Render the Robot, Generate the World: One Video World Model Across Embodiments

Yash Jangir[1] Karan Mirakhor[2] Renos Zabounidis[2] Muhammad Zubair Irshad[3]
Sergey Zakharov[3] Yonatan Bisk[2] Homanga Bharadhwaj[1]
[1]Johns Hopkins University [2]Carnegie Mellon University [3]Toyota Research Institute







Abstract.
Predicting the consequences of an agent's actions requires modeling both the embodiment's motion and the environment's response. Most video world models attempt to learn both jointly using only numerical action vectors, even though robot geometry and kinematics already provide 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, Bridge, ABC, T-Rex, GigaHands, OakInk2, and Ego-Exo4D, spanning Franka, UR5e, WidowX, and bimanual YAM arms, an egocentric humanoid with dexterous hands, and human hands, upper bodies, and full bodies, from about 600 hours of video and without an embodiment label. On the three robot-arm datasets, Doppel improves PSNR over action-only baselines by 1.00 dB on unseen trajectories and 1.20 dB under OOD evaluation while predicting more consistent object motion, and training on all eight embodiments (Doppel-8) improves robot predictions over the three-robot model (Doppel-3) on four of six robot evaluation sets. Project page: https://doppelwm.github.io/.

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 [1]. 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

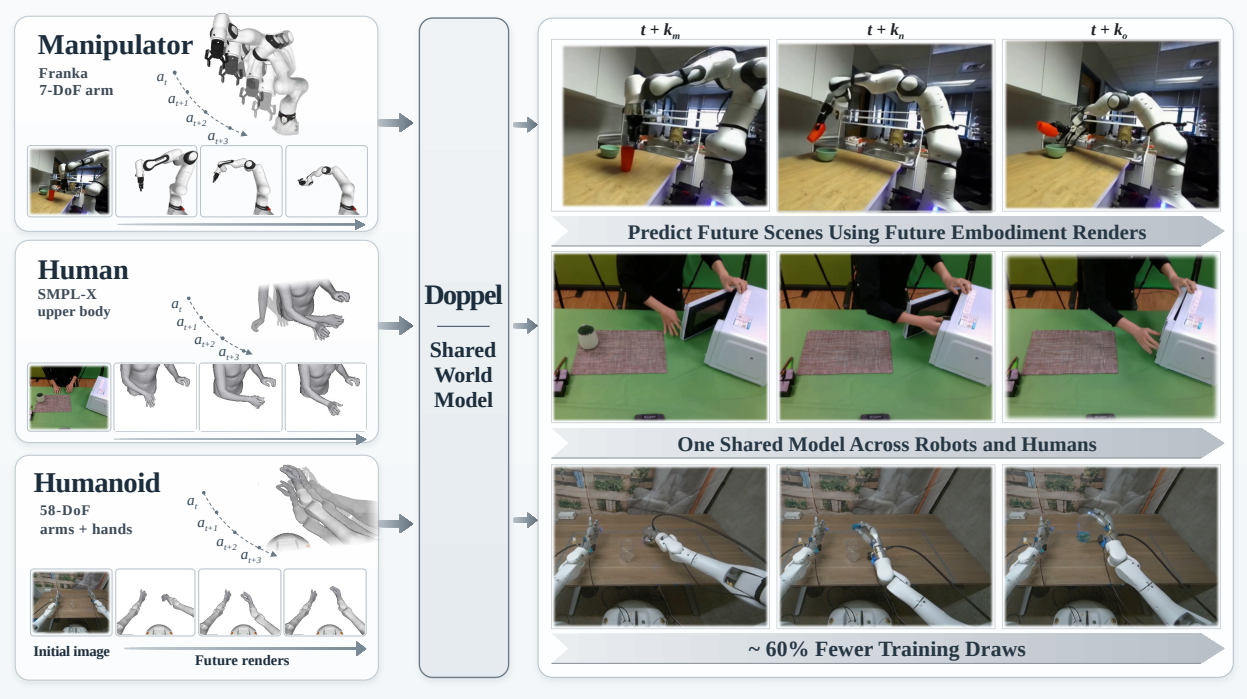


Figure 1: DOPPEL is a surprisingly simple approach for incorporating future embodiment renderings and improving robot-object interactions in action-conditioned video world models.

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) [2]. 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), a model we call Doppel-3, and evaluate generalization to unseen trajectories and environments. Because the conditioning only needs a rendering of the body, the same model extends beyond robot arms: we also train it on eight embodiments (Doppel-8), adding a bimanual robot (ABC), a humanoid with dexterous hands (T-Rex), and human hands and bodies (GigaHands, OakInk2, Ego-Exo4D) rendered from fitted body and hand meshes. 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 multi-embodiment learning.** We train one set of weights across three robot arms, outperforming action-only experts, and achieve higher average PSNR and SCS than a shared action-only baseline with 63.8% fewer training draws. Scaling the same model to eight embodiments (Doppel-8), including a humanoid and human hands and bodies, with no embodiment label, further improves robot predictions on Droid and RoboMind (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 45k demonstrations across WidowX, UR5e, and bimanual YAM, along with calibration code for 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 [3–5]. In robotics, action-conditioned video prediction has benefits in model-predictive control [6, 7] and policy improvement [8, 9]. Recent large-scale models extend this idea to interactive simulators learned from video [10–12]. Since pretrained video generators [13–15] provide strong visual priors, they can be adapted into action-conditioned world models [16], used to synthesize demonstrations from which actions are

inferred for policy learning [17, 18], or used as learned simulators for policy evaluation [19, 20].

Most robot world models condition on numerical actions per frame [21], with multi-view memory [20], or through timestep modulation [2]; others jointly model video and actions [22]. 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 [23], 3D robot point flow [24], occupancy-derived visual priors [25], depth- and normal-aware 4D representations [26], embodiment masks injected through a ControlNet branch [27], projected end-effector pose maps with delta-action embeddings [28], robot pointmaps with joint RGB and pointmap generation [29], egocentric hand-mesh and static-scene renderings [30], and rendered robot motion for demonstration synthesis [31, 32]. Unlike these, we render the full articulated body along the trajectory and pair it with numerical actions; ORV [25] and EnerVerse-AC [28] also keep numerical actions, but alongside occupancy renderings or end-effector maps rather than the body itself. Render tokens attend to action tokens, and the result modulates every transformer block per token, in one model shared across robots, humans, and a humanoid without an embodiment label. We can use joint-space trajectories from a policy's action chunks or an off-the-shelf planner.

Multi-embodiment learning. Pooling data across robots lets policies share experience across embodiments [33–36], typically through embodiment-specific tokenizers or action heads [37, 38]. Other work closes the embodiment gap in image space by swapping one embodiment for another, at test time or in the training data [39–41], or by replacing both with a shared segmentation mask [42]. Motion can also be expressed visually, as trajectory sketches [43], object flow [44], point trajectories [45, 46], projected skeletons [23, 47], or shared particle representations for dynamics [48].

We bring both joint space and image-space embodiment representations together in video world models. Differentiable rendering [49] 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 trajectories $a^e_{0:T-1}$, 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:

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

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 [15]. We use a frozen causal VAE to encode a clip of T RGB frames into a latent tensor $\mathbf{z} \in \mathbb{R}^{C\times T_z\times H_z\times W_z}$,

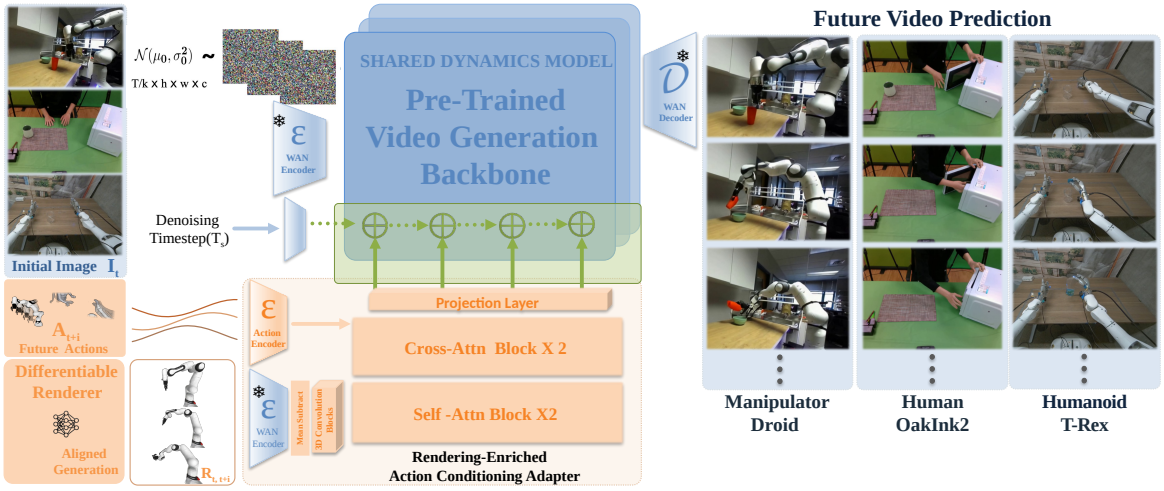


Figure 2: DOPPEL Architecture. We encode camera-aligned robot renderings with a frozen VAE and fuse their features with numerical actions. We use the resulting representation to modulate the video backbone, which predicts future frames from the initial observation.

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 $\boldsymbol{\epsilon} - \mathbf{z}$ [50].

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 $j > 0$:

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

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 [49] to render the robot at a given configuration. For each embodiment e, we use a set of 3D Gaussians $\mathcal{G}_e^0$ to represent the robot in a canonical pose. Given a configuration a_t^e, 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:

$$\mathcal{G}_e(a_t^e) = X_e\Big(D_e(\mathcal{G}_e^0, a_t^e), a_t^e\Big). \tag{4}$$

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

$$r_t^e = \pi_{K_i,C_i}(\mathcal{G}_e(a_t^e)), \tag{5}$$

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 $a_{0:T-1}^e$ to obtain a rendering sequence $r_{0:T-1}^e$ 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.

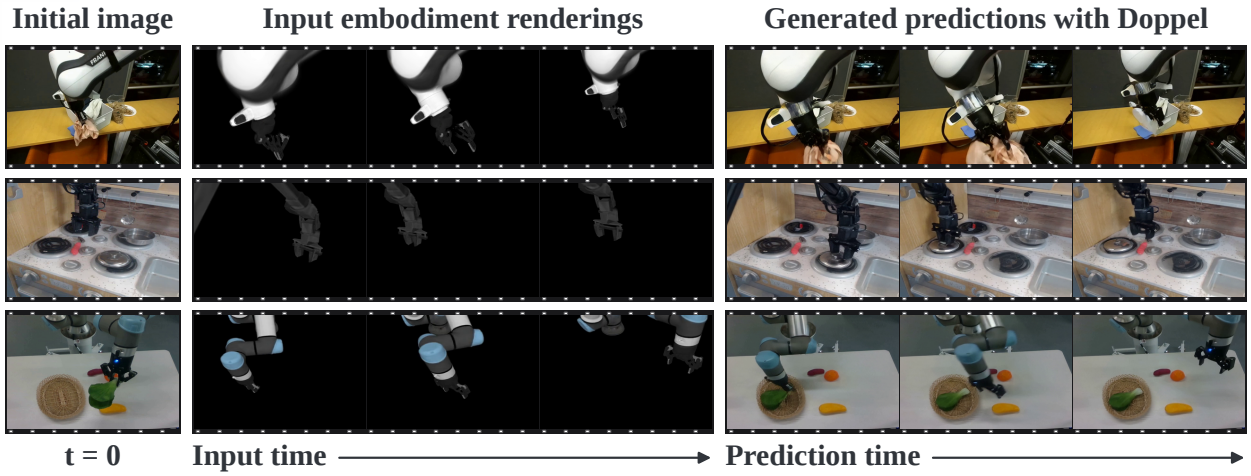


Figure 3: We show predictions with Doppel on Franka, WidowX, and UR5e arms in out-of-domain scenes with unseen objects (top to bottom). Each row shows the initial image, future robot renderings used as conditioning, and our predicted frames. We use the same model across all three embodiments. Our model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

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. [51]. 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 [52]. 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 [53] 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. 20. See App. A.9.

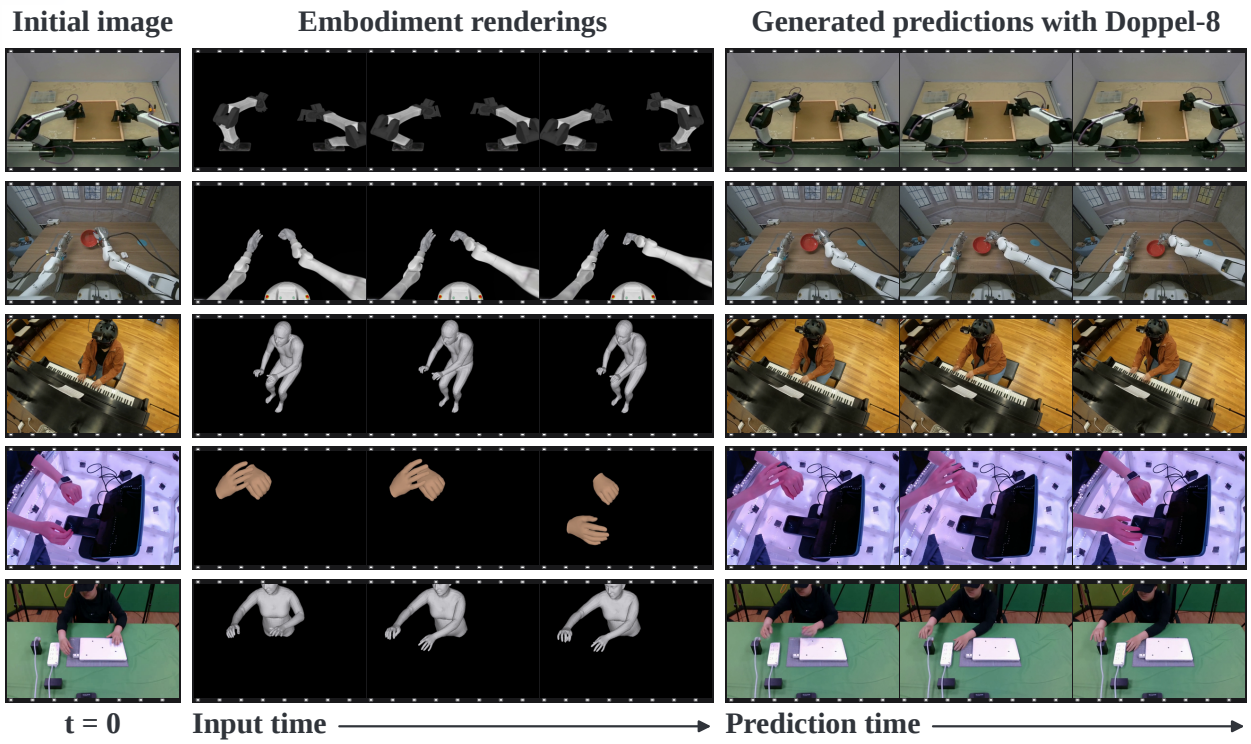


Figure 4: We show predictions with DOPPEL-8 on held-out test clips from the five added datasets: a bimanual YAM robot (ABC), a humanoid with dexterous hands (T-Rex), a human body (Ego-Exo4D), human hands (GigaHands), and a human upper body (OakInk2), from top to bottom. Each row shows the initial image, future embodiment renderings used as conditioning, and our predicted frames. We use the same model across all five embodiments, with no embodiment label.

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

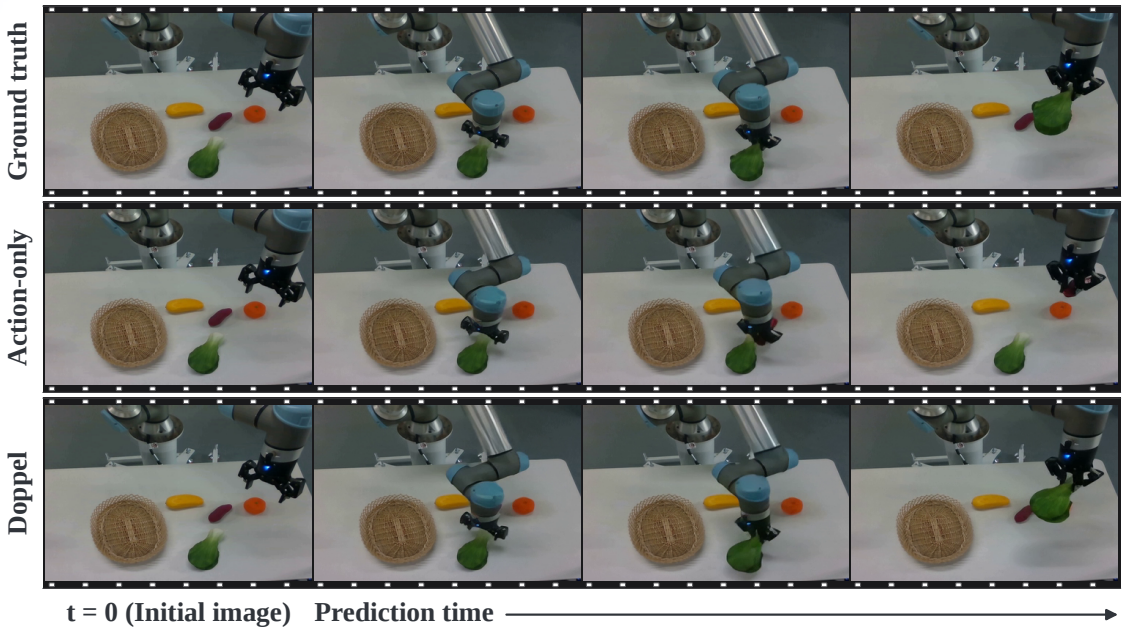


Figure 5: Qualitative comparison across trajectory and OOD generalization settings. Compared to the baseline we observe that DOPPEL generates much plausible robot-object interactions as intended by the provided action trajectory. Additional qualitative results are available on our project website (Appendix A.1) and a trajectory generalization illustration in Fig. 21.

configuration. We denote the complete render–action adapter by g_ϕ:

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

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. [2] 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 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

Table 1: Training corpus. Validation clips come from held-out trajectories in seen settings; test clips hold out the listed shift.

Dataset	Embodiment	Train episodes	Train clips	Val. clips	Test clips	Test distribution shift
DROID	Franka Panda	16726	83511	200	200	Unseen lab
BRIDGE	WidowX	12387	61935	200	200	Unseen kitchen
ROBOMIND	UR5e + 2F-85	14801	74005	200	200	Unseen objects
ABC	Bimanual YAM	9115	274523	200	200	Unseen tasks
T-Rex	Vega-1 + Sharpa hands	4784	58120	200	200	Unseen backgrounds
Ego-Exo4D	Human body (SMPL-H)	2041	213292	200	200	Unseen tasks
GigaHands	Human hands (MANO)	8491	31218	200	200	Unseen activities
OakInk2	Human upper body (SMPL-X)	403	6998	200	200	Unseen tasks
Three robot datasets	—	43914	219451	600	600	—
All eight datasets	—	68748	803602	1600	1600	—

Table 2: Prediction quality on the three robot datasets. *Input* is what the model is conditioned on besides the first frame; *Trained on* is its training data. All our models are evaluated at their final checkpoint and saw a similar amount of each robot dataset during training.

				Validation: unseen trajectories				OOD test			
Dataset	Method	Input	Trained on	PSNR ↑	SSIM ↑	LPIPS ↓	SCS ↑	PSNR ↑	SSIM ↑	LPIPS ↓	SCS ↑
DROID	Action-only	Actions	This dataset	22.888	0.841	0.078	0.927	20.069	0.779	0.131	0.873
	OSCAR (fine-tuned)	Skeleton	This dataset	20.427	0.751	0.135	0.903	18.619	0.705	0.174	0.854
	DOPPEL	Render + actions	This dataset	24.357	0.864	0.065	0.940	21.276	0.802	0.113	0.902
	DOPPEL-3	Render + actions	3 robot datasets	24.297	0.862	0.066	0.940	21.250	0.803	0.111	0.899
	DOPPEL-8	Render + actions	All 8 datasets	**24.819**	**0.870**	**0.062**	**0.943**	**21.460**	**0.806**	**0.108**	**0.903**
BRIDGE	Action-only	Actions	This dataset	26.290	0.871	0.050	0.946	20.584	0.755	0.131	0.866
	OSCAR (fine-tuned)	Skeleton	This dataset	22.559	0.771	0.117	0.914	19.487	0.633	0.211	0.854
	DOPPEL	Render + actions	This dataset	**26.618**	**0.875**	**0.048**	**0.949**	21.722	0.771	0.111	0.901
	DOPPEL-3	Render + actions	3 robot datasets	26.551	0.873	0.049	**0.949**	**22.474**	**0.786**	**0.097**	**0.905**
	DOPPEL-8	Render + actions	All 8 datasets	26.544	0.873	0.049	0.948	22.238	0.784	0.099	0.899
ROBOMIND	Action-only	Actions	This dataset	30.474	0.923	0.038	0.940	30.098	0.918	0.039	0.932
	OSCAR (fine-tuned)	Skeleton	This dataset	27.456	0.882	0.079	0.914	27.661	0.878	0.076	0.920
	DOPPEL	Render + actions	This dataset	31.104	0.928	0.037	**0.942**	30.290	0.919	0.039	0.928
	DOPPEL-3	Render + actions	3 robot datasets	31.065	0.928	0.037	0.941	30.442	0.920	**0.038**	0.935
	DOPPEL-8	Render + actions	All 8 datasets	**31.302**	**0.930**	**0.036**	**0.942**	**30.659**	**0.921**	**0.038**	**0.936**

Best in **bold**, second-best underlined; ties share emphasis. 200 clips per split; every method is scored on the same clips.

Datasets and splits. We train jointly on DROID (Franka), BRIDGE (WidowX), and ROBOMIND (UR5e) [54–56]. 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). We call this model DOPPEL-3. For DOPPEL-8, we add ABC [57] (bimanual YAM), T-Rex [58] (Vega-1 humanoid with Sharpa hands), GigaHands [59] (human hands), OakInk2 [60] (human upper body), and Ego-Exo4D [61] (human body). The two new robots are rendered from their kinematic models; people are rendered from fitted MANO [62], SMPL-X [63], and SMPL-H [62] meshes. Their test splits hold out tasks (ABC, OakInk2, Ego-Exo4D), activities (GigaHands), or wall backgrounds (T-Rex). All actions are zero-padded to a common 159-dimensional vector and enter one shared action encoder without an embodiment label, and we sample each dataset in proportion to its training clips. 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.

Table 3: Doppel-8, evaluated on the five datasets added beyond the three robot-arm datasets: a bimanual robot (ABC), a humanoid with dexterous hands (T-Rex), human bodies (Ego-Exo4D), human hands (GigaHands), and human upper bodies (OakInk2). These are the same weights as the last row of each block in Table 2. Validation holds out trajectories; the test split holds out the shift listed in Table 1.

	Validation: unseen trajectories				Test: held-out shift			
Dataset	PSNR ↑	SSIM ↑	LPIPS ↓	SCS ↑	PSNR ↑	SSIM ↑	LPIPS ↓	SCS ↑
ABC	24.653	0.850	0.096	0.920	25.217	0.861	0.089	0.927
T-Rex	28.327	0.871	0.050	0.961	28.346	0.867	0.051	0.956
Ego-Exo4D	27.935	0.895	0.045	0.966	27.611	0.892	0.044	0.965
GigaHands	25.622	0.885	0.053	0.941	25.784	0.897	0.049	0.944
OakInk2	29.014	0.918	0.036	0.970	29.090	0.921	0.033	0.970

Baselines. We compare Doppel against both an action-only [2] and geometric-conditioned baselines [23]. The action-only baseline follows EgoWM-style [2] 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 [23] 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 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 [64], and LPIPS [65], along with object SCS [2]. We compute SCS as the overlap of SAM2 masks propagated from the same initial frame through predicted and reference videos [66]. 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-dataset training and 32 GPUs for the multi-dataset runs at a global batch size of 256: Doppel-3 trains for 49.5k steps and Doppel-8 for 189k steps. We give complete training recipes in Appendix Tables 6–9.

6 Results and Analysis

Prediction across embodiments. We improve whole-frame prediction quality over action-only conditioning and fine-tuned OSCAR [23] on all three robot datasets, across validation and OOD splits (Table 2). With the same training data, adding the robot rendering to the action input improves PSNR over action-only conditioning by 0.2–1.5 dB. Our shared models outperform separately trained action-only experts, and Doppel-8 has the best PSNR, SSIM, and LPIPS on Droid and RoboMind in both splits. Compared with Doppel-3 on the same clips, adding the five other datasets raises PSNR by 0.52 and 0.21 dB on Droid validation and OOD and by 0.24 and 0.22 dB on RoboMind, leaves Bridge validation unchanged, and lowers Bridge OOD by 0.24 dB (Appendix Table 10). With every model at ∼51 passes over each robot dataset, Doppel-8 has the highest PSNR on five of the six robot evaluation sets and trails Doppel-3 by 0.05 dB on Bridge OOD (Appendix

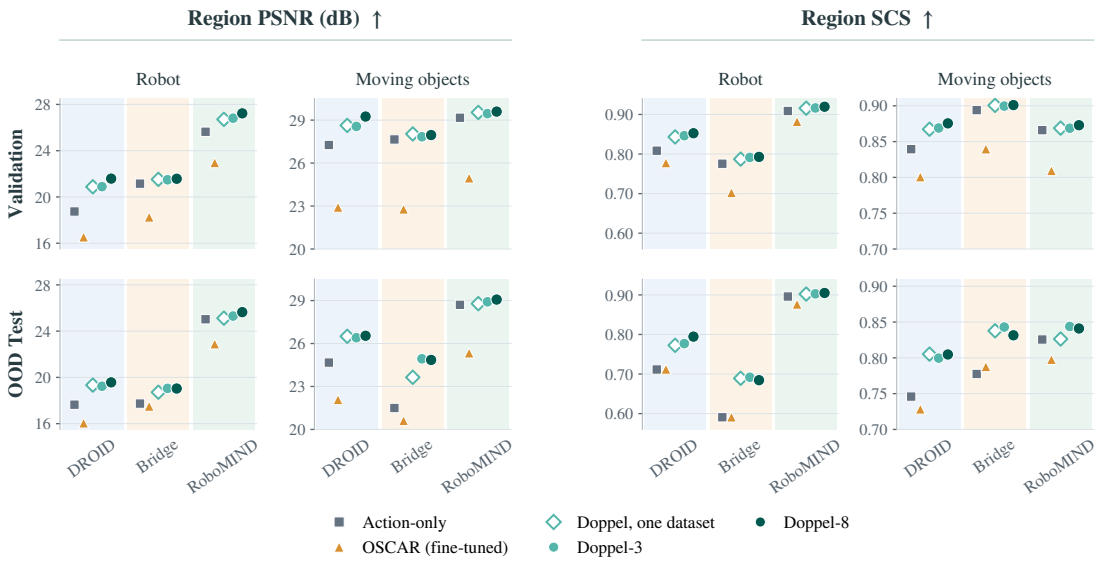


Figure 6: Prediction quality by image region (robot and moving objects) on the three robot datasets, for every model at its final checkpoint, on the same clips as Table 2. Values are in Appendix Table 13.

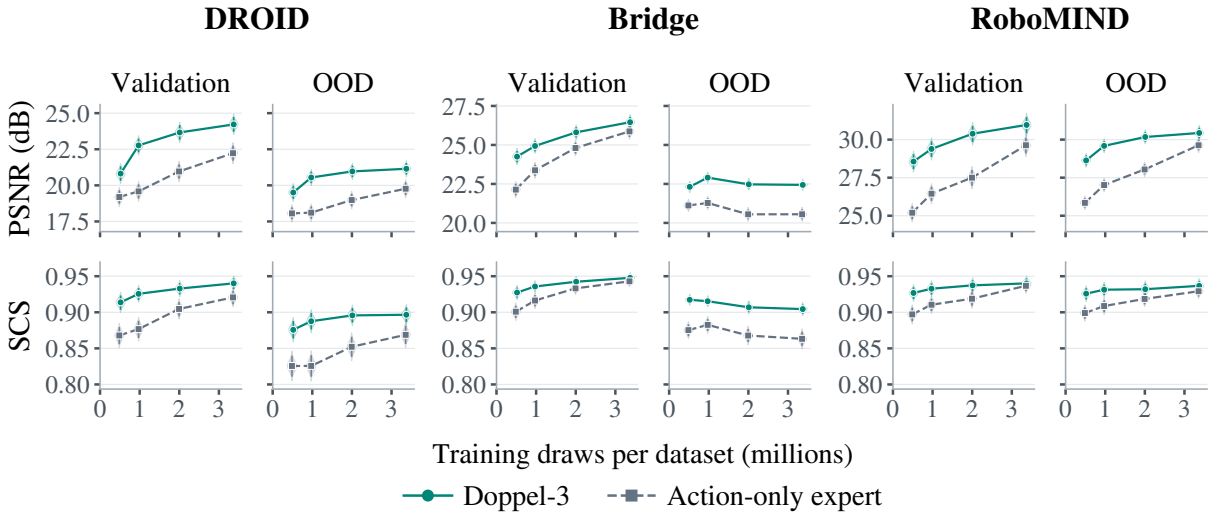


Figure 7: Training efficiency of DOPPEL versus single-dataset action-only experts. DOPPEL achieves higher validation and OOD prediction quality at comparable training draws.

Table 12). On BRIDGE OOD, both multi-dataset models peak before the end of training and then decline while BRIDGE validation keeps improving, which suggests that late training specializes to the seen kitchens. Table 3 reports DOPPEL-8 on the five added datasets, and Fig. 4 shows held-out test examples. We show predictions of DOPPEL-3 in held-out environments and with unseen objects in Fig. 3.

What joint multi-embodiment training contributes. The five added datasets contain no Franka, UR5e, or WidowX data, yet they improve robot prediction. The model is never told which body it sees: the rendering identifies the embodiment, and the shared action encoder receives zero-padded vectors whose slots mean different joints for different bodies. Overall, DOPPEL-8 improves PSNR over DOPPEL-3 on four of the six robot evaluation sets (Table 2; Appendix Table 10), so training

on other embodiments helps robot prediction. The exception is Bridge OOD, where Doppel-8 is 0.24 dB below Doppel-3. We suspect that the models begin to overfit to Bridge, the smallest of the three robot datasets, and specialize to its seen kitchens: Bridge OOD PSNR peaks early and then drops while Bridge validation PSNR keeps rising. In our evaluation, Doppel-3 peaks on Bridge OOD at 22.94 dB after ∼10 passes over Bridge and ends at 22.47 dB, while its validation PSNR rises from 24.40 to 26.55 dB; Doppel-8 peaks at 22.57 dB after ∼30 passes and ends at 22.24 dB, while its validation PSNR rises from 26.14 to 26.54 dB.

Predicting object motion. We also improve predictions of the robot and moving objects, measured separately in Fig. 6. On the Bridge OOD split, Doppel-8 raises moving-object PSNR from 21.50 to 24.85 dB and SCS from 0.778 to 0.831 over the action-only baseline. We show more consistent object motion on unseen objects in Fig. 5. 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. 21.

Training efficiency. We outperform separate action-only experts at comparable numbers of training draws per dataset (Fig. 7). We also compare with a shared action-only model, matching the batch size and optimization schedule and counting total draws across datasets (Fig. 9). 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.

Long-horizon rollouts. We chain 3.2 s predictions to roll out held-out test episodes for 25.6 s from a single real frame: each window starts from the model's own last predicted frame and receives that window's rendering and actions, so after the first frame the model never observes the real video. Fig. 8 shows two long horizon rollouts.

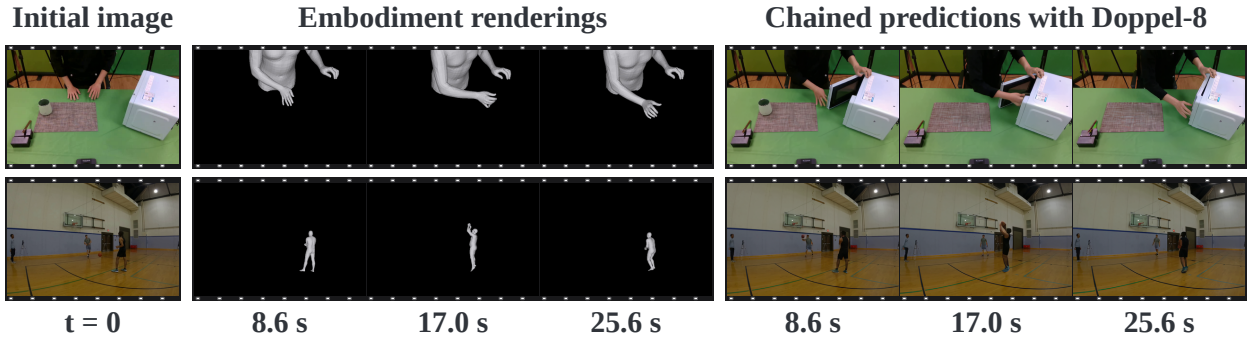


Figure 8: Long-horizon rollouts of Doppel-8 on held-out test episodes from OakInk2 (top) and Ego-Exo4D (bottom). Each episode is predicted for 25.6 s from its first frame alone by chaining 3.2 s windows on the model's own predictions. Each row shows the initial image, the embodiment renderings given as conditioning, and our predictions at 8.6, 17.0, and 25.6 s.

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,

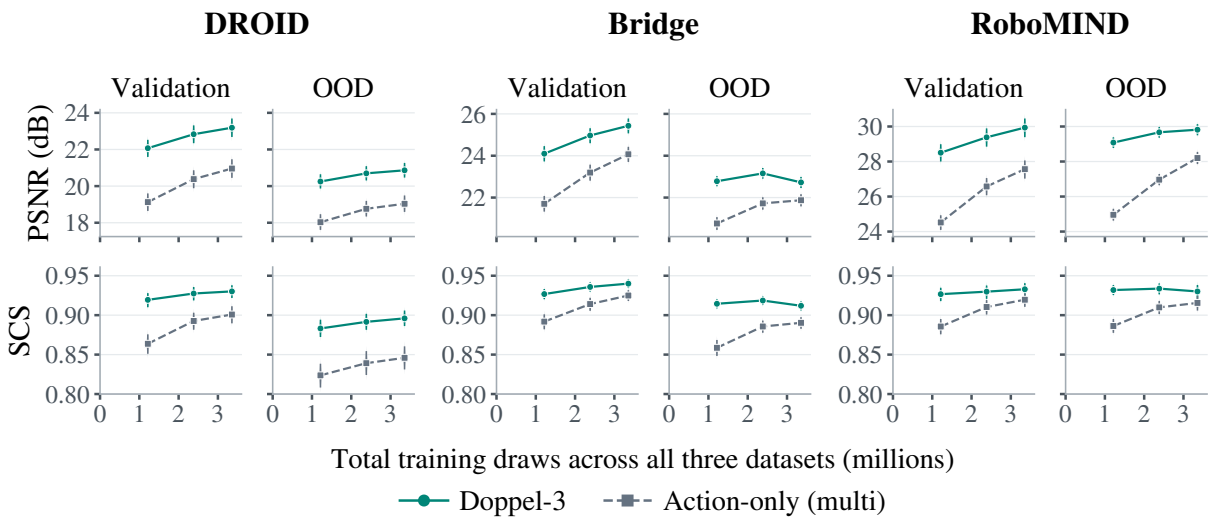


Figure 9: Training efficiency of DOPPEL-3 versus a three-robot action-only baseline. DOPPEL achieves higher validation and OOD prediction quality throughout 3.36M training draws.

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. Extending training to eight embodiments, including human hands and bodies, further improves robot predictions on DROID and ROBOMIND, suggesting that rendered bodies let a world model learn from data well beyond a single robot. 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 embodiments, multi-view prediction, and longer-horizon prediction to future work. Combining pretrained video models with learned 3D representations may further reduce the need for explicit pixel-level alignment.

References

[1] 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] 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)*, pages 472–489, 2026. doi: 10.1007/978-3-032-37098-3_26.

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

[4] 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.

[5] 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, pages 2226–2240, 2022.

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

[7] 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.

[8] 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 *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, volume 306 of *Proceedings of Machine Learning Research*, pages 37998–38011, 2026.

[9] 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.

[10] 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)*, pages 45210–45234, 2024.

[11] 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, pages 4603–4623, 2024.

[12] 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.

[13] 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.

[14] 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)*, pages 83048–83077, 2025.

[15] 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.

[16] 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)*, pages 10763–10806, 2026.

[17] 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, pages 9156–9172, 2023.

[18] 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, pages 5170–5194, 2025.

[19] 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)*, pages 78932–78957, 2026.

[20] 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)*, pages 6121–6138, 2026.

[21] 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)*, pages 9834–9844, 2025. doi: 10.1109/ICCV51701.2025.00917.

[22] 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)*, 2025. doi: 10.15607/RSS.2025.XXI.015.

[23] Zhuoyuan Wu and Jun Gao. OSCAR: Omni-embodiment action-conditioned world model for robotics. *arXiv preprint arXiv:2606.04463*, 2026.

[24] 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)*, pages 20765–20779, 2026.

[25] 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)*, pages 1053–1066, 2026.

[26] 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)*, pages 5337–5347, 2025. doi: 10.1109/ICCV51701.2025.00507.

[27] Yixiang Chen, Peiyan Li, Jiabing Yang, Keji He, Xiangnan Wu, Yuan Xu, Kai Wang, Jing Liu, Nianfeng Liu, Yan Huang, and Liang Wang. BridgeV2W: Bridging video generation models to embodied world models via embodiment masks. *arXiv preprint arXiv:2602.03793*, 2026.

[28] Yuxin Jiang, Shengcong Chen, Siyuan Huang, Liliang Chen, Pengfei Zhou, Yue Liao, Xindong He, Chiming Liu, Hongsheng Li, Maoqing Yao, and Guanghui Ren. EnerVerse-AC: Envisioning embodied environments with action condition. *arXiv preprint arXiv:2505.09723*, 2025.

[29] Mutian Xu, Tianbao Zhang, Tianqi Liu, Zhaoxi Chen, Xiaoguang Han, and Ziwei Liu. Kinema4D: Kinematic 4D world modeling for spatiotemporal embodied simulation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 4495–4500, 2026.

[30] Byungjun Kim, Taeksoo Kim, Junyoung Lee, and Hanbyul Joo. Dexterous world models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 29663–29673, 2026.

[31] 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)*, pages 6137–6144, 2026. doi: 10.1109/ICRA57385.2026.11697505.

[32] 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)*, 2026.

[33] 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)*, pages 6892–6903, 2024. doi: 10.1109/ICRA57147.2024.10611477.

[34] 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.

[35] 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, pages 2679–2713, 2024.

[36] 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.

[37] 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, pages 124420–124450, 2024.

[38] 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, pages 496–512, 2024.

[39] 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)*, 2024. doi: 10.15607/RSS.2024.XX.069.

[40] 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, pages 209–233, 2024.

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

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

[43] 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)*, pages 44918–44941, 2024.

[44] 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, pages 2475–2499, 2024.

[45] 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)*, pages 331–348, 2024. doi: 10.1007/978-3-031-72670-5_19.

[46] 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)*, pages 1–12, 2025. doi: 10.1109/CVPR52734.2025.00010.

[47] 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)*, pages 12713–12724, 2025. doi: 10.1109/ICCV51701.2025.01181.

[48] 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.

[49] Ruoshi Liu, Alper Canberk, Shuran Song, and Carl Vondrick. Differentiable robot rendering. In *Conference on Robot Learning (CoRL)*, volume 270, pages 117–129, 2024.

[50] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations (ICLR)*, 2023.

[51] Yash Jangir, Yidi Zhang, Pang-Chi Lo, 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)*, pages 25017–25043, 2026.

[52] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024.

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

[54] 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.

[55] 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, pages 1723–1736, 2023.

[56] 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.

[57] Arthur Allshire, Himanshu Gaurav Singh, Ritvik Singh, Adam Rashid, Hongsuk Choi, David McAllister, Justin Yu, Yiyuan Chen, Huang Huang, Pieter Abbeel, Xi Chen, Rocky Duan, Phillip Isola, Jitendra Malik, Fred Shentu, Guanya Shi, Philipp Wu, and Angjoo Kanazawa. Scalable behavior cloning with open data, training, and evaluation. In *Conference on Robot Learning (CoRL)*, 2026.

[58] Dantong Niu, Zhuoyang Liu, Zekai Wang, Boning Shao, Zhao-Heng Yin, Anirudh Pai, Yuvan Sharma, Stefano Saravalle, Ruijie Zheng, Jing Wang, Ryan Punamiya, Mengda Xu, Yuqi Xie, Yunfan Jiang, Letian Fu, Konstantinos Kallidromitis, Matteo Gioia, Junyi Zhang, Jiaxin Ge, Haiwen Feng, Fabio Galasso, Wei Zhan, David M. Chan, Yutong Bai, Roei Herzig, Jiahui Lei, Li Fei-Fei, Ken Goldberg, Jitendra Malik, Pieter Abbeel, Yuke Zhu, Danfei Xu, Linxi Fan, and Trevor Darrell. T-Rex: Tactile-reactive dexterous manipulation. In *Conference on Robot Learning (CoRL)*, 2026.

[59] Rao Fu, Dingxi Zhang, Alex Jiang, Wanjia Fu, Austin Funk, Daniel Ritchie, and Srinath Sridhar. GigaHands: A massive annotated dataset of bimanual hand activities. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17461–17474, 2025.

[60] Xinyu Zhan, Lixin Yang, Yifei Zhao, Kangrui Mao, Hanlin Xu, Zenan Lin, Kailin Li, and Cewu Lu. OakInk2: A dataset of bimanual hands-object manipulation in complex task completion. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 445–456, 2024.

[61] Kristen Grauman, Andrew Westbury, Lorenzo Torresani, Kris Kitani, Jitendra Malik, Triantafyllos Afouras, Kumar Ashutosh, Vijay Baiyya, Siddhant Bansal, Bikram Boote, Eugene Byrne, Zach Chavis, Joya Chen, Feng Cheng, Fu-Jen Chu, Sean Crane, Avijit Dasgupta, Jing Dong, Maria Escobar, Cristhian Forigua, Abrham Gebreselasie, Sanjay Haresh, Jing Huang, Md Mohaiminul Islam, Suyog Jain, Rawal Khirodkar, Devansh Kukreja, Kevin J Liang, Jia-Wei Liu, Sagnik Majumder, Yongsen Mao, Miguel Martin, Effrosyni Mavroudi, Tushar Nagarajan, Francesco Ragusa, Santhosh Kumar Ramakrishnan, Luigi Seminara, Arjun Somayazulu, Yale Song, Shan Su, Zihui Xue, Edward Zhang, Jinxu Zhang, Angela Castillo, Changan Chen, Xinzhu Fu, Ryosuke Furuta, Cristina Gonzalez, Prince Gupta, Jiabo Hu, Yifei Huang, Yiming Huang, Weslie Khoo, Anush Kumar, Robert Kuo, Sach Lakhavani, Miao Liu, Mi Luo, Zhengyi Luo, Brighid Meredith, Austin Miller, Oluwatumininu Oguntola, Xiaqing Pan, Penny Peng, Shraman Pramanick, Merey Ramazanova, Fiona Ryan, Wei Shan, Kiran Somasundaram, Chenan Song, Audrey Southerland, Masatoshi Tateno, Huiyu Wang, Yuchen Wang, Takuma Yagi, Mingfei Yan, Xitong Yang, Zecheng Yu, Shengxin Cindy Zha, Chen Zhao, Ziwei Zhao, Zhifan Zhu, Jeff Zhuo, Pablo Arbelaez, Gedas Bertasius, Dima Damen, Jakob Engel, Giovanni Maria Farinella, Antonino Furnari, Bernard Ghanem, Judy Hoffman, C.V. Jawahar, Richard Newcombe, Hyun Soo Park, James M. Rehg, Yoichi Sato, Manolis Savva, Jianbo Shi, Mike Zheng Shou, and Michael Wray. Ego-Exo4D: Understanding skilled human activity from first- and third-person perspectives. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 19383–19400, 2024.

[62] Javier Romero, Dimitrios Tzionas, and Michael J. Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics*, 36(6):245:1–245:17, 2017.

[63] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed A. A. Osman, Dimitrios Tzionas, and Michael J. Black. Expressive body capture: 3D hands, face, and body from a single image. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10967–10977, 2019.

[64] Zhou Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[65] 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)*, pages 586–595, 2018. doi: 10.1109/CVPR.2018.00068.

[66] 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)*, pages 28085–28128, 2025.

[67] Abhiram Maddukuri and Georgios Pavlakos. Ego-Exo4D human meshes dataset: 4D human motion reconstruction for ego-exo captures. *arXiv preprint arXiv:2609.30187*, 2026.

[68] NVIDIA, Arslan Ali, Junjie Bai, Maciej Bala, Yogesh Balaji, Aaron Blakeman, Tiffany Cai, Jiaxin Cao, Tianshi Cao, Elizabeth Cha, Yu-Wei Chao, Prithvijit Chattopadhyay, Mike Chen, Yongxin Chen, Yu Chen, Shuai Cheng, Yin Cui, Jenna Diamond, Yifan Ding, Jiaojiao Fan, Linxi Fan, Liang Feng, Francesco Ferroni, Sanja Fidler, Xiao Fu, Ruiyuan Gao, Yunhao Ge, Jinwei Gu, Aryaman Gupta, Siddharth Gururani, Imad El Hanafi, Ali Hassani, Zekun Hao, Jacob Huffman, Joel Jang, Pooya Jannaty, Jan Kautz, Grace Lam, Xuan Li, Zhaoshuo Li, Maosheng Liao, Chen-Hsuan Lin, Tsung-Yi Lin, Yen-Chen Lin, Huan Ling, Ming-Yu Liu, Xian Liu, Yifan Lu, Alice Luo, Qianli Ma, Hanzi Mao, Kaichun Mo, Seungjun Nah, Yashraj Narang, Abhijeet Panaskar, Lindsey Pavao, Trung Pham, Morteza Ramezanali, Fitsum Reda, Scott Reed, Xuanchi Ren, Haonan Shao, Yue Shen, Stella Shi, Shuran Song, Bartosz Stefaniak, Shangkun Sun, Shitao Tang, Sameena Tasmeen, Lyne Tchapmi, Wei-Cheng Tseng, Jibin Varghese, Andrew Z. Wang, Hao Wang, Haoxiang Wang, Heng Wang, Ting-Chun Wang, Fangyin Wei, Jiashu Xu, Dinghao Yang, Xiaodong Yang, Haotian Ye, Seonghyeon Ye, Xiaohui Zeng, Jing Zhang, Qinsheng Zhang, Kaiwen Zheng, Andrew Zhu, and Yuke Zhu. World simulation with video foundation models for physical AI. *arXiv preprint arXiv:2511.00062*, 2025.

[69] Hao-Shu Fang, Hongjie Fang, Zhenyu Tang, Jirong Liu, Chenxi Wang, Junbo Wang, Haoyi Zhu, and Cewu Lu. RH20T: A comprehensive robotic dataset for learning diverse skills in one-shot. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 653–660, 2024.

[70] Yang Tian, Yuyin Yang, Yiman Xie, Zetao Cai, Xu Shi, Ning Gao, Hangxu Liu, Xuekun Jiang, Zherui Qiu, Feng Yuan, Yaping Li, Ping Wang, Junhao Cai, Jia Zeng, Hao Dong, and Jiangmiao Pang. InternData-A1: Pioneering high-fidelity synthetic data for pre-training generalist policy. *arXiv preprint arXiv:2511.16651*, 2025.

[71] AgiBot World Contributors, Qingwen Bu, Jisong Cai, Li Chen, Xiuqi Cui, Yan Ding, Siyuan Feng, Shenyuan Gao, Xindong He, Xuan Hu, Xu Huang, Shu Jiang, Yuxin Jiang, Cheng Jing, Hongyang Li, Jialu Li, Chiming Liu, Yi Liu, Yuxiang Lu, Jianlan Luo, Ping Luo, Yao Mu, Yuehan Niu, Yixuan Pan, Jiangmiao Pang, Yu Qiao, Guanghui Ren, Cheng Ruan, Jiaqi Shan, Yongjian Shen, Chengshi Shi, Mingkang Shi, Modi Shi, Chonghao Sima, Jianheng Song, Huijie Wang, Wenhao Wang, Dafeng Wei, Chengen Xie, Guo Xu, Junchi Yan, Cunbiao Yang, Lei Yang, Shukai Yang, Maoqing Yao, Jia Zeng, Chi Zhang, Qinglin Zhang, Bin Zhao, Chengyue Zhao, Jiaqi Zhao, and Jianchao Zhu. AgiBot World Colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3549–3556, 2025.

[72] Ryosuke Takanami, Petr Khrapchenkov, Shu Morikuni, Jumpei Arima, Yuta Takaba, Shunsuke Maeda, Takuya Okubo, Genki Sano, Satoshi Sekioka, Aoi Kadoya, Motonari Kambara, Naoya Nishiura, Haruto Suzuki, Takanori Yoshimoto, Koya Sakamoto, Shinnosuke Ono, Hu Yang, Daichi Yashima, Aoi Horo, Tomohiro Motoda, Kensuke Chiyoma, Hiroshi Ito, Koki Fukuda, Akihito Goto, Kazumi Morinaga, Yuya Ikeda, Riko Kawada, Masaki Yoshikawa, Norio Kosuge, Yuki Noguchi, Kei Ota, Tatsuya Matsushima, Yusuke Iwasawa, Yutaka Matsuo, and Tetsuya

Ogata. AIRoA MoMa dataset: A large-scale hierarchical dataset for mobile manipulation. *arXiv preprint arXiv:2509.25032*, 2025.

[73] Ryan Hoque, Peide Huang, David J. Yoon, Mouli Sivapurapu, and Jian Zhang. EgoDex: Learning dexterous manipulation from large-scale egocentric video. In *International Conference on Learning Representations (ICLR)*, pages 4218–4237, 2026.

[74] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Antonino Furnari, Evangelos Kazakos, Jian Ma, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Rescaling egocentric vision: Collection, pipeline and challenges for EPIC-KITCHENS-100. *International Journal of Computer Vision*, 130(1):33–55, 2022.

[75] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

[76] 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)*, pages 5261–5271, 2025. doi: 10.1109/CVPR52734.2025.00496.

A Appendix

A.1 Website and Video

Additional qualitative results are available on our project webpage, including extended video comparisons across all eight datasets, long-horizon rollouts, examples of trajectory and environment generalization, and side-by-side comparisons with our baselines. We encourage readers to view the webpage for the full video results, which are difficult to represent comprehensively in static paper figures.
https://doppelwm.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. Figures 10 and 11 show 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. [49]. 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.

Additional embodiments. The bimanual YAM arms in ABC are rendered with a DrRobot model trained for the YAM in the same way. The Vega-1 humanoid and its Sharpa hands in T-Rex are rendered from the robot's URDF meshes with the dataset's released head-camera calibration. People are rendered from fitted body and hand meshes: MANO hands for GigaHands [62], SMPL-X for OakInk2 [63], and SMPL-H for Ego-Exo4D, using the meshes of Maddukuri and Pavlakos [67]; each mesh is rasterized through the dataset's camera model, including lens distortion. All renderings use a black background and contain only the embodiment.

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 three-robot training, we sample uniformly across the three datasets, such that each embodiment contributes approximately one third of the optimization batches.

Additional embodiments. For ABC, T-Rex, GigaHands, OakInk2, and Ego-Exo4D, we cut each video into back-to-back 3.2 s windows at 5 fps, plus equidistant segment clips and one whole-video clip; for Ego-Exo4D we use two exocentric cameras per take. Whole-video clips of ABC and Ego-Exo4D are evaluated but not used for training. Validation holds out a random 5% of episodes. The test splits hold out: 19 of ABC's tasks; two of T-Rex's twelve wall backdrops (400 of 5,464

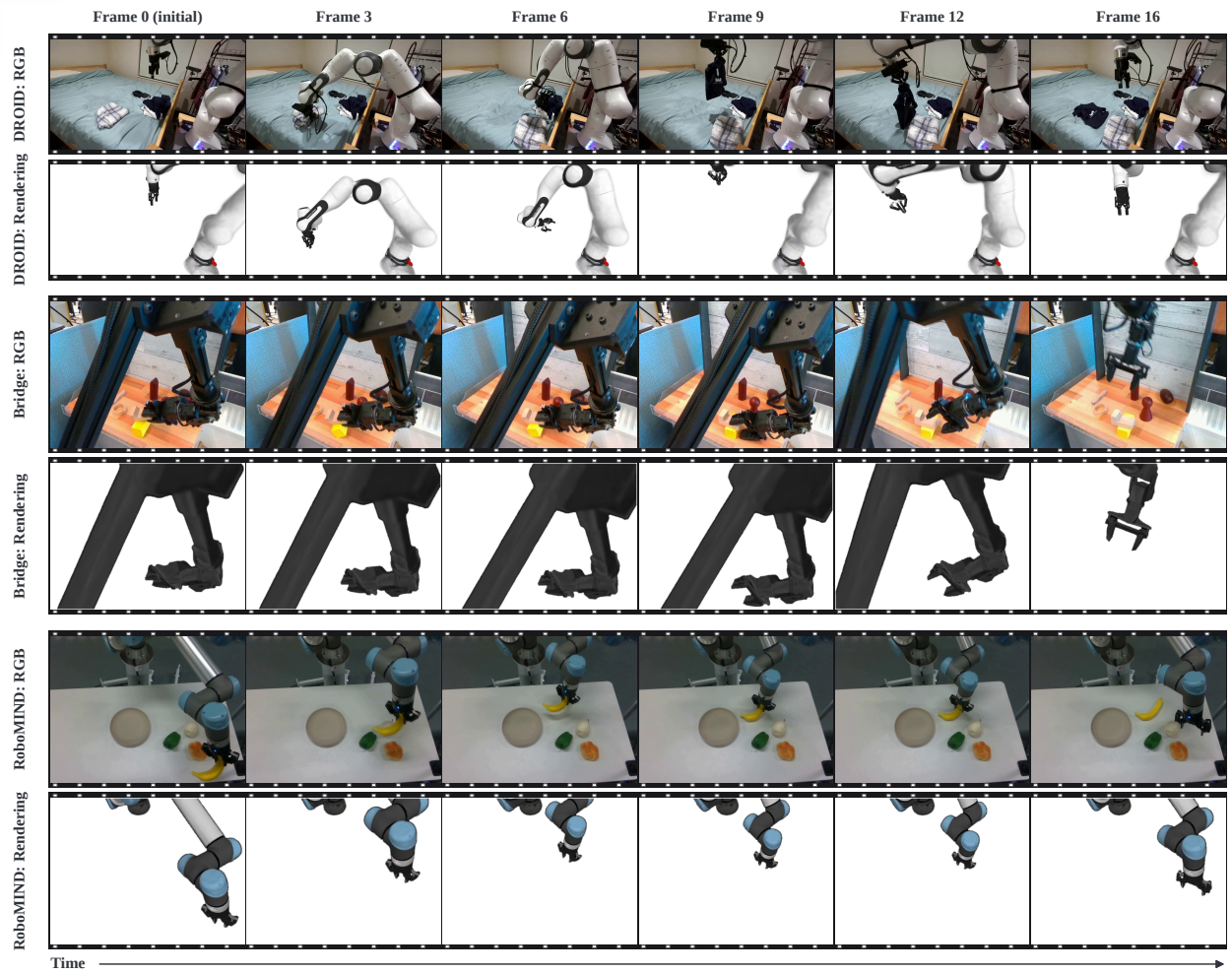


Figure 10: Alignment between the observed and rendered robot trajectories. The top row shows frames from the real RGB trajectory, and the bottom row shows the corresponding target-view robot renderings. The renderings express the commanded embodiment motion in the same image coordinates as the observations, providing the dense visual component of Doppel.

episodes), never seen in training or validation; one task per Ego-Exo4D activity (Mikan layup, chain cleaning, latte making, advanced salsa, CPR, piano scales, a bouldering route, and penalty kicks); five GigaHands activities (tea, stone, laptop, phone, crochet); and 51 OakInk2 tasks across its four scenes. Table 4 gives the amount of video per split.

Doppel-8 samples each dataset in proportion to its training clips, which gives about 60 passes over every dataset.

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

Table 4: Hours of video per dataset and split, from the full length of each source episode; each episode counts once, and Ego-Exo4D is counted per take (training uses up to two exocentric cameras per take, 176.3 h of camera video).

Dataset	Train	Val.	Test	Total
DROID	98.9	5.4	11.6	115.9
BRIDGE	26.4	1.4	3.2	31.1
ROBOMIND	62.1	3.2	6.8	72.1
ABC	242.6	12.7	31.4	286.7
T-Rex	44.2	2.4	3.9	50.5
Ego-Exo4D	88.3	5.1	10.4	103.8
GigaHands	25.4	1.2	2.1	28.6
OakInk2	5.2	1.2	1.5	7.9
All eight	593.1	32.6	70.8	696.6

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.

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 about 30.5M 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. $\text{Conv3D}(256, 512, 3 \times 3 \times 3)$
3. $\text{Conv3D}(512, 512, 1 \times 3 \times 3)$

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 a configuration vector: up to seven joint values and one gripper value for the robot arms, and up to 159 values for the other embodiments. We zero-pad every vector to 159 dimensions. We normalize each embodiment using its own training-set statistics and zero-pad configurations with fewer degrees of freedom to the shared 159-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:

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

where $W_1 : \mathbb{R}^{159} \to \mathbb{R}^{512}$, $W_2 : \mathbb{R}^{512} \to \mathbb{R}^{512}$, 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:

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

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:

$$G = \text{Linear}_{2048\to5120}(\text{GELU}(\text{Linear}_{512\to2048}(\text{LayerNorm}(H_{\text{fused}})))\,. \tag{13}$$

The resulting tensor has shape $G \in \mathbb{R}^{2025\times5120}$. 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 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 [2]. 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 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 6. This baseline therefore provides a controlled comparison between numerical action-only conditioning and DOPPEL conditioning.

OSCAR. For OSCAR [23], 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 5. 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 12, 13, and 14, respectively, and from the five added datasets in Figures 15–19. These examples illustrate the diversity of embodiments, environments, and viewpoints across the eight datasets.

A.7 Implementation Details

We provide the training configurations for all our models in Tables 6, 7 and 8. 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.

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.

Table 5: OSCAR configurations used in our experiments. The fine-tuned models are initialized from the released OSCAR checkpoint [23].

	Released OSCAR	**OSCAR (Fine-tuned)**
Initialization	Cosmos-Predict2.5 [68], 2B	Released OSCAR checkpoint
Conditioning	2D skeleton + text	2D skeleton + empty text prompt
Training data	180657 episodes: 94830 robot episodes (RH20T [69], InternData-A1 [70], DROID [54] 21904, AgiBot-Beta [71], AIROA-MoMa [72]) and 85827 human episodes (EgoDex [73], EPIC-Kitchens [74])	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× GH200	8× A100
Sampling	35 steps, guidance scale 6	35 steps, guidance scale 6

Table 6: 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 [75] 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 [50].

Whole-frame visual fidelity. We report peak signal-to-noise ratio (PSNR), structural similarity (SSIM) [64], 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.

Table 7: Training configuration for the single-node multi-embodiment model. Settings not listed are identical to Table 6.

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.

Table 8: Training configuration for the four-node multi-embodiment render–action model. Settings not listed are identical to Table 6.

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.

Table 9: Training configuration for DOPPEL-8. Settings not listed are identical to Table 8.

Data	Joint training on the eight datasets of Table 1 (803,602 training clips, 593 hours of training video, Table 4). Datasets are sampled in proportion to their training clips, and clips uniformly within each dataset.
Actions	Every action vector is zero-padded to 159 dimensions and passed to one shared action encoder; there is no embodiment label or embodiment-specific parameter.
Hardware	32 GPUs across four nodes, one FSDP rank per GPU: H100-80GB.

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.

Table 10: Doppel-3 versus Doppel-8, paired on the same 200 clips per set (final checkpoints). Δ is Doppel-8 minus Doppel-3 with its standard error; *Better* is the share of clips where Doppel-8 has higher PSNR; arm and moving-object Δ use the region metrics of Appendix A.8.

Dataset	Split	Doppel-3	Doppel-8	Δ PSNR	Better	Δ arm	Δ moving obj.
Droid	Val.	24.30	24.82	$+0.52 \pm 0.03$	92%	+0.70	+0.69
	OOD	21.25	21.46	$+0.21 \pm 0.03$	73%	+0.34	+0.13
Bridge	Val.	26.55	26.54	-0.01 ± 0.06	57%	+0.09	+0.12
	OOD	22.47	22.24	-0.24 ± 0.05	38%	−0.01	−0.08
RoboMind	Val.	31.07	31.30	$+0.24 \pm 0.04$	75%	+0.42	+0.14
	OOD	30.44	30.66	$+0.22 \pm 0.04$	72%	+0.34	+0.16

A.9 Camera extrinsics dataset

We release the camera extrinsics and intrinsics we recover for 45,345 demonstrations: Bridge 14,824, RoboMind 17,588, and ABC 12,933. The other five datasets ship their own cameras, which we use and do not re-release; for GigaHands we refine the released cameras per recording session. Droid extrinsics and ZED intrinsics are used as provided by Khazatsky et al. [54]. 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 [56] and MoGe-estimated [76] focal lengths for Bridge. We show sample camera-calibration overlays in Fig. 20. ABC's top-down cameras also ship without extrinsics and move between recordings, so we calibrate each ABC video separately by fitting the camera pose to segmentation masks of the two YAM arms and to photometric agreement with the rendering. We fit 13,518 videos, of which 12,933 (95.7%) pass our alignment check.

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. 21.

A.10.2 From three to eight embodiments

Table 10 compares Doppel-3 with Doppel-8 on the same clips. Both models see a similar amount of each robot dataset (Table 2); Doppel-8 additionally trains on the five other datasets. Table 12 repeats the comparison of Table 2 at matched training exposure, with every model at its saved checkpoint closest to 51 passes over each robot dataset. Table 11 follows the same comparison over training.

A.10.3 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.

Table 11: PSNR of Doppel-8 minus Doppel-3, on the same 200 clips per set, at three points in training.

		Doppel-3	Δ PSNR (Doppel-8 − Doppel-3)		
Dataset	Split	passes	≈30 passes	51 passes	final
Droid	Val.	24.8	+0.70	+0.45	+0.52
	OOD	24.8	+0.51	+0.17	+0.21
Bridge	Val.	33.4	+0.25	+0.13	−0.01
	OOD	33.4	−0.10	−0.05	−0.24
RoboMind	Val.	27.9	+0.52	+0.21	+0.24
	OOD	27.9	+0.25	+0.13	+0.22

Table 12: Prediction quality on the three robot datasets at matched training exposure: every model is evaluated at its saved checkpoint closest to 51 passes over each robot dataset.

				Validation: unseen trajectories				OOD test			
Dataset	Method	Input	Trained on	PSNR ↑	SSIM ↑	LPIPS ↓	SCS ↑	PSNR ↑	SSIM ↑	LPIPS ↓	SCS ↑
Droid	Action-only	Actions	This dataset	22.847	0.841	0.079	0.927	20.047	0.780	0.131	0.873
	Doppel	Render + actions	This dataset	24.322	0.864	0.065	0.940	21.278	0.802	0.113	**0.903**
	Doppel-3	Render + actions	3 robot datasets	24.297	0.862	0.066	0.940	21.250	0.803	0.111	0.899
	Doppel-8	Render + actions	All 8 datasets	**24.749**	**0.869**	**0.063**	**0.943**	**21.421**	**0.805**	**0.109**	0.901
Bridge	Action-only	Actions	This dataset	25.625	0.864	0.054	0.941	20.665	0.756	0.129	0.869
	Doppel	Render + actions	This dataset	26.311	0.871	0.050	0.947	21.824	0.773	0.109	**0.905**
	Doppel-3	Render + actions	3 robot datasets	26.389	0.871	0.050	0.947	**22.348**	0.784	**0.099**	**0.905**
	Doppel-8	Render + actions	All 8 datasets	**26.521**	**0.873**	**0.049**	**0.948**	22.295	**0.785**	**0.099**	0.900
RoboMind	Action-only	Actions	This dataset	29.992	0.919	0.040	0.938	29.805	0.915	0.040	0.930
	Doppel	Render + actions	This dataset	31.007	0.928	**0.037**	**0.942**	30.387	0.920	0.038	0.932
	Doppel-3	Render + actions	3 robot datasets	31.013	0.927	0.038	0.940	30.530	0.920	**0.037**	**0.939**
	Doppel-8	Render + actions	All 8 datasets	**31.221**	**0.929**	**0.037**	**0.942**	**30.662**	**0.921**	0.038	0.936

Single-dataset experts versus Doppel-3. Tables 14 and 15 give the four-checkpoint curves for the separately trained action-only experts and Doppel-3 (batch 256). The training budgets are measured per dataset. The expert rows combine the corresponding single-dataset model for each dataset.

Shared action-only versus Doppel-3 (batch 64). Tables 16 and 17 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 Doppel-3 (batch 64) run differs from the batch-256 Doppel-3 run in the expert comparison above.

Table 13: Prediction quality by image region on the three robot datasets, for the same models, checkpoints and clips as Table 2. *Arm* is the robot; *Obj.* is the objects that move during the clip (non-robot regions segmented in the first frame with SAM 2 and tracked through the clip). PSNR is computed inside each region and averaged over the 16 predicted frames; SCS is structural consistency of the same regions.

				Validation: unseen trajectories				OOD test			
Dataset	Method	Input	Trained on	$PSNR_{arm}$	$PSNR_{obj}$	SCS_{arm}	SCS_{obj}	$PSNR_{arm}$	$PSNR_{obj}$	SCS_{arm}	SCS_{obj}
DROID	Action-only	Actions	This dataset	18.74	27.26	0.808	0.839	17.64	24.67	0.712	0.746
	OSCAR (fine-tuned)	Skeleton	This dataset	16.54	22.92	0.778	0.801	16.05	22.08	0.712	0.728
	DOPPEL	Render + actions	This dataset	20.88	28.63	0.843	0.867	19.33	26.50	0.773	**0.805**
	DOPPEL-3	Render + actions	3 robot datasets	20.89	28.56	0.846	0.869	19.23	26.41	0.777	0.800
	DOPPEL-8	Render + actions	All 8 datasets	**21.59**	**29.25**	**0.853**	**0.876**	**19.57**	**26.54**	**0.795**	**0.805**
BRIDGE	Action-only	Actions	This dataset	21.15	27.65	0.776	0.894	17.74	21.50	0.591	0.778
	OSCAR (fine-tuned)	Skeleton	This dataset	18.25	22.78	0.702	0.840	17.50	20.62	0.591	0.788
	DOPPEL	Render + actions	This dataset	21.52	**28.04**	0.787	**0.901**	18.72	23.64	0.689	0.838
	DOPPEL-3	Render + actions	3 robot datasets	21.49	27.84	0.791	0.900	**19.05**	**24.93**	**0.692**	**0.843**
	DOPPEL-8	Render + actions	All 8 datasets	**21.57**	27.96	**0.793**	**0.901**	19.04	24.85	0.685	0.831
ROBOMIND	Action-only	Actions	This dataset	25.64	29.16	0.909	0.866	25.03	28.70	0.896	0.826
	OSCAR (fine-tuned)	Skeleton	This dataset	22.96	24.96	0.882	0.809	22.89	25.34	0.876	0.798
	DOPPEL	Render + actions	This dataset	26.71	29.53	0.916	0.869	25.12	28.78	0.902	0.826
	DOPPEL-3	Render + actions	3 robot datasets	26.81	29.46	0.917	0.869	25.30	28.91	0.903	**0.844**
	DOPPEL-8	Render + actions	All 8 datasets	**27.23**	**29.60**	**0.919**	**0.873**	**25.64**	**29.07**	**0.905**	0.841

Best in **bold**, second-best underlined; ties at displayed precision share emphasis. Higher is better for all metrics. 200 clips per split, the same clips for every method.

Table 14: PSNR at every checkpoint in the expert-versus-shared-model scaling comparison. Action-only experts are trained separately on each dataset; DOPPEL is trained jointly on all three.

		DROID		BRIDGE		ROBOMIND	
Method	Draws/dataset (M)	Val	OOD	Val	OOD	Val	OOD
Action-only expert	~ 0.5	19.194	18.058	22.136	21.119	25.199	25.848
DOPPEL-3	~ 0.5	20.815	19.505	24.253	22.314	28.560	28.634
Action-only expert	~ 1.0	19.583	18.110	23.374	21.279	26.449	27.016
DOPPEL-3	~ 1.0	22.768	20.545	24.935	22.907	29.386	29.589
Action-only expert	~ 2.0	20.963	18.982	24.805	20.552	27.501	28.039
DOPPEL-3	~ 2.0	23.664	20.965	25.809	22.472	30.387	30.172
Action-only expert	~ 3.4	22.228	19.767	25.869	20.555	29.629†	29.629†
DOPPEL-3	~ 3.4	24.220	21.147	26.458	22.438	30.964	30.436

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

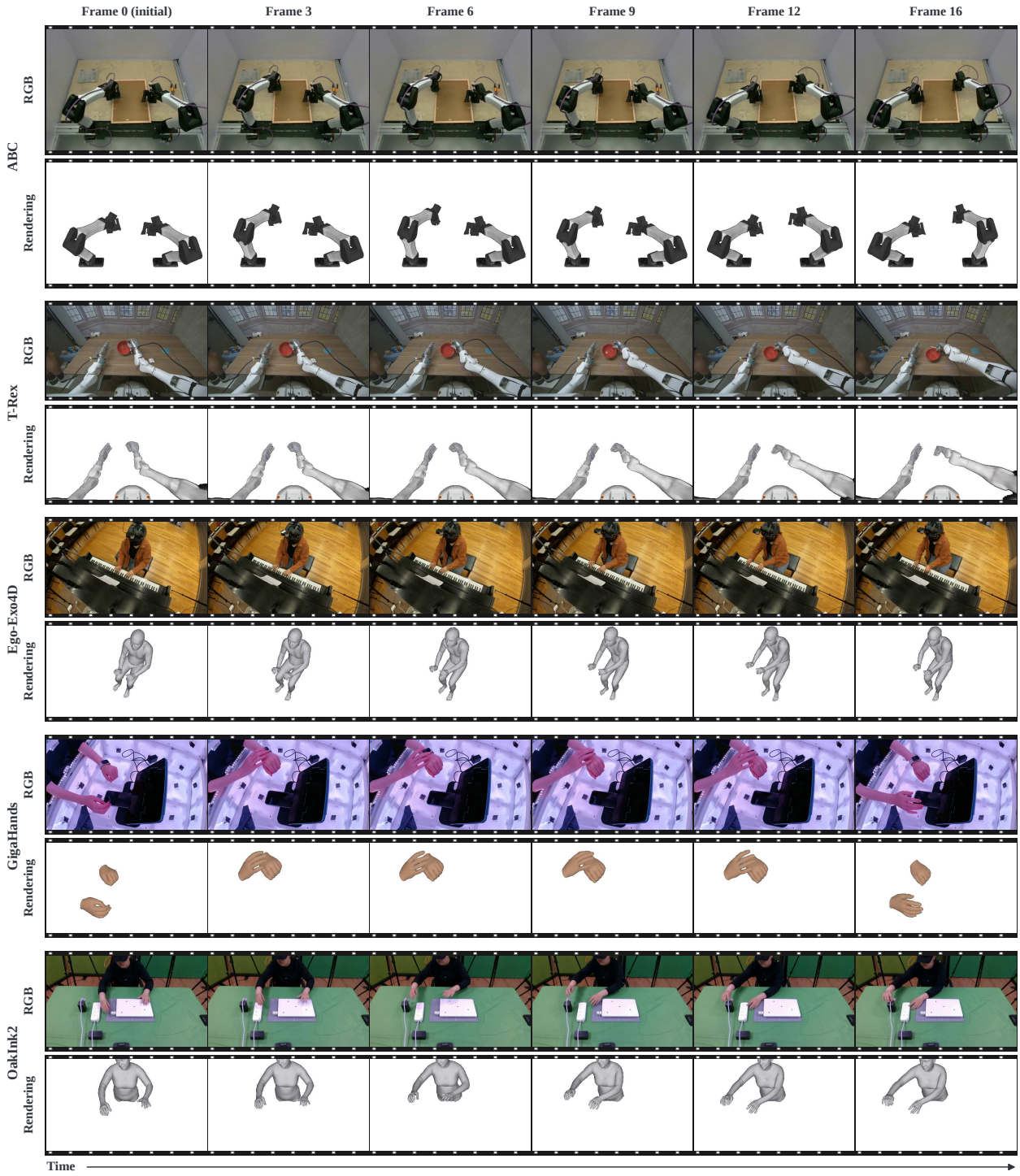


Figure 11: Alignment between the observed and rendered trajectories for the five added embodiments: a bimanual YAM robot (ABC), a humanoid with dexterous hands (T-Rex), a human body (Ego-Exo4D), human hands (GigaHands), and a human upper body (OakInk2). For each dataset, the top row shows the real RGB frames and the bottom row the corresponding camera-aligned renderings.

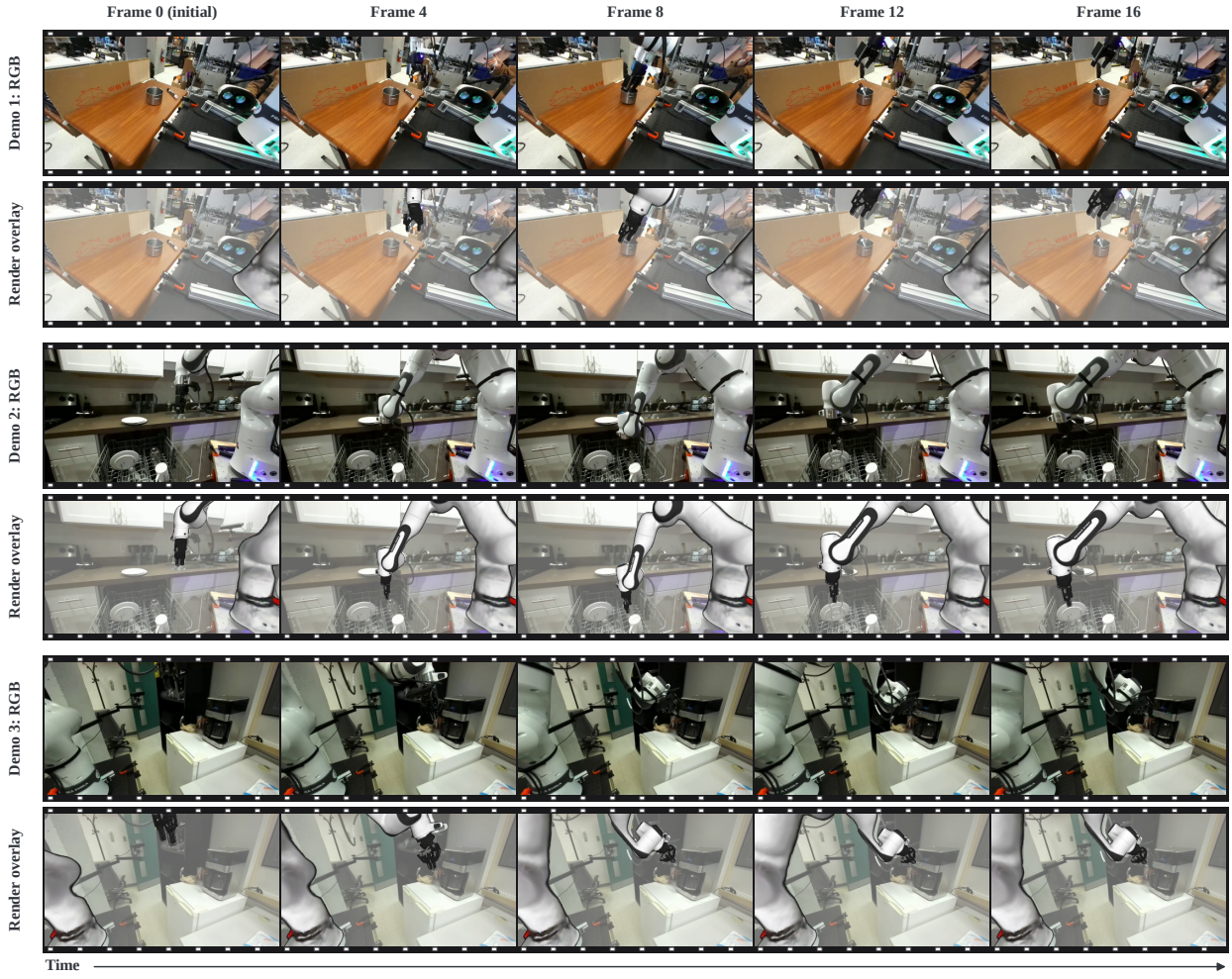


Figure 12: Representative examples from DROID with corresponding camera-aligned robot renderings.

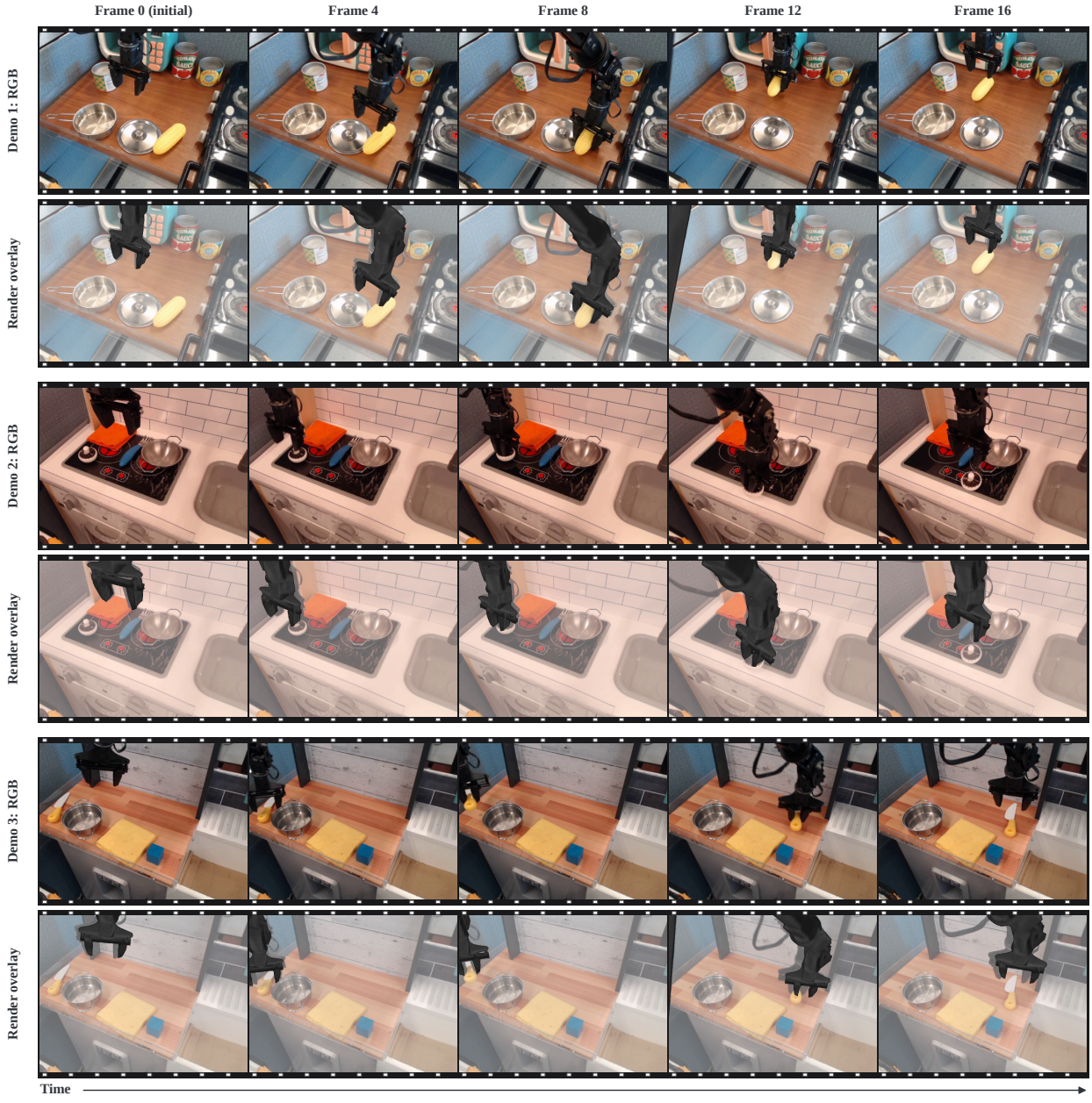


Figure 13: Representative examples from BRIDGE with corresponding camera-aligned robot renderings.

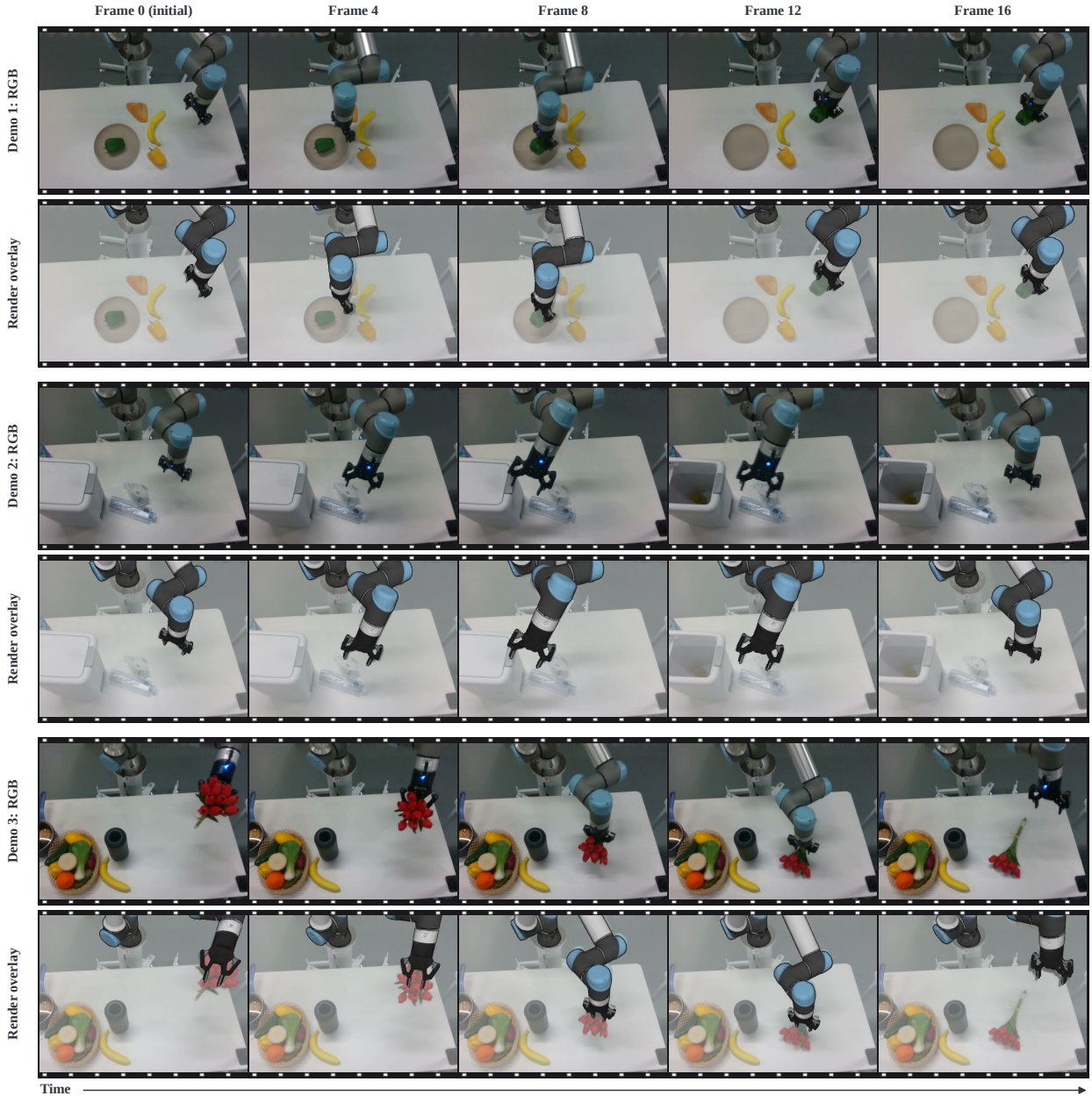


Figure 14: Representative examples from RoboMind with corresponding camera-aligned robot renderings.

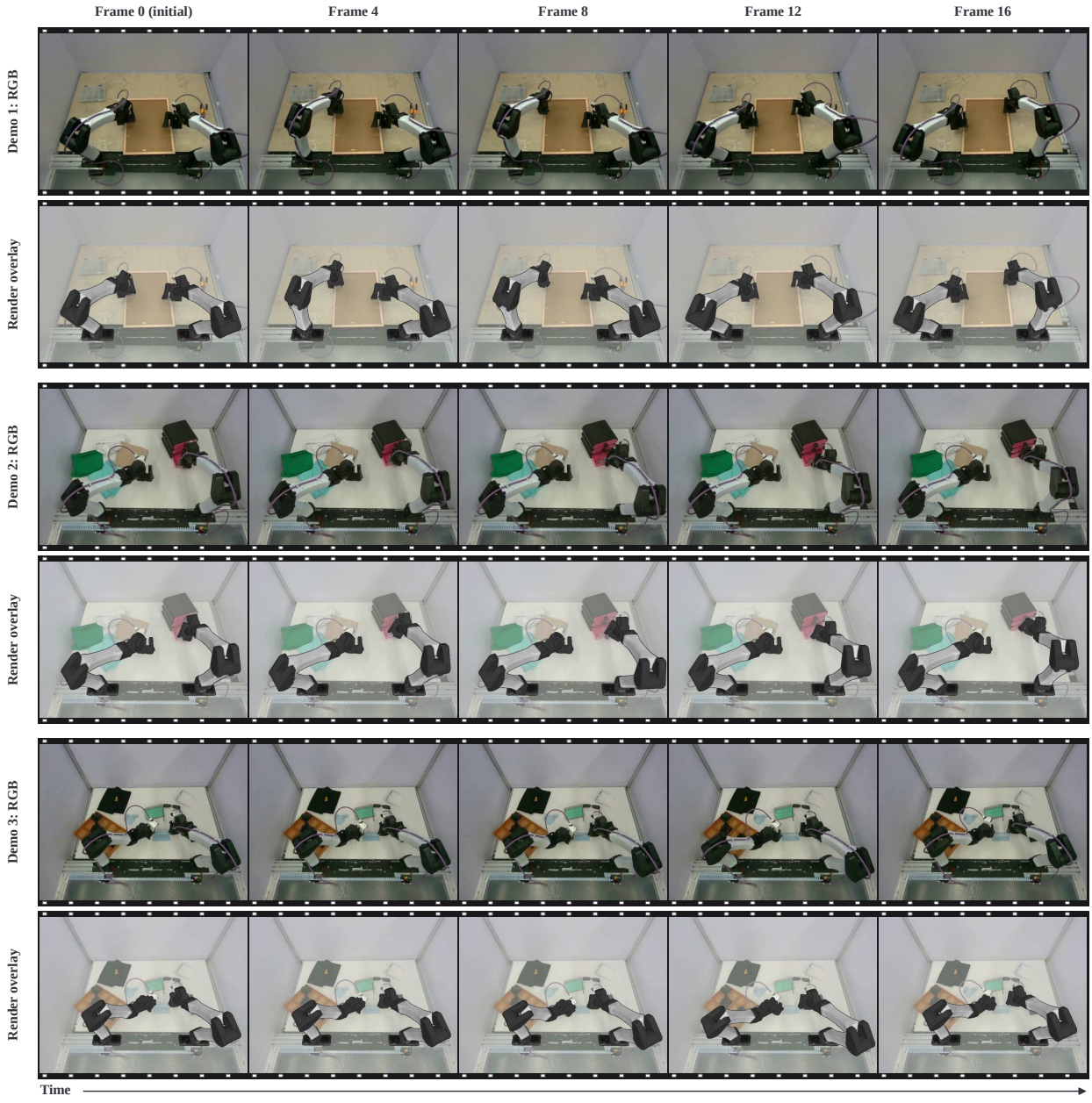


Figure 15: Representative examples from ABC (bimanual YAM) with corresponding camera-aligned renderings.



Figure 16: Representative examples from T-Rex (humanoid with dexterous hands) with corresponding camera-aligned renderings.

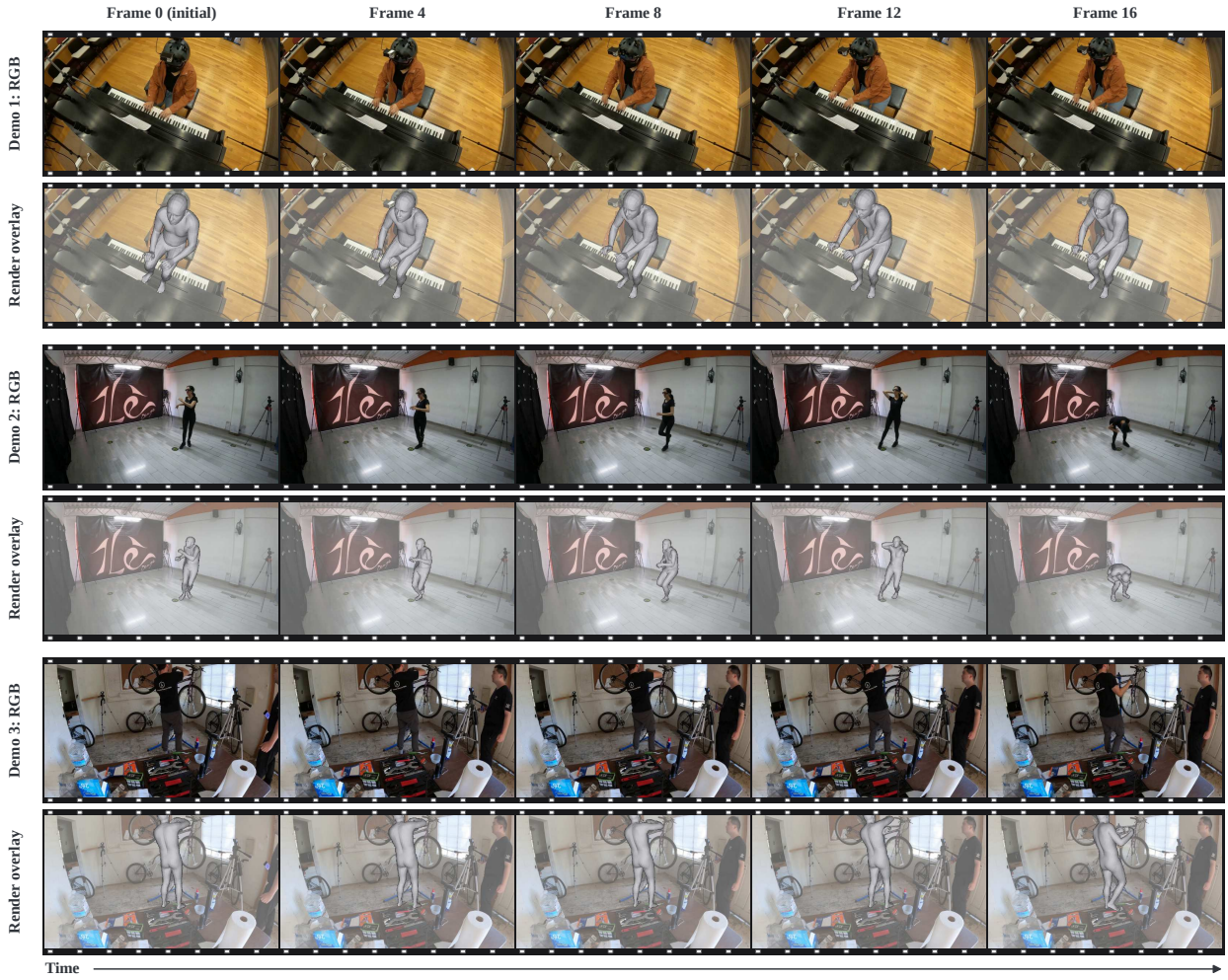


Figure 17: Representative examples from Ego-Exo4D (human body) with corresponding camera-aligned renderings.



Figure 18: Representative examples from GigaHands (human hands) with corresponding camera-aligned renderings.



Figure 19: Representative examples from OakInk2 (human upper body) with corresponding camera-aligned renderings.

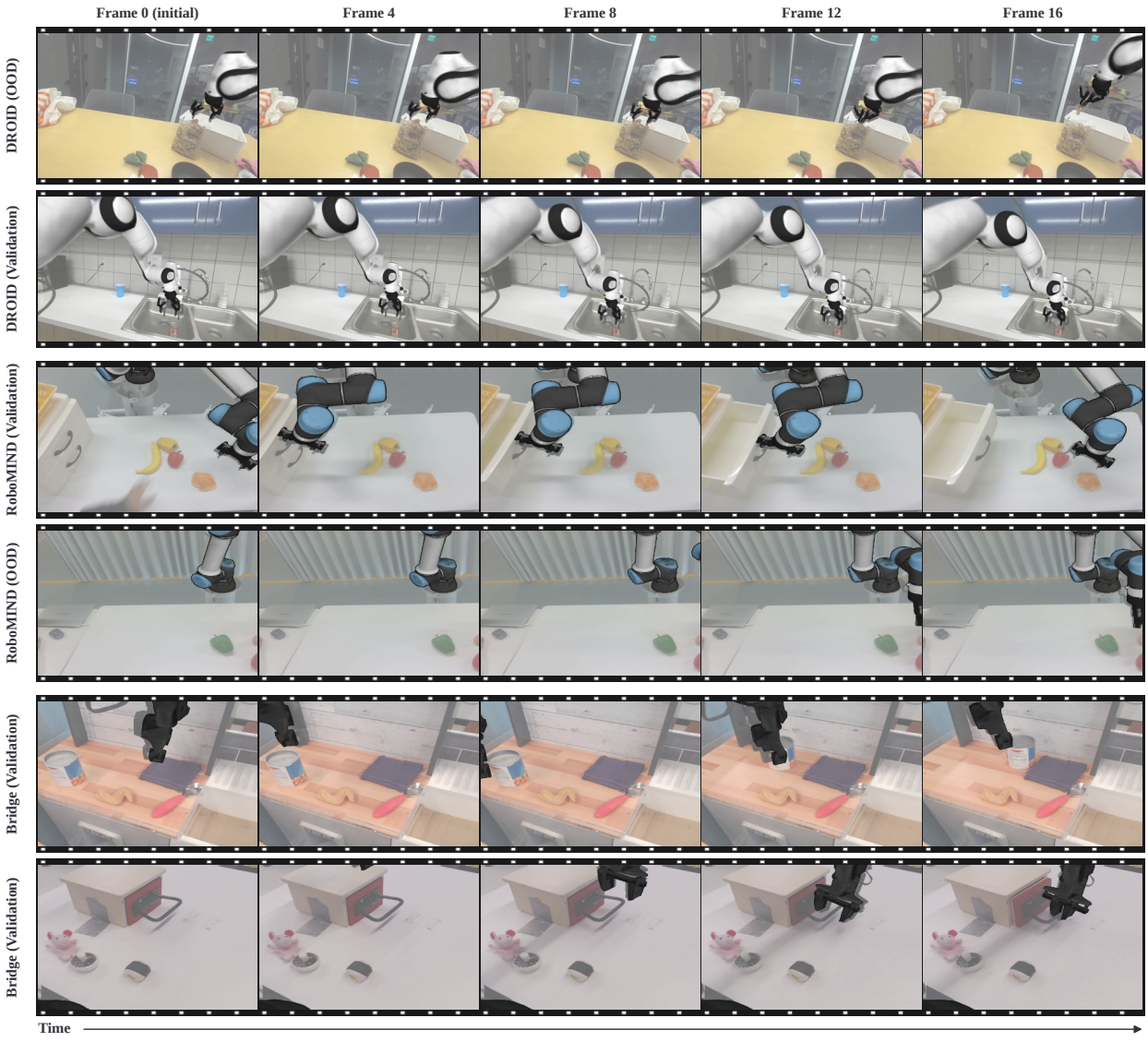


Figure 20: Overlays of camera calibration results.

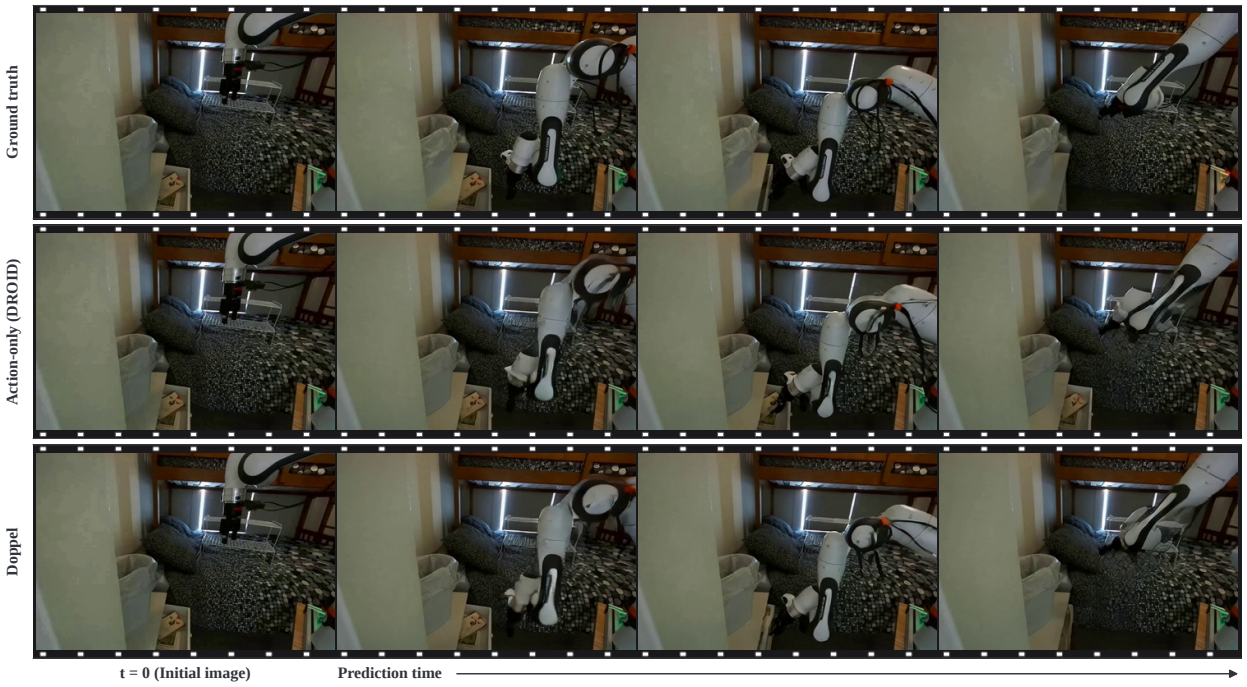


Figure 21: Additional trajectory-generalization example on DROID. We compare DOPPEL with the action-only baseline on a held-out trajectory from a familiar environment.

Table 15: SCS at every checkpoint in the expert-versus-shared-model scaling comparison. Action-only experts are trained separately on each dataset; DOPPEL is trained jointly on all three.

		DROID		BRIDGE		ROBOMIND	
Method	Draws/dataset (M)	Val	OOD	Val	OOD	Val	OOD
Action-only expert	~ 0.5	0.868	0.826	0.901	0.875	0.897	0.899
DOPPEL-3	~ 0.5	0.914	0.876	0.927	0.917	0.927	0.926
Action-only expert	~ 1.0	0.877	0.826	0.916	0.883	0.911	0.909
DOPPEL-3	~ 1.0	0.926	0.888	0.936	0.915	0.933	0.931
Action-only expert	~ 2.0	0.905	0.852	0.933	0.868	0.919	0.919
DOPPEL-3	~ 2.0	0.933	0.896	0.942	0.907	0.937	0.932
Action-only expert	~ 3.4	0.921	0.869	0.943	0.863	0.937	0.929
DOPPEL-3	~ 3.4	0.940	0.897	0.948	0.904	0.940	0.937

Table 16: PSNR at every checkpoint in the matched shared-model scaling comparison. Both models are trained jointly on all three datasets.

Method	Total draws (M)	DROID Val	DROID OOD	BRIDGE Val	BRIDGE OOD	ROBOMIND Val	ROBOMIND OOD
Action-only (multi)	~ 1.2	19.126	18.029	21.703	20.762	24.520	24.958
DOPPEL-3 (batch 64)	~ 1.2	22.070	20.251	24.098	22.781	28.504	29.082
Action-only (multi)	~ 2.4	20.381	18.762	23.196	21.722	26.575	26.965
DOPPEL-3 (batch 64)	~ 2.4	22.832	20.697	24.968	23.156	29.382	29.670
Action-only (multi)	~ 3.4	20.959	19.036	24.070	21.873	27.564	28.200
DOPPEL-3 (batch 64)	~ 3.4	23.188	20.862	25.431	22.726	29.937	29.818

Table 17: SCS at every checkpoint in the matched shared-model scaling comparison. Both models are trained jointly on all three datasets.

Method	Total draws (M)	DROID Val	DROID OOD	BRIDGE Val	BRIDGE OOD	ROBOMIND Val	ROBOMIND OOD
Action-only (multi)	~ 1.2	0.864	0.824	0.892	0.859	0.886	0.886
DOPPEL-3 (batch 64)	~ 1.2	0.919	0.883	0.927	0.914	0.927	0.932
Action-only (multi)	~ 2.4	0.893	0.839	0.914	0.886	0.910	0.910
DOPPEL-3 (batch 64)	~ 2.4	0.927	0.892	0.936	0.919	0.930	0.934
Action-only (multi)	~ 3.4	0.901	0.846	0.925	0.890	0.920	0.916
DOPPEL-3 (batch 64)	~ 3.4	0.930	0.896	0.940	0.912	0.933	0.930