% | 91% | 87% | **97%** | 95% | 93% | **99%** | 95% | 88% |
| *Place Bread Basket* | 77% | 64% | 91% | 94% | **97%** | **95%** | 91% | 93% | 89% | 94% |
| *Place Bread Skillet* | 85% | 66% | 86% | 83% | **95%** | 90% | 90% | **93%** | 86% | 86% |
| *Place Burger Fries* | 94% | 87% | 98% | 98% | 97% | 95% | 96% | **99%** | **99%** | 97% |
| *Place Can Basket* | 62% | 62% | 81% | 76% | 81% | **84%** | 71% | 69% | **90%** | 74% |
| *Place Cans Plasticbox* | 94% | 84% | 98% | 94% | **100%** | **99%** | 99% | 96% | 97% | 98% |
| *Place Container Plate* | 99% | 95% | 98% | 99% | 99% | 97% | 96% | **100%** | **100%** | 98% |
| *Place Dual Shoes* | 75% | 75% | 93% | 87% | **94%** | **89%** | **94%** | 88% | 43% | 52% |
| *Place Empty Cup* | **100%** | 99% | 99% | 98% | **100%** | **100%** | **100%** | **100%** | **100%** | 99% |
| *Place Fan* | 87% | 85% | 91% | 87% | **99%** | 93% | 96% | **96%** | 95% | 92% |
| *Place Mouse Pad* | 60% | 39% | 66% | 68% | **93%** | **96%** | 83% | 89% | 89% | 90% |
| *Place Object Basket* | 80% | 76% | 81% | 87% | **91%** | **88%** | 89% | **88%** | 87% | 84% |
| *Place Object Scale* | 86% | 80% | 88% | 85% | **96%** | 95% | 90% | **97%** | 92% | 81% |
| *Place Object Stand* | 91% | 85% | 98% | **97%** | **99%** | 96% | 90% | 94% | **99%** | **97%** |
| *Place Phone Stand* | 81% | 81% | 87% | 86% | **97%** | 97% | **97%** | **99%** | 96% | 94% |
| *Place Shoe* | 92% | 93% | 99% | 97% | 98% | 98% | 96% | **99%** | **100%** | 94% |
| *Press Stapler* | 87% | 83% | 93% | **98%** | 85% | 82% | 90% | 97% | **96%** | 97% |
| *Put Bottles Dustbin* | 84% | 79% | 81% | 79% | 87% | **91%** | **95%** | 90% | 93% | 86% |
| *Put Object Cabinet* | 80% | 79% | 88% | 71% | 85% | 87% | **94%** | **89%** | 89% | 78% |
| *Rotate QRcode* | 89% | 87% | 89% | 73% | **96%** | **91%** | 93% | 89% | 93% | 87% |
| *Scan Object* | 72% | 65% | 67% | 66% | **96%** | 91% | 89% | **92%** | 84% | 68% |
| *Shake Bottle Horizontally* | 99% | 99% | **100%** | 98% | **100%** | 99% | **100%** | **100%** | 98% | 97% |
| *Shake Bottle* | 99% | 97% | **100%** | 97% | **100%** | 97% | **100%** | **100%** | **100%** | 97% |
| *Stack Blocks Three* | 91% | 76% | 91% | 95% | **99%** | **98%** | 95% | 97% | 95% | 83% |
| *Stack Blocks Two* | 97% | **100%** | **100%** | 98% | **100%** | 98% | **100%** | **100%** | **100%** | 98% |
| *Stack Bowls Three* | 77% | 71% | 79% | **87%** | **86%** | 83% | 80% | 81% | 84% | 81% |
| *Stack Bowls Two* | 95% | 96% | **98%** | **98%** | 94% | **98%** | 92% | **98%** | **98%** | 96% |
| *Stamp Seal* | 79% | 55% | 93% | 92% | 96% | 97% | 90% | 94% | **97%** | **99%** |
| *Turn Switch* | 62% | 54% | 84% | 78% | 44% | 45% | 61% | 59% | **91%** | **80%** |
| **Average (%)** | 82.74 | 76.76 | 88.66 | 87.02 | **92.90** | 91.50 | 91.88 | **91.78** | 90.50 | 87.46 |

Table 11: Per-task success rates (%) across five robot embodiments on 31 shared RoboTwin 2.0 tasks under clean and randomized settings. Each task is evaluated over 25 episodes per embodiment and setting.

| **Simulation Task** | **Aloha** | | **Franka** | | **Piper** | | **ARX-X5** | | **UR5** | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Clean | Random | Clean | Random | Clean | Random | Clean | Random | Clean | Random |
| *Beat Block Hammer* | 100% | 100% | 60% | 64% | 60% | 76% | 56% | 68% | 56% | 48% |
| *Blocks Ranking RGB* | 0% | 0% | 0% | 0% | 4% | 0% | 0% | 0% | 0% | 0% |
| *Blocks Ranking Size* | 64% | 64% | 68% | 56% | 80% | 84% | 52% | 60% | 12% | 8% |
| *Grab Roller* | 100% | 96% | 96% | 96% | 100% | 100% | 100% | 100% | 32% | 48% |
| *Handover Mic* | 80% | 80% | 40% | 28% | 96% | 100% | 100% | 96% | 48% | 40% |
| *Lift Pot* | 100% | 96% | 72% | 68% | 96% | 100% | 96% | 96% | 0% | 0% |
| *Move Can Pot* | 72% | 72% | 48% | 68% | 68% | 60% | 84% | 84% | 24% | 8% |
| *Move Pillbottle Pad* | 84% | 88% | 68% | 64% | 100% | 92% | 96% | 100% | 4% | 0% |
| *Move Playingcard Away* | 96% | 100% | 96% | 100% | 100% | 100% | 100% | 100% | 88% | 88% |
| *Move Stapler Pad* | 92% | 96% | 52% | 32% | 84% | 80% | 88% | 92% | 16% | 24% |
| *Open Laptop* | 96% | 96% | 56% | 64% | 60% | 80% | 96% | 96% | 48% | 64% |
| *Open Microwave* | 28% | 36% | 16% | 36% | 16% | 24% | 12% | 20% | 20% | 28% |
| *Place A2B Left* | 92% | 84% | 92% | 72% | 96% | 84% | 92% | 76% | 20% | 12% |
| *Place A2B Right* | 76% | 84% | 80% | 92% | 84% | 76% | 96% | 92% | 20% | 20% |
| *Place Bread Basket* | 84% | 68% | 76% | 44% | 64% | 64% | 80% | 76% | 56% | 40% |
| *Place Burger Fries* | 96% | 100% | 84% | 80% | 92% | 100% | 96% | 96% | 20% | 24% |
| *Place Container Plate* | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 68% | 48% |
| *Place Dual Shoes* | 64% | 64% | 16% | 40% | 76% | 96% | 48% | 36% | 4% | 0% |
| *Place Empty Cup* | 100% | 100% | 100% | 96% | 96% | 68% | 92% | 52% | 68% | 36% |
| *Place Mouse Pad* | 80% | 88% | 84% | 68% | 92% | 100% | 100% | 96% | 4% | 8% |
| *Place Object Scale* | 84% | 88% | 84% | 72% | 80% | 100% | 80% | 96% | 44% | 32% |
| *Place Object Stand* | 100% | 100% | 92% | 84% | 96% | 96% | 96% | 88% | 16% | 20% |
| *Place Phone Stand* | 96% | 96% | 8% | 16% | 92% | 88% | 100% | 92% | 12% | 0% |
| *Place Shoe* | 96% | 100% | 4% | 4% | 88% | 92% | 92% | 96% | 4% | 8% |
| *Press Stapler* | 92% | 100% | 100% | 96% | 100% | 92% | 100% | 92% | 96% | 84% |
| *Shake Bottle Horizontally* | 100% | 100% | 68% | 52% | 100% | 100% | 100% | 100% | 36% | 44% |
| *Shake Bottle* | 100% | 96% | 80% | 68% | 100% | 92% | 100% | 92% | 48% | 68% |
| *Stack Blocks Two* | 16% | 4% | 16% | 20% | 12% | 16% | 4% | 8% | 8% | 16% |
| *Stack Bowls Two* | 92% | 92% | 88% | 92% | 100% | 100% | 96% | 96% | 60% | 52% |
| *Stamp Seal* | 76% | 88% | 56% | 64% | 100% | 92% | 100% | 100% | 8% | 28% |
| *Turn Switch* | 84% | 80% | 100% | 100% | 96% | 100% | 96% | 92% | 48% | 36% |
| **Average (%)** | 81.94 | 82.45 | 64.52 | 62.45 | 81.55 | 82.32 | 82.19 | 80.26 | 31.87 | 30.06 |