RENDER THE ROBOT, GENERATE THE WORLD: ONE VIDEO WORLD MODEL ACROSS EMBODIMENTS

Anonymous authors
Paper under double-blind review

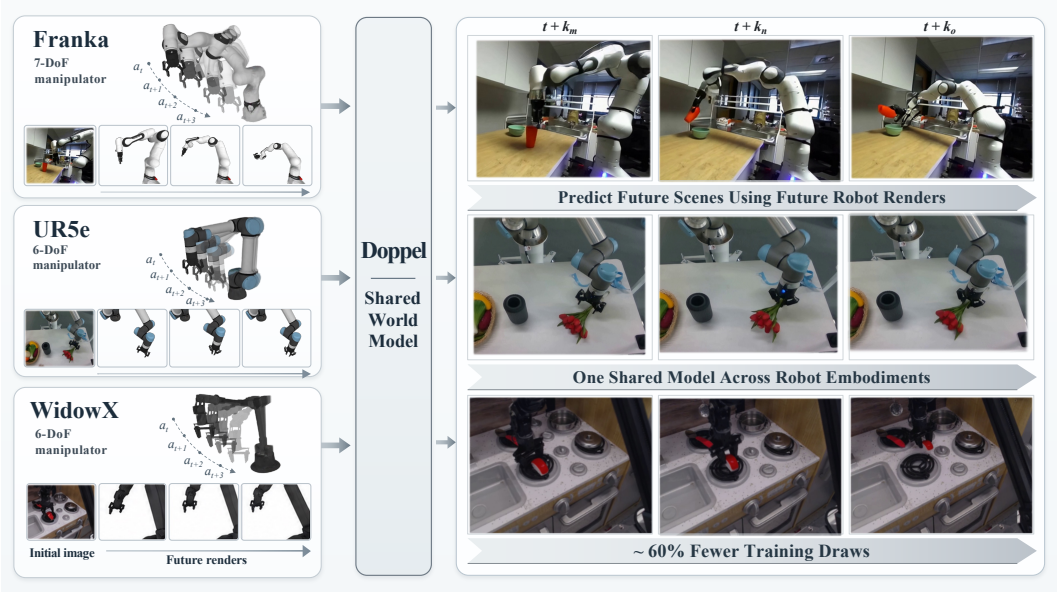

Figure 1: DOPPEL is a surprisingly simple approach for incorporating future embodiment renderings and improving robot-object interactions in action-conditioned video world models.

ABSTRACT

Predicting the consequences of an agent's actions requires modeling both the embodiment's motion and the environment's response. Existing video world models typically learn both jointly from numerical action vectors, even though known robot geometry and kinematics already provide much of the information needed to specify the intended robot motion. A general-purpose world model should capture these dynamics across multiple embodiments within a single model. Our key idea is to render the robot along a proposed joint-space trajectory and condition the video model directly on these future robot configurations. To this end, we introduce DOPPEL, an embodiment-agnostic video world model for learning a dynamics model shared across embodiments. DOPPEL jointly conditions on camera-aligned robot renderings and numerical action trajectories. Using forward kinematics and differentiable rendering, we explicitly provide the model with the robot's intended motion. This allows the shared model to focus its capacity on object motion and interaction-driven changes in the environment. Representing robot motion in image space also provides a common conditioning interface across embodiments with different geometries and action spaces. To enable this approach at scale, we recover missing camera extrinsics from offline robot datasets and align the resulting renderings with recorded observations. Using these aligned renderings, we train a single shared video backbone and conditioning adapter jointly on DROID, RoboMIND, and Bridge, spanning Franka, UR5e, and WidowX embodiments. Across three embodiments, DOPPEL improves PSNR over action-only baselines by 1.31 dB on unseen trajectories and 1.36 dB under OOD evaluation, while predicting more consistent object motion.

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.

1 INTRODUCTION

"Civilization advances by extending the number of important operations which we can perform without thinking about them."
Alfred North Whitehead, *An Introduction to Mathematics* (1911)

When we reach for a cup, we have a sense of where our arm is and can anticipate how it will move. Proprioceptive feedback and internal predictions of movement help us estimate the position of our limbs, even when we cannot see them (Wolpert et al., 1995). Knowing our own motion, however, does not fully determine what happens to the cup: it may slide, tip over, or stay in our grasp, depending on how we make contact. In everyday physical interactions, we intuitively combine knowledge of our own body with predictions of how the surrounding world will respond. We build on this factorization by using known robot kinematics and camera extrinsics to render the intended robot motion, allowing the world model to focus on modeling the environment's response.

For a robot, its geometry and kinematics provide an explicit description of its body. We study how to incorporate this knowledge in video world models, which *predict future observations from an initial image and a proposed action trajectory.* Recent methods adapt pretrained video generators to this task by conditioning them on numerical action vectors (for example robot joint poses) (Bagchi et al., 2026). The model must then learn both how the actions move the robot in the image and how that motion affects the environment. However, much of the first mapping can already be computed from the robot's geometry, joint configurations, and camera parameters.

Our key idea is to render the robot along the proposed action trajectory and condition the video model on these future frames. We use forward kinematics to compute link poses from joint configurations and the robot URDF, and a learned differentiable rendering model to generate camera-aligned robot renderings from the observed camera viewpoint. The resulting frames show where the arm and gripper would be if the robot followed the provided action trajectory. *In the spirit of Whitehead's observation, we use this known structure to simplify what must be learned*. With the intended robot motion provided explicitly through visual renderings, the video model can focus on predicting its consequences in the environment: whether a grasp would remain stable, how far would a pushed object move, and which way would a cup rotate.

We introduce **DOPPEL**, an embodiment-agnostic video world model that uses this conditioning. DOPPEL combines future robot renderings with the numerical action trajectory. Like a doppelgänger, the rendered robot is a faithful double of the real one; Doppel lets this double show where the robot will move, so the model can focus on how the world responds. Renderings describe the robot's geometry and motion in the image, while the numerical inputs retain information that may be visually ambiguous. Expressing actions this way also enables learning across robots with different joint structures and control spaces. Although Franka, UR5e, and WidowX move differently, their renderings describe the intended motion in a common image space. We therefore train a shared conditioning adapter and video backbone across these embodiments, using robot-specific geometry and normalization only to construct the inputs. This yields a single set of model weights that can represent multiple robot embodiments, avoiding the need to train and maintain a separate world model for each robot.

Realizing this representation across large-scale robot datasets requires accurate alignment between the rendered robot and the observation camera. This alignment depends on the camera pose relative to the robot base, which is often missing from manipulation datasets. We recover these missing extrinsics by optimizing the differentiable renderer to align rendered robot motion with the observations. This allows us to construct camera-aligned conditioning videos across datasets with different robots and camera setups.

Finally, because our formulation explicitly separates the robot's intended motion from its effects on the environment, we evaluate these components separately. We measure prediction quality over robot and moving-object regions, allowing us to distinguish accurate generation of the robot's motion from accurate prediction of the resulting environment dynamics. We train DOPPEL jointly on DROID (Franka), ROBOMIND (UR5e), and BRIDGE (WidowX), and evaluate generalization to unseen trajectories and environments. Our contributions are:

- **Embodiment-agnostic conditioning.** We introduce DOPPEL, a shared video world model conditioned on future robot renderings and numerical actions. This representation describes robot motion in a common image space across embodiments (Sec. 4).

- **Joint cross-embodiment learning.** We train one set of weights across all three robots, outperforming action-only experts. Against a shared action-only baseline, we achieve higher average PSNR and SCS with 63.8% fewer training draws (Sec. 6).
- **Interaction-focused evaluation.** We evaluate robot motion and object dynamics separately using region-specific metrics. Our model improves visual fidelity and motion consistency over action-only and skeleton-conditioned baselines.
- **A camera-calibrated corpus.** We release recovered camera extrinsics for over 32k WidowX and UR5e demos, along with code to calibrate additional datasets and embodiments (Sec. 4.1).

2 Related Work

Video world models. World models predict how an environment evolves under an agent's actions and use these predictions for planning and policy learning (Ha & Schmidhuber, 2018; Hafner et al., 2025; Wu et al., 2022). In robotics, action-conditioned video prediction has benefits in model-predictive control (Finn & Levine, 2017; Ebert et al., 2018) and policy improvement (Guo et al., 2026a; Seo et al., 2026). Recent large-scale models extend this idea to interactive simulators learned from video (Yang et al., 2024; Bruce et al., 2024; NVIDIA et al., 2025; Assran et al., 2025). Since pretrained video generators (Blattmann et al., 2023; Yang et al., 2025; Wan et al., 2025) provide strong visual priors, they can be adapted into action-conditioned world models (Huang et al., 2026a; Ye et al., 2026a;b), used to synthesize demonstrations from which actions are inferred for policy learning (Du et al., 2023; Jang et al., 2025), or used as learned simulators for policy evaluation (Quevedo et al., 2026; Guo et al., 2026b).

Most robot world models condition on numerical actions per frame (Zhu et al., 2025b), with multi-view memory (Guo et al., 2026b), or through timestep modulation (Bagchi et al., 2026); others jointly model video and actions (Zhu et al., 2025a). These models must learn how actions translate into image-space robot motion, although kinematics and camera geometry largely determine this mapping. Recent work introduces geometry through skeletons (Wu & Gao, 2026), 3D robot point flow (Huang et al., 2026b), occupancy-derived visual priors (Yang et al., 2026), depth- and normal-aware 4D representations (Zhen et al., 2025), and rendered robot motion for demonstration synthesis (Ye et al., 2026b;a). We combine camera-aligned robot renderings with numerical actions in one shared model, providing robot motion explicitly so it can focus on the environment's response. We can use joint-space trajectories from a policy's action chunks or an off-the-shelf planner.

Cross-embodiment learning. Pooling data across robots lets policies share experience across embodiments (O'Neill et al., 2024; Ghosh et al., 2024; Kim et al., 2024; Black et al., 2025), typically through embodiment-specific tokenizers or action heads (Wang et al., 2024; Doshi et al., 2024). Other work closes the embodiment gap in image space by rendering the target robot in place of the source (Chen et al., 2024a;b; Lepert et al., 2025) or replacing both with a shared segmentation mask (Lepert et al., 2024). Motion can also be expressed visually, as trajectory sketches (Gu et al., 2024), object flow (Xu et al., 2024), point trajectories (Wu et al., 2024; Geng et al., 2025), projected skeletons (Wang et al., 2025b; Wu & Gao, 2026), or shared particle representations for dynamics (He et al., 2026).

We bring both joint space and image-space embodiment representations together in video world models. Differentiable rendering (Liu et al., 2024) places each robot's motion in a common camera-aligned space, showing shape, gripper geometry, and self-occlusion, while numerical actions retain joint and gripper details that are ambiguous in the image. This lets us train one backbone and adapter jointly across robots, keeping robot-specific assets and normalization and padding actions to a common size. Many offline datasets lack camera extrinsics, so we recover them before training for joint training across embodiments.

3 Setting and Preliminaries

We learn a single visual forward-dynamics model across robots with different geometries and kinematics. Let $e \in \mathcal{E}$ denote an embodiment with a d_e-dimensional configuration space. For each embodiment, we use episodes of time-aligned RGB observations $\mathbf{o}_{0:T-1}$ and configuration trajecto-

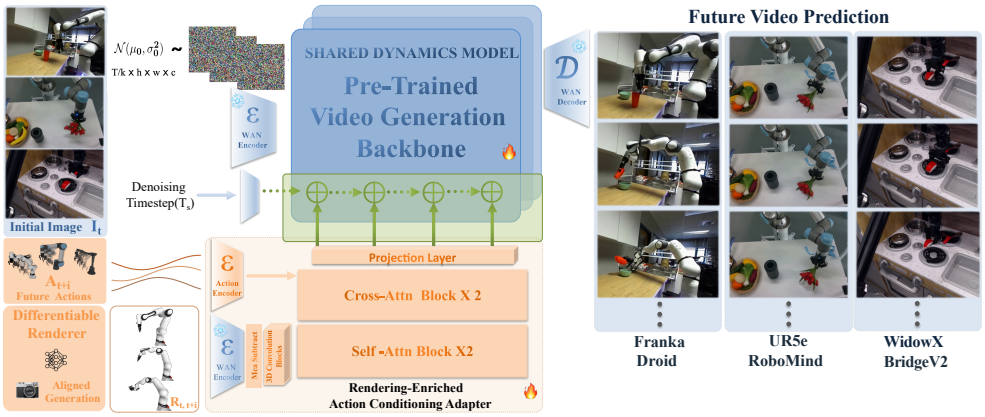

Figure 2: DOPPEL Architecture. We encode camera-aligned robot renderings with a frozen VAE and fuse their features with numerical actions. We use the resulting representation to modulate the video backbone, which predicts future frames from the initial observation.

ries $a^e_{0:T-1}$, where $a^e_t \in \mathbb{R}^{d_e}$ contains the joint positions and gripper state. We also assume access to a URDF $\mathcal{U}_e$ describing the robot's geometry and kinematics.

Given an initial image $\mathbf{o}_0$ and a planned configuration trajectory $a^e_{1:T-1}$, we predict future images showing how the robot moves and how the environment responds:

$$p_\theta\left(\mathbf{o}_{1:T-1} \mid \mathbf{o}_0,\, a^e_{1:T-1},\, \mathcal{U}_e\right), \tag{1}$$

where we share θ across embodiments and model uncertainty in physical properties and object states not visible in the initial image. Since joint vectors differ in size and meaning across robots, we need a common action representation. We evaluate unseen trajectories in familiar environments and shifts to completely new objects, scenes, and environments.

Latent video flow matching. We build on a pretrained image-to-video rectified-flow transformer (Wan et al., 2025). We use a frozen causal VAE to encode a clip of T RGB frames into a latent tensor $\mathbf{z} \in \mathbb{R}^{C\times T_z\times H_z\times W_z}$, where T_z, H_z, and W_z are the compressed temporal and spatial dimensions. Given Gaussian noise $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$, we interpolate between the clean latent and noise at flow timestep $\tau \in [0, 1]$:

$$\mathbf{z}_\tau = (1-\tau)\mathbf{z} + \tau\boldsymbol{\epsilon}, \tag{2}$$

where τ is distinct from video time. We train the model to predict the target velocity $\boldsymbol{\epsilon} - \mathbf{z}$.

To condition on the initial observation, we replace the first temporal slice of $\mathbf{z}_\tau$ with its clean value. We compute the loss over the remaining slices using a mask $M \in \{0,1\}^{1\times T_z\times 1\times 1}$, with $M_0 = 0$ and $M_j = 1$ for $j > 0$:

$$\mathcal{L}_{\text{flow}} = \mathbb{E}_{\tau,\boldsymbol{\epsilon}}\left[\left\|M \odot \left(f_\theta(\mathbf{z}_\tau, \tau, c) - (\boldsymbol{\epsilon} - \mathbf{z})\right)\right\|_2^2\right], \tag{3}$$

where c is the conditioning provided to the transformer. We pass an empty string to the text encoder throughout training and evaluation. Our predictions therefore depend on the initial image and the robot conditioning described below.

Differentiable robot rendering. We use Dr. Robot (Liu et al., 2024) to render the robot at a given configuration. For each embodiment e, we use a set of 3D Gaussians $\mathcal{G}^0_e$ to represent the robot in a canonical pose. Given a configuration a^e_t, we compute the joint transformations through forward kinematics and deform the Gaussians using an implicit linear blend skinning function D_e. We then apply the pose-conditioned appearance model X_e, which adjusts their orientation, scale, opacity, and spherical-harmonic coefficients:

$$\mathcal{G}_e(a^e_t) = X_e\left(D_e(\mathcal{G}^0_e, a^e_t), a^e_t\right). \tag{4}$$

We render the posed Gaussians with a differentiable rasterizer using the camera intrinsics K_i and extrinsics C_i:

$$r^e_t = \pi_{K_i,C_i}(\mathcal{G}_e(a^e_t)), \tag{5}$$

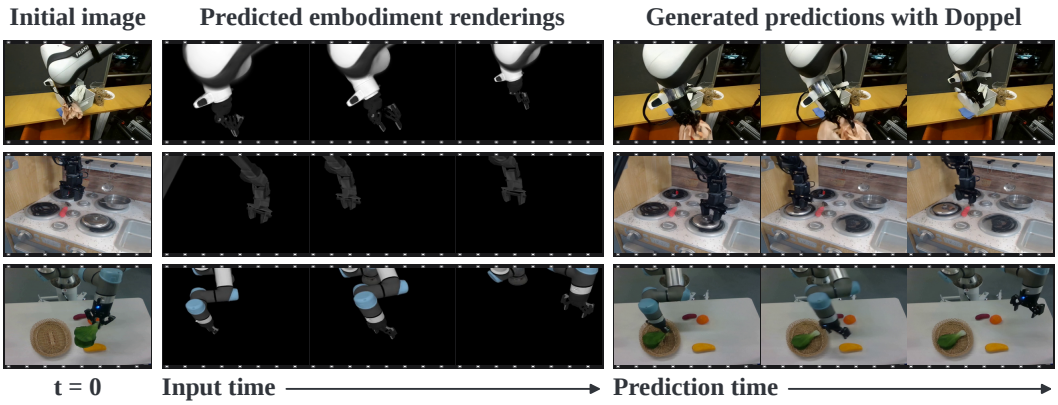

Figure 3: We show predictions with DOPPEL on Franka, WidowX, and UR5e arms in out-of-domain scenes with unseen objects (top to bottom). Each row shows the initial image, future robot renderings used as conditioning, and our predicted frames. We use the same model across all three embodiments. Our model preserves embodiment-consistent motion while predicting plausible environment responses under substantial distribution shift.

where $r_t^e \in \mathbb{R}^{H\times W\times 3}$ is the robot image for episode i at video time t. We repeat this for the configuration trajectory $a^e_{0:T-1}$ to obtain a rendering sequence $r^e_{0:T-1}$ aligned with the observation camera. The renderer is differentiable with respect to both robot configurations and camera parameters. We use its camera gradients to recover missing extrinsics.

4 DOPPEL

Our key idea is to provide the robot's intended motion explicitly so the video model can focus on predicting its effects on the environment. We first recover the camera pose relative to the robot base. We then render the robot along the supplied configuration trajectory from this viewpoint. DOPPEL uses our render-enriched action conditioning adapter to combine these renderings with the numerical joint and gripper trajectories. The renderings show where the robot moves in the image, while the numerical inputs retain configuration information that may be difficult to see. We use this representation to condition DOPPEL, a single image-to-video model shared across embodiments.

4.1 CAMERA CALIBRATION PREPROCESSING

We need to render the robot from the same viewpoint as the recorded observations. However, the source datasets have missing or inconsistent camera extrinsics. We recover these extrinsics using the calibration procedure of Jangir et al. (2026). We optimize a six-degree-of-freedom camera-to-robot transform by aligning rendered robot trajectories with the recorded video using RGB, optical flow, and DINOv2 features. We keep the robot renderer fixed and optimize only the camera pose using analysis by synthesis approach.

We calibrate each fixed camera once using a recorded RGB trajectory, the corresponding robot configuration trajectory, the robot's URDF, and the camera intrinsics. We reuse the same differentiable renderer that produces our conditioning videos and recover extrinsics without calibration targets. The recovered camera-to-robot transform is then reused for all demonstrations captured with that camera and remains fixed unless the camera viewpoint changes. We replace the original procedure's optical-flow estimator with RAFT (Teed & Deng, 2020) and optimize multiple camera-pose initializations in parallel. We use a coarse pose search for the initialization. After optimization, we select the pose with the lowest alignment loss and keep it fixed while training with DOPPEL. This procedure removes the need for manual camera calibration by automatically recovering missing camera extrinsics during dataset preprocessing at scale. Qualitative results of the camera calibration are shown in Fig. 12. See App. A.9.

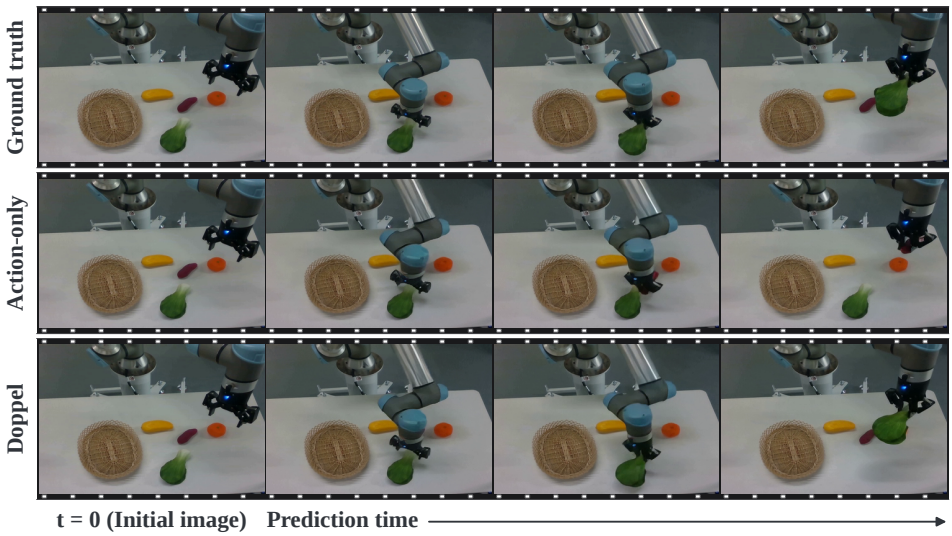

Figure 4: Qualitative comparison across trajectory and OOD generalization settings. Compared to the baseline we observe that DOPPEL generates much plausible robot-object interactions as intended by the provided action trajectory. Additional qualitative results are available on our project website (Appendix A.1) and a trajectory generalization illustration in Fig. 13.

4.2 RENDERING-ENRICHED DENSE ACTION REPRESENTATION

We combine robot renderings and numerical actions to produce one conditioning vector for each video token. Renderings describe the robot's motion in image space, while numerical actions provide joint and gripper information hidden by occlusion or ambiguous from a single view.

Render and action encoding. We encode the rendering sequence $\mathbf{r}_{0:T-1}$ with the same frozen causal VAE used for the target video. The encoded renderings have the same latent resolution as the target video. Most of each rendering is empty background, so we center each latent channel by subtracting its mean across frames and spatial locations. We then use a small 3D convolutional encoder to produce features on the backbone's token grid. We add separate positional embeddings for time, height, and width, and use self-attention to let tokens exchange information across the rendering sequence. We normalize the configuration trajectory $\mathbf{a}^e_{0:T-1}$ using the training statistics for each robot. We pad these vectors so all robots use inputs of the same size, although the joint represented by each coordinate can differ across robots. We apply a shared MLP to each configuration and add a learned embedding for its time index.

Render-action fusion. We let the render tokens attend to the action sequence. In this cross-attention step, render tokens supply the queries and action tokens supply the keys and values. Each output token keeps its location on the video grid and also contains information about the robot's configuration. We denote the complete render–action adapter by g_ϕ:

$$G = g_\phi\big(\mathbf{r}_{0:T-1}, \mathbf{a}^e_{0:T-1}\big) \in \mathbb{R}^{N\times d}, \tag{6}$$

where N is the number of video tokens and d is the backbone width. We write the conditioning vector at video token j as g_j.

Per-token timestep modulation. We follow Bagchi et al. (2026) and condition the video model through its flow-timestep embedding. We use a separate conditioning vector for each video token so the model receives information about robot motion at different positions and times. For token j, we add g_j to the embedding $e(s) \in \mathbb{R}^d$ of flow timestep s:

$$\widetilde{e}_j(s) = e(s) + g_j. \tag{7}$$

We pass $\widetilde{e}_j(s)$ through the backbone's existing projection to obtain the shift, scale, and gate parameters for the attention and feed-forward sublayers in every transformer block. We retain the pretrained

Table 1: Multi-embodiment corpus. DOPPEL is trained jointly on all three datasets, whereas each single-embodiment baseline is trained on one dataset.

Dataset	Embodiment	Train episodes	Train clips	Val. clips	Test clips	Test distribution shift
DROID	Franka Panda	16726	83511	200	200	Unseen lab environments
ROBOMIND	UR5e + 2F-85	14801	74005	200	200	Unseen objects
BRIDGE	WidowX	12387	61935	200	200	Unseen kitchen environments
Combined	Three embodiments	43914	219451	600	600	—

Table 2: Whole-frame prediction quality on the evaluation sets. Each single-dataset expert sees approximately 3.36M training draws, while multi-embodiment DOPPEL matches the per-dataset training exposure. OSCAR is fine-tuned over its pretrained checkpoint.

		Validation: unseen trajectories				OOD test			
Dataset	Method	PSNR ↑	SSIM ↑	LPIPS ↓	SCS ↑	PSNR ↑	SSIM ↑	LPIPS ↓	SCS ↑
DROID	Action-only	22.228	0.832	0.085	0.921	19.767	0.775	0.135	0.869
	OSCAR (Finetuned)	20.427	0.751	0.135	0.903	18.619	0.705	0.174	0.854
	DOPPEL (single)	24.077	0.860	0.068	0.938	**21.151**	0.799	0.115	**0.898**
	DOPPEL (multi)	**24.220**	**0.861**	**0.067**	**0.940**	21.147	**0.800**	**0.113**	0.897
BRIDGE	Action-only	25.869	0.866	0.052	0.943	20.555	0.752	0.132	0.863
	OSCAR (Finetuned)	22.559	0.771	0.117	0.914	19.487	0.633	0.211	0.854
	DOPPEL (multi)	**26.458**	**0.873**	**0.049**	**0.948**	**22.438**	**0.785**	**0.098**	**0.904**
ROBOMIND	Action-only	29.629†	0.916	0.041	0.937	29.629†	0.914	0.041	0.929
	OSCAR (Finetuned)	27.456	0.882	0.079	0.914	27.661	0.878	0.076	0.920
	DOPPEL (multi)	**30.964**	**0.927**	**0.037**	**0.940**	**30.436**	**0.919**	**0.038**	**0.937**

Best in **bold**, second-best underlined; ties share emphasis. †Val. and OOD PSNR differ only beyond 3 decimals.

attention and feed-forward architecture and give the adapter details and tensor dimensions in Appendix A.4.

Sharing across embodiments. We train a single render–action adapter and video backbone across all embodiments. Each robot uses its own frozen renderer and action-normalization statistics. We share all trainable parameters and use no embodiment tokens or robot-specific heads. The renderer supplies the geometry and kinematics of each robot, and we use the same world model to learn how its motion changes the scene.

5 EXPERIMENTAL SETUP

Datasets and splits. We train jointly on DROID (Franka), BRIDGE (WidowX), and ROBOMIND (UR5e) (Khazatsky et al., 2024; Walke et al., 2023; Wu et al., 2025). Franka and UR5e use Robotiq 2F-85 grippers. We sample datasets uniformly and balance environment categories, using short contiguous and temporally subsampled full-span clips. We evaluate held-out trajectories in familiar environments and OOD shifts to a new lab (DROID), a new kitchen (BRIDGE), and unseen objects and tasks (ROBOMIND). All evaluated embodiments appear in training. We give corpus sizes in Table 1, construction details in Appendix A.3, and examples in Appendix A.6.

Baselines. We compare DOPPEL against both an action-only (Bagchi et al., 2026) and geometric-conditioned baselines (Wu & Gao, 2026). The action-only baseline follows EgoWM-style (Bagchi et al., 2026) timestep-modulated action conditioning using the same Wan 2.1 backbone and corresponding dataset's training corpus. We compare single dataset experts with our multi-embodiment DOPPEL at approximately matched training exposure per dataset. In separate multi embodiment comparison, both models use same training recipe. OSCAR (Wu & Gao, 2026) serves as a geometry-conditioned system-level baseline; we fine-tune one model per embodiment on the corresponding training demonstrations using OSCAR's temporal windowing and evaluate on the same splits. Because OSCAR uses a different backbone and training formulation, we retain its native architecture

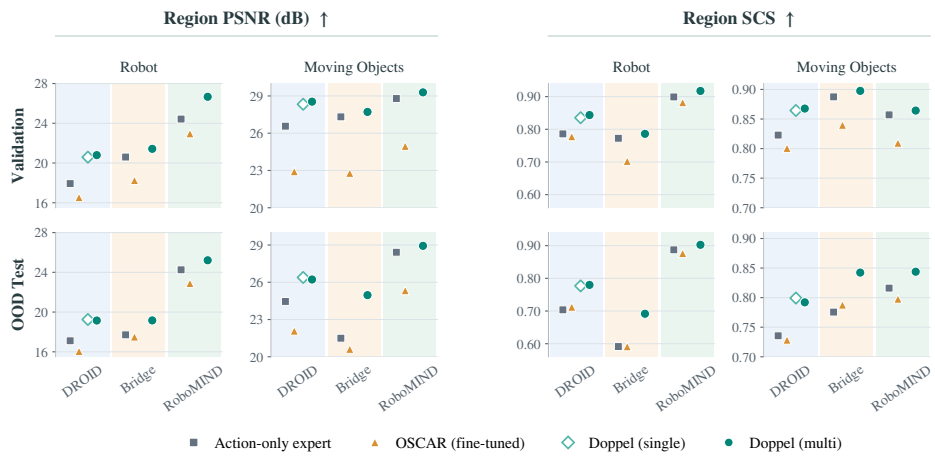

Figure 5: Region-specific prediction metrics across datasets. Single-dataset experts use 3.36M training draws; multi-embodiment DOPPEL matches per-dataset exposure. See A.10 for more details.

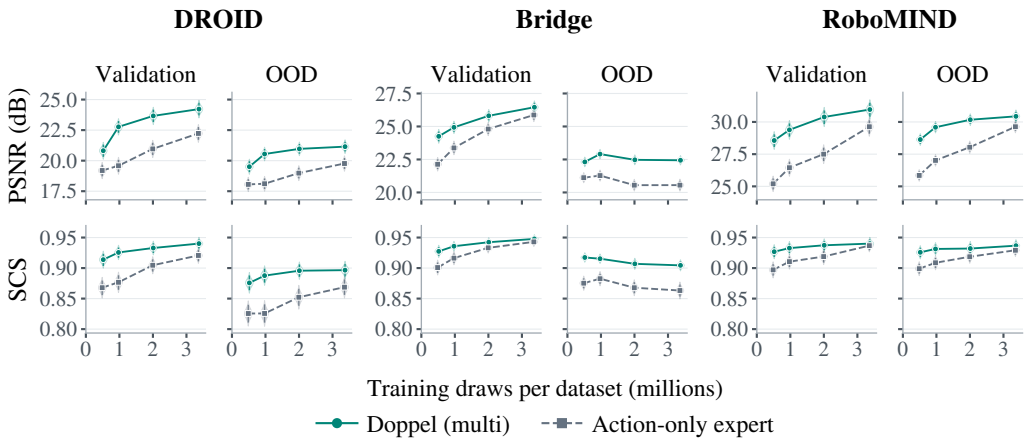

Figure 6: Training efficiency of DOPPEL versus single-dataset action-only experts. DOPPEL achieves higher validation and OOD prediction quality at comparable training draws.

while matching the evaluation resolution and removing text prompts for a fairer comparison. Full training configurations for all baselines are provided in Appendix A.5.

Evaluation metrics. We report whole-frame PSNR, SSIM, and LPIPS (Zhang et al., 2018), along with object SCS (Bagchi et al., 2026). We compute SCS as the overlap of SAM2 masks propagated from the same initial frame through predicted and reference videos (Ravi et al., 2025). We also measure PSNR separately over the robot arm, moving objects, and SCS over the arm and moving objects. We exclude the conditioning frame from all metrics and give definitions in Appendix A.8.

Implementation details. We use 8 H100 GPUs for single-embodiment training and 32 for the larger-batch multi-embodiment run. We give complete training recipes in Appendix Tables 4- 6.

6 RESULTS AND ANALYSIS

Prediction across embodiments. We improve whole-frame prediction quality over action-only conditioning and fine-tuned OSCAR (Wu & Gao, 2026) on all three datasets, across validation and OOD splits (Table 2). Our shared model outperforms separately trained action-only experts. On DROID, we obtain slightly better validation scores and comparable OOD performance to our single-embodiment model. We show predictions in held-out environments and with unseen objects in Fig. 3, using the same model across all three robots.

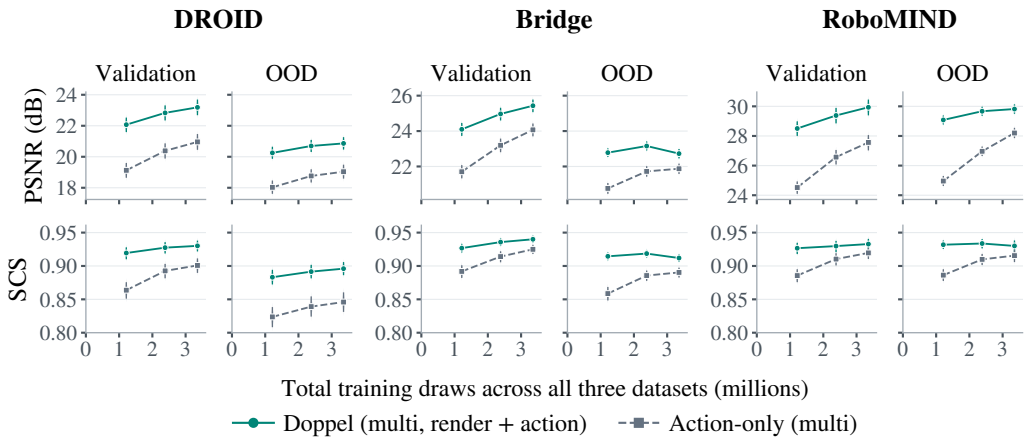

Figure 7: Training efficiency of DOPPEL versus the multi-embodiment action-only baseline. DOPPEL achieves higher validation and OOD prediction quality throughout 3.36M training draws.

Predicting object motion. We also improve predictions of the robot and moving objects, measured separately in Fig. 5. On the BRIDGE OOD split, we raise moving-object PSNR from 21.50 to 24.96 dB and SCS from 0.776 to 0.842 over the action-only baseline. We show more consistent object motion on unseen objects in Fig. 4. We can see that the baseline moves the arm but leaves the object on the table, while we predict the lift. We provide further examples of trajectory generalization in Fig. 13.

Training efficiency. We outperform separate action-only experts at comparable numbers of training draws per dataset (Fig. 6). We also compare with a shared action-only model, matching the batch size and optimization schedule and counting total draws across datasets (Fig. 7). Averaged across datasets, we obtain higher validation and OOD PSNR and SCS after 1.216M draws than this baseline obtains after 3.356M draws, using 63.8% fewer draws. These counts include repeated examples. We provide additional quantitative results in Appendix A.10.

7 DISCUSSION, LIMITATIONS, AND CONCLUSION

We presented DOPPEL, a shared video world model that uses known robot geometry to predict interactions across embodiments. DOPPEL combines numerical actions with camera-aligned future robot renderings. This provides the intended robot motion explicitly, reducing what the model must learn about how joint configurations translate into image-space motion. Across DROID, BRIDGE, and ROBOMIND, we improve whole-frame prediction over action-only and skeleton-conditioned baselines. Our region-specific evaluation also shows better predictions of moving-object appearance and motion, suggesting that explicit robot motion helps the model learn the environment's response. We observe these gains on held-out trajectories and OOD interactions with new objects and scenes. We retain them with one model trained across all three embodiments and outperform the shared action-only baseline with fewer training draws. This supports using geometric action representations to share dynamics learning across robots with different bodies and configuration spaces.

We require accurate robot geometry, a URDF, and camera calibration to construct the renderings. Although we recover missing extrinsics in offline datasets, unknown or rapidly changing viewpoints remain a limitation. Camera-aware or camera-invariant representations could reduce this dependence. We also condition on configuration trajectories; handling imperfect geometry or lower-level control commands would require modeling uncertainty in robot motion and execution. Our evaluation covers embodiments seen during training and one camera per trajectory. We leave transfer to unseen robots, multi-view prediction, and training across more robots and environments to future work. Combining pretrained video models with learned 3D representations may further reduce the need for explicit pixel-level alignment.

AI USE STATEMENT

For our work, generative AI tools were used to assist with drafting and editing text, improving clarity and presentation, formatting references, and coding assistance. The full manuscript was reviewed and verified by the authors. The authors take full responsibility for the content of this work, including any text or artifacts produced with the assistance of generative AI.

ETHICS STATEMENT

This work studies video world models for robotic manipulation using existing robot-learning datasets and does not involve the collection of new human-subject data. As with other learned world models for robotics, inaccurate predictions could lead to unsafe behavior if used directly for real-world control without appropriate validation and safeguards. Our experiments are conducted for research purposes, and we encourage appropriate safety evaluation before deploying such models in physical robotic systems. We follow the licenses and usage requirements of the datasets and models used in this work.

REPRODUCIBILITY STATEMENT

We provide implementation and evaluation details throughout the paper and supplementary material to facilitate reproduction of our results. The model architecture and training procedure are described in Sec. 4 and Sec. 5, including separate training configurations for single- and multi-embodiment models given in Appendix A.7. We document the datasets, evaluation splits, metrics, and camera-calibration procedure used to construct DOPPEL. Additional quantitative results, qualitative examples, and experimental details are provided in Appendix A.10. We will release code with trained checkpoints, calibration parameters, and the camera extrinsics used in our experiments to support reproduction and future research.

REFERENCES

Mahmoud 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. 3

Anurag Bagchi, Zhipeng Bao, Homanga Bharadhwaj, Yu-Xiong Wang, Pavel Tokmakov, and Martial Hebert. Walk through paintings: Egocentric world models from Internet priors. In *European Conference on Computer Vision (ECCV)*, pp. 472–489, 2026. doi: 10.1007/978-3-032-37098-3_26. 2, 3, 6, 7, 8, 19

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control. In *Robotics: Science and Systems (RSS)*, 2025. doi: 10.15607/RSS.2025.XXI.010. 3

Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, Varun Jampani, and Robin Rombach. Stable Video Diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023. 3

Jake Bruce, Michael Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, Yusuf Aytar, Sarah Bechtle, Feryal Behbahani, Stephanie Chan, Nicolas Heess, Lucy Gonzalez, Simon Osindero, Sherjil Ozair, Scott Reed, Jingwei Zhang, Konrad Zolna, Jeff Clune, Nando de Freitas, Satinder Singh, and Tim Rocktäschel. Genie: Generative interactive environments. In *International Conference on Machine Learning (ICML)*, volume 235, pp. 4603–4623, 2024. 3

Lawrence Yunliang Chen, Kush Hari, Karthik Dharmarajan, Chenfeng Xu, Quan Vuong, and Ken Goldberg. Mirage: Cross-embodiment zero-shot policy transfer with cross-painting. In *Robotics: Science and Systems (RSS)*, 2024a. doi: 10.15607/RSS.2024.XX.069. 3

Lawrence Yunliang Chen, Chenfeng Xu, Karthik Dharmarajan, Muhammad Zubair Irshad, Richard Cheng, Kurt Keutzer, Masayoshi Tomizuka, Quan Vuong, and Ken Goldberg. RoVi-Aug: Robot and viewpoint augmentation for cross-embodiment robot learning. In *Conference on Robot Learning (CoRL)*, volume 270, pp. 209–233, 2024b. 3

Ria Doshi, Homer Walke, Oier Mees, Sudeep Dasari, and Sergey Levine. Scaling cross-embodied learning: One policy for manipulation, navigation, locomotion and aviation. In *Conference on Robot Learning (CoRL)*, volume 270, pp. 496–512, 2024. 3

Yilun Du, Mengjiao Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Joshua B. Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 9156–9172, 2023. 3

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. 3

Chelsea Finn and Sergey Levine. Deep visual foresight for planning robot motion. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2786–2793, 2017. doi: 10.1109/ICRA.2017.7989324. 3

Daniel Geng, Charles Herrmann, Junhwa Hur, Forrester Cole, Serena Zhang, Tobias Pfaff, Tatiana Lopez-Guevara, Yusuf Aytar, Michael Rubinstein, Chen Sun, Oliver Wang, Andrew Owens, and

Deqing Sun. Motion prompting: Controlling video generation with motion trajectories. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–12, 2025. doi: 10.1109/CVPR52734.2025.00010. 3

Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Pannag Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy. In *Robotics: Science and Systems (RSS)*, 2024. doi: 10.15607/RSS.2024.XX.090. 3

Jiayuan Gu, Sean Kirmani, Paul Wohlhart, Yao Lu, Montserrat Gonzalez Arenas, Kanishka Rao, Wenhao Yu, Chuyuan Fu, Keerthana Gopalakrishnan, Zhuo Xu, Priya Sundaresan, Peng Xu, Hao Su, Karol Hausman, Chelsea Finn, Quan Vuong, and Ted Xiao. RT-Trajectory: Robotic task generalization via hindsight trajectory sketches. In *International Conference on Learning Representations (ICLR)*, pp. 44918–44941, 2024. 3

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. In *International Conference on Machine Learning (ICML)*, 2026a. URL https://openreview.net/forum?id=68zOaa2gOf. 3

Yanjiang Guo, Lucy Xiaoyang Shi, Jianyu Chen, and Chelsea Finn. Ctrl-World: A controllable generative world model for robot manipulation. In *International Conference on Learning Representations (ICLR)*, pp. 6121–6138, 2026b. 3

David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31, pp. 2450–2462, 2018. 3

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640(8059):647–653, 2025. doi: 10.1038/s41586-025-08744-2. 3

Zihao He, Bo Ai, Tongzhou Mu, Yulin Liu, Weikang Wan, Jiawei Fu, Yilun Du, Henrik I. Christensen, and Hao Su. Scaling cross-embodiment world models for dexterous manipulation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2026. URL https://arxiv.org/abs/2511.01177. 3

Siqiao Huang, Jialong Wu, Qixing Zhou, Shangchen Miao, and Mingsheng Long. Vid2World: Crafting video diffusion models to interactive world models. In *International Conference on Learning Representations (ICLR)*, pp. 10763–10806, 2026a. 3

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. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20765–20779, 2026b. 3

Joel Jang, Seonghyeon Ye, Zongyu Lin, Jiannan Xiang, Johan Bjorck, Yu Fang, Fengyuan Hu, Spencer Huang, Kaushil Kundalia, Yen-Chen Lin, Loïc Magne, Ajay Mandlekar, Avnish Narayan, You Liang Tan, Guanzhi Wang, Jing Wang, Qi Wang, Yinzhen Xu, Xiaohui Zeng, Kaiyuan Zheng, Ruijie Zheng, Ming-Yu Liu, Luke Zettlemoyer, Dieter Fox, Jan Kautz, Scott Reed, Yuke Zhu, and Linxi Fan. DreamGen: Unlocking generalization in robot learning through video world models. In *Conference on Robot Learning (CoRL)*, volume 305, pp. 5170–5194, 2025. 3

Yash Jangir, Yidi Zhang, Kashu Yamazaki, Chenyu Zhang, Kuan-Hsun Tu, Tsung-Wei Ke, Lei Ke, Yonatan Bisk, and Katerina Fragkiadaki. RobotArena ∞: Scalable robot benchmarking via real-to-sim translation. In *International Conference on Learning Representations (ICLR)*, pp. 25017–25043, 2026. 5

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, Peter David Fagan, Joey Hejna, Masha Itkina, Marion Lepert, Yecheng Jason Ma, Patrick Tree Miller, Jimmy Wu, Suneel Belkhale, Shivin Dass, Huy Ha, Arhan Jain, Abraham Lee, Youngwoon Lee, Marius Memmel, Sungjae Park, Ilija Radosavovic, Kaiyuan Wang, Albert Zhan, Kevin

Black, Cheng Chi, Kyle Beltran Hatch, Shan Lin, Jingpei Lu, Jean Mercat, Abdul Rehman, Pannag R Sanketi, Archit Sharma, Cody Simpson, Quan Vuong, Homer Rich Walke, Blake Wulfe, Ted Xiao, Jonathan Heewon Yang, Arefeh Yavary, Tony Z. Zhao, Christopher Agia, Rohan Baijal, Mateo Guaman Castro, Daphne Chen, Qiuyu Chen, Trinity Chung, Jaimyn Drake, Ethan Paul Foster, Jensen Gao, Vitor Guizilini, David Antonio Herrera, Minho Heo, Kyle Hsu, Jiaheng Hu, Muhammad Zubair Irshad, Donovon Jackson, Charlotte Le, Yunshuang Li, Kevin Lin, Roy Lin, Zehan Ma, Abhiram Maddukuri, Suvir Mirchandani, Daniel Morton, Tony Nguyen, Abigail O'Neill, Rosario Scalise, Derick Seale, Victor Son, Stephen Tian, Emi Tran, Andrew E. Wang, Yilin Wu, Annie Xie, Jingyun Yang, Patrick Yin, Yunchu Zhang, Osbert Bastani, Glen Berseth, Jeannette Bohg, Ken Goldberg, Abhinav Gupta, Abhishek Gupta, Dinesh Jayaraman, Joseph J Lim, Jitendra Malik, Roberto Martín-Martín, Subramanian Ramamoorthy, Dorsa Sadigh, Shuran Song, Jiajun Wu, Michael C. Yip, Yuke Zhu, Thomas Kollar, Sergey Levine, and Chelsea Finn. DROID: A large-scale in-the-wild robot manipulation dataset. In *Robotics: Science and Systems (RSS)*, 2024. doi: 10.15607/RSS.2024.XX.120. 7, 24

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Pannag Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An open-source vision-language-action model. In *Conference on Robot Learning (CoRL)*, volume 270, pp. 2679–2713, 2024. 3

Marion Lepert, Ria Doshi, and Jeannette Bohg. SHADOW: Leveraging segmentation masks for cross-embodiment policy transfer. In *Conference on Robot Learning (CoRL)*, volume 270, pp. 3536–3550, 2024. 3

Marion Lepert, Jiaying Fang, and Jeannette Bohg. Phantom: Training robots without robots using only human videos. In *Conference on Robot Learning (CoRL)*, volume 305, pp. 4545–4565, 2025. 3

Ruoshi Liu, Alper Canberk, Shuran Song, and Carl Vondrick. Differentiable robot rendering. In *Conference on Robot Learning (CoRL)*, volume 270, pp. 117–129, 2024. 3, 4, 17

NVIDIA, Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, Daniel Dworakowski, Jiaojiao Fan, Michele Fenzi, Francesco Ferroni, Sanja Fidler, Dieter Fox, Songwei Ge, Yunhao Ge, Jinwei Gu, Siddharth Gururani, Ethan He, Jiahui Huang, Jacob Huffman, Pooya Jannaty, Jingyi Jin, Seung Wook Kim, Gergely Klár, Grace Lam, Shiyi Lan, Laura Leal-Taixe, Anqi Li, Zhaoshuo Li, Chen-Hsuan Lin, Tsung-Yi Lin, Huan Ling, Ming-Yu Liu, Xian Liu, Alice Luo, Qianli Ma, Hanzi Mao, Kaichun Mo, Arsalan Mousavian, Seungjun Nah, Sriharsha Niverty, David Page, Despoina Paschalidou, Zeeshan Patel, Lindsey Pavao, Morteza Ramezanali, Fitsum Reda, Xiaowei Ren, Vasanth Rao Naik Sabavat, Ed Schmerling, Stella Shi, Bartosz Stefaniak, Shitao Tang, Lyne Tchapmi, Przemek Tredak, Wei-Cheng Tseng, Jibin Varghese, Hao Wang, Haoxiang Wang, Heng Wang, Ting-Chun Wang, Fangyin Wei, Xinyue Wei, Jay Zhangjie Wu, Jiashu Xu, Wei Yang, Lin Yen-Chen, Xiaohui Zeng, Yu Zeng, Jing Zhang, Qinsheng Zhang, Yuxuan Zhang, Qingqing Zhao, and Artur Zolkowski. Cosmos world foundation model platform for physical AI. *arXiv preprint arXiv:2501.03575*, 2025. 3

Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, Albert Tung, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anchit Gupta, Andrew Wang, Anikait Singh, Animesh Garg, Aniruddha Kembhavi, Annie Xie, Anthony Brohan, Antonin Raffin, Archit Sharma, Arefeh Yavary, Arhan Jain, Ashwin Balakrishna, Ayzaan Wahid, Ben Burgess-Limerick, Beomjoon Kim, Bernhard Schölkopf, Blake Wulfe, Brian Ichter, Cewu Lu, Charles Xu, Charlotte Le, Chelsea Finn, Chen Wang, Chenfeng Xu, Cheng Chi, Chenguang Huang, Christine Chan, Christopher Agia, Chuer Pan, Chuyuan Fu, Coline Devin, Danfei Xu, Daniel Morton, Danny Driess, Daphne Chen, Deepak Pathak, Dhruv Shah, Dieter Büchler, Dinesh Jayaraman, Dmitry Kalashnikov, Dorsa Sadigh, Edward Johns, Ethan Foster, Fangchen Liu, Federico Ceola, Fei Xia, Feiyu Zhao, Freek Stulp, Gaoyue Zhou, Gaurav S. Sukhatme, Gautam Salhotra, Ge Yan, Gilbert Feng, Giulio Schiavi, Glen Berseth, Gregory Kahn, Guanzhi Wang, Hao Su, Hao-Shu Fang, Haochen Shi, Henghui Bao, Heni Ben Amor, Henrik I Christensen, Hiroki Furuta, Homer Walke, Hongjie Fang, Huy Ha, Igor Mordatch, Ilija Radosavovic, Isabel Leal, Jacky Liang, Jad

Abou-Chakra, Jaehyung Kim, Jaimyn Drake, Jan Peters, Jan Schneider, Jasmine Hsu, Jeannette Bohg, Jeffrey Bingham, Jeffrey Wu, Jensen Gao, Jiaheng Hu, Jiajun Wu, Jialin Wu, Jiankai Sun, Jianlan Luo, Jiayuan Gu, Jie Tan, Jihoon Oh, Jimmy Wu, Jingpei Lu, Jingyun Yang, Jitendra Malik, João Silvério, Joey Hejna, Jonathan Booher, Jonathan Tompson, Jonathan Yang, Jordi Salvador, Joseph J. Lim, Junhyek Han, Kaiyuan Wang, Kanishka Rao, Karl Pertsch, Karol Hausman, Keegan Go, Keerthana Gopalakrishnan, Ken Goldberg, Kendra Byrne, Kenneth Oslund, Kento Kawaharazuka, Kevin Black, Kevin Lin, Kevin Zhang, Kiana Ehsani, Kiran Lekkala, Kirsty Ellis, Krishan Rana, Krishnan Srinivasan, Kuan Fang, Kunal Pratap Singh, Kuo-Hao Zeng, Kyle Hatch, Kyle Hsu, Laurent Itti, Lawrence Yunliang Chen, Lerrel Pinto, Li Fei-Fei, Liam Tan, Linxi Jim Fan, Lionel Ott, Lisa Lee, Luca Weihs, Magnum Chen, Marion Lepert, Marius Memmel, Masayoshi Tomizuka, Masha Itkina, Mateo Guaman Castro, Max Spero, Maximilian Du, Michael Ahn, Michael C. Yip, Mingtong Zhang, Mingyu Ding, Minho Heo, Mohan Kumar Srirama, Mohit Sharma, Moo Jin Kim, Naoaki Kanazawa, Nicklas Hansen, Nicolas Heess, Nikhil J Joshi, Niko Suenderhauf, Ning Liu, Norman Di Palo, Nur Muhammad Mahi Shafiullah, Oier Mees, Oliver Kroemer, Osbert Bastani, Pannag R Sanketi, Patrick Tree Miller, Patrick Yin, Paul Wohlhart, Peng Xu, Peter David Fagan, Peter Mitrano, Pierre Sermanet, Pieter Abbeel, Priya Sundaresan, Qiuyu Chen, Quan Vuong, Rafael Rafailov, Ran Tian, Ria Doshi, Roberto Martín-Martín, Rohan Baijal, Rosario Scalise, Rose Hendrix, Roy Lin, Runjia Qian, Ruohan Zhang, Russell Mendonca, Rutav Shah, Ryan Hoque, Ryan Julian, Samuel Bustamante, Sean Kirmani, Sergey Levine, Shan Lin, Sherry Moore, Shikhar Bahl, Shivin Dass, Shubham Sonawani, Shuran Song, Sichun Xu, Siddhant Haldar, Siddharth Karamcheti, Simeon Adebola, Simon Guist, Soroush Nasiriany, Stefan Schaal, Stefan Welker, Stephen Tian, Subramanian Ramamoorthy, Sudeep Dasari, Suneel Belkhale, Sungjae Park, Suraj Nair, Suvir Mirchandani, Takayuki Osa, Tanmay Gupta, Tatsuya Harada, Tatsuya Matsushima, Ted Xiao, Thomas Kollar, Tianhe Yu, Tianli Ding, Todor Davchev, Tony Z. Zhao, Travis Armstrong, Trevor Darrell, Trinity Chung, Vidhi Jain, Vincent Vanhoucke, Wei Zhan, Wenxuan Zhou, Wolfram Burgard, Xi Chen, Xiaolong Wang, Xinghao Zhu, Xinyang Geng, Xiyuan Liu, Xu Liangwei, Xuanlin Li, Yao Lu, Yecheng Jason Ma, Yejin Kim, Yevgen Chebotar, Yifan Zhou, Yifeng Zhu, Yilin Wu, Ying Xu, Yixuan Wang, Yonatan Bisk, Yoonyoung Cho, Youngwoon Lee, Yuchen Cui, Yue Cao, Yueh-Hua Wu, Yujin Tang, Yuke Zhu, Yunchu Zhang, Yunfan Jiang, Yunshuang Li, Yunzhu Li, Yusuke Iwasawa, Yutaka Matsuo, Zehan Ma, Zhuo Xu, Zichen Jeff Cui, Zichen Zhang, and Zipeng Lin. Open X-Embodiment: Robotic learning datasets and RT-X models. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6892–6903, 2024. doi: 10.1109/ICRA57147.2024.10611477. 3

Julian Quevedo, Ansh Kumar Sharma, Yixiang Sun, Varad Suryavanshi, Percy Liang, and Sherry Yang. WorldGym: World model as an environment for policy evaluation. In *International Conference on Learning Representations (ICLR)*, pp. 78932–78957, 2026. 3

Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. SAM 2: Segment anything in images and videos. In *International Conference on Learning Representations (ICLR)*, pp. 28085–28128, 2025. 8

Junwon Seo, Sushant Veer, Ran Tian, Wenhao Ding, Apoorva Sharma, Karen Leung, Edward Schmerling, Marco Pavone, and Andrea Bajcsy. StressDream: Steering video world models for robust policy evaluation and improvement. In *Conference on Robot Learning (CoRL)*, 2026. URL https://arxiv.org/abs/2606.00267. 3

Zachary Teed and Jia Deng. RAFT: Recurrent all-pairs field transforms for optical flow. In *European Conference on Computer Vision (ECCV)*, pp. 402–419, 2020. doi: 10.1007/978-3-030-58536-5_24. 5

Homer Walke, Kevin Black, Abraham Lee, Moo Jin Kim, Max Du, Chongyi Zheng, Tony Zhao, Philippe Hansen-Estruch, Quan Vuong, Andre He, Vivek Myers, Kuan Fang, Chelsea Finn, and Sergey Levine. BridgeData V2: A dataset for robot learning at scale. In *Conference on Robot Learning (CoRL)*, volume 229, pp. 1723–1736, 2023. 7

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Mengyang Feng, Ningyi

Zhang, Pandeng Li, Pingyu Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang, Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng Zhou, Wente Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong, Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025. 3, 4

Lirui Wang, Xinlei Chen, Jialiang Zhao, and Kaiming He. Scaling proprioceptive-visual learning with heterogeneous pre-trained transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 124420–124450, 2024. 3

Ruicheng Wang, Sicheng Xu, Cassie Dai, Jianfeng Xiang, Yu Deng, Xin Tong, and Jiaolong Yang. MoGe: Unlocking accurate monocular geometry estimation for open-domain images with optimal training supervision. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5261–5271, 2025a. doi: 10.1109/CVPR52734.2025.00496. 24

Yuang Wang, Chao Wen, Haoyu Guo, Sida Peng, Minghan Qin, Hujun Bao, Xiaowei Zhou, and Ruizhen Hu. Precise action-to-video generation through visual action prompts. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 12713–12724, 2025b. doi: 10.1109/ICCV51701.2025.01181. 3

Daniel M. Wolpert, Zoubin Ghahramani, and Michael I. Jordan. An internal model for sensorimotor integration. *Science*, 269(5232):1880–1882, 1995. doi: 10.1126/science.7569931. 2

Kun Wu, Chengkai Hou, Jiaming Liu, Zhengping Che, Xiaozhu Ju, Zhuqin Yang, Meng Li, Yinuo Zhao, Zhiyuan Xu, Guang Yang, Shichao Fan, Xinhua Wang, Fei Liao, Zhen Zhao, Guangyu Li, Zhao Jin, Lecheng Wang, Jilei Mao, Ning Liu, Pei Ren, Qiang Zhang, Yaoxu Lyu, Mengzhen Liu, Jingyang He, Yulin Luo, Zeyu Gao, Chenxuan Li, Chenyang Gu, Yankai Fu, Di Wu, Xingyu Wang, Sixiang Chen, Zhenyu Wang, Pengju An, Siyuan Qian, Shanghang Zhang, and Jian Tang. RoboMIND: Benchmark on multi-embodiment intelligence normative data for robot manipulation. In *Robotics: Science and Systems (RSS)*, 2025. doi: 10.15607/RSS.2025.XXI.152. 7

Philipp Wu, Alejandro Escontrela, Danijar Hafner, Ken Goldberg, and Pieter Abbeel. DayDreamer: World models for physical robot learning. In *Conference on Robot Learning (CoRL)*, volume 205, pp. 2226–2240, 2022. 3

Weijia Wu, Zhuang Li, Yuchao Gu, Rui Zhao, Yefei He, David Junhao Zhang, Mike Zheng Shou, Yan Li, Tingting Gao, and Di Zhang. DragAnything: Motion control for anything using entity representation. In *European Conference on Computer Vision (ECCV)*, pp. 331–348, 2024. doi: 10.1007/978-3-031-72670-5_19. 3

Zhuoyuan Wu and Jun Gao. OSCAR: Omni-embodiment action-conditioned world model for robotics. *arXiv preprint arXiv:2606.04463*, 2026. 3, 7, 8, 20

Mengda Xu, Zhenjia Xu, Yinghao Xu, Cheng Chi, Gordon Wetzstein, Manuela Veloso, and Shuran Song. Flow as the cross-domain manipulation interface. In *Conference on Robot Learning (CoRL)*, volume 270, pp. 2475–2499, 2024. 3

Sherry Yang, Yilun Du, Kamyar Ghasemipour, Jonathan Tompson, Leslie Kaelbling, Dale Schuurmans, and Pieter Abbeel. Learning interactive real-world simulators. In *International Conference on Learning Representations (ICLR)*, pp. 45210–45234, 2024. 3

Xiuyu Yang, Bohan Li, Shaocong Xu, Nan Wang, Chongjie Ye, Zhaoxi Chen, Minghan Qin, Yikang Ding, Zheng Zhu, Xin Jin, Hang Zhao, and Hao Zhao. ORV: 4D occupancy-centric robot video generation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1053–1066, 2026. 3

Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, Da Yin, Yuxuan Zhang, Weihan Wang, Yean Cheng, Bin Xu, Xiaotao Gu, Yuxiao Dong, and Jie Tang. CogVideoX: Text-to-video diffusion models with an expert transformer. In *International Conference on Learning Representations (ICLR)*, pp. 83048–83077, 2025. 3

Junjie Ye, Rong Xue, Basile Van Hoorick, Runhao Li, Harshitha Rajaprakash, Pavel Tokmakov, Muhammad Zubair Irshad, Vitor Guizilini, and Yue Wang. RoboDream: Compositional world models for scalable robot data synthesis. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2026a. 3

Junjie Ye, Rong Xue, Basile Van Hoorick, Pavel Tokmakov, Muhammad Zubair Irshad, Yue Wang, and Vitor Guizilini. AnchorDream: Repurposing video diffusion for embodiment-aware robot data synthesis. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2026b. 3

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 586–595, 2018. doi: 10.1109/CVPR.2018.00068. 8

Haoyu Zhen, Qiao Sun, Hongxin Zhang, Junyan Li, Siyuan Zhou, Yilun Du, and Chuang Gan. TesserAct: Learning 4D embodied world models. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5337–5347, 2025. doi: 10.1109/ICCV51701.2025.00507. 3

Chuning Zhu, Raymond Yu, Siyuan Feng, Benjamin Burchfiel, Paarth Shah, and Abhishek Gupta. Unified world models: Coupling video and action diffusion for pretraining on large robotic datasets. In *Robotics: Science and Systems (RSS)*, 2025a. doi: 10.15607/RSS.2025.XXI.015. 3

Fangqi Zhu, Hongtao Wu, Song Guo, Yuxiao Liu, Chilam Cheang, and Tao Kong. IRASim: A fine-grained world model for robot manipulation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9834–9844, 2025b. doi: 10.1109/ICCV51701.2025.00917. 3

A Appendix

A.1 Website and Video

Additional qualitative results are available on our project webpage, including extended video comparisons across DROID, BRIDGE, and ROBOMIND, trajectory- and environment-generalization examples, and side-by-side comparisons with our baselines. We encourage reviewers to view the webpage for the full video results, which are difficult to represent comprehensively in static paper figures.

https://submit15330.github.io/

A.2 Rendering Pipeline

We render the robot along the supplied joint and gripper trajectory using an articulated Gaussian robot model and the target camera calibration. The renderer is fixed during world-model training. RGB observations, robot configurations, and rendered frames are constructed using the same sampled source-frame indices, ensuring temporal alignment across all conditioning signals. When constructing training windows, we additionally verify that cached renderings correspond to the same temporal sampling grid.

The robot-only renderings encode the embodiment's geometry and motion directly in image coordinates; they contain no future scene or object information. Figure 8 shows representative alignment between the rendered robot trajectories and the corresponding RGB observations across all three datasets.

For each embodiment, we use the differentiable robot-rendering framework of Liu et al. (2024). Starting from the robot's URDF, we train an articulated Gaussian model using the original renderer architecture and training procedure provided by the authors. We train a separate renderer for each robot embodiment and keep its parameters frozen when training DOPPEL.

A.3 Dataset Construction and Evaluation Splits

Training data. We construct training clips from demonstrations in DROID, ROBOMIND, and BRIDGE. Within each dataset, we balance sampling across environment categories. From each demonstration, we extract up to five clips: four contiguous windows capturing local interaction dynamics and one full-span window covering the entire demonstration at a coarser temporal resolution. Demonstrations that are too short contribute fewer clips. Table 1 reports the resulting numbers of demonstrations and training clips. During multi-embodiment training, we sample uniformly across the three datasets, such that each embodiment contributes approximately one third of the optimization batches.

Evaluation splits. We evaluate generalization along two distinct axes. The validation split measures *trajectory generalization*: for each embodiment, we reserve 200 clips from held-out demonstrations collected in environments that remain represented in the training set. These comprise 100 contiguous-window clips and 100 full-span clips. This split tests whether the model can predict previously unseen robot trajectories within familiar environment distributions.

The test split measures *environment generalization* under three complementary distribution shifts. For DROID, we hold out an entire laboratory; no scene, camera, operator, or task from this laboratory appears in the training set. For BRIDGE, whose demonstrations are collected in toy-kitchen environments, we hold out one complete kitchen environment. For ROBOMIND, we hold out four object-centric tasks and evaluate interactions involving object instances absent from the training set. These splits therefore test generalization to a completely new laboratory, an unseen instance of a semantically familiar environment, and unseen objects and tasks, respectively.

Each dataset contributes 200 test clips, evenly divided between 100 contiguous-window clips and 100 full-span clips. We report validation and test results separately because predicting a novel trajectory in a familiar environment and predicting interaction dynamics under an environmental distribution shift capture different generalization capabilities.

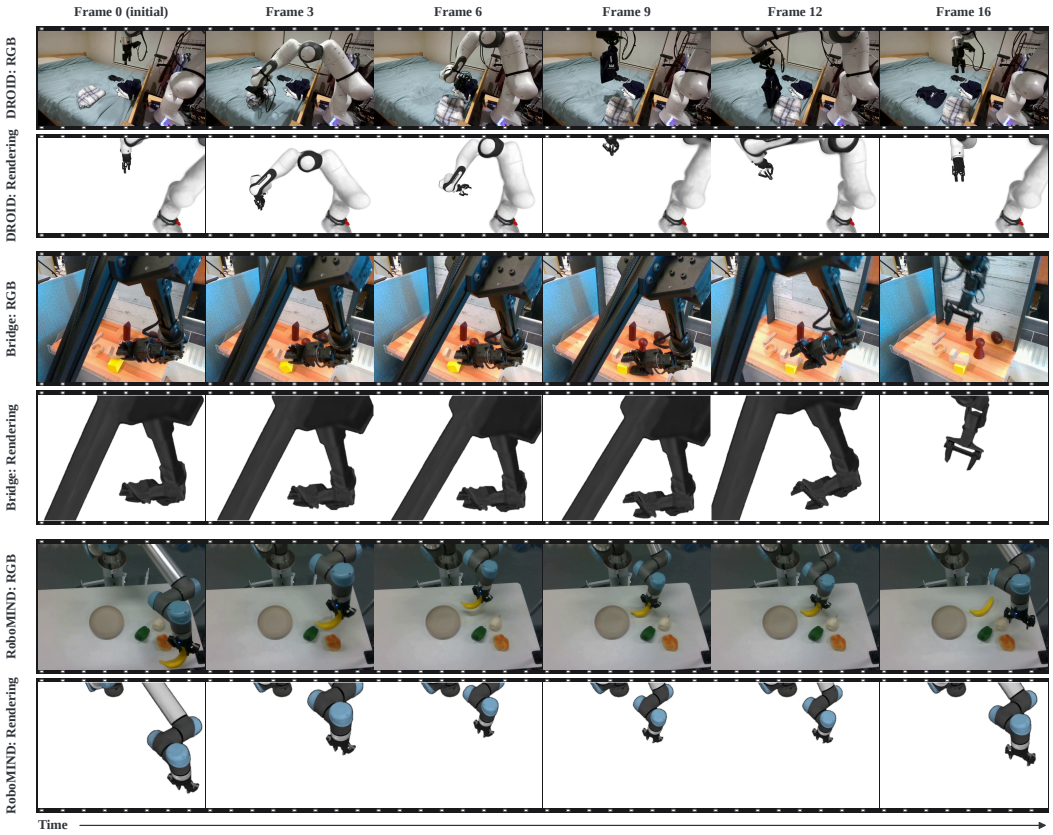

Figure 8: Alignment between the observed and rendered robot trajectories. The top row shows frames from the real RGB trajectory, and the bottom row shows the corresponding target-view robot renderings. The renderings express the commanded embodiment motion in the same image coordinates as the observations, providing the dense visual component of DOPPEL.

A.4 ADAPTER ARCHITECTURE DETAILS

This section provides the architectural details of the render–action adapter g_ϕ introduced in Sec. 4.2. The adapter contains 30.49M parameters and is shared across all embodiments.

Render normalization. Robot-only renderings are sparse, with most spatial locations containing empty background. We first encode the rendering sequence $\mathbf{r}_{0:T-1}$ using the same frozen causal VAE of Wan 2.1:

$$z^{\text{rend}} = E_{\text{vae}}(r_{0:T-1})\,. \tag{8}$$

Before applying the trainable render encoder, we subtract the per-sample, per-channel spatiotemporal mean:

$$\bar{z}^{\text{rend}}_{c,t,h,w} = z^{\text{rend}}_{c,t,h,w} - \frac{1}{T_z H_z W_z} \sum_{t',h',w'} z^{\text{rend}}_{c,t',h',w'}. \tag{9}$$

This prevents the empty-backgrounds from making the renders tokens very similar. If this mean subtractions is skipped the adapter can collapse to zero weights. We compute this operation in fp32, since the robot signal can be small relative to the shared latent response of the empty rendering background.

Render encoder. The centered render latent is processed by three 3D convolutions with GELU activations:

1. $\text{Conv3D}(16, 256, 1 \times 3 \times 3)$

2. $\mathrm{Conv3D}(256, 512, 3 \times 3 \times 3)$
3. $\mathrm{Conv3D}(512, 512, 1 \times 3 \times 3)$

The output grid is flattened into 2025 tokens of width 512. We then add factorized learned spatiotemporal positional embeddings,

$$p_{t,h,w} = p_t^{\text{time}} + p_h^{\text{height}} + p_w^{\text{width}}, \tag{10}$$

using temporal, height, and width embedding tables of maximum sizes 32, 32, and 64, respectively.

Two pre-normalization self-attention blocks contextualize the render tokens. Each block uses eight attention heads, an FFN expansion factor of four, and residual connections around both the attention and FFN sublayers. This global mixing allows information from the spatially sparse robot region to reach tokens outside the convolutional encoder's local receptive field.

Action encoder. Each frame is associated with an eight-dimensional configuration vector containing up to seven joint values and one gripper value. We normalize each embodiment using its own training-set statistics and zero-pad configurations with fewer degrees of freedom to the shared eight-dimensional layout. Padding provides a common input shape but does not establish correspondence between the joint dimensions of different embodiments.

A shared framewise MLP maps each action vector to width 512:

$$h_t^{\text{act}} = W_2 \ \mathrm{GELU}(W_1 a_t^e) + p_t^{\text{act}}. \tag{11}$$

where $W_1 : \mathbb{R}^8 \to \mathbb{R}^{512}$, $W_2 : \mathbb{R}^{512} \to \mathbb{R}^{512}$, and p_t^{act} is selected from a learned temporal position table supporting up to 128 action frames.

Render–action fusion. We fuse the encoded streams using two pre-normalization cross-attention blocks. The 2025 render tokens serve as queries, while the framewise action tokens serve as keys and values:

$$H_{\text{fused}} = \mathrm{CrossAttn}(H_{\text{rend}}, H_{\text{act}}, H_{\text{act}}) \,. \tag{12}$$

Each block uses eight attention heads, an FFN expansion factor of four, and residual connections around both sublayers. A final projection maps each fused token to the Wan hidden dimension:

$$G = \mathrm{Linear}_{2048 \to 5120}(\mathrm{GELU}(\mathrm{Linear}_{512 \to 2048}(\mathrm{LayerNorm}(H_{\text{fused}})))) \,. \tag{13}$$

The resulting tensor has shape $G \in \mathbb{R}^{2025 \times 5120}$. Neither the output projection nor its preceding layers are zero-initialized.

Backbone injection. Let $g_j \in \mathbb{R}^{5120}$ denote the adapter output for video token j, and let $e(s)$ denote the embedded flow timestep. We form the token-specific timestep embedding

$$\widetilde{e}_j(s) = e(s) + g_j. \tag{14}$$

The backbone's existing $\mathrm{time_proj}$ maps this embedding to six modulation vectors per token: the shift, scale, and gate applied to the attention and FFN sublayers. The same token-specific embedding is used in all 40 transformer blocks and determines the shift and scale of the final output normalization. Thus, rendering-enriched actions enters exclusively through the pretrained timestep-modulation pathway and does not modify the backbone's attention or FFN architecture.

During training, we set the adapter contribution G to zero with probability 0.1, while retaining the initial-frame conditioning. This conditioning dropout regularizes the adapter and provides an action-unconditioned branch for evaluation controls.

A.5 Baseline Re-implementation

Action-only baseline. Our action-only baseline follows the conditioning mechanism of EgoWM (Bagchi et al., 2026). Starting from the same Wan 2.1 image-to-video backbone used by DOPPEL, we remove the rendering encoder and render–action fusion modules entirely. The numerical robot configuration at each timestep is normalized using the same per-embodiment statistics as DOPPEL, projected with an MLP followed by a 1D temporal convolution, and injected through

Table 3: OSCAR configurations used in our experiments. The fine-tuned models are initialized from the released OSCAR checkpoint.

	Released OSCAR	**OSCAR (Fine-tuned)**
Initialization	Cosmos-Predict2.5, 2B	Released OSCAR checkpoint
Conditioning	2D skeleton + text	2D skeleton + empty text prompt
Training data	180657 episodes: 94830 robot episodes (RH20T, InternData-A1, DROID 21904, AgiBot-Beta, AIROA-MoMa) and 85827 human episodes (EgoDex, EPIC-Kitchens)	Our training split (Table 1), using 81-frame windows at the selected start positions, with overlapping windows permitted
Clip length	81 frames, native frame rate	81 frames, native frame rate
Resolution	640×480	432×240
Optimizer	AdamW	AdamW, $\beta = (0.9, 0.99)$, weight decay 10^{-3}
Learning rate	3×10^{-5}	3×10^{-5} with 500-step warmup and cosine decay to 3×10^{-6}
Batch size	16	16
Training steps	15k robot-only steps, followed by robot + human training	10000
Hardware	$1\times$ GH200	$8\times$ A100
Sampling	35 steps, guidance scale 6	35 steps, guidance scale 6

the backbone's timestep-modulation pathway. Thus, the baseline receives the same numerical robot trajectory as DOPPEL but no camera-aligned robot renderings.

All remaining components are kept identical to the corresponding DOPPEL configuration, including backbone initialization, training clips, clip construction, optimization schedule, conditioning dropout, and training budget. All models are evaluated on exactly the same validation and test splits. The complete single-embodiment training recipe is given in Table 4. This baseline therefore provides a controlled comparison between numerical action-only conditioning and DOPPEL conditioning.

OSCAR. For OSCAR (Wu & Gao, 2026), we retain the released architecture and its 2D kinematic-skeleton conditioning rather than modifying its conditioning formulation. We initialize from the released OSCAR checkpoint and fine-tune a separate model for each of DROID, BRIDGE, and ROBOMIND using the corresponding training demonstrations and selected start positions, with OSCAR's 81-frame temporal windows. We do not report zero-shot results on BRIDGE or ROBOMIND, since these datasets are not part of OSCAR's original training corpus.

To align OSCAR more closely with our evaluation setting, we fine-tune it at the same 432×240 resolution used by DOPPEL and use an empty text prompt throughout training and evaluation. We preserve its native skeleton-conditioning mechanism and otherwise follow the fine-tuning configuration summarized in Table 3. All models are evaluated on the same validation and test clips. Because OSCAR differs from DOPPEL in backbone, pretraining data, model size, conditioning representation, and optimization history, we do not attempt to equalize architecture-level compute and treat OSCAR as a system-level baseline rather than a controlled ablation of DOPPEL.

A.6 DATASET VISUALIZATIONS

We show representative RGB observations and corresponding camera-aligned robot renderings from DROID, BRIDGE, and ROBOMIND in Figures 9, 10, and 11, respectively. These examples illustrate the diversity of embodiments, environments, and viewpoints across the three datasets.

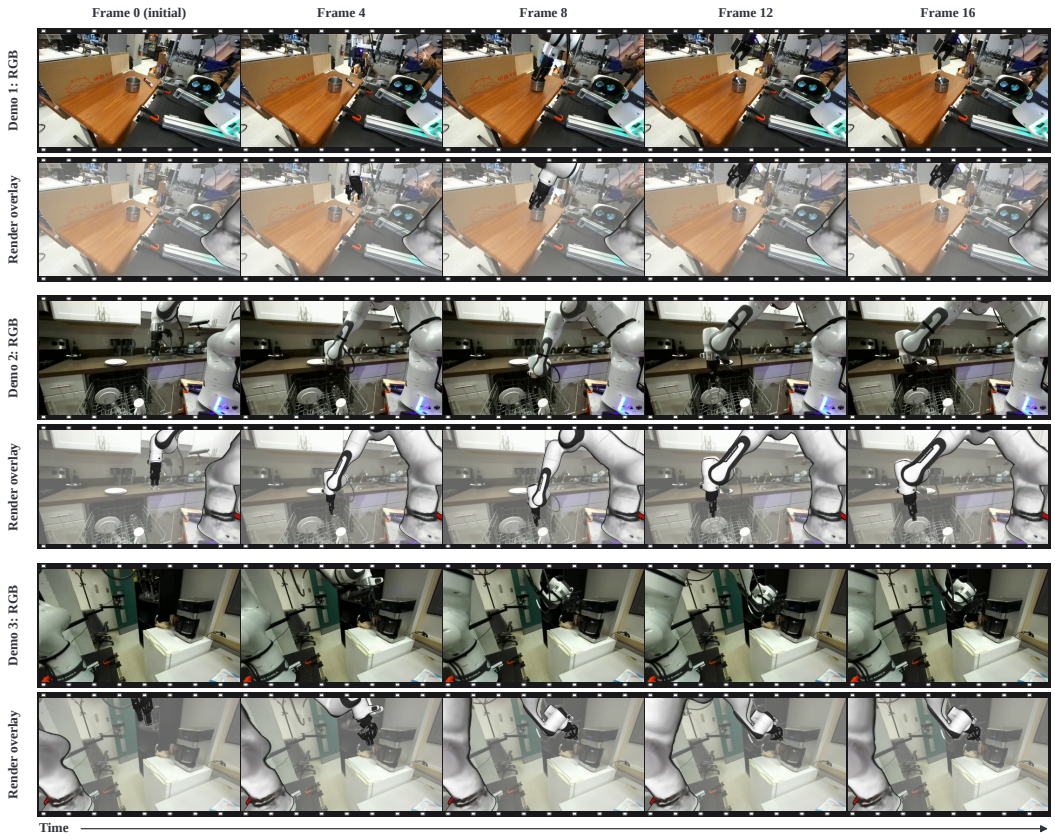

Figure 9: Representative examples from DROID with corresponding camera-aligned robot renderings.

Table 4: Training configuration for the single-embodiment models.

Backbone	Wan 2.1 I2V 14B (16.4B parameters). We train all 40 transformer blocks together with the output and timestep projections, yielding 16.3B trainable backbone parameters.
Adapters	Render–action adapter: 30.5M parameters; action-only adapter: 4.2M parameters.
Data	One dataset per model: see Table 1.
Input	17 frames at 240×432. Contiguous windows use 0.2 s temporal spacing, while full-span windows cover the complete demonstration. The text prompt is always empty.
Optimizer	AdamW with weight decay 10^{-3} and gradient clipping at 1.0. Training uses bf16, fp32 master weights, FSDP, and activation checkpointing.
Learning rates	5×10^{-6} for the backbone and 2×10^{-5} for the conditioning adapter.
Schedule	Cosine decay with 500 warm-up steps and a minimum learning-rate ratio of 0.05.
Batch size	8 examples per GPU on 8 H100-80GB GPUs, giving a global batch size of 64.
Weight EMA	Decay of 0.9975, updated every 10 steps after a 500-step warm-up.
Conditioning dropout	Probability 0.1 per training step.
Objective	Latent flow matching.

A.7 IMPLEMENTATION DETAILS

We provide the training configurations for all our models in Tables 4, 5 and 6. To ensure that the multi-embodiment model observes a comparable number of draws per dataset as the corresponding single-dataset action-only and baseline runs, we use multi-node training with a larger global batch size. Unless otherwise noted, the multi-embodiment models use the same architecture and optimization settings as the single-embodiment models.

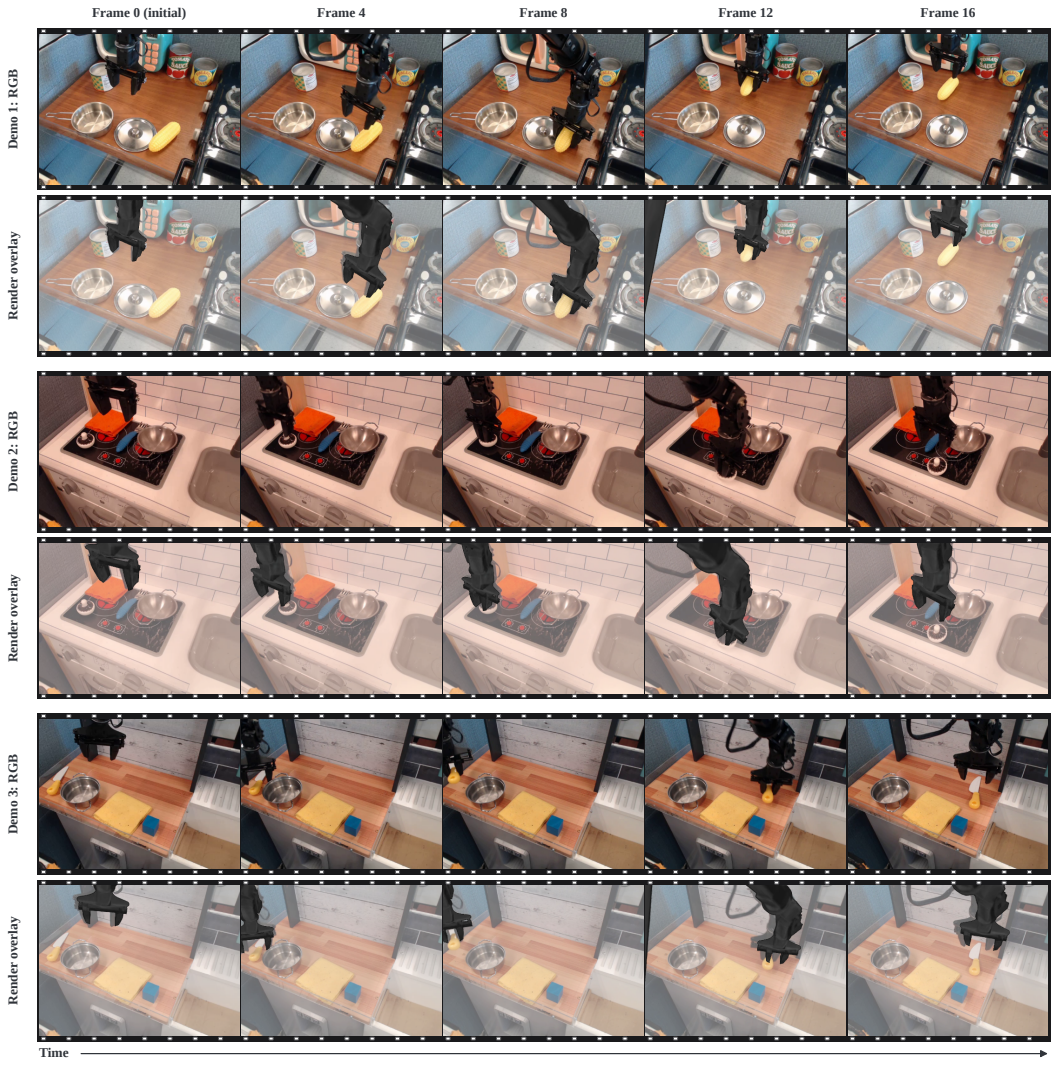

Figure 10: Representative examples from BRIDGE with corresponding camera-aligned robot renderings.

Table 5: Training configuration for the single-node multi-embodiment model. Settings not listed are identical to Table 4.

Data	Joint training on DROID, ROBOMIND, and BRIDGE. Datasets are sampled with equal probability, and clips are sampled uniformly within each dataset.
Hardware	8 H100-80GB GPUs on one node, with one FSDP rank per GPU.
Batch size	8 examples per GPU, giving a global batch size of 64.
Learning rates	Backbone: 5×10^{-6}; adapter: 2×10^{-5}.
Schedule	Cosine decay with 500 warm-up steps and a minimum learning-rate ratio of 0.05.
EMA	Decay of 0.9975, updated every 10 steps after a 500-step warm-up.

A.8 EVALUATION METRICS

We evaluate generated videos against the reference videos at 240×432 resolution. All metrics exclude the conditioning frame and are computed over the sixteen predicted frames. Unless otherwise specified, scores are first averaged over time within each clip and then averaged across clips with equal weight. Undefined scores are omitted.

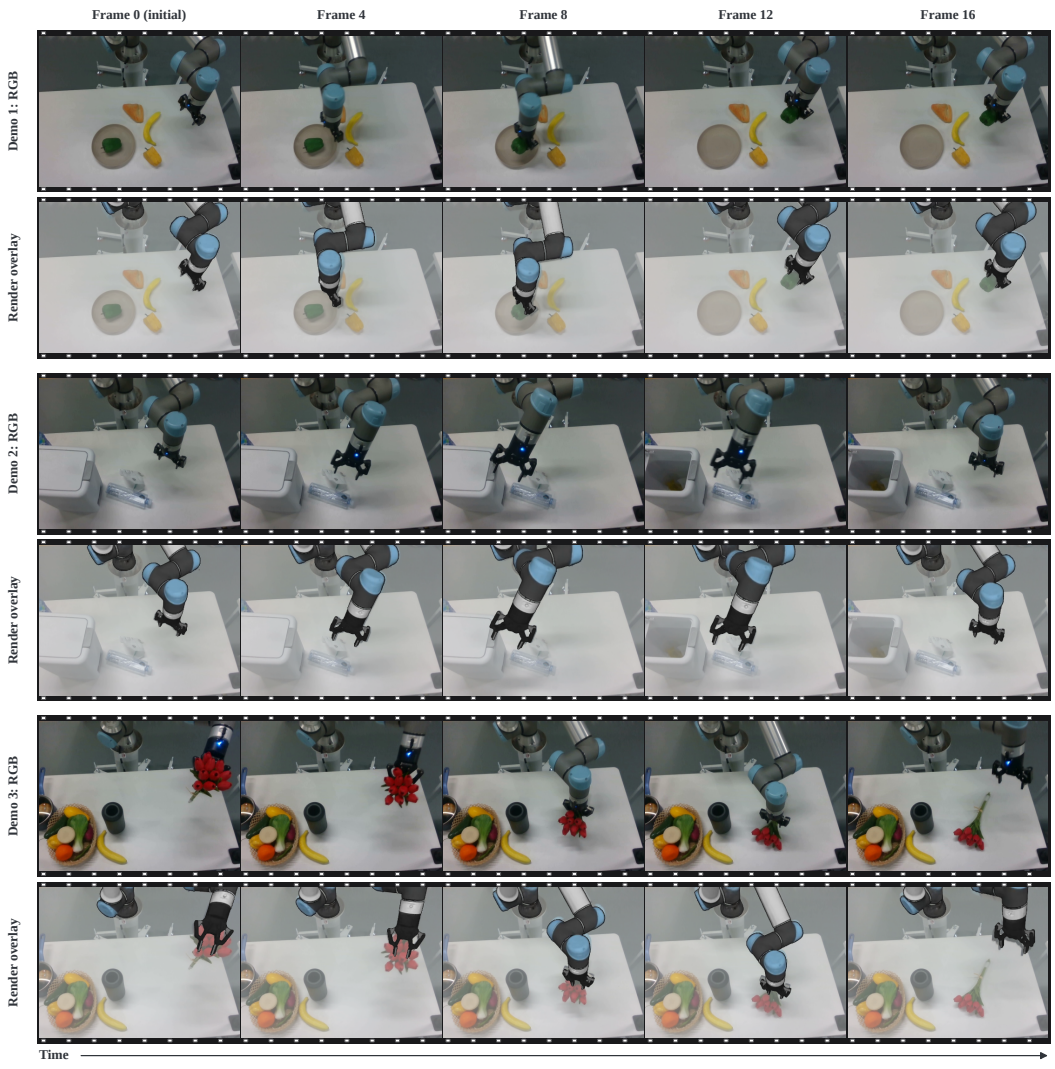

Figure 11: Representative examples from RoboMind with corresponding camera-aligned robot renderings.

Table 6: Training configuration for the four-node multi-embodiment render–action model. Settings not listed are identical to Table 4.

Data	Joint training on DROID, RoboMind, and BRIDGE. Datasets are sampled with equal probability, and clips are sampled uniformly within each dataset.
Hardware	32 H100-80GB GPUs across four nodes, with one FSDP rank per GPU.
Batch size	8 examples per GPU, giving a global batch size of 256.
Learning rates	Backbone: 1×10^{-5}; adapter: 4×10^{-5}.
Schedule	Cosine decay with 250 warm-up steps and a minimum learning-rate ratio of 0.05.
Weight EMA	Decay of 0.995, updated every 10 steps after a 250-step warm-up.

Whole-frame visual fidelity. We report peak signal-to-noise ratio (PSNR), structural similarity (SSIM), and learned perceptual image patch similarity (LPIPS). PSNR measures pixel-level reconstruction fidelity, SSIM measures local agreement in luminance, contrast, and structure, and LPIPS measures perceptual discrepancy using pretrained AlexNet features. Each metric is computed independently for every predicted frame and averaged over the prediction horizon. Higher PSNR and SSIM are better, while lower LPIPS is better.

Region-specific reconstruction. To separate robot reconstruction from changes in the surrounding scene, we compute PSNR over the robot and moving non-robot objects. We initialize the robot mask from the first rendered frame and obtain non-robot object masks using SAM 2 on the first reference frame. These masks are propagated through the reference video to define the evaluation regions.

A non-robot object is classified as moving when its maximum centroid displacement, normalized by the square root of its initial mask area, is at least 0.25. We compute $\mathrm{PSNR}_{\mathrm{arm}}$ over the reference robot mask and $\mathrm{PSNR}_{\mathrm{obj}}$ over moving non-robot object masks. For object metrics, scores are first averaged over time for each object and then averaged equally across objects.

Structural consistency. Structural consistency score (SCS) measures agreement between predicted and reference region trajectories. Starting from identical first-frame masks, SAM 2 independently propagates each region through the generated and reference videos. We compute the intersection-over-union between corresponding masks at each predicted frame and average over time and regions. Higher SCS indicates greater consistency in region position and shape.

In the whole-frame results table, SCS is averaged over all non-robot objects. In the region-specific results, $\mathrm{SCS}_{\mathrm{arm}}$ evaluates the robot and $\mathrm{SCS}_{\mathrm{obj}}$ evaluates moving non-robot objects. The training-budget comparisons use SCS averaged over all tracked non-robot objects, as in the Tab. 2.

A.9 Camera extrinsics dataset

We will provide camera extrinsics and intrinsics for all 52,249 demonstrations in our corpus (DROID 19,837, RoboMind 17,588, BRIDGE 14,824), covering 1,133 distinct camera poses. DROID extrinsics and ZED intrinsics are used as provided by Khazatsky et al. (2024). RoboMind and BRIDGE ship no extrinsics, so we treat each camera as fixed within a recording session and recover one pose per session with the procedure of Sec. 4.1: 49 poses for RoboMind and 56 for BRIDGE. Intrinsics are the RealSense D435i values for RoboMind and MoGe-estimated (Wang et al., 2025a) focal lengths for BRIDGE. We show sample camera-calibration overlays in Fig. 12.

A.10 Results

A.10.1 Additional Qualitative Results

We provide additional qualitative results across all our experiments. Please see A.1 for the complete set of results. We also include an example of trajectory generalization, comparing DOPPEL with the baseline in Fig. 13.

A.10.2 Numerical results for the training curves

We report the numerical values underlying both training-curve comparisons. Each entry averages 200 held-out clips for the corresponding dataset and split. PSNR excludes the conditioning frame and averages over the 16 predicted frames before averaging across clips; SCS averages all tracked non-robot objects. All values are reported to three decimal places. Training draws include repeated examples. The two comparisons use different budget conventions, explicitly indicated in the tables. The uncertainty bars in the figures use 5,000 source-episode bootstrap resamples; the tables below report the corresponding point estimates.

Single-dataset experts versus shared DOPPEL. Tables 8 and 9 give the four-checkpoint curves for the separately trained action-only experts and the larger-batch shared DOPPEL model. The training budgets are measured per dataset. The expert rows combine the corresponding single-dataset model for each dataset.

Shared action-only versus shared DOPPEL. Tables 10 and 11 give the three-checkpoint curves for the matched batch-64 shared models. Here, training budgets count total draws across all three datasets. The two models use the same training exposure and optimization schedule at each checkpoint. This shared DOPPEL run differs from the larger-batch run in the expert comparison above.

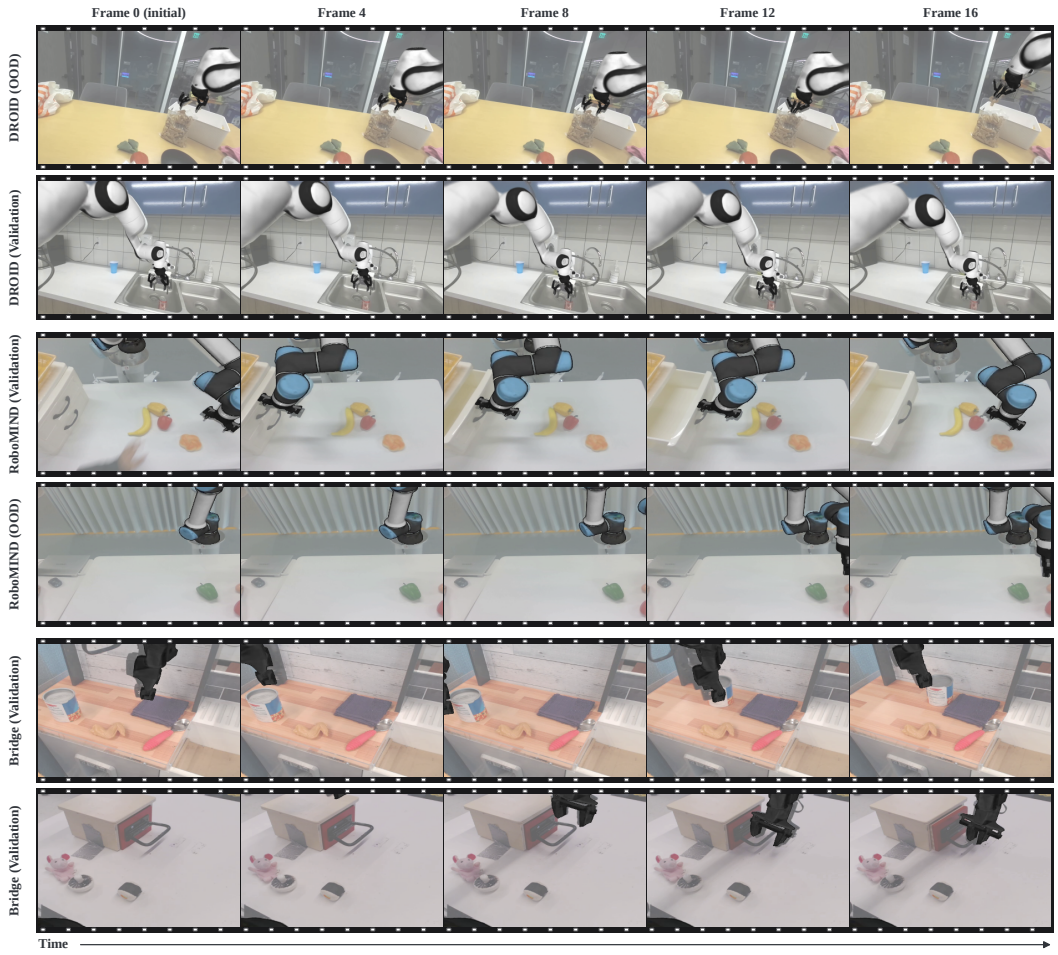

Figure 12: Overlays of camera calibration results.

Table 7: Region-specific prediction metrics. Each single-dataset expert sees approximately 3.36M training draws, while multi-embodiment DOPPEL matches the per-dataset training exposure. OSCAR is fine-tuned over its pretrained checkpoint.

		Validation: unseen trajectories				OOD test			
Dataset	Method	$\mathrm{PSNR}_{\mathrm{arm}}$	$\mathrm{PSNR}_{\mathrm{obj}}$	$\mathrm{SCS}_{\mathrm{arm}}$	$\mathrm{SCS}_{\mathrm{obj}}$	$\mathrm{PSNR}_{\mathrm{arm}}$	$\mathrm{PSNR}_{\mathrm{obj}}$	$\mathrm{SCS}_{\mathrm{arm}}$	$\mathrm{SCS}_{\mathrm{obj}}$
DROID	Action-only	17.929	26.568	0.786	0.823	17.112	24.455	0.704	0.735
	OSCAR (Finetuned)	16.537	22.915	0.778	0.801	16.053	22.080	0.712	0.728
	DOPPEL (single)	20.590	28.332	0.835	0.864	**19.252**	**26.386**	0.777	**0.799**
	DOPPEL (multi)	**20.794**	**28.540**	**0.844**	**0.868**	19.150	26.222	**0.780**	0.792
BRIDGE	Action-only	20.603	27.317	0.773	0.887	17.709	21.497	0.591	0.776
	OSCAR (Finetuned)	18.255	22.784	0.702	0.840	17.504	20.615	0.591	0.788
	DOPPEL (multi)	**21.428**	**27.715**	**0.786**	**0.898**	**19.161**	**24.963**	**0.692**	**0.842**
ROBOMIND	Action-only	24.427	28.794	0.899	0.857	24.254	28.417	0.887	0.816
	OSCAR (Finetuned)	22.960	24.956	0.882	0.809	22.890	25.336	0.876	0.798
	DOPPEL (multi)	**26.667**	**29.289**	**0.917**	**0.864**	**25.214**	**28.932**	**0.903**	**0.844**

Best in **bold**, second-best underlined; ties at displayed precision share emphasis. Higher is better for all metrics.

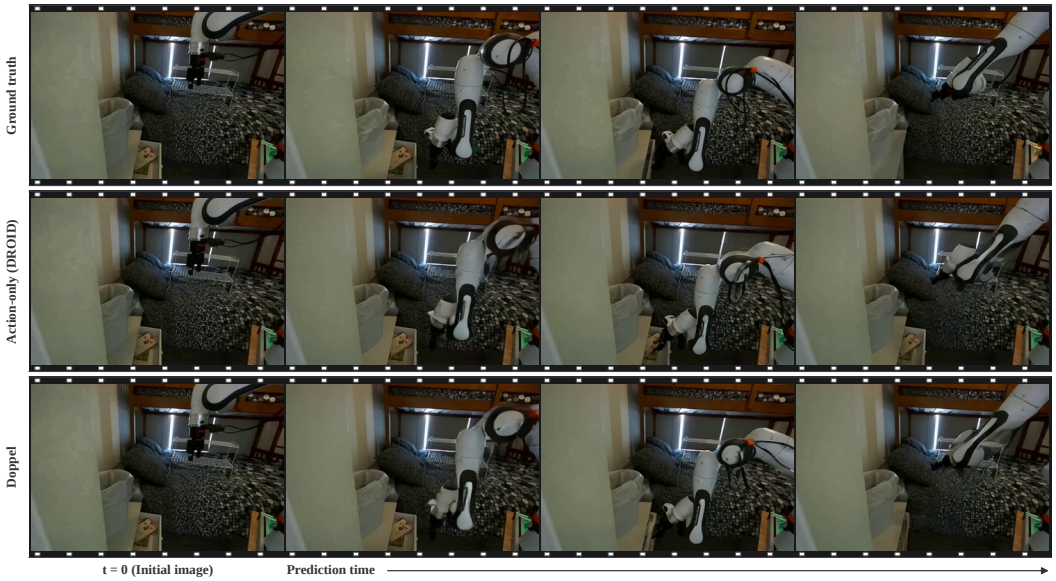

Figure 13: Additional trajectory-generalization example on DROID. We compare DOPPEL with the action-only baseline on a held-out trajectory from a familiar environment.

Table 8: PSNR at every checkpoint in the expert-versus-shared-model scaling comparison. Action-only experts are trained separately on each dataset with global batch 64; DOPPEL is trained jointly with global batch 256. Draws are in millions *per dataset*; shared-model counts are expectations under the dataset sampler. Training budgets are rounded to approximate milestones; the curves use the actual training-draw counts. PSNR is reported in dB.

			DROID		BRIDGE		ROBOMIND	
Method	Checkpoint	Draws/dataset (M)	Val	OOD	Val	OOD	Val	OOD
Action-only expert	99	~ 0.5	19.194	18.058	22.136	21.119	25.199	25.848
DOPPEL (multi)	79	~ 0.5	20.815	19.505	24.253	22.314	28.560	28.634
Action-only expert	199	~ 1.0	19.583	18.110	23.374	21.279	26.449	27.016
DOPPEL (multi)	149	~ 1.0	22.768	20.545	24.935	22.907	29.386	29.589
Action-only expert	409	~ 2.0	20.963	18.982	24.805	20.552	27.501	28.039
DOPPEL (multi)	309	~ 2.0	23.664	20.965	25.809	22.472	30.387	30.172
Action-only expert	689	~ 3.4	22.228	19.767	25.869	20.555	29.629†	29.629†
DOPPEL (multi)	519	~ 3.4	24.220	21.147	26.458	22.438	30.964	30.436

†Val. and OOD PSNR differ only beyond 3 decimals.

Table 9: SCS at every checkpoint in the expert-versus-shared-model scaling comparison. Action-only experts are trained separately on each dataset with global batch 64; DOPPEL is trained jointly with global batch 256 to match the per-dataset training exposure of the single-dataset experts. Draws are in millions *per dataset*. Training budgets are rounded to approximate milestones; the curves use the actual training-draw counts. SCS averages all tracked non-robot objects and is reported on the 0–1 scale.

			DROID		BRIDGE		ROBOMIND	
Method	Checkpoint	Draws/dataset (M)	Val	OOD	Val	OOD	Val	OOD
Action-only expert	99	~ 0.5	0.868	0.826	0.901	0.875	0.897	0.899
DOPPEL (multi)	79	~ 0.5	0.914	0.876	0.927	0.917	0.927	0.926
Action-only expert	199	~ 1.0	0.877	0.826	0.916	0.883	0.911	0.909
DOPPEL (multi)	149	~ 1.0	0.926	0.888	0.936	0.915	0.933	0.931
Action-only expert	409	~ 2.0	0.905	0.852	0.933	0.868	0.919	0.919
DOPPEL (multi)	309	~ 2.0	0.933	0.896	0.942	0.907	0.937	0.932
Action-only expert	689	~ 3.4	0.921	0.869	0.943	0.863	0.937	0.929
DOPPEL (multi)	519	~ 3.4	0.940	0.897	0.948	0.904	0.940	0.937

Table 10: PSNR at every checkpoint in the matched shared-model scaling comparison. Both models are trained jointly on all three datasets with global batch 64 and the same optimization schedule. Draws are in millions *total across all three datasets*, matching the horizontal axis of the shared-model figure. PSNR is reported in dB.

			DROID		BRIDGE		ROBOMIND	
Method	Checkpoint	Total draws (M)	Val	OOD	Val	OOD	Val	OOD
Action-only (multi)	249	~ 1.2	19.126	18.029	21.703	20.762	24.520	24.958
DOPPEL (multi)	249	~ 1.2	22.070	20.251	24.098	22.781	28.504	29.082
Action-only (multi)	489	~ 2.4	20.381	18.762	23.196	21.722	26.575	26.965
DOPPEL (multi)	489	~ 2.4	22.832	20.697	24.968	23.156	29.382	29.670
Action-only (multi)	689	~ 3.4	20.959	19.036	24.070	21.873	27.564	28.200
DOPPEL (multi)	689	~ 3.4	23.188	20.862	25.431	22.726	29.937	29.818

Table 11: SCS at every checkpoint in the matched shared-model scaling comparison. Both models are trained jointly on all three datasets with global batch 64 and the same optimization schedule. Draws are in millions *total across all three datasets*, matching the horizontal axis of the shared-model figure. SCS averages all tracked non-robot objects and is reported on the 0–1 scale.

			DROID		BRIDGE		ROBOMIND	
Method	Checkpoint	Total draws (M)	Val	OOD	Val	OOD	Val	OOD
Action-only (multi)	249	~ 1.2	0.864	0.824	0.892	0.859	0.886	0.886
DOPPEL (multi)	249	~ 1.2	0.919	0.883	0.927	0.914	0.927	0.932
Action-only (multi)	489	~ 2.4	0.893	0.839	0.914	0.886	0.910	0.910
DOPPEL (multi)	489	~ 2.4	0.927	0.892	0.936	0.919	0.930	0.934
Action-only (multi)	689	~ 3.4	0.901	0.846	0.925	0.890	0.920	0.916
DOPPEL (multi)	689	~ 3.4	0.930	0.896	0.940	0.912	0.933	0.930