POINTWORLD: SCALING 3D WORLD MODELS FOR IN-THE-WILD ROBOTIC MANIPULATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Humans anticipate, from a glance and a contemplated action of their bodies, how the 3D world will respond, a capability that is equally vital for robotic manipulation. We introduce **POINTWORLD**, a large pre-trained 3D world model that unifies state and action in a shared 3D space as 3D point flows: given one or few RGB-D images and a sequence of low-level robot action commands, POINTWORLD forecasts per-pixel displacements in 3D that respond to the given actions. By representing actions as 3D point flows instead of embodiment-specific action spaces (e.g., joint positions), this formulation directly conditions on physical geometries of robots, while seamlessly integrating learning across embodiments. To train our 3D world model, we curate a large-scale dataset spanning real and simulated robotic manipulation in open-world environments, enabled by recent advances in 3D vision and simulated environments, totaling about 2M trajectories and 500 hours across a single-arm Franka and a bimanual humanoid. Through rigorous, large-scale empirical studies of backbones, action representations, learning objectives, partial observability, data mixtures, domain transfers, and scaling, we distill design principles for large-scale 3D world modeling. With a real-time (0.1s) inference speed, POINTWORLD can be efficiently integrated in the model-predictive control (MPC) framework for manipulation. We demonstrate that a **single pre-trained checkpoint** enables a real-world Franka robot to perform rigid-body pushing, deformable and articulated object manipulation, and tool use, without requiring any demonstrations or post-training and all from a single image captured in-the-wild.

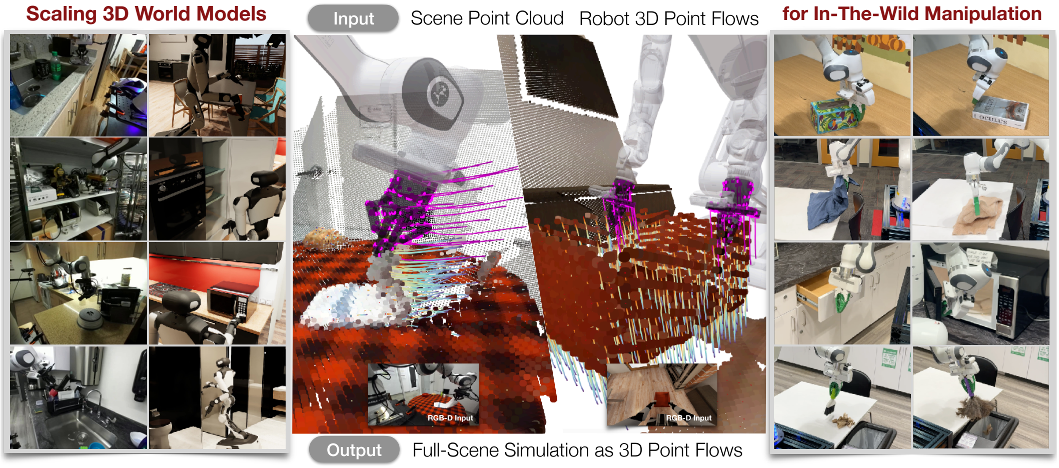

Figure 1: **POINTWORLD** is a large pre-trained 3D world model that predicts full-scene 3D point flows from a static point cloud and an embodiment-agnostic description of robot actions, represented also as 3D point flows. Project page at pointworld-iclr26.github.io.

1 INTRODUCTION

World modeling in unstructured environments is imperative for general-purpose robots: predicting how the world evolves from what the robot sees and intends to do with its body. Humans do this from a glance and a grasp, forecasting deformation, articulation, stability, and contact, revealing how much a world-modeling objective captures when conditioned on a contemplated action in 3D

(Appendix Fig. 7). Actions unfold where physics lives, in space and time: our aim is a predictive model that makes such spatially grounded, action-conditioned predictions from only perceptual inputs in open-world settings, a pinnacle goal of spatial intelligence Fei-Fei (2025).

A large body of work has studied world modeling from complementary angles. Physics-based models (Todorov et al., 2012), while capable of highly accurate predictions, face sim-to-real gaps and require curated, environment-specific modeling. Learning-based dynamics models (Li et al., 2018) address this by learning from observed interaction, yet often depend on domain-specific inductive bias (e.g., full observability, objectness priors, or material specification). In parallel, large video generative models trained at scale (Brooks et al., 2024) are capable of producing photorealistic predictions but lack explicit action conditioning and often fall short on physical consistency. See Ai et al. (2025) for a recent survey. Despite progress, a gap remains between what current models predict and what humans can foresee from visual observations in the wild and a contemplated action.

Our philosophy is unification for scaling: represent *state* and *action* in the same modality of 3D physical space. State is represented by a full-scene 3D point cloud built from RGB-D captures; actions are dense 3D point trajectories instantiated from the agent's own embodiment, typically known a priori (e.g., a robot description file), and thus forecastable over time. Under this representation, 3D world modeling equates to modeling *full-scene 3D point flow* under perturbations from a temporal sequence of robot points: given partially observed 3D scene points and those action points, predict per-point scene displacements over a horizon. While conceptually simple, this formulation ties raw sensory observation and an *embodiment-agnostic* action space in a shared representation through dynamics (what moves, how, and where) and implicitly captures objectness, articulation, and material properties, all through interaction between the robot's specific geometry (e.g., grippers, fingers) and the partially-observed scene. By modeling the *geometries of interaction* independent of goals, POINTWORLD aims to capture the single source of truth of the physical world, while naturally learning from heterogeneous embodiments, tasks, and trajectories (regardless of success or failure), akin to "next-token prediction" (Brown et al., 2020) but for interaction over 3D space and time. We term our approach **POINTWORLD**.

To provide supervision, we curate a large-scale dataset for 3D dynamics modeling, spanning hundreds of in-the-wild scenes with single-arm, bimanual, and whole-body interactions across both real and simulated domains. The dataset was built from existing robotic manipulation datasets, DROID (Khazatsky et al., 2024) and BEHAVIOR-1K (Li et al., 2024). Since accurate 3D annotations are crucial for capturing precise contact in physical interactions, significant efforts were spent to build a custom pipeline to extract 3D point flows from the real-world dataset, enabled by recent advances in metric depth estimation (Wen et al., 2025), camera pose estimation (Wang et al., 2025a), and point tracking (Karaev et al., 2025). Leveraging the dataset, we distill important design decisions for large-scale 3D dynamics learning through rigorous investigations of backbone architectures, action representations, objectives, partial observability, data mixtures, scaling laws, and domain transfers under zero-shot and finetuned settings.

To demonstrate POINTWORLD's potential for manipulation, we integrate it with a model-predictive controller (MPC) for action inference on a real robot. As POINTWORLD predicts scene dynamics jointly over short action chunks in a single forward pass at a real-time latency (0.1s), it provides a natural and efficient integration with sampling-based MPC (e.g., MPPI (Williams et al., 2017)). We show that a **single pre-trained checkpoint** enables a real-world robot to perform rigid-body pushing, deformable and articulated object manipulation, and tool use, without requiring any demonstrations or post-training and all from a single image captured in-the-wild.

Contributions. **(i)** We introduce a large pre-trained 3D world model, POINTWORLD, that unifies state and action in a shared representation of 3D point flows, and present rigorous studies of its modeling recipe. **(ii)** We curate and open-source a large-scale high-quality 3D interaction dataset used for training POINTWORLD, totaling $\sim$ 2M trajectories or $\sim$ 500 hours. **(iii)** We demonstrate a single pre-trained POINTWORLD enables a real robot to perform diverse manipulation tasks from a single in-the-wild RGB-D capture, without requiring additional demonstrations or training.

2 Related Work

A world model or dynamic model is an engineered or learned transition map from current state and action to next state. With a world model, one can leverage several model-based methods like online model-predictive control or offline policy optimization to determine the appropriate robot actions to tackle a given robotic manipulation task. A substantial body of research has explored world modeling from various complementary perspectives. Physics-based models (Todorov et al., 2012; Coumans, 2015; Tedrake & the Drake Development Team, 2019; Makoviychuk et al., 2021; Koenig & Howard, 2004; Hu et al., 2018; 2020; Huang et al., 2021; Sulsky et al., 1993) can deliver highly accurate predictions but often suffer from sim-to-real transfer issues and require carefully crafted, environment-specific designs. Learning-based dynamics models (Battaglia et al., 2016; Sanchez-Gonzalez et al., 2020; Pfaff et al., 2020; Huang et al., 2025; Ha & Schmidhuber, 2018; Hafner et al., 2018; 2019; 2020; Janner et al., 2019; Chua et al., 2018) learn directly from observed interactions, mitigating some of these challenges. However, they typically rely on strong domain-specific inductive biases—such as full observability, object-centric representations, or predefined material properties—which can hinder scalability and fall short of the generalization seen in humans. Large-scale video generative models (Lee et al., 2018; Oh et al., 2015; Lotter et al., 2016; Denton & Fergus, 2018; Yan et al., 2021; Ho et al., 2022b; Singer et al., 2022; Ho et al., 2022a; Guo et al., 2023; Yin et al., 2023b; Chen et al., 2023b; Li et al., 2025d; Chen et al., 2024; Song et al., 2025b; Hong et al., 2024; Bai et al., 2025; Bardes et al., 2024; Assran et al., 2025; Agarwal et al., 2025; Yang et al., 2023; 2025) can produce photorealistic future frames but usually lack explicit action conditioning and struggle with maintaining physical plausibility. For a recent overview of learning-based dynamics models in the context of manipulation, see Ai et al. (2025). In this work, we aim to develop a foundational 3D world model using large-scale real-world data across diverse tasks, enabling zero-shot deployment in the wild on real robots with varying embodiments.

3 Method

We formulate 3D world modeling as action-conditioned full-scene 3D point flow prediction (Section 3.1; Figure 2). We then describe how PointWorld may be used for action inference and discuss its use case in the framework of model predictive control that we explore in this work (Section 3.2).

3.1 3D World Modeling with **PointWorld**

We model environment dynamics as a neural network $\mathcal{F}_\theta : \mathbf{S} \times \mathbf{A} \to \mathbf{S}$ parametrized by θ that predicts next state given current state and robot action, where $\mathbf{S}$ and $\mathbf{A}$ denote state and action spaces. Existing approaches Ai et al. (2025) typically formulate this as a single-step update $\mathbf{s}_{t+1} = \mathcal{F}_\theta(\mathbf{s}_t, \mathbf{a}_t)$. In contrast, we adopt a multi-step (chunked) formulation for data-driven modeling Zhao et al. (2023): the model predicts future states over a horizon H in a single forward pass $\mathcal{F}_\theta^H : (\mathbf{s}_t, \mathbf{a}_{t:t+H-1}) \to \mathbf{s}_{t+1:t+H}$, which improves temporal consistency and amortizes computation. We use $H = 10$ steps and 0.1s per step.

State Representation. Building a world model requires a deliberate choice of a state space $\mathbf{S}$, with state at time t denoted by $\mathbf{s}_t \in \mathbf{S}$. In this work, we use *point flows* (referred also as particles Li et al. (2018); Zhang et al. (2025a)) as the environment state. Formally, let $\mathbf{s}_t = \{(\mathbf{p}_{t,i}, \mathbf{f}_i^S)\}_{i=1}^{N_S}$ denote the point flows at time t, consisting of N_S points with positions $\mathbf{p}_{t,i} \in \mathbb{R}^3$ and time-constant features $\mathbf{f}_i^S \in \mathbb{R}^{D_S}$ of dimension D_S for each point.

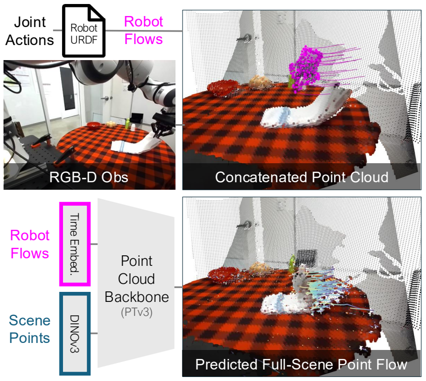

Figure 2: **Overview of PointWorld.** Given calibrated RGB-D, robot joint-space actions, and a robot description file (URDF), we convert actions to robot flows and concatenate with scene to form a single point cloud serving as an *embodiment-agnostic interaction geometry*. Scene points are featurized with a frozen DINOv3 encoder, robot points with temporal embeddings, and a point cloud backbone predicts *full-scene* 3D point flows.

Compared to alternative representations, point flows offer the following advantages for world modeling in manipulation: (i) emphasis on physical interactions between 3D geometries instead of appearance, akin to the role of physics simulators rather than renderers; (ii) accessibility from any RGB-D captures in partially observable environments Kaelbling et al. (1998) while not assuming objectness or material priors; (iii) simple and stable training via L2 losses on displacements, without permutation matching; (iv) expressiveness to capture diverse fine-grained contact dynamics. To obtain the point flows, from one or a few calibrated RGB-D views, we mask robot pixels via forward kinematics (using the URDF and joint configuration) and back-project the remaining pixels to obtain $\mathbf{p}_{t,i}$. Note that since the model takes in a static point set from the environment as input, and correspondence is preserved only within the model's forward pass (i.e., its "imagination"), no separate point tracker is required for inference, and point count may vary between forward passes.

Action Representation. To learn from heterogeneous embodiments (different kinematics, gripper geometries, and even different numbers of grippers), we again use 3D point flows. However, unlike scene point flows which are obtained from RGB-D captures, robot point flows are generated by forecasting the robot's own geometry via forward kinematics using its URDF (known a priori). This is an intentional design for ensuring "imagined actions" are *fully*, rather than partially, observable while being represented in an *embodiment-agnostic* way–crucial in cases where contact occurs in occluded regions (e.g., holding and transporting a large box with egocentric view). Specifically, given a sequence of joint configurations $\{\mathbf{q}_{t+k}\}_{k=0}^{H}$, we sample robot surface points once at time t, attach each to its corresponding link, and propagate them with forward kinematics to obtain an ordered set of N_R robot points $\{\,(\mathbf{r}_{t+k,j},\, \mathbf{f}^{R}_{t+k,j})\,\}_{j=1}^{N_R}$ at each time step $t{+}k$, where $\mathbf{r}_{t+k,j} \in \mathbb{R}^3$ denotes the position of point j at time $t{+}k$ and $\mathbf{f}^{R}_{t+k,j} \in \mathbb{R}^{D_R}$ is its time-varying feature vector of dimension D_R. We treat this collection as the action at time $t{+}k$ and denote it by $\mathbf{a}_{t+k}$. This yields an embodiment-agnostic description of *interaction geometry* over the horizon. In practice, most robot surface points never contact the scene; we therefore sample robot point flows only from the grippers (a few hundred points per gripper). See Section 5.2.

Dynamics Prediction. Given the above state-action representations, we now have a static full-scene point cloud $\mathbf{s}_t$ and a temporal sequence of robot point-flow actions $\mathbf{a}_{t:t+H-1}$ as inputs to the model. Instead of designing custom architectures, we deliberately build on top of state-of-the-art point cloud backbones Wu et al. (2023b) to distill the core principles that enable scalable, large-scale 3D world modeling. Towards this goal, we concatenate the initial scene points with the time-stacked robot points to form a single point cloud processed by the backbone. Scene points are featurized with frozen DINOv3 (Siméoni et al., 2025; Duisterhof et al., 2025) by projecting them to 2D views, while robot points are featurized with temporal embedding. The point cloud backbone processes the concatenated point cloud and outputs features for all points. A shared MLP head then predicts per-point displacements of the scene points at each step within a chunk of length H in a single forward pass. This chunked formulation delivers extremely efficient inference capable of evaluating many candidate trajectories with a real-time latency ($0.1\,\mathrm{s}$ per batched forward pass), which stands in contrast to pixel-based approaches that typically require seconds-long inference Assran et al. (2025); Agarwal et al. (2025) due to the use of diffusion objectives.

Training Objective. Appendix Fig. 8 visualizes movement weighting and uncertainty regularization on a real-world example. While the formulation lends itself to standard regression objectives, 3D world modeling introduces two distinctive challenges that require careful design: (i) due to full-scene prediction, the robot often manipulates only a small subset of the scene, so most points are static and standard L2 loss leads to very sparse training signal; (ii) real-world data is noisy, so we need to regularize the model to be robust to this noise. To address challenge (i), we adopt a weighted regression objective, reweighting each point at each timestep by a soft movement likelihood $m_{k,i} \in [0,1]$ computed from ground-truth motion so as to focus the loss on moving points. Letting $\delta_{k,i} \geq 0$ denote the norm of the ground-truth displacement vector for point i at step k, we set $m_{k,i} = \sigma\big(\kappa(\delta_{k,i} - \tau)\big)$, where σ is the logistic sigmoid, and τ and κ are non-negative displacement-threshold and temperature parameters, respectively. We then normalize these likelihoods to obtain weights $w_{k,i} = m_{k,i} / \sum_{k,i} m_{k,i}$ for each point i at step k. To address challenge (ii), we adopt aleatoric uncertainty regularization Kendall & Gal (2017); Novotny et al. (2017); Wang et al. (2025a) by predicting a scalar log-variance $s_{k,i}$ for each point i at step k and further using a Huber loss on the residual. Formally, the full training objective becomes:

$$\tfrac{1}{2} \sum_{k,i}^{H,N_S} \underbrace{w_{k,i}}_{\substack{\text{movement}\\ \text{weight}}} \Big(\underbrace{\rho_\delta\big(\hat{\mathbf{P}}_{t+k,i} - \mathbf{P}_{t+k,i}\big)}_{\substack{\text{Huber loss}\\ \text{on 3D residual}}} \underbrace{e^{-s_{k,i}}}_{\substack{\text{uncertainty}\\ \text{weight}}} + \underbrace{s_{k,i}}_{\substack{\text{uncertainty}\\ \text{reg.}}} \Big), \tag{1}$$

where ρ_δ is the elementwise Huber loss, and $\hat{\mathbf{P}}_{t+k,i}$ and $\mathbf{P}_{t+k,i}$ are the predicted and ground-truth positions of point i at step k, respectively. In practice, we also ignore the points that are deemed not visible by the 2D tracker used to provide the pseudo ground-truth (more details in Section 4).

3.2 PointWorld for Robotic Manipulation

A pre-trained PointWorld enables diverse use cases in robotics, as discussed in Section A.9. In this work, we specifically investigate whether a single pre-trained PointWorld can enable action inference in unseen, in-the-wild real-world environments from only a single RGB-D capture, **without any additional demonstrations or post-training at deployment time**. To this end, we integrate PointWorld in an MPC framework with a sampling-based planner MPPI (Williams et al., 2017) that plans a sequence of T end-effector pose targets in $\mathrm{SE}(3)$ given a cost function defined in the model's state space. Specifically, given a calibrated RGB-D capture, we first form a scene point set as described in Section 3.1, yielding an initial state $\mathbf{s}_0$. We then sample K action perturbations $\ell_{1:K}$ using a time-correlated (cubic-spline) noise distribution, which are added to a nominal end-effector trajectory. For each sampled trajectory $\mathbf{E}^{(\ell)}_{1:T}$, the corresponding robot point-flow actions $\mathbf{a}^{(\ell)}_{1:T}$ are constructed, scene flows are rolled out by PointWorld conditioned on $\mathbf{a}^{(\ell)}_{1:T}$, and a trajectory cost $J^{(\ell)}$ is accumulated. The nominal trajectory is iteratively refined by computing exponentiated weights $\omega_\ell \propto \exp(-J^{(\ell)}/\beta)$ over samples and updating the nominal as a weighted average of sampled trajectories, where β is non-negative temperature.

To define the cost function, we separate task objectives from control regularization. Let $\mathcal{I}_{\text{task}} \subseteq \{1, \dots, N_S\}$ denote a set of task-relevant scene points, with associated target positions $\{\mathbf{g}_i\}_{i \in \mathcal{I}_{\text{task}}}$. Task cost on a predicted state $\mathbf{s}_k$ at time k is $c_{\text{task}}(\mathbf{s}_k) = \frac{1}{|\mathcal{I}_{\text{task}}|} \sum_{i \in \mathcal{I}_{\text{task}}} \|\mathbf{p}_{k,i} - \mathbf{g}_i\|_2^2$. Such pointwise goal costs apply broadly across rigid, deformable, and articulated objects. Task-relevant points can be specified by either human via GUI or by VLMs Huang et al. (2024). The overall optimization problem is formulated as a global trajectory optimization:

$$\begin{aligned} \arg\min_{\mathbf{E}_{0:T}} \sum_{k=1}^{T} \Big[c_{\text{task}}(\mathbf{s}_k) \;+\; c_{\text{ctrl}}(\mathbf{E}_k) \Big] \\ \text{s.t. } \mathbf{s}_{1:T} \;=\; \mathcal{F}_\theta^T\big(\mathbf{s}_0,\; \mathbf{a}_{1:T}\big),\; \mathbf{E}_0 \;=\; \mathbf{E}_{\text{measured}}, \end{aligned} \tag{2}$$

where c_{ctrl} subsumes path length and reachability regularization, $\mathbf{E}_k$ denotes end-effector pose at step k, and $\mathbf{E}_{\text{measured}}$ is the current end-effector pose. Further details in Appendix.

4 Dataset Curation and Evaluation Protocol

Accurate, large-scale 3D data is essential for the world model in Section 3 to generalize in the wild. Apart from requiring action labels, the dataset needs to also have accurate spatial perception (i.e., high-fidelity depth), hand-eye calibration (i.e., camera extrinsics in robot base frame), and per-pixel correspondence matching amid occlusions (i.e., point tracking). While large efforts have been made for collecting diverse real-world manipulation datasets Khazatsky et al. (2024); O'Neill et al. (2024), obtaining their 3D annotations has previously been challenging. Our key observation is that recent advances in 3D vision—metric depth estimation, camera pose estimation, and dense point tracking—are maturing to provide a markerless offline pipeline that operates purely on recorded data to produce such a dataset of interest (Appendix Fig. 9, top-left).

Photorealistic simulation complements this with ground-truth supervision. Combining both, we curate a dataset of about 2M trajectories (500 hours) spanning single-arm, bimanual, and whole-body teleoperated interactions across in-the-wild real scenes and simulated home-scale environments. To our knowledge, this is the largest 3D dynamics modeling dataset, fully open-sourced. Appendix Fig. 11 shows unseen rollouts from a single pre-trained PointWorld across diverse domains.

3D Annotation for Real-World Data. We leverage DROID Khazatsky et al. (2024), a robot manipulation dataset with diverse in-the-wild interactions recorded by two external cameras and a wrist-mounted camera. Although DROID provides sensor depth and camera extrinsics, the depth often degrades in open-world environments and camera poses are inaccurate due to imperfect calibration. Frontier 3D reconstruction models such as VGGT (Wang et al., 2025a) jointly estimate depth and camera parameters from RGB images and often look visually plausible, but yield overly smoothed depth maps and camera poses that can deviate from ground-truth by tens of centimeters.

After extensive experimentation, we adopt a three-stage annotation pipeline that combines several learned models with a dedicated optimization procedure. First, we replace sensor depth with stereo-estimated depth from FoundationStereo (Wen et al., 2025), which is particularly effective at the close working distances typical of manipulation. Second, we compute camera extrinsics by refining VGGT-initialized camera poses with an optimization procedure that aligns robot depth observations to the known robot mesh. Third, given accurate depth and extrinsics, we perform per-pixel point tracking using CoTracker3 (Karaev et al., 2025). CoTracker3 is a 2D point tracker that outputs image-space correspondences and a visibility map; we lift these tracks to 3D using the refined depth and camera poses and carry over the visibility labels so that occluded points are excluded from supervision during model training. With this pipeline, we recover reliable tracked 3D point flows for over 60% of DROID (nearly 200 hours of raw human teleoperation) and obtain reconstructed point clouds that both qualitatively and quantitatively improve over both original dataset and alternative annotation methods (Appendix Fig. 9). To further assess extrinsics accuracy in the absence of ground truth in the real world, we treat the best 1% of scenes under the original dataset extrinsics (as measured by depth reprojection loss) as a proxy for the true extrinsics. Relative to this reference, our optimized extrinsics achieve a median translation and rotation error of 1.8 cm and 1.9 degrees.

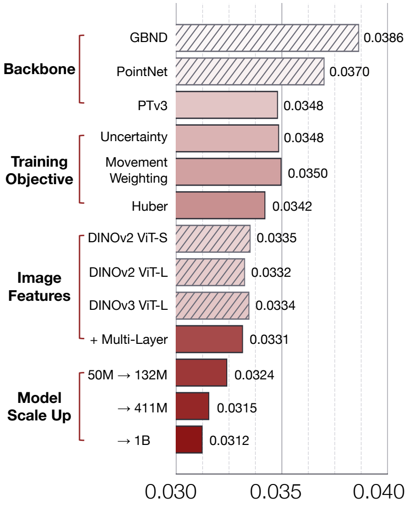

Figure 3: **Roadmap for Scaling 3D World Models**, measured by ℓ_2 error on moving scene points on DROID test set. Starting from an existing baseline Ai et al. (2025), we progressively modernize the backbone, stabilize training objectives, leverage pre-trained features, and scale model size, yielding consistent gains in accuracy. Hatched bars indicate settings that are not adopted in the final model.

Simulation (BEHAVIOR-1K). To complement real-world data, we use BEHAVIOR-1K (Li et al., 2024) (B1K), which provides about 1100 hours of teleoperated (pre-filtering) interaction in photorealistic home-scale environments with bimanual, whole-body, and mobile manipulation. We obtain ground-truth 3D point flows by leveraging known simulation state. Because the dataset focuses on long-horizon activities while POINTWORLD focuses on short-horizon interaction dynamics, we filter trajectories using privileged information accessible in simulation. We retain only trajectories with active contacts between robot and objects and those with nonzero object motion. Details in Appendix.

Model Evaluation Protocol. We evaluate predicted point flow from POINTWORLD and other baselines using a per-point, per-timestep ℓ_2 distance over the prediction horizon. Because most scene points remain static during robot interaction, we focus on the metric on moving points (ℓ_2 mover), as measured by ground-truth data and filter the full set of points using the movement likelihood introduced in Section 3. For real-world domains, we further denoise the evaluation data by training a separate expert model only on the held-out test split to flag unreliable flows via the uncertainty objective from Section 3, retaining only the top 80% of points measured by model confidence. All evaluated models are trained exclusively on the imperfect training set and are evaluated on the expert-filtered test set. Details in Appendix.

Interpretation of the Metric. This dense per-point ℓ_2 metric is highly discriminative when comparing methods and reveals systematic differences in rollout fidelity that task-level success rates often fail to expose Barreiros et al. (2025). Because all errors are measured over one-second horizons, absolute metric differences can appear modest, since even large motions move points by only a few centimeters, yet we empirically observe that small numerical differences often correspond to pronounced qualitative gains in rollout fidelity. Given the scale of the evaluation set (approx. 40,000 robot trajectories with 10,000 point flows each), standard errors for the ℓ_2 metrics are negligible ($\leq 10^{-5}$m), so we report means only.

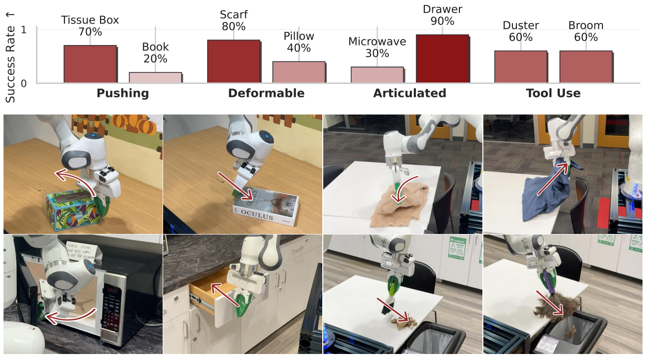

Figure 4: **Real-World Action Inference**. POINTWORLD runs zero-shot with MPC for rigid, deformable, articulated, and tool-use tasks in the wild. Success rates are on top.

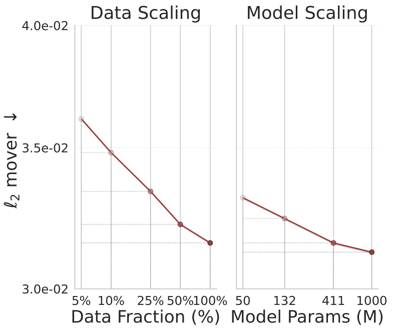

Figure 5: **Scaling Study**. Scaling POINTWORLD in either data or model size yields roughly log-linear gains in prediction accuracy.

5 EXPERIMENTS

Focusing on real-world data, we chart a roadmap of empirical lessons we learned for scaling 3D world models (Section 5.1, Figure 3) Liu et al. (2022); Wu et al. (2025). We then discuss targeted ablations along complementary design axes for POINTWORLD (Section 5.2). Using real and simulated data, we quantify in-domain, cross-domain, and held-out generalization under zero-shot and finetuned settings (Section 5.3). Finally, we study POINTWORLD for MPC-based action inference on a physical robot in the wild without extra demonstrations or finetuning (Section 5.4). All experiments isolate a single modeling choice under controlled setups against baselines unless noted. Additional ablations on chunked prediction and partial observability are in Appendix (Sections A.6 and A.7).

5.1 SCALING 3D WORLD MODELS: A ROADMAP

Modern point cloud backbone (PTv3 Wu et al. (2023b)) is effective, efficient, and scalable for 3D world modeling. Graph-based neural dynamics (GBND) models are widely used for dynamics modeling due to their relational inductive bias Ai et al. (2025). Scaling a GBND baseline to our dataset reveals two challenges (Table 1). Memory consumption grows rapidly because maintaining high-dim features for all points in a scene is expensive. Purely local message passing struggles under partial observability, since long-range effects must traverse noisy hops.

Backbone	Params	Mem.	FLOPs	Latency	ℓ_2 mov.	ℓ_2 stat.
○ GBND	1.00x	1.00x	1.00x	13.46	0.0390	0.0066
○ PointNet	1.03x	0.34x	0.04x	5.93	0.0369	0.0084
○ PointNet++	1.07x	0.67x	0.06x	327.08	0.0368	0.0073
○ SparseConv	33.31x	7.18x	1.32x	17.70	0.0396	0.0076
○ Transformer	41.06x	0.31x	3.38x	30.43	0.0339	0.0071
● PTv3-50M	49.14x	0.30x	0.34x	59.60	0.0331	0.0067
● PTv3-132M	127.22x	0.69x	1.04x	69.60	0.0324	0.0061
● PTv3-411M	398.67x	1.89x	1.90x	102.47	0.0315	0.0059
● PTv3-1B	**957.71**x	4.30x	3.57x	123.65	**0.0312**	**0.0056**

Table 1: **Backbone Comparisons.** PTv3 Wu et al. (2023b) enables massive parameter scaling while retaining similar memory and efficient inference (latency in milliseconds, PTv3-1B ≈ 0.12 s).

Motivated by these limitations, we study alternative point cloud architectures, moving from PointNet Qi et al. (2017a), PointNet++ Qi et al. (2017b), sparse convolutional nets Contributors (2022) to transformers Vaswani et al. (2017). Among these, PointTransformerV3 Wu et al. (2023b) (PTv3) delivers the strongest modeling power. Its point serialization mirrors GBND's local grouping, while U-net hierarchy enables attention over progressively coarser point sets for long-range modeling and substantial parameter growth. Table 1 shows that it scales to 957× GBND while keeping modest memory and runtime increases. These results motivate PTv3 as the default backbone.

Movement weighting, uncertainty regularization, Huber loss stabilize 3D world model learning on real-world data. Discussed in Section 3, naïve ℓ_2 loss is hard to optimize because only a fraction of points move (1–5%). Noisy real-world data exacerbates this. We therefore adopt movement weighting, uncertainty regularization, and a Huber loss on 3D residuals. Movement weighting alone over-emphasizes noisy signals, but the uncertainty head and robust loss temper the weights and reduce overfitting. Together, these changes stabilize training and improve accuracy over an unweighted ℓ_2 baseline.

Pre-trained 2D features offer critical priors and substantial gains. High-quality pretrained 3D representations remain scarce despite compelling 3D geometry. Methods such as Sonata Wu et al.

		In-Domain		Cross-Domain		Held-Out Real			From
		D → D	B → B	D → B	B → D	D → H	B → H	D+B → H	Scratch
Zero-Shot	ℓ_2 mover ↓	0.0315	0.0087	0.1460	0.0558	0.0305	0.0531	0.0300	0.0293
	ℓ_2 static ↓	0.0059	0.0010	0.0050	0.0058	0.0049	0.0057	0.0063	0.0043
Finetuned	ℓ_2 mover ↓	–	–	0.0107	0.0378	0.0271	0.0299	0.0272	0.0293
	ℓ_2 static ↓	–	–	0.0003	0.0086	0.0040	0.0046	0.0040	0.0043

Table 2: **Generalization of PointWorld across in-domain, cross-domain, held-out real environments under zero-shot and fine-tuned settings.** D denotes DROID, B denotes B1K, H denotes held-out real-world scenes. "From Scratch" denotes specialist trained on the held-out lab's data. Evaluations are done on unseen samples from the corresponding dataset. PointWorld generalizes within domains, zero-shot transfers to unseen real environments, surpasses specialists when finetuned with 20x fewer updates, and benefits from real-sim co-training.

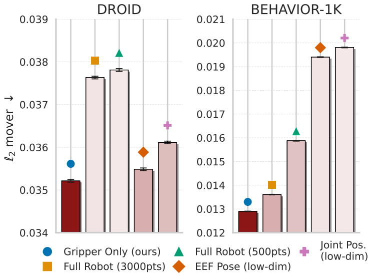

Figure 6: **Action representations.** Representing actions as point flows on grippers balances effective, efficient contact reasoning and enables positive transfer across heterogeneous embodiments.

(2025) make encouraging progress but often lag behind in fine-grained scenes. Following Wen et al. (2025); Wang et al. (2025a), we hypothesize that dense features from DINOv3 Siméoni et al. (2025) provide objectness priors without explicit segmentation. We therefore project points into calibrated cameras and attach features from multiple layers from a frozen DINOv3. This simple addition substantially boosts accuracy over the baseline.

Model size scaling is necessary to ingest large-scale world modeling data. With architecture, objective, and features in place, we expand depth and width within the same PTv3 blueprint. Aligned with scaling-law observations in vision and language modeling Kaplan et al. (2020), scaling model size from 50M to 1B parameters yields smooth, log-linear gains (Figure 5) for 3D world modeling.

Taken together, all these levers—backbone, training objective, pre-trained feature, and model scaling—yield substantial gains over the original GBND baseline Ai et al. (2025).

5.2 Ablations

Representing actions as point flows on grippers balances effective, efficient contact reasoning and enables positive transfer across heterogeneous embodiments. In PointWorld, robot actions are dense point flows over grippers with 300–500 points per gripper. We compare against four baselines: (i) whole-body point flows with the same number of points (sparser coverage), (ii) whole-body point clouds with 2000 points (similar density as ours), (iii) 6-DoF end-effector pose and gripper openness, and (iv) joint positions and gripper openness. The last two low-dim variants omit robot points, which the flow-based models concatenate with scene features. We train all models jointly on both DROID and B1K data, where DROID uses a single-arm Franka and B1K uses a bimanual humanoid. Results are in Figure 6. On B1K, representing contact spatially lets point-flow actions outperform low-dim alternatives (end-effector poses and joint positions). Sparse whole-body flows underperform gripper-only flows, likely due to insufficient resolution to capture precise contact. Dense whole-body flows help but still lag behind, as gradients must pass through inactive points and incur compute overhead. On real-world DROID, both whole-body point-flow baselines underperform low-dim baselines. A plausible explanation is that extensive robot points obscure already-sparse learning signals from noisy real-world data. Gripper-only flows address this issue and attain the best performance, underscoring their effectiveness on real-world data and their ability to obtain positive transfer across heterogeneous embodiments in both domains.

Prediction error decreases roughly log-linearly with both model size and data. Inspired by scaling laws from language and vision Kaplan et al. (2020); Chowdhery et al. (2022); Alayrac et al. (2022), we test whether PointWorld follows similar trends. On DROID, we vary model capacity (50M–1B) and data fraction (5%–100%). Each curve sweeps one axis only. In log space we observe approximately linear behavior for both axes (Figure 5), suggesting predictable gains from extra data and capacity.

5.3 Generalization and Transfer

We study PointWorld's generalization across in-domain, cross-domain, and to held-out real-world environments under zero-shot and finetuned settings. Each finetuning uses $1/20$ of the original training iterations. Results are in Table 2; Appendix Fig. 10 visualizes held-out real-world rollouts.

PointWorld generalizes within domains. We study in-domain transfer on held-out splits of DROID and B1K that are unseen during training. On B1K the model achieves sub-centimeter mover error on held-out trajectories, while DROID performance on held-out remains similar to training despite real-world variations. This indicates that PointWorld does not simply memorize training samples.

Pre-trained PointWorld can be efficiently finetuned (20x fewer updates) for both real-to-sim and sim-to-real transfer. We study cross-domain transfer by evaluating how a model pre-trained on DROID generalizes to B1K, and vice versa. Zero-shot transfer between simulation and real domains remains challenging. Yet, finetuning with only 5% of the original training steps rapidly narrows the gap to domain-specific models trained from scratch using $20\times$ more updates. The effect is symmetric: real-to-sim and sim-to-real transfers both benefit. Empirically, we observe training on real-world data provide better transfer than reverse, plausibly due to the higher scene diversity of the real-world data.

PointWorld zero-shot generalizes to unseen real-world environments, surpasses specialists if finetuned with 20x fewer updates, and benefits from real-sim co-training. To study held-out real generalization, we hold out data from the CLVR lab within DROID and evaluate how well a model pre-trained on the remaining DROID data generalizes to that lab. The held-out set is split into 90% train and 10% test. Zero-shot models never see these frames, while finetuned variants access only the 90% subset. PointWorld pre-trained on the remaining DROID data achieves on-par performance with specialists trained on the held-out lab despite changes in background, lighting, object, and possibly motion distribution. With finetuning, it quickly surpasses the specialist. We observe simulation-pretrained models do not outperform scratch baselines yet reach comparable accuracy with finetuning. Finally, a model pre-trained on combined DROID and B1K mix delivers mildly stronger zero-shot performance than DROID-only.

5.4 Model-Based Planning with PointWorld

Pre-trained on diverse interactions, we test whether PointWorld can be zero-shot deployed for manipulation on a physical robot in the wild. For evaluation, we use a Franka setup similar to DROID, mounted on a wheeled base and equipped with one RealSense D435 camera. Depth is estimated using FoundationStereo (Wen et al., 2025). For each task, we manually draw an object mask and specify target positions through a GUI tool. Each optimization rolls out 30 steps (3 autoregressive forward passes). With only the pre-trained model and a shared MPC framework, PointWorld optimizes actions for real-world tasks: non-prehensile pushing of rigid objects (tissue box, book), deformable manipulation (folding a scarf, placing a pillow), articulated manipulation (opening a microwave and closing a drawer, with revolute and prismatic joints), and tool use (sweeping with a duster or broom). Tasks and success rates are shown in Figure 4, indicating the pre-trained PointWorld captures transferable interaction dynamics, including contact reasoning under partial observability (rigid pushing), implicitly inferring articulation and deformation of objects (articulated and deformable manipulation), and object-object interactions (tool use).

6 Conclusion

We introduced PointWorld, a large pre-trained 3D world model, that predicts 3D environment dynamics given in-the-wild RGB-D capture(s) and robot actions under a shared representation of 3D point flows. To train the model, we leveraged recent advances in 3D vision and curated a large-scale dataset for action-conditioned 3D world modeling, with high-quality depth maps, camera poses, and 3D tracks. Through empirical evaluations, we rigorously studied the recipe for scaling 3D world model training, including backbone designs, action representations, learning objectives, partial observability, data mixtures, domain transfers, and scaling laws. Pre-trained on diverse data, a single PointWorld model enabled practical manipulation behaviors in the real world, including non-prehensile pushing, deformable and articulated object manipulation, and tool use.

REFERENCES

Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, pp. 1, 2004. 25

Jad Abou-Chakra et al. Particlenerf: A particle-based encoding for online neural radiance fields. *arXiv preprint arXiv:2211.04041*, 2022. 27

Jad Abou-Chakra et al. Physically embodied gaussian splatting: A realtime correctable world model for robotics. *arXiv preprint arXiv:2406.10788*, 2024. 27

Jad Abou-Chakra et al. Real-is-sim: Bridging the sim-to-real gap with a dynamic digital twin. *arXiv preprint arXiv:2504.03597*, 2025. 27

Hassan Abu Alhaija, Jose Alvarez, Maciej Bala, Tiffany Cai, Tianshi Cao, Liz Cha, Joshua Chen, Mike Chen, Francesco Ferroni, Sanja Fidler, et al. Cosmos-transfer1: Conditional world generation with adaptive multimodal control. *arXiv preprint arXiv:2503.14492*, 2025. 27

Constructions Aeronautiques, Adele Howe, Craig Knoblock, ISI Drew McDermott, Ashwin Ram, Manuela Veloso, Daniel Weld, David Wilkins Sri, Anthony Barrett, Dave Christianson, et al. Pddl—the planning domain definition language. *Technical Report, Tech. Rep.*, 1998. 27

Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, et al. Cosmos world foundation model platform for physical ai. *arXiv preprint arXiv:2501.03575*, 2025. 3, 4, 27

Pulkit Agrawal, Ashvin V Nair, Pieter Abbeel, Jitendra Malik, and Sergey Levine. Learning to poke by poking: Experiential learning of intuitive physics. *Advances in neural information processing systems*, 29, 2016. 27

Bo Ai, Stephen Tian, Haochen Shi, Yixuan Wang, Tobias Pfaff, Cheston Tan, Henrik I. Christensen, Hao Su, Jiajun Wu, and Yunzhu Li. A review of learning-based dynamics models for robotic manipulation. *Science Robotics*, 10(106), 2025. doi: 10.1126/scirobotics.adt1497. 2, 3, 6, 7, 8, 27

Jean-Baptiste Alayrac, Dustin Donato, Antoine Ortega, Jacob Menick, Aidan Clark, et al. Flamingo: a visual language model for few-shot learning. *arXiv preprint arXiv:2204.14198*, 2022. 8

Aaron D Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. Control barrier functions: Theory and applications. In *2019 18th European control conference (ECC)*, pp. 3420–3431. Ieee, 2019. 27

OpenAI: Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020. 27

Mahmoud Assran, Adrien Bardes, Xinlei Chen, Yann LeCun, et al. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025. 3, 4, 27

Yutong Bai, Danny Tran, Amir Bar, Yann LeCun, Trevor Darrell, and Jitendra Malik. Whole-body conditioned egocentric video prediction. *arXiv preprint arXiv:2506.21552*, 2025. 3

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mahmoud Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. *arXiv preprint arXiv:2404.08471*, 2024. 3, 27

Jose Barreiros, Andrew Beaulieu, Aditya Bhat, Rick Cory, Eric Cousineau, Hongkai Dai, Ching-Hsin Fang, Kunimatsu Hashimoto, Muhammad Zubair Irshad, Masha Itkina, et al. A careful examination of large behavior models for multitask dexterous manipulation. *arXiv preprint arXiv:2507.05331*, 2025. 6, 27

Peter W. Battaglia, Razvan Pascanu, Matthew Lai, Danilo Rezende, and Koray Kavukcuoglu. Interaction networks for learning about objects, relations and physics. *arXiv preprint arXiv:1612.00222*, 2016. 3, 27

Homanga Bharadhwaj, Roozbeh Mottaghi, Abhinav Gupta, and Shubham Tulsiani. Track2act: Predicting point tracks from internet videos enables generalizable robot manipulation. In *European Conference on Computer Vision*, pp. 306–324. Springer, 2024. 28

Andreas Blattmann, Willi Menapace, Yiyi He, Arash Vahdat, Robin Rombach, et al., and Agrim Gupta. Photorealistic video generation with diffusion models. *arXiv preprint arXiv:2312.06662*, 2023. 27

Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. 2024. URL https://openai.com/research/video-generation-models-as-world-simulators. 2, 27

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020. 2

Jake Bruce, Michael D Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, et al. Genie: Generative interactive environments. In *Forty-first International Conference on Machine Learning*, 2024. 27

Ryan Burgert, Yuancheng Xu, Wenqi Xian, Oliver Pilarski, Pascal Clausen, Mingming He, Li Ma, Yitong Deng, Lingxiao Li, Mohsen Mousavi, et al. Go-with-the-flow: Motion-controllable video diffusion models using real-time warped noise. *arXiv preprint arXiv:2501.08331*, 2025. 27

Boyuan Chen, Diego Marti Monso, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. *arXiv preprint arXiv:2407.01392*, 2024. 3

Haonan Chen, Yilong Niu, Kaiwen Hong, Shuijing Liu, Yixuan Wang, Yunzhu Li, and Katherine Rose Driggs-Campbell. Predicting object interactions with behavior primitives: An application in stowing tasks. In *7th Annual Conference on Robot Learning*, 2023a. 27

Jiajun Chen, Yu Li, Zhaohui Fan, Fan Zhang, Yu Qiao, and Bo Dai. Videocrafter2: Overcoming data limitations for high-quality video diffusion models. *arXiv preprint arXiv:2303.19712*, 2023b. 3, 27

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2023. 39

Cheng Chi, Zhenjia Xu, Chuer Pan, Eric Cousineau, Benjamin Burchfiel, Siyuan Feng, Russ Tedrake, and Shuran Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. *arXiv preprint arXiv:2402.10329*, 2024. 39

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. In *International Conference on Machine Learning*, 2022. 8

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *arXiv preprint arXiv:1805.12114*, 2018. 3, 27

Spconv Contributors. Spconv: Spatially sparse convolution library. https://github.com/traveller59/spconv, 2022. 7

Erwin Coumans. Bullet physics simulation. In *ACM SIGGRAPH 2015 Courses*, 2015. doi: 10.1145/2776880.2792704. 3, 27

Pieter-Tjerk de Boer, Dirk P. Kroese, Shie Mannor, and Reuven Y. Rubinstein. A tutorial on the cross-entropy method. *Annals of Operations Research*, 134(1):19–67, 2005. doi: 10.1007/s10479-005-5724-z. 27

Emily Denton and Rob Fergus. Stochastic video generation with a learned prior. In *Proceedings of the 35th International Conference on Machine Learning*, 2018. 3, 27

Karthik Dharmarajan, Wenlong Huang, Ruohan Zhang, Jiajun Wu, and Li Fei-Fei. Dream2flow: Bridging video generation and open-world manipulation with 3d object flow, 2025. Preprint. 28

Carl Doersch, Pauline Luc, Yi Yang, Dilara Gokay, Skanda Koppula, Ankush Gupta, Joseph Heyward, Ignacio Rocco, Ross Goroshin, Joao Carreira, et al. Bootstap: Bootstrapped training for tracking-any-point. In *Proceedings of the Asian Conference on Computer Vision*, pp. 3257–3274, 2024. 27

Alexandre Donze. Breach, a toolbox for verification and parameter synthesis of hybrid systems. In *Computer Aided Verification*, pp. 167–170. Springer, 2010. 27

Danny Driess, Zhiao Huang, Yunzhu Li, Russ Tedrake, and Marc Toussaint. Learning multi-object dynamics with compositional neural radiance fields. In *Conference on robot learning*, pp. 1755–1768. PMLR, 2023. 27

Maximilian Du and Shuran Song. Dynaguide: Steering diffusion polices with active dynamic guidance. *arXiv preprint arXiv:2506.13922*, 2025. 27

Yilun Du, Sherry Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Josh Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. *Advances in neural information processing systems*, 36:9156–9172, 2023. 27

Bardienus P. Duisterhof, Zhao Mandi, Yunchao Yao, Jia-Wei Liu, Jenny Seidenschwarz, Mike Zheng Shou, Deva Ramanan, Shuran Song, Stan Birchfield, Bowen Wen, and Jeffrey Ichnowski. Deformgs: Scene flow in highly deformable scenes for deformable object manipulation. *arXiv preprint arXiv:2312.00583*, 2024. 28

Bardienus P. Duisterhof, Jan Oberst, Bowen Wen, Stan Birchfield, Deva Ramanan, and Jeffrey Ichnowski. Rayst3r: Predicting novel depth maps for zero-shot object completion. *arXiv preprint arXiv:2506.05285*, 2025. 4

Ben Eisner, Harry Zhang, and David Held. Flowbot3d: Learning 3d articulation flow to manipulate articulated objects. In *Robotics: Science and Systems (RSS)*, 2022. 28

Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pp. 226–231, 1996. 31

Li Fei-Fei. From words to worlds: Spatial intelligence is ai's next frontier. https://drfeifei.substack.com/p/from-words-to-worlds-spatial-intelligence, November 2025. Substack. 2

Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction. *Advances in neural information processing systems*, 29, 2016. 27

Daniel Geng, Charles Herrmann, Junhwa Hur, Forrester Cole, Serena Zhang, Tobias Pfaff, Tatiana Lopez-Guevara, Carl Doersch, Yusuf Aytar, Michael Rubinstein, et al. Motion prompting: Controlling video generation with motion trajectories. *arXiv preprint arXiv:2412.02700*, 2024. 27

Nate Gillman, Charles Herrmann, Michael Freeman, Daksh Aggarwal, Evan Luo, Deqing Sun, and Chen Sun. Force prompting: Video generation models can learn and generalize physics-based control signals. *arXiv preprint*, 2025. arXiv preprint. 27

Jun Guo, Xiaojian Ma, Yikai Wang, Min Yang, Huaping Liu, and Qing Li. Flowdreamer: A rgb-d world model with flow-based motion representations for robot manipulation. *arXiv preprint arXiv:2505.10075*, 2025a. 28

Yanjiang Guo, Lucy Xiaoyang Shi, Jianyu Chen, and Chelsea Finn. Ctrl-world: A controllable generative world model for robot manipulation. *arXiv preprint arXiv:2510.10125*, 2025b. 27

Yuwei Guo, Zhiyu Chen, Chenyang Lei, Chengyue Gao, Fang Lu, and Yi Zhang. Animatediff: Animate your personalized text-to-image diffusion models without specific tuning. *arXiv preprint arXiv:2307.04725*, 2023. 3, 27

Irmak Guzey, Yinlong Dai, Georgy Savva, Raunaq Bhirangi, and Lerrel Pinto. Bridging the human to robot dexterity gap through object-oriented rewards. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3344–3351. IEEE, 2025. 28

David Ha and J"urgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018. 3, 27

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. *arXiv preprint arXiv:1811.04551*, 2018. 3, 27

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019. 3, 25, 27

Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020. 3, 27

Danijar Hafner, Wilson Yan, and Timothy Lillicrap. Training agents inside of scalable world models. *arXiv preprint arXiv:2509.24527*, 2025. 27

Siddhant Haldar and Lerrel Pinto. Point policy: Unifying observations and actions with key points for robot manipulation. *arXiv preprint arXiv:2502.20391*, 2025. 28

Nicklas Hansen, Hao Su, and Xiaolong Wang. Td-mpc2: Scalable, robust world models for continuous control. *arXiv preprint arXiv:2310.16828*, 2023. 27

Hao He, Yinghao Xu, Yuwei Guo, Gordon Wetzstein, Bo Dai, Hongsheng Li, and Ceyuan Yang. Cameractrl: Enabling camera control for text-to-video generation. *arXiv preprint arXiv:2404.02101*, 2024. 27

Xianglong He, Chunli Peng, Zexiang Liu, Boyang Wang, Yifan Zhang, Qi Cui, Fei Kang, Biao Jiang, Mengyin An, Yangyang Ren, et al. Matrix-game 2.0: An open-source, real-time, and streaming interactive world model. *arXiv preprint arXiv:2508.13009*, 2025a. 27

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. *arXiv preprint arXiv:2511.01177*, 2025b. 27

Jonathan Ho, William Chan, Carl Doersch, Suman Ravuri, David J. Fleet, Mohammad Norouzi, and Tim Salimans. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022a. 3, 27

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J. Fleet. Video diffusion models. *arXiv preprint arXiv:2204.03458*, 2022b. 3, 27

Yining Hong, Beide Liu, Maxine Wu, Yuanhao Zhai, Kai-Wei Chang, Linjie Li, Kevin Lin, Chung-Ching Lin, Jianfeng Wang, Zhengyuan Yang, Yingnian Wu, and Lijuan Wang. Slow-fast learning for action-driven long video generation. *arXiv preprint arXiv:2410.23277*, 2024. 3

Li Hu, Xin Gao, Peng Zhang, Ke Sun, Bang Zhang, and Liefeng Bo. Animate anyone: Consistent and controllable image-to-video synthesis for character animation. *arXiv preprint arXiv:2311.17117*, 2023. 27

Yuanming Hu, Jiancheng Liu, Andrew Spielberg, Joshua B. Tenenbaum, William T. Freeman, Jiajun Wu, Daniela Rus, and Wojciech Matusik. Chainqueen: A real-time differentiable physical simulator for soft robotics. *arXiv preprint arXiv:1810.01054*, 2018. 3

Yuanming Hu, Luke Anderson, Tzu-Mao Li, Qi Sun, Nathan Carr, Jonathan Ragan-Kelley, and Frédo Durand. Difftaichi: Differentiable programming for physical simulation. *arXiv preprint arXiv:1910.00935*, 2020. 3, 27

Suning Huang, Qianzhong Chen, Xiaohan Zhang, Jiankai Sun, and Mac Schwager. Particleformer: A 3d point cloud world model for multi-object, multi-material robotic manipulation. *arXiv preprint arXiv:2506.23126*, 2025. 3, 27

Wenlong Huang, Chen Wang, Yunzhu Li, Ruohan Zhang, and Li Fei-Fei. Rekep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation. *arXiv preprint arXiv:2409.01652*, 2024. 5, 25, 28

Zhiao Huang, Yuanming Hu, Tao Du, Siyuan Zhou, Hao Su, Joshua B. Tenenbaum, and Chuang Gan. Plasticinelab: A soft-body manipulation benchmark with differentiable physics. *arXiv preprint arXiv:2104.03311*, 2021. 3

Joel Jang, Seonghyeon Ye, Zongyu Lin, Jiannan Xiang, Johan Bjorck, Yu Fang, Fengyuan Hu, Spencer Huang, Kaushil Kundalia, Yen-Chen Lin, Loic 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. *arXiv preprint arXiv:2505.12705*, 2025. 27

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. *arXiv preprint arXiv:1906.08253*, 2019. 3, 27

Hanxiao Jiang, Hao-Yu Hsu, Kaifeng Zhang, Hsin-Ni Yu, Shenlong Wang, and Yunzhu Li. Phystwin: Physics-informed reconstruction and simulation of deformable objects from videos. *arXiv preprint arXiv:2503.17973*, 2025. 26, 27

Leslie Pack Kaelbling and Tomás Lozano-Pérez. Hierarchical task and motion planning in the now. In *2011 IEEE international conference on robotics and automation*, pp. 1470–1477. IEEE, 2011. 27

Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2):99–134, 1998. 4, 27

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeff Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020. 8

Nikita Karaev, Yuri Makarov, Jianyuan Wang, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker3: Simpler and better point tracking by pseudo-labelling real videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6013–6022, 2025. 2, 6, 27, 28, 31, 36

Matthew Kelly. An introduction to trajectory optimization: How to do your own direct collocation. *SIAM Review*, 59(4):849–904, 2017. doi: 10.1137/16M1062569. 27

Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? *Advances in neural information processing systems*, 30, 2017. 4

Bernhard Kerbl, Georgios Kopanas, Thomas LeGendre, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), 2023. doi: 10.1145/3592433. 25, 27

Alex Khazatsky, Karl Pertsch, Ashvin Nair, Ajay Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Madhav Srirama, Annie Chen, Benjamin Ellis, Patrick Fagan, Joseph Hejna, Maria Itkina, Lucille Lepert, Henry Ma, Alex Miller, Guanzhi Wu, Suneel Belkhale, Anish Dass, Jeongseok Ha, Ayush Jain, Angelica Lee, Youngwoon Lee, Nicole Memmel, Jeongho Park, Ilya Radosavovic, Rose Wang, Xuchen Zhan, Michael Black, Chi Chi, Landon Hatch, Jinqiang Lin, Zhenjia Lu, Jean Mercat, Ali Rehman, Pratyusha Sanketi, Akshara Sharma, Stacey Simpson, Quan Vuong, Pranav Walke, Blake Wulfe, Chih-Yuan Xiao, Brian Yang, Arman Yavary, Tony Zhao, Cem Agia, Parv Baijal, Alec Castro, Huan Chen, Tao Chen, Jen Jen Chung, Joshua Drake, Matthew Foster, Zhe Gao, Andres Herrera, Jeongwon Heo, Andy Hsu, Siyan Hu, Gabriel Jackson, Brian Le, Canyu Li, Hugo Lin, Donghun Ma, Avinash Maddukuri, Mihir Mirchandani, John Morton, Duc Nguyen, Brian O'Neill, Vincent Scalise, Jacob Seale, Doyeon Son, Yeming Tian, Quang Tran, Henry Wang, Andy Wu, Ho Chit Billy Xie, Juncheng Yang, Xiaolong Yin, Wenlong Zhang, Osbert Bastani, Glen Berseth, Jeannette Bohg, Ken Goldberg, Abhinav Gupta, Anchit Gupta, Dinesh Jayaraman, Joseph J. Lim, Jitendra Malik, Roberto Martín-Martín, Subramanian Ramamoorthy, Dorsa Sadigh, Shuran Song, Kuan-Ting 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 XX*, 2024. doi: 10.15607/rss.2024.xx.120. 2, 5, 28, 29

Nathan Koenig and Andrew Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2149–2154, 2004. doi: 10.1109/IROS.2004.1389727. 3, 27

Danila Kondratyuk, Uriel Singer, Willi Menapace, Reza Mahjourian, Xiaohua Zhai Sun, et al., and Agrim Gupta. Videopoet: A large language model for zero-shot video generation. *arXiv preprint arXiv:2312.14125*, 2023. 27

Klemen Kotar, Wanhee Lee, Rahul Venkatesh, Honglin Chen, Daniel Bear, Jared Watrous, Simon Kim, Khai Loong Aw, Lilian Naing Chen, Stefan Stojanov, et al. World modeling with probabilistic structure integration. *arXiv preprint arXiv:2509.09737*, 2025. 27

Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. Rma: Rapid motor adaptation for legged robots. *arXiv preprint arXiv:2107.04034*, 2021. 27

Yann LeCun. A path towards autonomous machine intelligence version 0.9. 2, 2022-06-27. *Open Review*, 62(1): 1–62, 2022. 27

Alex X. Lee, Richard Zhang, Frederik Ebert, Pieter Abbeel, Chelsea Finn, and Sergey Levine. Stochastic adversarial video prediction. *arXiv preprint arXiv:1804.01523*, 2018. 3, 27

Chengshu Li, Ruohan Zhang, Josiah Wong, Cem Gokmen, Sanjana Srivastava, Roberto Martín-Martín, Chen Wang, Gabrael Levine, Wensi Ai, Benjamin Martinez, Hang Yin, Michael Lingelbach, Minjune Hwang, Ayano Hiranaka, Sujay Garlanka, Arman Aydin, Sharon Lee, Jiankai Sun, Mona Anvari, Manasi Sharma, Dhruva Bansal, Samuel Hunter, Kyu-Young Kim, Alan Lou, Caleb R Matthews, Ivan Villa-Renteria, Jerry Huayang Tang, Claire Tang, Fei Xia, Yunzhu Li, Silvio Savarese, Hyowon Gweon, C. Karen Liu, Jiajun Wu, and Li Fei-Fei. Behavior-1k: A human-centered, embodied ai benchmark with 1,000 everyday activities and realistic simulation. *arXiv preprint arXiv:2403.09227*, 2024. 2, 6, 32

Chenhao Li, Andreas Krause, and Marco Hutter. Robotic world model: A neural network simulator for robust policy optimization in robotics. *arXiv preprint arXiv:2501.10100*, 2025a. 27

Hongyu Li, Lingfeng Sun, Yafei Hu, Duy Ta, Jennifer Barry, George Konidaris, and Jiahui Fu. Novaflow: Zero-shot manipulation via actionable flow from generated videos. *arXiv preprint arXiv:2510.08568*, 2025b. 28

Jiaqi Li, Junshu Tang, Zhiyong Xu, Longhuang Wu, Yuan Zhou, Shuai Shao, Tianbao Yu, Zhiguo Cao, and Qinglin Lu. Hunyuan-gamecraft: High-dynamic interactive game video generation with hybrid history condition. *arXiv preprint*, 2025c. arXiv preprint. 27

Shuang Li, Yihuai Gao, Dorsa Sadigh, and Shuran Song. Unified video action model. *arXiv preprint arXiv:2503.00200*, 2025d. 3

Sizhe Lester Li, Annan Zhang, Boyuan Chen, Hanna Matusik, Chao Liu, Daniela Rus, and Vincent Sitzmann. Controlling diverse robots by inferring jacobian fields with deep networks. *Nature*, pp. 1–7, 2025e. 25

Yunzhu Li, Jiajun Wu, Russ Tedrake, Joshua B Tenenbaum, and Antonio Torralba. Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids. *arXiv preprint arXiv:1810.01566*, 2018. 2, 3, 27

Zizhang Li, Hong-Xing Yu, Wei Liu, Yin Yang, Charles Herrmann, Gordon Wetzstein, and Jiajun Wu. Wonderplay: Dynamic 3d scene generation from a single image and actions. *arXiv preprint arXiv:2505.18151*, 2025f. 27

Hao Liu, Wilson Yan, Matei Zaharia, and Pieter Abbeel. World model on million-length video and language with blockwise ringattention. *arXiv preprint arXiv:2402.08268*, 2024. 27

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11976–11986, 2022. 7

Alberta Longhini et al. Edo-net: Learning elastic properties of deformable objects from graph dynamics. *arXiv preprint arXiv:2209.08996*, 2022. 27

William Lotter, Gabriel Kreiman, and David Cox. Deep predictive coding networks for video prediction and unsupervised learning. *arXiv preprint arXiv:1605.08104*, 2016. 3, 27

Tomas Lozano-Perez. Spatial planning: A configuration space approach. *IEEE transactions on computers*, 32 (02):108–120, 1983. 27

Guanxing Lu, Baoxiong Jia, Puhao Li, et al. Gwm: Towards scalable gaussian world models for robotic manipulation. *arXiv preprint arXiv:2508.17600*, 2025. 27

Viktor Makoviychuk, Ilya Wawrzyniak, Sergii Gupta, Anurag Narang, Maciej Kayastha, Yashraj Henry, Guanyang Li, Igor Ling, Andreas Merentitis, Alexey Makoviychuk, Kaichun Tsai, Gabe State, Dieter Fox, and Ankur Handa. Isaac gym: High performance gpu based physics simulation for robot learning. *arXiv preprint arXiv:2108.10470*, 2021. 3, 27

Jiayuan Mao, Chuang Gan, Pushmeet Kohli, Joshua B Tenenbaum, and Jiajun Wu. The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision. *arXiv preprint arXiv:1904.12584*, 2019. 27

Tobia Marcucci, Jack Umenberger, Pablo Parrilo, and Russ Tedrake. Shortest paths in graphs of convex sets. *SIAM Journal on Optimization*, 34(1):507–532, 2024. 27

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *arXiv preprint arXiv:2003.08934*, 2020. 25, 27

Igor Mordatch and Emo Todorov. Combining the benefits of function approximation and trajectory optimization. In *Robotics: Science and Systems*, volume 4, pp. 23, 2014. 27

Muyao Niu, Xiaodong Cun, Xintao Wang, Yong Zhang, Ying Shan, and Yinqiang Zheng. Mofa-video: Controllable image animation via generative motion field adaptions in frozen image-to-video diffusion model. *arXiv preprint arXiv:2405.20222*, 2024. 27

David Novotny, Diane Larlus, and Andrea Vedaldi. Learning 3d object categories by looking around them. In *Proceedings of the IEEE international conference on computer vision*, pp. 5218–5227, 2017. 4

Junhyuk Oh, Xiaoxiao Guo, Honglak Lee, Richard L. Lewis, and Satinder Singh. Action-conditional video prediction using deep networks in atari games. In *Advances in Neural Information Processing Systems*, 2015. 3, 27

Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6892–6903. IEEE, 2024. 5

Shivansh Patel, Shraddhaa Mohan, Hanlin Mai, Unnat Jain, Svetlana Lazebnik, and Yunzhu Li. Robotic manipulation by imitating generated videos without physical demonstrations. *arXiv preprint arXiv:2507.00990*, 2025a. 28

Shivansh Patel, Xinchen Yin, Wenlong Huang, Shubham Garg, Hooshang Nayyeri, Li Fei-Fei, Svetlana Lazebnik, and Yunzhu Li. A real-to-sim-to-real approach to robotic manipulation with vlm-generated iterative keypoint rewards. *arXiv preprint arXiv:2502.08643*, 2025b. 28

Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pp. 2778–2787. PMLR, 2017. 27

Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 3803–3810. IEEE, 2018. 27

Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter W. Battaglia. Learning mesh-based simulation with graph networks. *arXiv preprint arXiv:2010.03409*, 2020. 3, 27

Joelle Pineau, Geoffrey Gordon, and Sebastian Thrun. Point-based value iteration: An anytime algorithm for pomdps. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI)*, 2003. 27

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 652–660, 2017a. 7

Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017b. 7

Han Qi, Haocheng Yin, Aris Zhu, Yilun Du, and Heng Yang. Strengthening generative robot policies through predictive world modeling. *arXiv preprint arXiv:2502.00622*, 2025. 27

Haitao Qiu, Xiaoyu Zhou, Liyang Zhang, Jiaqi Yang, Kailun Zhou, et al. Physgen3d: Crafting a miniature interactive world from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025. doi: 10.1109/cvpr52734.2025.00579. 27

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. *arXiv preprint arXiv:2408.00714*, 2024. URL `https://arxiv.org/abs/2408.00714`. 39

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter W. Battaglia. Learning to simulate complex physics with graph networks. *arXiv preprint arXiv:2002.09405*, 2020. 3, 27

Daniel Seita, Yufei Wang, Sarthak J Shetty, Edward Yao Li, Zackory Erickson, and David Held. Toolflownet: Robotic manipulation with tools via predicting tool flow from point clouds. *arXiv preprint arXiv:2211.09006*, 2022. 28

Ramanan Sekar, Oleh Rybkin, Kostas Daniilidis, Pieter Abbeel, Danijar Hafner, and Deepak Pathak. Planning to explore via self-supervised world models. In *International conference on machine learning*, pp. 8583–8592. PMLR, 2020. 27

Junyao Shi, Joshua Smith, Jianing Qian, and Dinesh Jayaraman. Points2reward: Robotic manipulation rewards from just one video. 2025. 28

Xiaoyu Shi, Zhaoyang Huang, Fu-Yun Wang, Weikang Bian, Dasong Li, Yi Zhang, Manyuan Zhang, Ka Chun Cheung, Simon See, Hongwei Qin, et al. Motion-i2v: Consistent and controllable image-to-video generation with explicit motion modeling. In *ACM SIGGRAPH Conference Papers*, 2024. 27

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. Dinov3. *arXiv preprint arXiv:2508.10104*, 2025. 4, 8

Uriel Singer, Adam Polyak, Timothy Hayes, Sagie Benaim, Oron Gafni, Oron Ashual, Yuval Atzmon, Tomer Shalev, Devi Parikh, Yaniv Taigman, Ronen Banner, and Eliya Nachmani. Make-a-video: Text-to-video generation without text-video data. *arXiv preprint arXiv:2209.14792*, 2022. 3, 27

Chonghyuk Song, Michal Stary, Boyuan Chen, George Kopanas, and Vincent Sitzmann. Generative view stitching. *arXiv preprint arXiv:2510.24718*, 2025a. 27

Kiwhan Song, Boyuan Chen, and Max Simchowitz. History-guided video diffusion. *arXiv preprint arXiv:2502.06764*, 2025b. 3

D. Sulsky, Z. Chen, and H. L. Schreyer. A particle method for history-dependent materials. Technical report, U.S. Department of Energy, OSTI, 1993. 3, 27

Zhanyi Sun and Shuran Song. Latent policy barrier: Learning robust visuomotor policies by staying in-distribution. *arXiv preprint arXiv:2508.05941*, 2025. 27

Gemini Robotics Team, Coline Devin, Yilun Du, Debidatta Dwibedi, Ruiqi Gao, Abhishek Jindal, Thomas Kipf, Sean Kirmani, Fangchen Liu, Anirudha Majumdar, et al. Evaluating gemini robotics policies in a veo world simulator. *arXiv preprint arXiv:2512.10675*, 2025. 27

Russ Tedrake and the Drake Development Team. Drake: A planning, control, and analysis toolbox for nonlinear systems. Technical Report, MIT CSAIL, `https://drake.mit.edu`, 2019. 3, 27

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5026–5033, 2012. doi: 10.1109/IROS.2012.6386109. 2, 3, 27

Marc Toussaint. Logic-geometric programming: An optimization-based approach to combined task and motion planning. In *IJCAI*, pp. 1930–1936, 2015. 27

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 7

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025a. 2, 4, 5, 8, 28, 29

Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689*, 2021. 27

Qineng Wang, Wenlong Huang, Yu Zhou, Hang Yin, Tianwei Bao, Jianwen Lyu, Weiyu Liu, Ruohan Zhang, Jiajun Wu, Li Fei-Fei, et al. Enact: Evaluating embodied cognition with world modeling of egocentric interaction. *arXiv preprint arXiv:2511.20937*, 2025b. 27

Shengjie Wang, Jiacheng You, Yihang Hu, Jiongye Li, and Yang Gao. Skil: Semantic keypoint imitation learning for generalizable data-efficient manipulation. In *Robotics: Science and Systems (RSS)*, 2025c. 28

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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12713–12724, 2025d. 27

Zhouxia Wang, Ziyang Yuan, Xintao Wang, Yaowei Li, Tianshui Chen, Menghan Xia, Ping Li, Ping Luo, and Ying Shan. Motionctrl: A unified and flexible motion controller for video generation. In *ACM SIGGRAPH Conference Papers*, 2024. 27

Bowen Wen, Matthew Trepte, Joseph Aribido, Jan Kautz, Orazio Gallo, and Stan Birchfield. Foundationstereo: Zero-shot stereo matching. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 5249–5260, 2025. 2, 6, 8, 9, 22, 28, 29, 39

Chuan Wen, Xingyu Lin, John So, Kai Chen, Qi Dou, Yang Gao, and Pieter Abbeel. Any-point trajectory modeling for policy learning. *arXiv preprint arXiv:2401.00025*, 2023. 28

Thomas Weng, Sujay Bajracharya, Yufei Wang, Khush Agrawal, and David Held. Fabricflownet: Bimanual cloth manipulation with a flow-based policy. *arXiv preprint arXiv:2111.05623*, 2021. 28

William F Whitney, Jacob Varley, Deepali Jain, Krzysztof Choromanski, Sumeet Singh, and Vikas Sindhwani. Modeling the real world with high-density visual particle dynamics. *arXiv preprint arXiv:2406.19800*, 2024. 27

Grady Williams, Andrew Aldrich, and Evangelos A. Theodorou. Model predictive path integral control: From theory to parallel computation. *Journal of Guidance, Control, and Dynamics*, 40(2):344–357, 2017. doi: 10.2514/1.g001921. 2, 5, 25, 27

Philipp Wu, Alejandro Escontrela, Danijar Hafner, Pieter Abbeel, and Ken Goldberg. Daydreamer: World models for physical robot learning. In *Conference on robot learning*, pp. 2226–2240. PMLR, 2023a. 27

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*, 2024. 27

Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. Point transformer v3: Simpler, faster, stronger. *arXiv preprint arXiv:2312.10035*, 2023b. 4, 7

Xiaoyang Wu, Daniel DeTone, Duncan Frost, Tianwei Shen, Chris Xie, Nan Yang, Jakob Engel, Richard Newcombe, Hengshuang Zhao, and Julian Straub. Sonata: Self-supervised learning of reliable point representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025. 7

Hongchi Xia, Zhi-Hao Lin, Wei-Chiu Ma, and Shenlong Wang. Video2game: Real-time interactive realistic and browser-compatible environment from a single video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4578–4588, 2024. 27

Hongchi Xia, Entong Su, Marius Memmel, Arhan Jain, Raymond Yu, Numfor Mbiziwo-Tiapo, Ali Farhadi, Abhishek Gupta, Shenlong Wang, and Wei-Chiu Ma. Drawer: Digital reconstruction and articulation with environment realism. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 21771–21782, 2025. 27

Guangyuan Xiang, Juntong Wang, Linhan Qiu, Jing Yang, Zhihao Liu, Jiahui Zhang, Zhen Li, et al. Physgaussian: Physics-integrated 3d gaussians for generative dynamics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. doi: 10.1109/cvpr52733.2024.00420. 27

Yuxi Xiao, Qianqian Wang, Shangzhan Zhang, Nan Xue, Sida Peng, Yujun Shen, and Xiaowei Zhou. Spatialtracker: Tracking any 2d pixels in 3d space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20406–20417, 2024. 27

Eric Xing, Mingkai Deng, Jinyu Hou, and Zhiting Hu. Critiques of world models. *arXiv preprint arXiv:2507.05169*, 2025. 27

Jinbo Xing, Hanyuan Liu, Menghan Xia, Yong Zhang, Xintao Wang, Ying Shan, and Tien-Tsin Wong. Tooncrafter: Generative cartoon interpolation. *ACM Transactions on Graphics*, 2024. 27

Mengda Xu, Zhenjia Xu, Yinghao Xu, Cheng Chi, Gordon Wetzstein, Manuela Veloso, and Shuran Song. Flow as the cross-domain manipulation interface. *arXiv preprint arXiv:2407.15208*, 2024a. 28

Xiaomeng Xu, Huy Ha, and Shuran Song. Dynamics-guided diffusion model for robot manipulator design. *CoRR*, 2024b. 27

Wilson Yan, Yunzhi Zhang, Pieter Abbeel, and Aravind Srinivas. Videogpt: Video generation using vq-vae and transformers. *arXiv preprint arXiv:2104.10157*, 2021. 3, 27

Sherry Yang, Yilun Du, Kamyar Ghasemipour, Jonathan Tompson, Leslie Kaelbling, Dale Schuurmans, and Pieter Abbeel. Learning interactive real-world simulators. *arXiv preprint arXiv:2310.06114*, 2023. 3, 27

Sherry Yang et al. Evaluating robot policies in a world model. *arXiv preprint arXiv:2506.00613*, 2025. 3, 27

Brent Yi, Chung Min Kim, Justin Kerr, Gina Wu, Rebecca Feng, Anthony Zhang, Jonas Kulhanek, Hongsuk Choi, Yi Ma, Matthew Tancik, and Angjoo Kanazawa. Viser: Imperative, web-based 3d visualization in python, 2025. URL `https://arxiv.org/abs/2507.22885`. 23

Shengming Yin, Chenfei Wu, Jian Liang, Jie Shi, Houqiang Li, Gong Ming, and Nan Duan. Dragnuwa: Fine-grained control in video generation by integrating text, image, and trajectory. *arXiv preprint arXiv:2308.08089*, 2023a. 27

Yichao Yin, Lei Qi, Hong Zhu, Jiakai Qin, Salman Khan, Luc Van Gool, Li Liu, and Philip Torr. Dynamicrafter: Animating open-domain images with video diffusion priors. *arXiv preprint arXiv:2310.12190*, 2023b. 3, 27

Zhao-Heng Yin, Sherry Yang, and Pieter Abbeel. Object-centric 3d motion field for robot learning from human videos. *arXiv preprint arXiv:2506.04227*, 2025. 28

Kevin Zakka, Baruch Tabanpour, Qiayuan Liao, Mustafa Haiderbhai, Samuel Holt, Jing Yuan Luo, Arthur Allshire, Erik Frey, Koushil Sreenath, Lueder A Kahrs, et al. Mujoco playground. *arXiv preprint arXiv:2502.08844*, 2025. 39

Jie Zhang, Yilun Du, et al. Learning to act from actionless videos through dense correspondences. *arXiv preprint arXiv:2310.08576*, 2023. 27, 28

Kaifeng Zhang, Baoyu Li, Kris Hauser, and Yunzhu Li. Adaptigraph: Material-adaptive graph-based neural dynamics for robotic manipulation. *arXiv preprint arXiv:2407.07889*, 2024a. 27

Kaifeng Zhang, Baoyu Li, Kris Hauser, and Yunzhu Li. Particle-grid neural dynamics for learning deformable object models from rgb-d videos. *arXiv preprint arXiv:2506.15680*, 2025a. 3, 27

Kaifeng Zhang, Shuo Sha, Hanxiao Jiang, Matthew Loper, Hyunjong Song, Guangyan Cai, Zhuo Xu, Xiaochen Hu, Changxi Zheng, and Yunzhu Li. Real-to-sim robot policy evaluation with gaussian splatting simulation of soft-body interactions. *arXiv preprint arXiv:2511.04665*, 2025b. 27

Tianyuan Zhang, Hong-Xing Yu, Rundi Wu, Brandon Y. Feng, Changxi Zheng, Noah Snavely, Jiajun Wu, and William T. Freeman. Physdreamer: Physics-based interaction with 3d objects via video generation. *arXiv preprint arXiv:2404.13026*, 2024b. 27

Zhenghao Zhang, Junchao Liao, Menghao Li, Zuozhuo Dai, Bingxue Qiu, Siyu Zhu, Long Qin, and Weizhi Wang. Tora: Trajectory-oriented diffusion transformer for video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025c. 27

Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023. 3

Haoyu Zhen, Qiao Sun, Hongxin Zhang, Junyan Li, Siyuan Zhou, Yilun Du, and Chuang Gan. Tesseract: learning 4d embodied world models. *arXiv preprint arXiv:2504.20995*, 2025. 27

Guangcong Zheng, Teng Li, Rui Jiang, Yehao Lu, Tao Wu, and Xi Li. Cami2v: Camera-controlled image-to-video diffusion model. *arXiv preprint arXiv:2410.15957*, 2024. 27

Ruijie Zheng, Jing Wang, Scott Reed, Johan Bjorck, Yu Fang, Fengyuan Hu, Joel Jang, Kaushil Kundalia, Zongyu Lin, Loic Magne, Avnish Narayan, You Liang Tan, Guanzhi Wang, Qi Wang, Jiannan Xiang, Yinzhen Xu, Seonghyeon Ye, Jan Kautz, Furong Huang, Yuke Zhu, and Linxi Fan. Flare: Robot learning with implicit world modeling. *arXiv preprint arXiv:2505.15659*, 2025. 27

Jensen Zhou, Hang Gao, Vikram Voleti, Aaryaman Vasishta, Chun-Han Yao, Mark Boss, Philip Torr, Christian Rupprecht, and Varun Jampani. Stable virtual camera: Generative view synthesis with diffusion models. *arXiv preprint arXiv:2503.14489*, 2025a. 27

Siyuan Zhou, Yilun Du, Yuncong Yang, Lei Han, Peihao Chen, Dit-Yan Yeung, and Chuang Gan. Learning 3d persistent embodied world models. *arXiv preprint arXiv:2505.05495*, 2025b. 27

Yifeng Zhu, Abhishek Joshi, Peter Stone, and Yuke Zhu. Viola: Imitation learning for vision-based manipulation with object proposal priors. *arXiv preprint arXiv:2210.11339*, 2022. doi: 10.48550/arXiv.2210.11339. 39

A APPENDIX

APPENDIX CONTENTS

A.1 Rich Supervision of 3D World Modeling

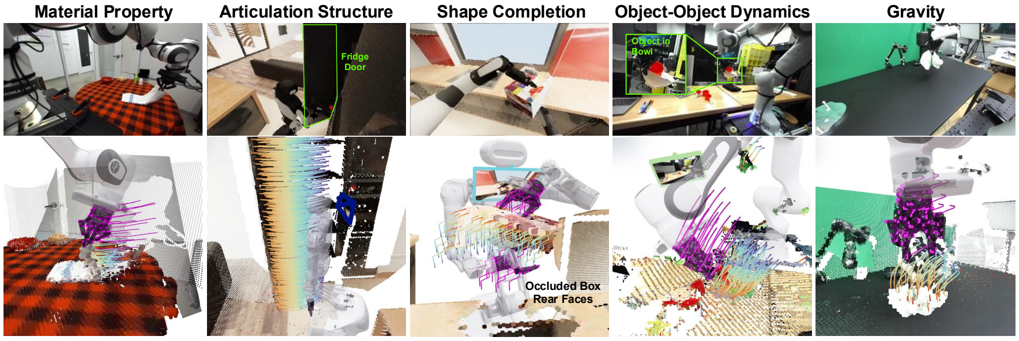

Figure 7: **Rich Supervision of 3D World Modeling for Physical Interactions**, when conditioned on 3D robot point flows and partial observable RGB-D. The 3D world modeling objective enjoys dense pixel-level supervision while encoding a wide range of capabilities central to robotic manipulation. To predict full-scene evolution, the model must implicitly segment objects, infer material or articulation, perform shape completion for contact reasoning, propagate robot-object interactions for object-object dynamics, and account for gravity—all in a single forward pass of the learned model.

A.2 Movement Weighting and Uncertainty Regularization

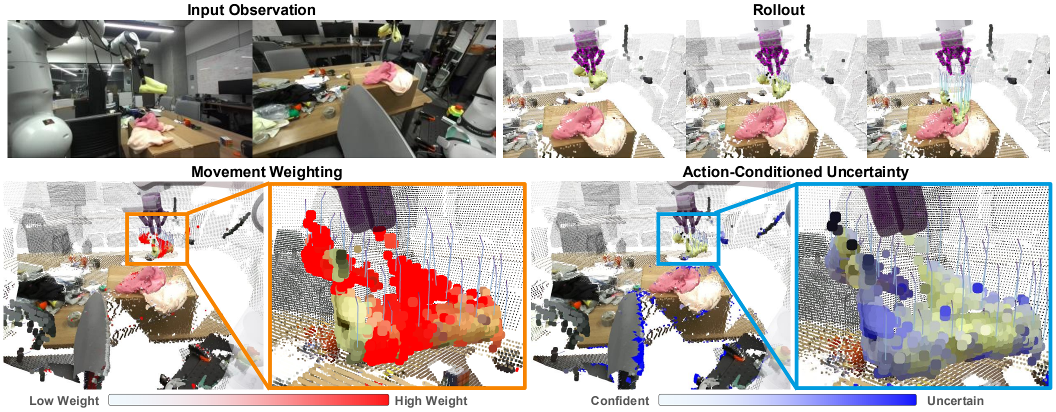

Figure 8: **Movement Weighting and Uncertainty Regularization**, where the robot releases and drops a yellow cloth. (**Bottom Left**) The movement weighting, used in the training objective, effectively biases the training towards scene points that are moving at each timestep, computed with the ground-truth flows. (**Bottom Right**) The uncertainty value, predicted by the model without any ground-truth, regularizes training to prevent overfitting to points that have unreliable ground-truth. Intriguingly, we observe that it also emerges to capture action-conditioned uncertainty arising from the object's physical properties (e.g., larger variability along the edge of the cloth).

A.3 3D Annotation Quality and Comparisons

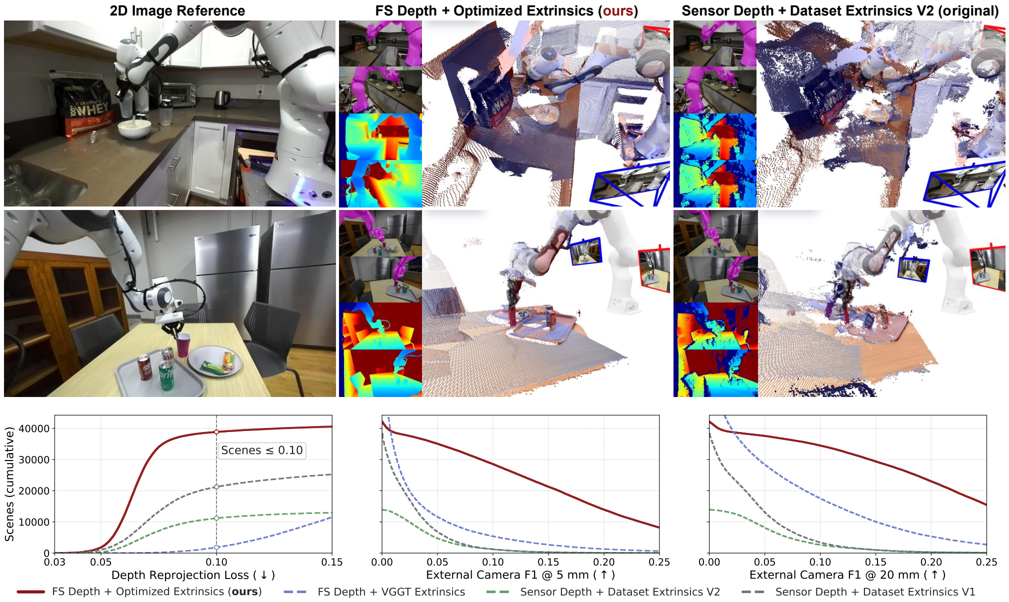

Figure 9: **3D Annotation Quality and Comparisons.** FS denotes FoundationStereo (Wen et al., 2025); Dataset Extrinsics V1/V2 are the two DROID extrinsics releases. **(Top)** Our pipeline yields higher-quality depth and camera poses than DROID, producing more accurate robot masks and better-aligned point clouds. **(Bottom)** We report depth reprojection loss (robot surface) and point-cloud F1 alignment. Existing models (FS, VGGT) are insufficient, and V2 improves over V1 by filtering misaligned scenes but at much lower counts. Our pipeline retains far more scenes below the 0.10 depth-loss criterion and dominates all metrics.

A.4 Zero-Shot Generalization to Held-Out Real-World Scenes

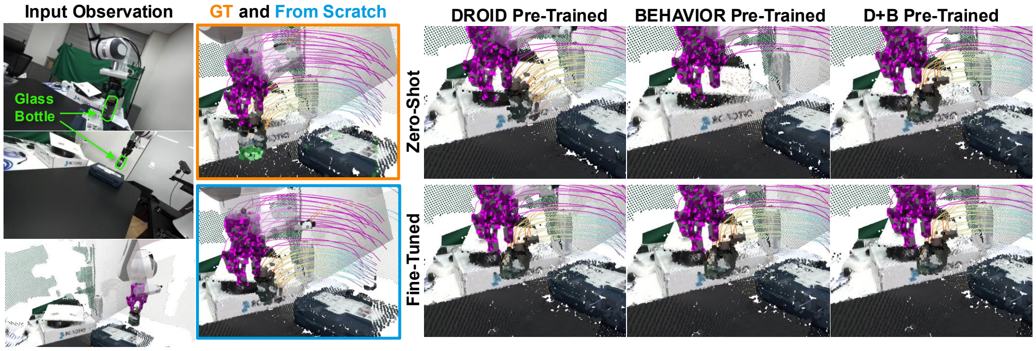

Figure 10: **Zero-Shot and Finetuned Generalization to Held-Out Real-World Scenes**, where the robot transports a reflective glass bottle. PointWorld pre-trained on DROID or jointly on DROID and BEHAVIOR (D+B) can zero-shot generalize to unseen environments and motion from a held-out DROID lab scene, closing the gap to the specialist trained on that lab's data. PointWorld pre-trained only on simulation data fail to generalize zero-shot. Further finetuning yields more accurate trajectories of grasped objects.

A.5 Unseen Rollouts from a Single Pre-trained PointWorld

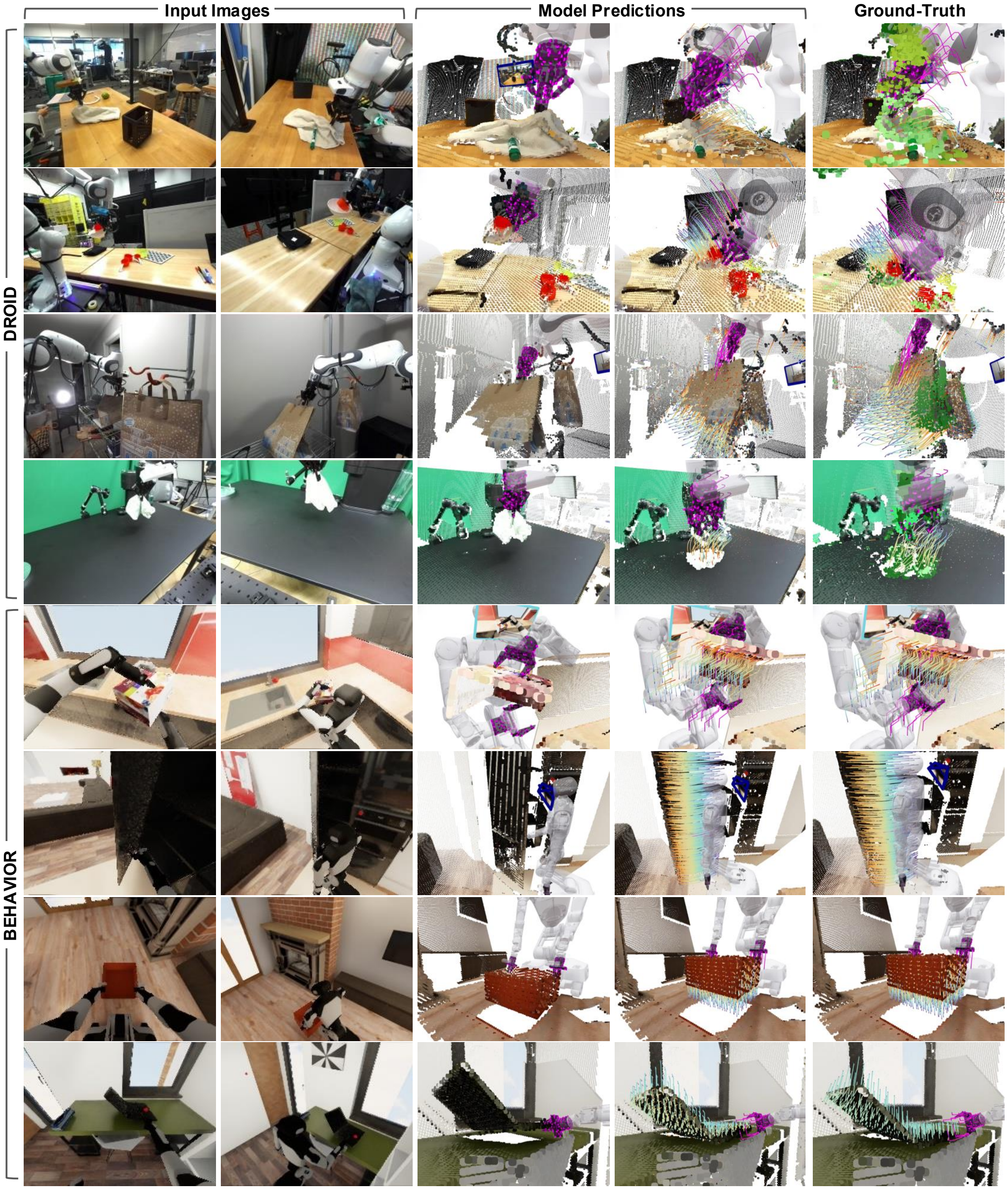

Figure 11: **Unseen rollouts from a single pre-trained PointWorld across diverse domains**, visualized with Viser Yi et al. (2025). Given RGB-D captures, PointWorld predicts 10-step point flows conditioned on robot flows. We show first prediction, last prediction, and last ground-truth. Green points in GT mark regions occluded during 2D point tracking, for which we observe model predictions are often more accurate because these points are not being supervised in model training. Due to grid downsampling (1.5cm) we apply to all point clouds, all 3D visualization is upsampled to image resolution for visual clarity via nearest neighbors. Interactive visualization provided in the supplementary material.

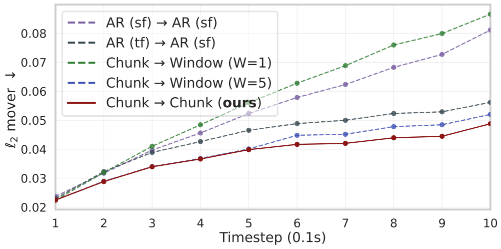

Figure 12: **Ablation on Chunked Prediction**, where we study different rollout strategies in training and testing. Chunked rollouts at both training and inference time lead to significantly less drift than other baselines while amortizing compute with only a single forward pass of the model.

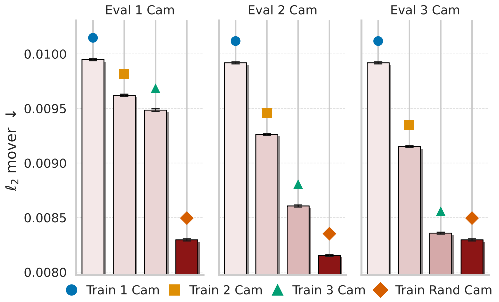

Figure 13: **Ablation on Partial Observability**, where we train variants of POINTWORLD with varying number of cameras and evaluate them on all settings at test time. POINTWORLD is robust to different levels of partial observability and benefits from additional cameras in both training and inference. Training with randomized camera counts yields the best performance across all test settings.

A.6 Chunked Prediction

We ablate 10-step chunked prediction against two autoregressive baselines: (i) teacher-forcing (GT input each step) and (ii) self-feeding with 10k warmup steps, plus sliding-window inference (W=1, 5) using the same chunked model. Teacher-forcing outperforms self-feeding when training and inference strategies are aligned. Evaluating a chunk-trained model with W=1 (equivalent to self-feeding) incurs the strongest performance degradation; W=5 recovers some accuracy but degrades after the trained window. Matching chunked prediction in training and testing over the full horizon minimizes drift while amortizing compute with a single forward pass (vs. 2–10 autoregressive), making chunking both more accurate and more efficient.

A.7 Partial Observability

We report how varying the number of cameras at train and test time affects error and robustness. We train four variants that differ only in camera count for RGB-D observations: one, two, three, or a random draw of up to three cameras. We then evaluate all models on three settings with up to three cameras. Error on moving points stays sub-centimeter with negligible standard errors, but using more cameras at train time consistently reduces error at test time. Models trained with fixed camera count perform better when more cameras are available at inference. The random-view model is most robust across all test camera counts, suggesting that exposure to varied observability helps the model infer objectness and physical properties under partial observability at inference time.

A.8 Extended Discussions on Limitations

Static Initial State. The model takes as input RGB-D point cloud together with a finite-horizon sequence of robot actions and predicts how points will move in response. Because no prior frames or velocities are provided, this formulation assumes that the world is static at the observation instant. Supporting fully dynamic initial conditions would require augmenting the input with externally tracked trajectories or recurrent state, which we leave as future work.

Reward/Cost Specification for Action Inference. In this work, we explore PointWorld's use case for action inference in manipulation by integrating it with a sampling-based planner, MPPI Williams et al. (2017), which requires an explicit specification of reward/cost functions in the same state-action space of 3D point flows. For the scope of this work, we restrict ourselves to manual specification (e.g., moving a subset of points to target locations). Future work may consider automatically specifying (single or multi-stage) reward via vision-language models Huang et al. (2024), or inferring reward from demonstrations using inverse reinforcement learning Abbeel & Ng (2004), while keeping PointWorld as the dynamics function. In addition to planning, action inference can also be alternatively performed by learning a parameterized policy by interacting with the model as the environment via reinforcement learning Hafner et al. (2019).

Fine-Scale Objects and Calibration Noise. Thin or very small objects (e.g., pens or cables) are challenging to annotate accurately in 3D: modest depth or extrinsic errors can be comparable to the object thickness and lead to ambiguous separation between robot and scene points during ground-truth annotation. In such scenarios, mis-registrations in the ground-truth flows propagate into training and can cause the model to confuse overlapping motions between the gripper and nearby scene points. Improved calibration and depth estimation could strengthen supervision for these fine-grained interactions.

Correlation vs. Causation. Given an observed context frame and a sequence of robot actions, PointWorld is trained to predict the subsequent sequence of scene states. As such, it primarily captures correlations present in the training distribution between robot actions, robot motion, and observed scene evolution. In settings where exogenous factors (such as other agents or environment changes not controlled by the robot) also influence the future, these influences are entangled with the robot-induced effects in the data and are not disentangled as separate causal mechanisms. Our experiments therefore evaluate predictive fidelity and planning performance under the observed action-conditioned distribution, rather than claiming to recover the underlying causal structure of the environment.

Lack of Photometric Dynamics. PointWorld only outputs displacements of 3D points captured from RGB-D inputs, which focus on geometry and physical interactions rather than appearance. While often visually plausible when rendered as point clouds, it is insufficient if one desires to reason about photometric changes of the environment caused by the robot actions, such as lights or screens turning on and off. Combining PointWorld with appearance models that predict emitted radiance, such as those from Gaussian Splatting (Kerbl et al., 2023) or Neural Radiance Fields (Mildenhall et al., 2020), may be necessary for tasks where such photometric dynamics are critical.

Rigid-Body Robot Assumption. Robot embodiment is represented as a kinematic tree of rigid links, and we propagate a fixed set of robot surface points by forward kinematics. This ignores deformations of soft, tendon-driven, or compliant structures (e.g., fin-ray grippers) and non-rigid effects of the robot body. As a result, PointWorld reasons only about how the scene moves in response to a forecasted robot geometry, rather than how contact may reshape the robot itself. Extending the representation to include deformable links Li et al. (2025e) would enable reasoning about how contact deforms the robot body and how those deformations, in turn, affect contact geometry.

Actuation and Tracking Assumptions. Our formulation treats the robot trajectory as a known, *fully realized* sequence of joint configurations and uses forward kinematics to construct robot flows. As a result, PointWorld effectively models "what the environment does if the robot body follows this path," rather than "whether and how the robot will actually realize this path" under a particular controller, actuation limits, or contact-induced tracking errors. This quasi-static, kinematic view of the robot action representation can break down for underactuated or compliant joints (e.g., tendon-driven or compliant hand fingers), or when strong contacts, payloads, or controller changes induce large tracking errors. Extending the method to jointly reason about robot and scene dynamics is an important avenue for future work.

Lack of Explicit Physics Priors. The current formulation is purely data-driven and does not incorporate explicit physics priors such as Newtonian mechanics or conservation-law constraints, to provide a focused study on scaling 3D world models without priors of objectness and of their material and physical properties. Despite this, we observe that POINTWORLD captures many aspects of rigid, articulated, and deformable behavior from data alone. Incorporating physics-informed regularization or hybrid simulators (Jiang et al., 2025) could further improve generalization and extrapolation, particularly in regimes that in-domain interaction data can be collected for accurate scene/object reconstruction, not only for their geometries but also for their physics parameters.

A.9 Extended Related Work

World Modeling. World models (Ha & Schmidhuber, 2018) are predictive models that simulate future states given current state and action, categorized often by their state-action representations. Video models use pixel-space state, trained either with photometric reconstruction (Finn et al., 2016; Lee et al., 2018; Oh et al., 2015; Lotter et al., 2016; Denton & Fergus, 2018; Yan et al., 2021; Ho et al., 2022b; Singer et al., 2022; Ho et al., 2022a; Guo et al., 2023; Yin et al., 2023b; Chen et al., 2023b; Brooks et al., 2024; Kondratyuk et al., 2023; Blattmann et al., 2023; Zhen et al., 2025; Kotar et al., 2025) or joint-embedding predictions (Bardes et al., 2024; Assran et al., 2025). 3D world models instead operate on meshes or explicit surfaces (Wang et al., 2021; Pfaff et al., 2020; Longhini et al., 2022; Jiang et al., 2025; Xia et al., 2025; 2024; Qiu et al., 2025), radiance fields or Gaussians (Mildenhall et al., 2020; Kerbl et al., 2023; Driess et al., 2023; Abou-Chakra et al., 2024; 2025; Zhang et al., 2024b; Lu et al., 2025; Zhang et al., 2025b; Xiang et al., 2024; Li et al., 2025f), or particles (Battaglia et al., 2016; Sanchez-Gonzalez et al., 2020; Li et al., 2018; Abou-Chakra et al., 2022; Huang et al., 2025; Zhang et al., 2025a; 2024a; He et al., 2025b; Whitney et al., 2024). Hybrid approaches additionally reason over hierarchical structures in world modeling (Aeronautiques et al., 1998; Kaelbling & Lozano-Pérez, 2011; Mao et al., 2019; LeCun, 2022; Xing et al., 2025; Liu et al., 2024; Wang et al., 2025b). Action parameterizations range from low-level joint-space commands (Finn et al., 2016; Hafner et al., 2018; 2019; 2020; Hansen et al., 2023; Qi et al., 2025; Guo et al., 2025b; Li et al., 2025a), to camera and navigation motions (Bruce et al., 2024; Zhou et al., 2025a; He et al., 2024; 2025a; Li et al., 2025c; Zheng et al., 2024; Song et al., 2025a; Zhou et al., 2025b), textual prompts (Yang et al., 2023; Brooks et al., 2024; Agarwal et al., 2025; Abu Alhaija et al., 2025; Zheng et al., 2025; Jang et al., 2025), and 2D cues (Burgert et al., 2025; Niu et al., 2024; Geng et al., 2024; Gillman et al., 2025; Hu et al., 2023; Shi et al., 2024; Wang et al., 2024; Wu et al., 2024; Yin et al., 2023a; Zhang et al., 2025c; Xing et al., 2024). Robot actions can then be produced by online planning (de Boer et al., 2005; Williams et al., 2017; Kelly, 2017; Pineau et al., 2003; Hafner et al., 2018; Li et al., 2018), offline policy synthesis (Hafner et al., 2018; 2019; 2020; Janner et al., 2019; Chua et al., 2018; Hafner et al., 2025; Hansen et al., 2023), or inverse-dynamics models (Agrawal et al., 2016; Du et al., 2023; Zhang et al., 2023). PointWorld uses 3D point flow as shared state-action representation, emphasizing contact and geometry rather than appearance, conditions on 3D actions with specific geometry of given robot/gripper, interaction beyond only visible regions compared to 2D cues, and doing so with one (or sparse) input images (with estimated depth) in a single, real-time forward pass of a large pre-trained model (Figure 7).

Dynamics Models in Robotics. Dynamics models in robotics instantiate world models with robot action spaces. They include physics-based simulators (Todorov et al., 2012; Coumans, 2015; Tedrake & the Drake Development Team, 2019; Makoviychuk et al., 2021; Koenig & Howard, 2004; Hu et al., 2020) and learning-based models (Ai et al., 2025; Battaglia et al., 2016; Sanchez-Gonzalez et al., 2020; Pfaff et al., 2020; Li et al., 2018; Abou-Chakra et al., 2022; Huang et al., 2025; Zhang et al., 2025a; 2024a; Guo et al., 2025b; Wu et al., 2023a). Crucial for robotics, they support policy learning Peng et al. (2018); Andrychowicz et al. (2020); Kumar et al. (2021), planning Lozano-Perez (1983); Kaelbling et al. (1998); Kaelbling & Lozano-Pérez (2011); Toussaint (2015); Marcucci et al. (2024), model-based RL (Hafner et al., 2018; 2019; 2020; Janner et al., 2019; Chua et al., 2018; Hafner et al., 2025; Hansen et al., 2023), exploration and online guidance (Pathak et al., 2017; Sekar et al., 2020; Mordatch & Todorov, 2014; Du & Song, 2025), safety filtering (Ames et al., 2019; Sun & Song, 2025), model-based design and verification Donze (2010); Tedrake & the Drake Development Team (2019); Xu et al. (2024b), and policy evaluation (Barreiros et al., 2025; Zhang et al., 2025b; Yang et al., 2025; Team et al., 2025). While existing dynamics models often require curated, scene-specific modeling Ai et al. (2025), our aim is to pre-train a single dynamics model that generalizes across diverse in-the-wild environments. Using 3D flows as state-action space, it naturally encapsulates many action parameterizations used in prior works in an embodiment-agnostic manner: joint-space commands Wu et al. (2023a); Guo et al. (2025b); Wang et al. (2025d), end-effector actions Zhang et al. (2025a), and motion primitives Li et al. (2018); Chen et al. (2023a), while operating on partially observable RGB-D image(s) in the wild without scene reconstruction Jiang et al. (2025), priors on objectness Zhang et al. (2025a) or materials Sulsky et al. (1993).

2D and 3D Flows for Manipulation. Flows (or point tracks), which address correspondences across space and time, provide a powerful interface between perception and control. With advances in point tracking Karaev et al. (2025); Doersch et al. (2024); Xiao et al. (2024), recent works explored them as structured representations for policy learning Wen et al. (2023); Haldar & Pinto (2025); Weng et al. (2021); Seita et al. (2022); Wang et al. (2025c); Guo et al. (2025a); Duisterhof et al. (2024); Yin et al. (2025), reward modeling Xu et al. (2024a); Patel et al. (2025b); Guzey et al. (2025); Shi et al. (2025), (sub-)goal specification Huang et al. (2024), or as visual servoing targets Eisner et al. (2022); Zhang et al. (2023); Bharadhwaj et al. (2024); Patel et al. (2025a); Li et al. (2025b); Dharmarajan et al. (2025). In this work, we leverage recent advances in 3D vision (depth (Wen et al., 2025), camera pose estimation (Wang et al., 2025a), and point tracking (Karaev et al., 2025)) to label 3D scene flows from large-scale real-world manipulation dataset Khazatsky et al. (2024) (with robot flows obtained from known robot geometry, kinematics, and proprioception), which enables training of a large 3D world models via stable regression losses to capture robotic interactions with diverse objects in open-world environments.

A.10 DROID 3D Annotation Pipeline

DROID (Khazatsky et al., 2024) is a large-scale robot manipulation dataset with human-teleoperated interactions collected with a wrist-mounted camera and two externally-mounted cameras (randomized over the left and right sides of the workspace). We use all DROID episodes for which raw camera streams are available, irrespective of task success or failure, since 3D world modeling depends only on the observed interactions rather than the task-specific outcomes of the manipulation. Each episode provides stereo RGB streams with ground-truth camera intrinsics for all three cameras, plus robot joint states and a known kinematic model of the robot. In this work, we use the recovered 3D scene flows from the two externally-mounted cameras. All data share a synchronized timestamp. We augment the robot model to include the Robotiq 2F-85 gripper and the custom camera mount used in the standardized DROID setup. For each scene, the pipeline aligns timestamps to a reference stream (binary search to nearest), downsamples by 2, then runs: (i) dense metric depth; (ii) external-camera extrinsic refinement by aligning rendered robot mesh to observed depth; (iii) 2D tracking under workspace and robot masks; (iv) 3D trajectory reconstruction, slicing, and postprocessing. Note that we do not store robot point flows because those can be reconstructed efficiently at training/inference time given known robot URDF and the given joint actions.

A.10.1 Depth Estimation

Per-view metric depth is obtained using a high-quality stereo estimator, FoundationStereo (Wen et al., 2025). Note that unlike typical sensor depth, the estimated depth from FoundationStereo does not have a minimum depth threshold for valid depth perception. However, it is still observed that its estimated depth can be inaccurate for distant, especially texture-less, regions (such as walls). Therefore, depth values are sanitized by clamping to a trusted range $[0, 4]\,\mathrm{m}$ and producing a per-pixel validity mask, which is also propagated to 3D points as a per-point depth-valid flag.

A.10.2 Camera Pose Estimation

We do not use dataset-provided extrinsics. Instead, we compute camera extrinsics using a two-stage procedure that leverages the accurate metric depth obtained from FoundationStereo discussed previously. First, we initialize the camera pose estimates using VGGT (Wang et al., 2025a). Second, we refine all camera poses of the two external cameras from all timesteps jointly by aligning rendered robot geometry to observed depth using the recorded robot joint states.

Camera Pose Initialization. Our goal is to estimate, for each externally mounted camera C_i, a single rigid transform $T_{C_i \leftarrow B} \in \mathrm{SE}(3)$ that maps 3D points from the robot base frame B into the camera frame and is fixed throughout a DROID episode. We denote the robot base frame by B, the wrist camera frame at time t by W_t, and the external cameras by C_i. A multi-view pose estimator (VGGT (Wang et al., 2025a)) is applied to time-aligned images from the two external cameras and the wrist camera; it treats the first external camera at an initial timestep as the reference frame, and returns rigid transforms $T_{E_0 \leftarrow C_i}$ and $T_{E_0 \leftarrow W_t}$ that map points from each camera C_i or W_t into this reference frame E_0. Independently, forward kinematics applied to the robot joint states provide the pose of the gripper in the base frame, $T_{G_t \leftarrow B}$. For each physical robot (given by the recorded robot serial), we assume that the wrist camera is rigidly mounted relative to the end effector and reused across all episodes. We empirically found that this assumption is largely valid for robots used in DROID as the averaged transforms exhibit sub-millimeter alignment error with each other. Under this assumption, we can obtain a known gripper-to-wrist transform $T_{W \leftarrow G}$ for each robot. Using this transform and the forward-kinematics model, we obtain a time-varying wrist pose in the base frame,

$$T_{W_t \leftarrow B} = T_{W \leftarrow G}\, T_{G_t \leftarrow B}.$$

Combining this with the estimator's transform between the wrist and the reference external camera yields per-frame estimates of the reference camera pose in the base frame,

$$T^{(t)}_{E_0 \leftarrow B} = T_{E_0 \leftarrow W_t}\, T_{W_t \leftarrow B},$$

which we average over all valid wrist frames to obtain a single $T_{E_0 \leftarrow B}$. For any other external camera C_i, the estimator provides a relative transform $T_{E_0 \leftarrow C_i}$. We convert this to a base-frame extrinsic

$$T_{C_i \leftarrow B} = T_{C_i \leftarrow E_0}\, T_{E_0 \leftarrow B}, \quad T_{C_i \leftarrow E_0} = T^{-1}_{E_0 \leftarrow C_i},$$

so that all external cameras are expressed in the same robot base frame before the refinement stage.

Camera Pose Refinement. Starting from the initialized base-frame extrinsics $T_{C_i \leftarrow B}$, we jointly refine the poses of all external cameras using robot-depth reprojection. Let external cameras be indexed by i, timesteps by t, and let k index valid robot pixels after filtering (in front of the camera, within image bounds, deduplicated in the image plane, and with observed depth in the trusted range). For each camera i we optimize a small 6-DoF update on top of the initialization, parameterized by translation and rotation and scaled such that optimization can be done within a good numerical range. Given an observed depth value $d^{\text{obs}}_{i,t,k}$ at a valid robot pixel and the corresponding predicted depth $d^{\text{pred}}_{i,t,k}$ obtained by projecting robot surface points from the base frame through the current extrinsics $T_{C_i \leftarrow B}$, we define the robot-depth reprojection loss

$$L_{\text{robot-depth}} = \frac{1}{K} \sum_{i,t,k} \big| d^{\text{obs}}_{i,t,k} - d^{\text{pred}}_{i,t,k} \big|,$$

where the sum runs over all valid robot pixels across cameras and frames and K is their total count. We optimize the 6-DoF updates for all external cameras jointly using a first-order optimizer (100 iterations, learning rate 10^{-3}), and restrict supervision to robot pixels whose observed depth lies in a trusted range of $[0.3, 2.0]$ m. To ensure reliable gradients, we further require that each camera–frame pair contributes at least $2{,}000$ valid robot points; frames that fail this criterion for any external camera are discarded before refinement. With these procedures, we can accurately label camera poses for around 60% of all episodes recorded in DROID. Quantitative metrics are reported in the main text.

A.10.3 Benchmark Metrics for 3D Annotation

Depth Reprojection. For each scene and calibration variant, we render the robot mesh into each external camera using the candidate extrinsics and recorded joint states, discard depth outside $[0.3, 2.0]$ m or out-of-bounds robot pixels, and compute an L1 depth reprojection loss over valid robot pixels. The per-frame losses are averaged over valid pixels and frames to produce a point-weighted value reported per scene and then aggregated into cumulative curves.

Two-view F1 @ 5/20 mm. For each external camera, we back-project valid depth into 3D using corresponding intrinsics/extrinsics, remove robot pixels, and crop to workspace. Given the resulting paired point clouds, we compute precision, recall, and F1 via symmetric nearest-neighbor matching: a point in cloud A (resp. B) is a true positive if its nearest neighbor in B (resp. A) lies within the threshold (5 mm or 20 mm); otherwise it contributes to the false-positive/false-negative count. Metrics are accumulated over frames, normalized by the number of valid points per view, and then aggregated per scene into the cumulative counts shown in Figure 9. Scenes with missing depth/extrinsics for a given combo are omitted from evaluation for that combo.

A.10.4 Occlusion-Aware Tracking in 3D

Given depth and camera poses, we herein describe how we can obtain 3D point flows. We describe the following in order: (i) filtering clips based on robot motion, (ii) tracking visible points within each retained clip, and (iii) postprocessing the resulting 3D trajectories.

Clip Filtering. We slice each episode into overlapping clips of length $F{=}16$ frames with stride $s{=}1$. Each clip covers approximately one second of the episode. Clips are retained if either the gripper changes state within the clip or end-effector motion exceeds thresholds. Thresholds depend on whether the gripper is predominantly open or closed within the clip: position/rotation thresholds are $(0.005\,\mathrm{m},\ 0.10\,\mathrm{rad})$ when open and $(0.002\,\mathrm{m},\ 0.05\,\mathrm{rad})$ when closed; either exceeding suffices to keep a clip, and any change in gripper state also keeps the clip.

Tracking. After obtaining depth and camera poses and slicing episodes into short clips, we perform dense tracking to obtain 3D scene flows on a per-clip basis. Clips cover roughly one-second windows and, depending on where they fall within a longer episode, may observe quite different regions of the workspace; tracking after clip selection ensures that each clip has its own consistent set of tracked regions and avoids mixing trajectories across widely separated time intervals. To improve efficiency and robustness, we perform tracking restricted to workspace and non-robot regions. To construct the workspace mask, we project a fixed 3D workspace volume to the image. To construct the non-robot mask, we render the robot's URDF and project the mesh to the image plane. We then track 2D points using 2D point trackers (CoTracker3 (Karaev et al., 2025)) on the masked regions only, producing dense 2D trajectories. The tracker also outputs a per-point visibility mask over time; we store this visibility for each 2D trajectory so that occluded points can later be excluded from supervision after lifting to 3D. Because POINTWORLD is trained on 3D point flows rather than image-plane trajectories, we lift each tracked 2D point to a 3D world-space trajectory by back-projecting it with the corresponding depth, intrinsics, and extrinsics at each timestep. We next reconstruct 3D trajectories and apply clip-level postprocessing. For each frame, valid depth, intrinsics, and extrinsics back-project tracked pixels to world-frame 3D points with RGB. Tracks across time yield temporally consistent per-point trajectories. We store per-camera trajectories to avoid mixing viewpoints prematurely and keep per-point visibility and depth-valid flags that are later used to mask supervision.

Postprocessing. To further improve the quality of the obtained 3D point flows, we apply two postprocessing steps: DBSCAN-based outlier removal and per-frame normal estimation. For each clip we first remove spatial outliers using multi-scale DBSCAN clustering (Ester et al., 1996) across all external cameras: at each timestep, we run DBSCAN with radii $\varepsilon \in \{0.02, 0.05\}$ m and minimum core size 5, and mark point flows that are classified as outliers in more than 20% of frames as outliers to be discarded. From the remaining trajectories, we estimate normals per frame using local neighborhoods (up to k=30 neighbors within radius 0.1 m) and orient them toward the camera, followed by a temporal consistency step that flips back-facing normals so that normal directions remain coherent over time.

A.11 BEHAVIOR-1K Data Generation

BEHAVIOR-1K (B1K) (Li et al., 2024) is a large-scale benchmark of everyday household activities in photorealistic simulation built on NVIDIA Isaac Sim. As part of the 2025 BEHAVIOR Challenge, it provides approximately $10{,}000$ human-teleoperated episodes (average length ≈ 6.6 minutes) across 50 tasks executed by a bimanual mobile robot (Galexea R1 Pro). We replay these episodes in the original simulator and attach three virtual cameras (left shoulder, right shoulder, and head) to extract short clips with meaningful interactions and dense 3D point flows, as detailed below.

A.11.1 Dataset Replay

For each BEHAVIOR-1K episode, we replay it in the simulator using the recorded sequence of environment states and actions. To prevent physics leakage and adhere to the original demonstrations, we iterate over the stored trajectory and, at every recorded step, load the corresponding simulator state and advance the simulator once with the recorded action. At every such step, we also render three external RGB-D cameras mounted on the robot: a left and right shoulder camera attached near the base, and a head-mounted camera. All three cameras have ground-truth intrinsics and extrinsics, and produce per-pixel depth, surface normals, and per-link segmentation in addition to RGB. All extrinsics are recorded in the robot base frame in the first timestep of each clip.

A.11.2 Clip Filtering

We aim to extract short clips of fixed length $F{=}11$ frames that contain meaningful interaction between the robot and the scene while discarding static or uninteresting intervals. To generate candidates, we slide an overlapping window of length F over each replayed episode at a fixed temporal stride; any window for which *all* external cameras have no visible, in-workspace scene objects—that is, no non-robot, non-ground meshes with valid depth inside the workspace bounds—is immediately discarded.

For each remaining candidate window, we maintain a set of *motion indicators* and *contact indicators* that are updated over the clip. Let M_g denote the event that at least one arm's end-effector exhibits sufficient translational or rotational motion over the clip or undergoes a change in gripper open/closed state, with thresholds that depend on whether the gripper is predominantly open or closed. Let M_j denote the event that any non-base robot joint moves more than a fixed threshold over the clip. Using the object trajectories described in Section A.11.3, we define M_o as the event that at least one object moves more than an object-movement threshold in position or orientation relative to its pose at the first frame of the clip.

From the ground-truth simulation state, we further construct contact indicators. Let C_t denote the event that any trunk or arm link experiences a nonzero contact impulse during the clip, and let C_f denote the event that any gripper finger link experiences contact. Clips that contain large simulator-induced discontinuities (such as scene resets) are filtered internally before applying the following logical criterion.

At the end of the horizon, a remaining candidate is accepted as a valid clip if and only if

$$\neg C_t \wedge \big((M_o \wedge M_j) \vee (M_o \wedge C_f) \vee (\neg M_o \wedge M_g \wedge M_j)\big). \tag{3}$$

The term $\neg C_t$ discards clips that contain trunk or arm collisions. The first disjunct in equation 3 retains clips where object motion is causally associated with non-base joint motion. The second disjunct retains clips where object motion primarily arises through gripper-finger contacts, which covers behaviors such as pushing an object with only base motion rather than arm motion (e.g., pushing a door by moving the base). The third disjunct retains "negative" clips in which the robot moves but no objects move, providing supervision on background dynamics and free-space motions.

A.11.3 3D Point Flows from Simulation

For each accepted clip and each external camera, we construct a compact representation of 3D point flows that exploits three properties of the simulation setting: (i) the environment is composed of rigid objects decomposed into rigid links; (ii) we have access to ground-truth link-level instance segmentation in rendered images; and (iii) we can query the exact rigid trajectory of every link throughout the clip. At the first frame of the clip, we back-project depth for each visible link to

Operation	Description
Camera subsampling	Sample randomized calibrated RGB-D views per timestep and concatenate their 3D points into a single scene cloud.
Bounds filtering	Retain only scene points that stay within a workspace cube (approx. $[-3, 3]^3$ m) for the entire clip, dropping particles that ever exit the bounds.
Centering	Mean-center first-frame scene and robot points.
Image resize	Downscale RGB-D images to 320×180.
Voxel downsampling	Voxel-grid sampling at $1.5\,\text{cm}$; select one point per occupied voxel at $t{=}0$, and apply the same indices to all timesteps.
Multi-sphere cropping	Iteratively remove spheres of points far from the robot (up to three spheres, radii in $[0.10, 0.80]$ m with buffer 0.25 m) until the scene falls below the target budget.
Max scene / robot points	Randomly subsample scene points if more than $12\,000$ remain after cropping; robot points are capped at 500 by construction.
Random yaw	Uniform rotation about the vertical axis over $[-\pi, \pi]$.
Uniform scaling	Isotropic scaling with factor sampled uniformly from $[0.9, 1.1]$.
Random reflection	With probability 0.5, reflect the scene and robot across either the x- or y-axis.
Chromatic auto-contrast	Apply chromatic auto-contrast to RGB channels with probability 0.2 and blend factor up to 0.2.
Chromatic translation	Add a global RGB offset with magnitude 2% with probability 0.95.
Chromatic jitter	Add per-point RGB noise with standard deviation 2% with probability 0.95.

Table 3: **Data Preprocessing and Augmentations.**

obtain a set of surface points in that link's local frame, together with associated colors and normals, after filtering out background and robot meshes and enforcing workspace bounds in the robot base frame at the clip start. We then record the time-varying poses of all visible links and cameras in this same clip-start robot base frame. This factorized representation, local link points plus per-link trajectories, allows us to reconstruct exact per-point 3D trajectories for any clip while remaining far more storage-efficient than storing dense point clouds at every frame. Note that while we access ground-truth simulator state for obtaining ground-truth 3D point flows, the simulator state is never exposed to the model.

Point set	Feature	Definition
Robot	Position $p_{t,j}^{\text{robot}}$	3D coordinates of robot points over time.
Robot	Color c_j^{robot}	Constant magenta color $(1, 0, 1)$ indicating robot identity, shared across timesteps.
Robot	Normal $n_{t,j}^{\text{robot}}$	Surface normals of robot points from the known robot URDF.
Robot	Gripper openness $\tilde{g}_t$	Scalar gripper open value per timestep, broadcast to all robot points.
Robot	Velocity $v_{t,j}^{\text{robot}}$	Per-point velocity from mid-point finite differences over $p_{t,j}^{\text{robot}}$ across time.
Robot	Acceleration $a_{t,j}^{\text{robot}}$	Per-point acceleration from mid-point finite differences over $v_{t,j}^{\text{robot}}$ across time.
Scene	Position $x_{0,i}$	3D coordinates of scene points at the first frame after preprocessing.
Scene	Color $c_{0,i}^{\text{scene}}$	RGB color of scene points at the first frame.
Scene	Normal $n_{0,i}^{\text{scene}}$	(Estimated) surface normals of scene points at the first frame.
Scene	Gripper openness sequence $g_{0:T-1}$	Sequence of gripper openness values over the context and prediction horizon, broadcast to every scene point.
Scene	Distance-to-robot $d_{0:T-1,i}$	For each timestep, distance from the first-frame position of scene point i to the closest robot point, stacked across time.

Table 4: **Per-Point Input Features.**

A.12 Model Training Details

Data Preprocessing and Augmentations. Here we describe the data preprocessing and augmentations used in our experiments. Each training sample fuses calibrated RGB-D views before passing through workspace filtering, centering, and deterministic voxel sampling with multi-sphere cropping so that the fused cloud respects fixed budgets for scene and robot points. Geometric augmentations consist of random yaw rotations, isotropic scaling, and reflections; photometric augmentations apply auto-contrast, global color shifts, and per-point jitter to the RGB channels. For evaluation, we ensure the pipeline to be fully deterministic and disable all augmentations.

Per-Point Input Features. Here we describe the per-point features produced as part of the data pipeline, prior to their consumption by the model. Details are listed in Table 4. Robot features stack positions, surface normals, a gripper scalar, and velocity and acceleration terms:

$$\phi_{t,j}^{\text{robot}} = \big[p_{t,j}^{\text{robot}},\, c_{j}^{\text{robot}},\, n_{t,j}^{\text{robot}},\, \tilde{g}_t,\, v_{t,j}^{\text{robot}},\, a_{t,j}^{\text{robot}}\big],$$

where $p_{t,j}^{\text{robot}}$ and $n_{t,j}^{\text{robot}}$ are position and normal, c_{j}^{robot} is a fixed color tag, $\tilde{g}_t$ is the normalized gripper openness, and $(v_{t,j}^{\text{robot}}, a_{t,j}^{\text{robot}})$ come from mid-point finite differences across the horizon with zero-velocity boundary conditions at the first and last timestep, i.e., we assume the robot is stationary at the boundaries of each model window. Scene features are computed for only the first frame $t{=}0$ and combine positions, colors, estimated normals, gripper openness sequence, and distances to the nearest robot point:

$$\phi_{i}^{\text{scene}} = \big[x_{0,i},\, c_{0,i}^{\text{scene}},\, n_{0,i}^{\text{scene}},\, g_{0:T-1},\, d_{0:T-1,i}\big],$$

where $c_{0,i}^{\text{scene}}$ and $n_{0,i}^{\text{scene}}$ are the RGB color and normal at the first frame, $g_{0:T-1} \in \mathbb{R}^T$ is the sequence of gripper openness values over the context-plus-prediction horizon broadcast to every scene point, and $d_{t,i}$ is the distance from scene point i to the closest robot point at timestep t,

$$d_{t,i} = \min_j \big\| x_{0,i} - r_{t,j} \big\|_2, \qquad d_{0:T-1,i} \in \mathbb{R}^T.$$

The distance field $d_{0:T-1,i}$ is obtained from nearest-neighbor queries between first-frame scene points and robot points at every timestep.

3D Scene Featurization with DINOv3. Prior to the point cloud backbone, PointWorld uses a 2D scene encoder based on DINOv3 ViT-L/16 by aggregating its multi-layer features. To featurize the 3D scene points with the image-based encoder, the first-frame scene coordinates $x_{0,i} \in \mathbb{R}^3$ are projected into each chosen camera. For camera c with intrinsics K_c and extrinsics (R_c, t_c) we form $\tilde{u}_{c,i} = K_c(R_c x_{0,i} + t_c)$ and obtain the pixel coordinate as

$$u_{c,i} = \big[\tilde{u}_{c,i}^{(1)}/\tilde{u}_{c,i}^{(3)},\, \tilde{u}_{c,i}^{(2)}/\tilde{u}_{c,i}^{(3)}\big]^\top.$$

The same intrinsics and extrinsics support a depth-consistency mask that compares the projected depth of each point with the given depth image, so only views whose discrepancy is below a few millimeters contribute features.

For each visible point-camera pair (i, c), DINOv3 patch tokens are sampled at $u_{c,i}$ by bilinear interpolation on the patch-token grid, using a coordinate mapping that aligns token centers with pixel centers. Let $f_{c,i} \in \mathbb{R}^{D_{\text{patch}}}$ denote the concatenated multi-layer patch feature for point i in camera c, and let $m_{c,i} \in \{0, 1\}$ indicate visibility and depth consistency. Features are aggregated across cameras by averaging over the contributing views,

$$f_i = \frac{1}{\max\big(1, \sum_c m_{c,i}\big)} \sum_c m_{c,i}\, f_{c,i}.$$

The averaged token is mapped to the backbone width (256 channels) by a learned projection and fused with a separately projected version of the raw scene features from Table 4; layer normalization is applied to each stream before concatenation, and a final linear layer produces the per-point embedding supplied to the dynamics backbone. The 2D encoder is kept frozen during training and evaluation.

Visibility-Aware Supervision. For real-world domains, we restrict training on 3D point trajectories to correspondences that are both geometrically and photometrically reliable. The annotation pipeline supplies per-point visibility (from 2D trackers Karaev et al. (2025) on real data and from ground-truth simulator state on synthetic data) together with per-pixel depth-validity; both signals are propagated to the lifted 3D trajectories and stored as binary flags per scene point and timestep. During training, we construct a per-timestep mask that selects scene points that are visible in the camera view and have valid depth support. The weighted dynamics objective from Section 3 is then evaluated only over this subset of correspondences (points filtered out receive zero loss weight), so that gradients are driven by non-occluded, depth-valid 3D flows. For simulation domains, where trajectories and depth are noise-free and occlusions are explicitly modeled, all scene points contribute to the loss.

Training Configuration. We train the 1B-parameter version of POINTWORLD on both BEHAVIOR-1K and DROID, with configuration and PointTransformerV3 (PTv3) design summarized in Table 5 and Table 6, respectively. For the main experiments in the paper, training configuration and PTv3 design are summarized in Table 7 and Table 8.

Aleatoric Uncertainty on Simulation Data. When training on mixtures of real and simulated domains, directly learning per-point uncertainty everywhere can collapse the model because simulated trajectories are noise-free. In the objective from Section 3 the residual term for point i at step k is weighted as $w_{k,i}\,\rho_\delta(\hat{\mathbf{P}}_{t+k,i} - \mathbf{P}_{t+k,i})\,\mathrm{e}^{-s_{k,i}} + w_{k,i}\,s_{k,i}$. For vanishing residuals (typical in simulation), minimizing the loss drives $s_{k,i}$ toward $\log \rho_\delta(\cdot)$. Since $\rho_\delta(\cdot)$ approaches zero, the optimal $s_{k,i}$ becomes a large negative number, i.e., the predicted variance $\sigma^2_{k,i} = \exp(s_{k,i})$ collapses toward zero. As a consequence, $\mathrm{e}^{-s_{k,i}}$ explodes, so any small numerical discrepancy in simulated residuals produces excessively large gradients that overwhelm the real-data contributions and destabilize joint training. To stabilize training, we treat aleatoric variance on simulation domains as a constant: the uncertainty head is trained normally on real data, but for simulated domains its log-variance is replaced by a batch-wise constant that matches the average variance observed on real samples (or a small fixed value when only simulation is present). This preserves heteroscedastic weighting where it is most useful (real, noisy supervision) while preventing the model from exploiting the uncertainty head to down-weight clean simulated gradients.

Setting	Value
Optimizer	AdamW
Learning rate	1×10^{-4}
Epochs	300
Weight decay	10^{-2}
Global batch size	1920 sequences
Gradient clipping	Global ℓ_2 norm capped at 5
Loss	Huber loss with $\delta = 5.0$ with movement weighting and aleatoric uncertainty
Prediction horizon	10 steps
Training GPUs	128 NVIDIA H100 GPUs
Training time	20 days

Table 5: **Training Configuration for PointWorld-1B.**

Component	Values
Grid size	1.5 cm
Encoder depth	$(4,\ 4,\ 8,\ 8,\ 12,\ 12,\ 4)$
Encoder channels	$(256,\ 384,\ 384,\ 512,\ 512,\ 768,\ 1024)$
Encoder heads	$(8,\ 12,\ 12,\ 16,\ 16,\ 24,\ 32)$
Encoder stride	$(1,\ 2,\ 2,\ 2,\ 2,\ 2,\ 2)$
Encoder patch size	$(256,\ 256,\ 256,\ 256,\ 256,\ 256,\ 256)$
Decoder depth	$(4,\ 4,\ 4,\ 4,\ 4,\ 4)$
Decoder channels	$(256,\ 384,\ 384,\ 512,\ 512,\ 768)$
Decoder heads	$(8,\ 12,\ 12,\ 16,\ 16,\ 24)$
Decoder patch size	$(256,\ 256,\ 256,\ 256,\ 256,\ 256)$

Table 6: **PointTransformerV3 (PTv3) Architecture for PointWorld-1B. Encoder and decoder configurations are ordered by stage.**

Setting	Value
Optimizer	AdamW
Learning rate	1×10^{-4}
Epochs	200
Weight decay	10^{-2}
Global batch size	176 sequences
Gradient clipping	Global ℓ_2 norm capped at 5
Loss	Huber loss with $\delta = 5.0$ with movement weighting and aleatoric uncertainty
Prediction horizon	10 steps
Training GPUs	8 NVIDIA H100 GPUs
Training time	7 days

Table 7: **Training Configuration for PointWorld-411M.**

Component	Values
Grid size	1.5 cm
Encoder depth	$(4,\ 4,\ 4,\ 8,\ 8,\ 12,\ 4)$
Encoder channels	$(256,\ 256,\ 256,\ 384,\ 384,\ 512,\ 768)$
Encoder heads	$(4,\ 4,\ 4,\ 8,\ 8,\ 16,\ 24)$
Encoder stride	$(1,\ 2,\ 2,\ 2,\ 2,\ 2,\ 2)$
Encoder patch size	$(256,\ 256,\ 256,\ 256,\ 256,\ 256,\ 256)$
Decoder depth	$(2,\ 2,\ 2,\ 2,\ 2,\ 2)$
Decoder channels	$(256,\ 256,\ 256,\ 384,\ 384,\ 512)$
Decoder heads	$(4,\ 4,\ 4,\ 8,\ 8,\ 16)$
Decoder patch size	$(256,\ 256,\ 256,\ 256,\ 256,\ 256)$

Table 8: **PointTransformerV3 (PTv3) Architecture for PointWorld-411M. Encoder and decoder configurations are ordered by stage.**

A.13 DROID Evaluation Protocol

Following the protocol in Section 5, we measure per-sequence losses and aggregate them into dataset-level summaries. Alongside the overall per-point, per-timestep ℓ_2 distance, we report the same metric separately on moved and static points, since movers form a minority of the points but dominate perceived quality. We use these metrics directly for simulation data (BEHAVIOR-1K) since they are noiseless. However, for real-world data (DROID), we further apply expert confidence filtering to obtain filtered metrics. Details are described below.

Expert Confidence Filtering. Although the mover-ℓ_2 score highlights the behavior we care about most, imperfect real-world annotations mean that a noticeable fraction of mover points correspond to outliers or background clutter, because those points tend to have large movement magnitudes due to unstable depth estimation. During training, the aleatoric uncertainty regularization down-weights those points, but at evaluation time different models produce their own uncertainty predictions, making comparisons challenging. To obtain a model-agnostic notion of trustworthy ground-truth, we train an expert model only on the evaluation split with uncertainty predictions, convert the predicted variance into a per-point confidence in $[0, 1]$, and threshold this per-timestep per-point confidence at the 0.8 quantile over all points. Points below this confidence are treated as low-confidence outliers. We voxelize these low-confidence sets in world coordinates using the training grid size g and cache the resulting voxel grids for each evaluation sample so that the same filtering masks can be reused across subsequent evaluation runs and model variants. Note that the expert model is only used to compute the low-confidence voxel grids and does not share any training data or parameters with any evaluated models.

Filtered Evaluation. To evaluate a model, for each sample, we first reconstruct world-coordinate voxel indices of scene points and then determine whether each point lies inside a precomputed low-confidence voxel. This yields a binary filter mask so that only high-confidence points at high-confidence timesteps (deemed by the shared expert model) contribute to the filtered metrics.

Mover/Static Splits. Let $\hat{P}_{t,i}$ and $P_{t,i}$ denote predicted and ground-truth 3D positions. We compute per-point error $e_{t,i} = \|\hat{P}_{t,i} - P_{t,i}\|_2$ and report

$$\ell_2 = \frac{1}{T} \sum_t \frac{1}{|V_t|} \sum_{i \in V_t} e_{t,i},$$

where V_t denotes valid points at timestep t. Mover-ℓ_2 and static-ℓ_2 use the same definition but restrict V_t to moved or static points identified from the ground-truth trajectories via a small displacement threshold.

A.14 Real-Robot Experiment Details

Real-robot experiments use a 7-DoF Franka arm equipped with a 3D-printed fin ray gripper Chi et al. (2023; 2024). The robot is mounted on a wheeled, non-motorized base for in-the-wild deployments. Since PointWorld is trained on data containing Robotiq 2F-85 and Galexea R1 Pro grippers, the fin ray gripper geometry remains fully unseen by the model, illustrating cross-gripper geometry generalization. Since the pipeline predicts 6-DoF end-effector poses, we run position control at 20 Hz: each predicted target pose is clipped to a predefined workspace, then linearly interpolated from the current pose with steps of 5 mm in translation and 1° in rotation. For every interpolated pose, inverse kinematics (PyBullet solver) produces target joint positions that are tracked with the Deoxys joint-impedance controller Zhu et al. (2022). We use one RealSense D435 mounted on the left shoulder of the robot to capture RGB and the stereo IR images. The stereo IR images are used to estimate the metric depth using FoundationStereo (Wen et al., 2025), given known baseline and camera intrinsics.

A.14.1 Model-Based Planning

In this work, we use a single pre-trained PointWorld as the dynamics model. The model is pre-trained jointly on both real-world and simulated data. We use a sampling-based model-predictive path integral (MPPI) controller that samples action sequences around a zero-initialized nominal using cubic splines with $n_{\text{knots}}{=}4$ and degree 3. Noise scales are scheduled between $\sigma_{\min}{=}0.05$ and $\sigma_{\max}{=}0.50$ (in normalized action units). Each refinement iteration draws 256 samples; importance weights use temperature $\beta{=}0.05$, and the nominal is updated with an exponential moving average (EMA $= 0.9$). We perform planning for 30 steps into the future, and the horizon is chunked to match the prediction window of the dynamics model. We perform 20 refinement iterations. The planning time is typically around a few seconds depending on task complexity and specific model size variant used. While we do not perform replanning in this work, replanning can be done at a real-time frequency by warm-starting from the previous nominal trajectory.

A.14.2 Task Specification

We specify tasks through a GUI tool that allows users to select object masks using SAM2 Ravi et al. (2024) and specify target positions in the world frame. We find this simple objective as a unified interface for specifying diverse real-world tasks including rigid pushing, deformable manipulation, articulated-object interaction, and tool use. Following common practices in reward design Zakka et al. (2025), we add a mild end-effector proximity term to encourage exploration in the object's neighborhood without prescribing a particular contact pattern. For deformable and tool-use tasks we begin from a pre-grasped configuration so that subsequent motion primarily probes deformable dynamics and object-object contacts. All tasks share same control regularization comprising SE(3) path-length penalties and IK-based reachability residual.

A.14.3 Evaluation Protocol

We conduct evaluations on the following tasks: rigid pushing (tissue box, book), deformable manipulation (scarf fold, pillow place), articulated manipulation (microwave open, drawer close), and tool use (duster sweep, broom sweep). Each task is evaluated with ten randomly sampled initial configurations. The configurations are sampled prior to evaluation and verified to be kinematically feasible for the robot. For each trial, a human operator restores the scene to the designated configuration and triggers execution. We consider the trial successful if the task objective is met. Otherwise we declare failure. If the optimization produces a solution that is considered unsafe for execution, the trial is considered failure too. The success rates are reported in the main paper.

A.14.4 Effect of Training Mixture

Beyond the quantitative success rates for real-world deployment, we observe interesting qualitative traits when using different variants of PointWorld pre-trained on different data mixtures. We empirically observe that models trained only on real data tend to be conservative: a common failure mode is for scene points to remain static even when the robot establishes contact, which we attribute to heavy regularization coping with annotation noise. On the other hand, models trained only on simulation data excel on rigid objects but frequently mis-segment cluttered real scenes implicitly, causing background points to move together with the target. Models trained on both real and simulated domains yield the most balanced behavior in practice, combining realistic contact handling with the ability to generalize to novel real-world scenes. A systematic study of how training-mixture design shapes deployment-time behavior, e.g., by varying real/sim proportions or task/domain coverage under controlled conditions, remains an important direction for future work.

A.15 Additional 3D Annotation Examples

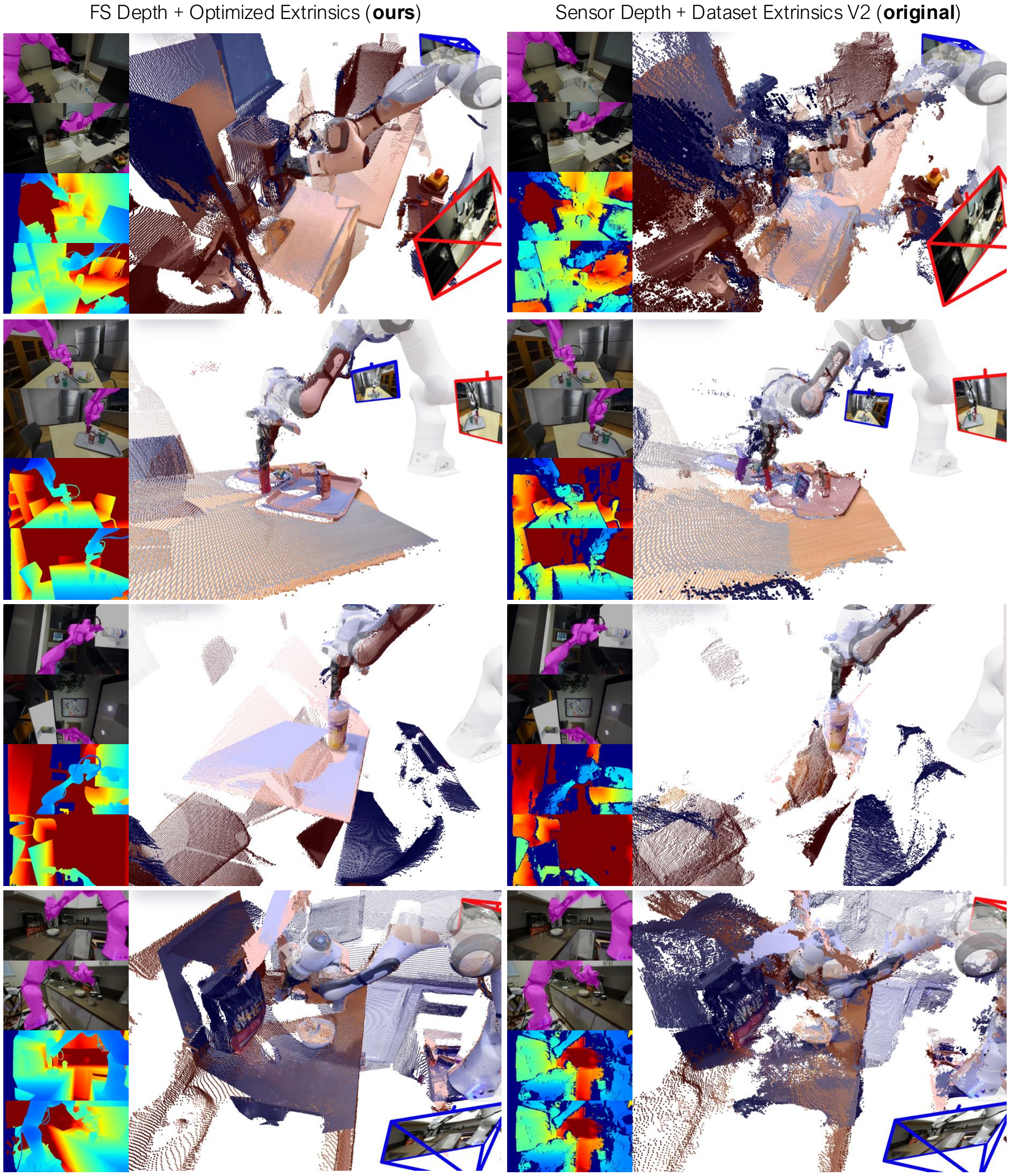
FS Depth + Optimized Extrinsics (**ours**)
Sensor Depth + Dataset Extrinsics V2 (**original**)

Figure 14: **DROID 3D annotations**, including robot-overlaid RGBs, depths, point clouds, and comparisons to original dataset.

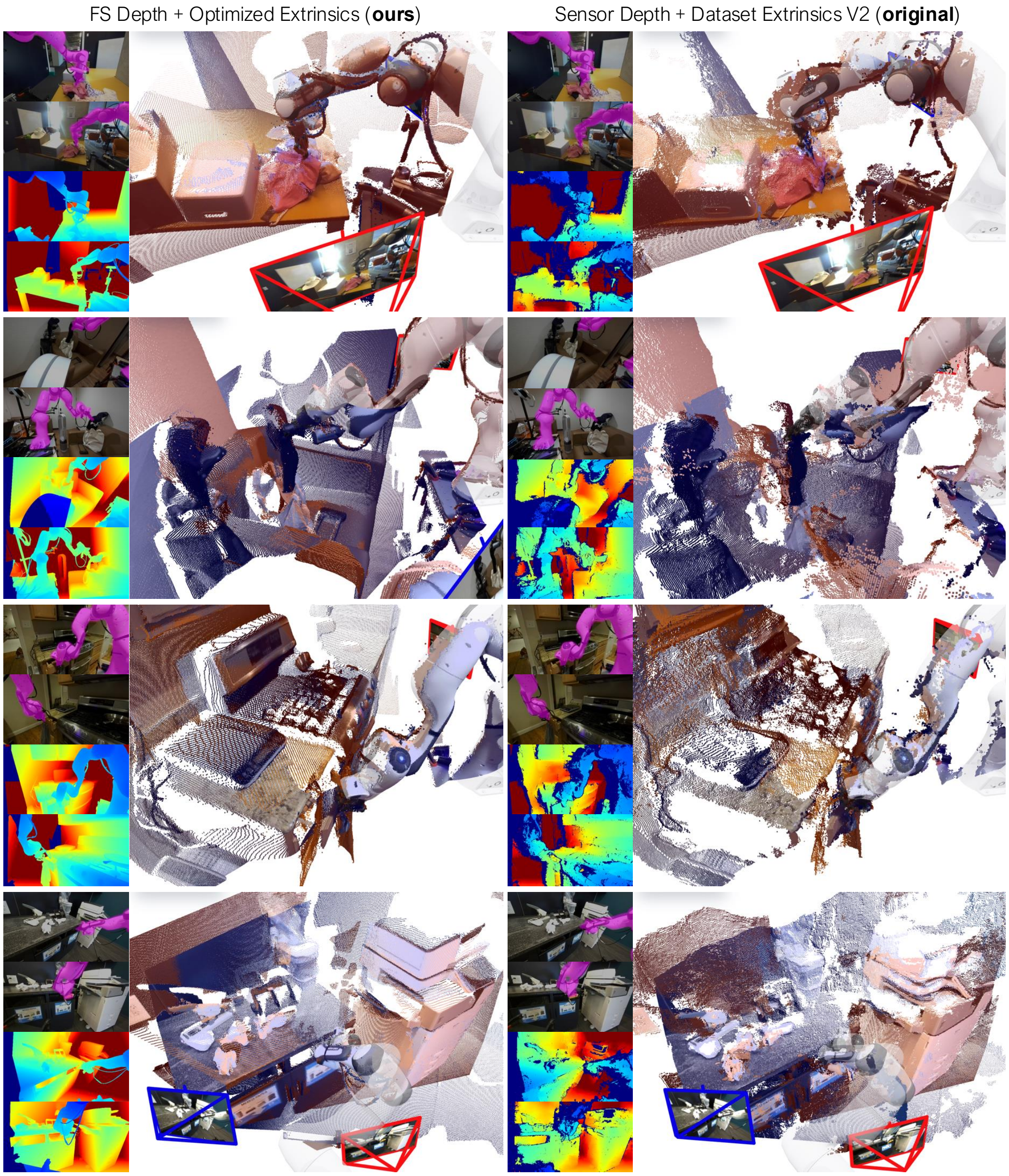
FS Depth + Optimized Extrinsics (**ours**)
Sensor Depth + Dataset Extrinsics V2 (**original**)

Figure 15: **DROID 3D annotations**, including robot-overlaid RGBs, depths, point clouds, and comparisons to original dataset.

A.16 Additional Model Rollouts

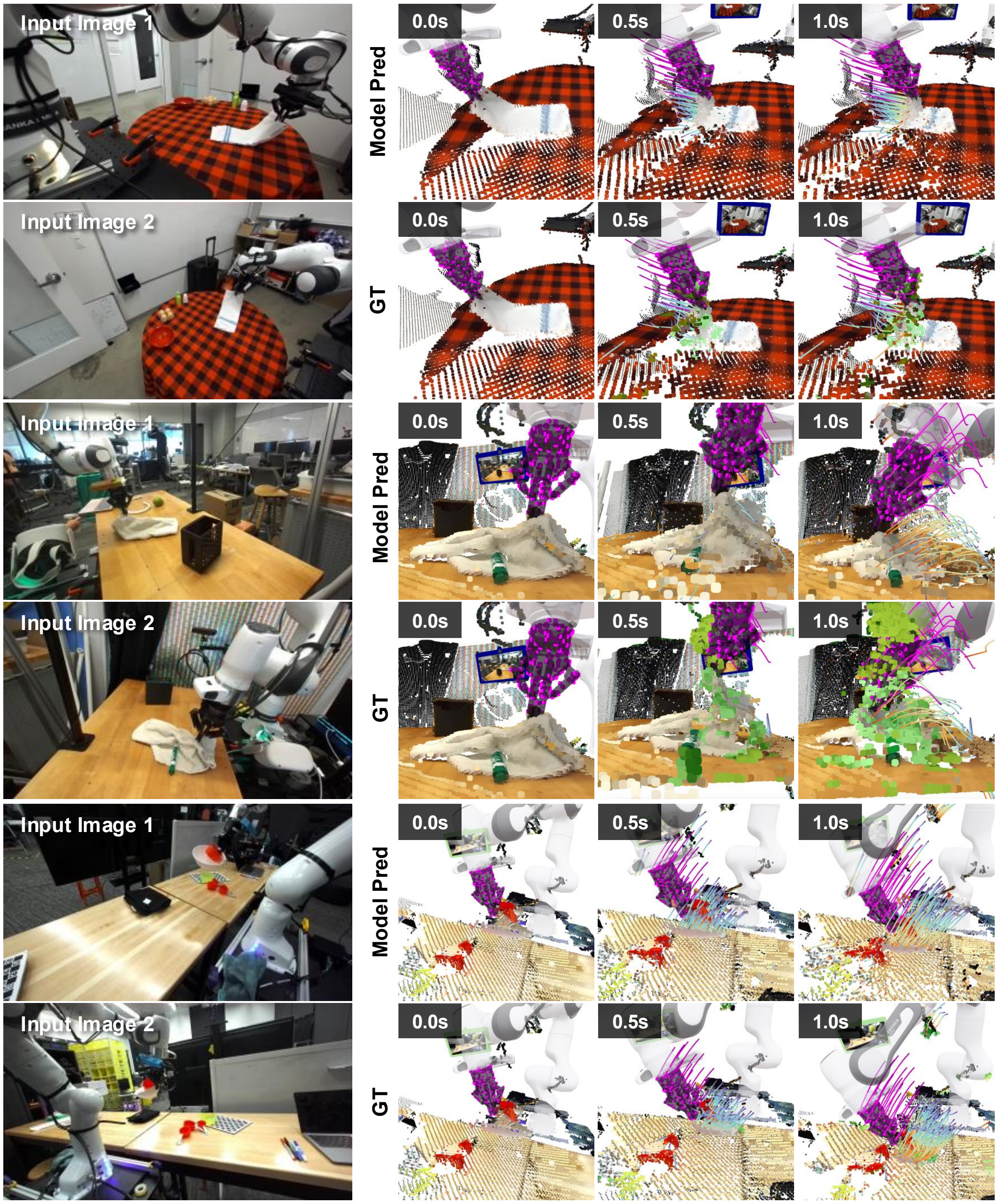

Figure 16: **DROID Unseen Rollouts**, including deformable manipulation, robot-object interactions, and object-object interactions.

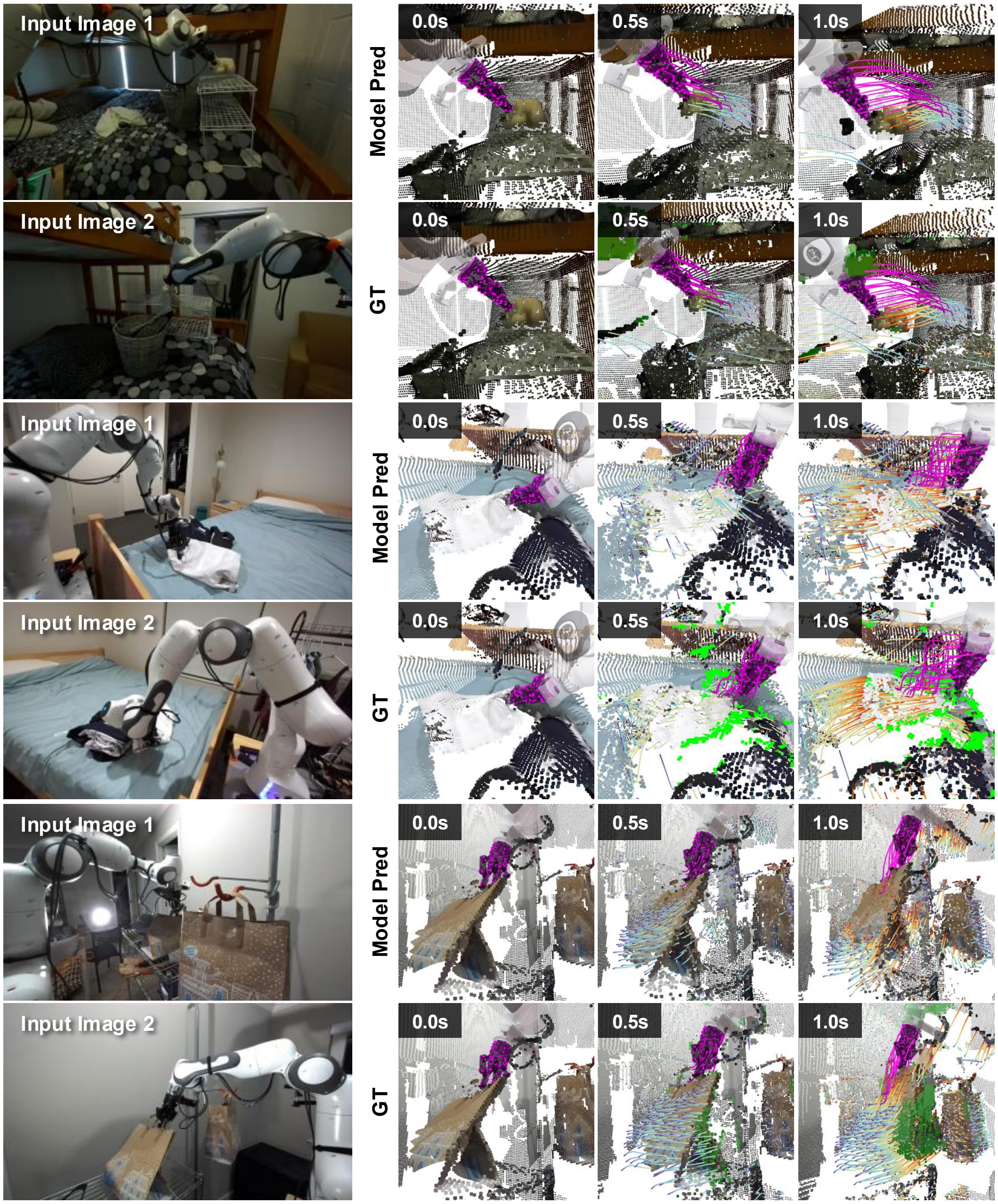

Figure 17: **DROID Unseen Rollouts**, including deformable manipulation, and grasping behaviors.

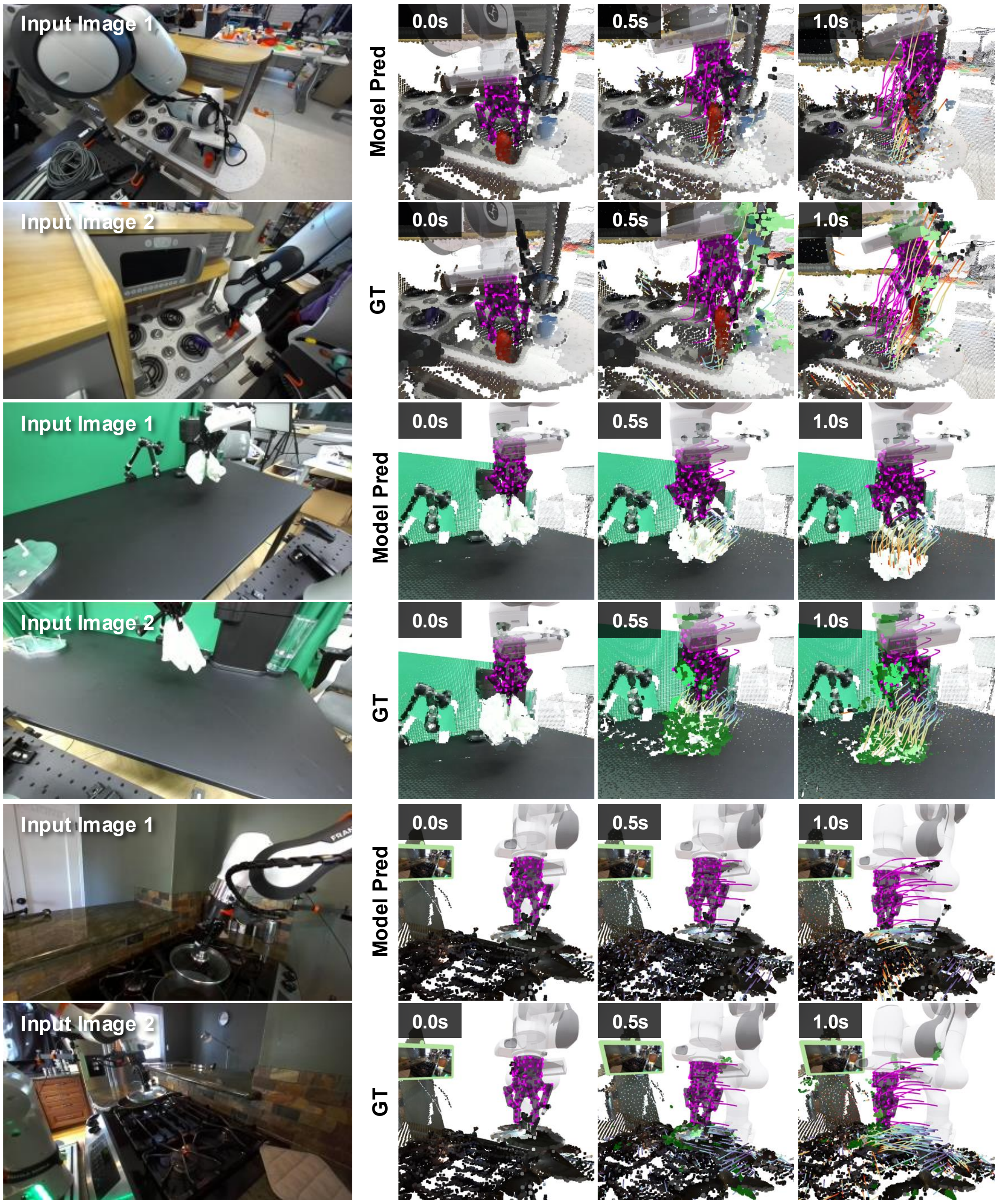

Figure 18: **DROID Unseen Rollouts**, including grasping behaviors, gravity effects, and glass objects.

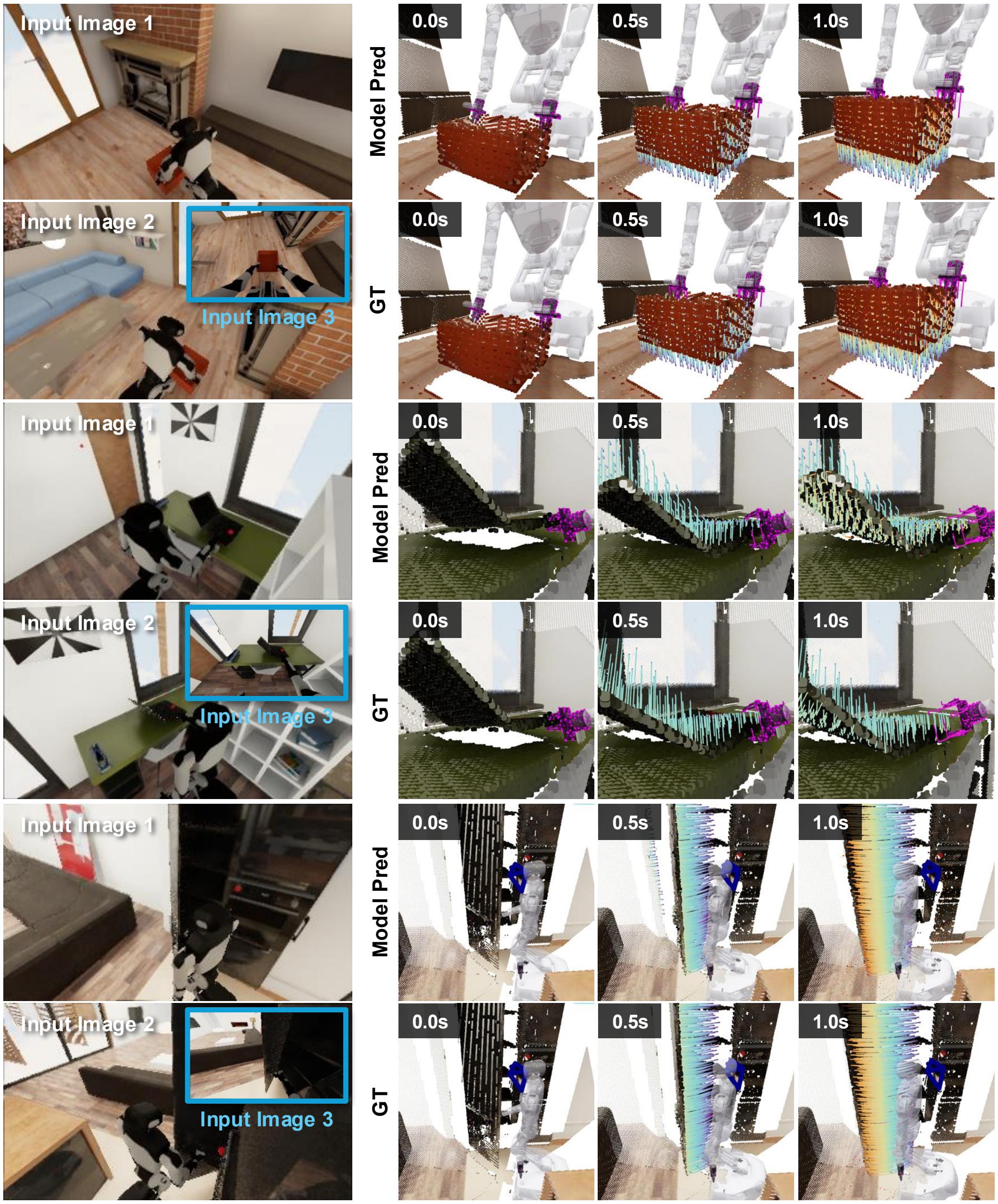

Figure 19: **BEHAVIOR-1K Unseen Rollouts**, including constrained bimanual lifting, gravity effects (dropped laptop), object-object interactions (laptop v.s. table), and articulated manipulation (fridge).

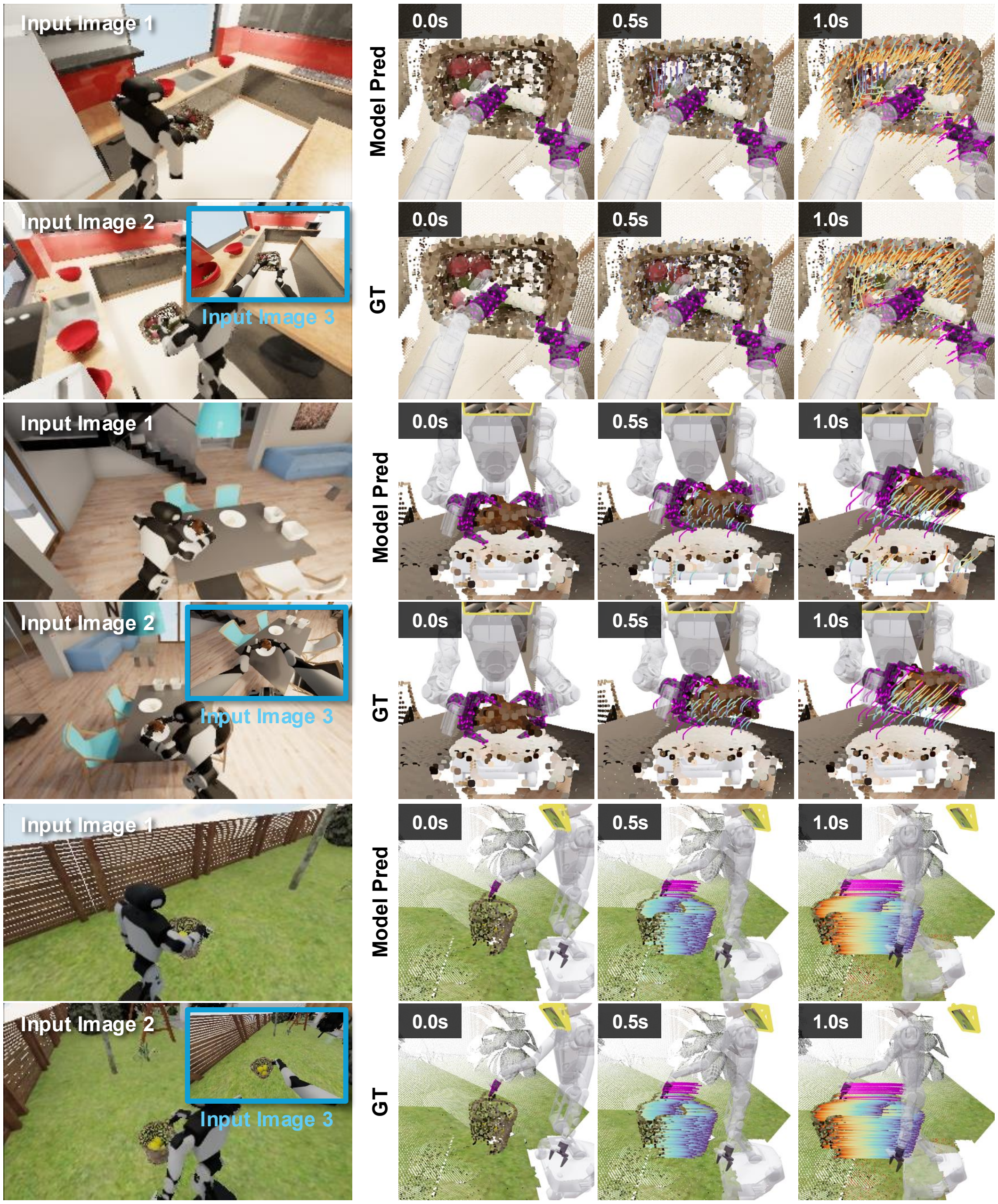

Figure 20: **BEHAVIOR-1K Unseen Rollouts**, including object-object interactions (within basket), gravity effects (within basket), and whole-body behaviors.

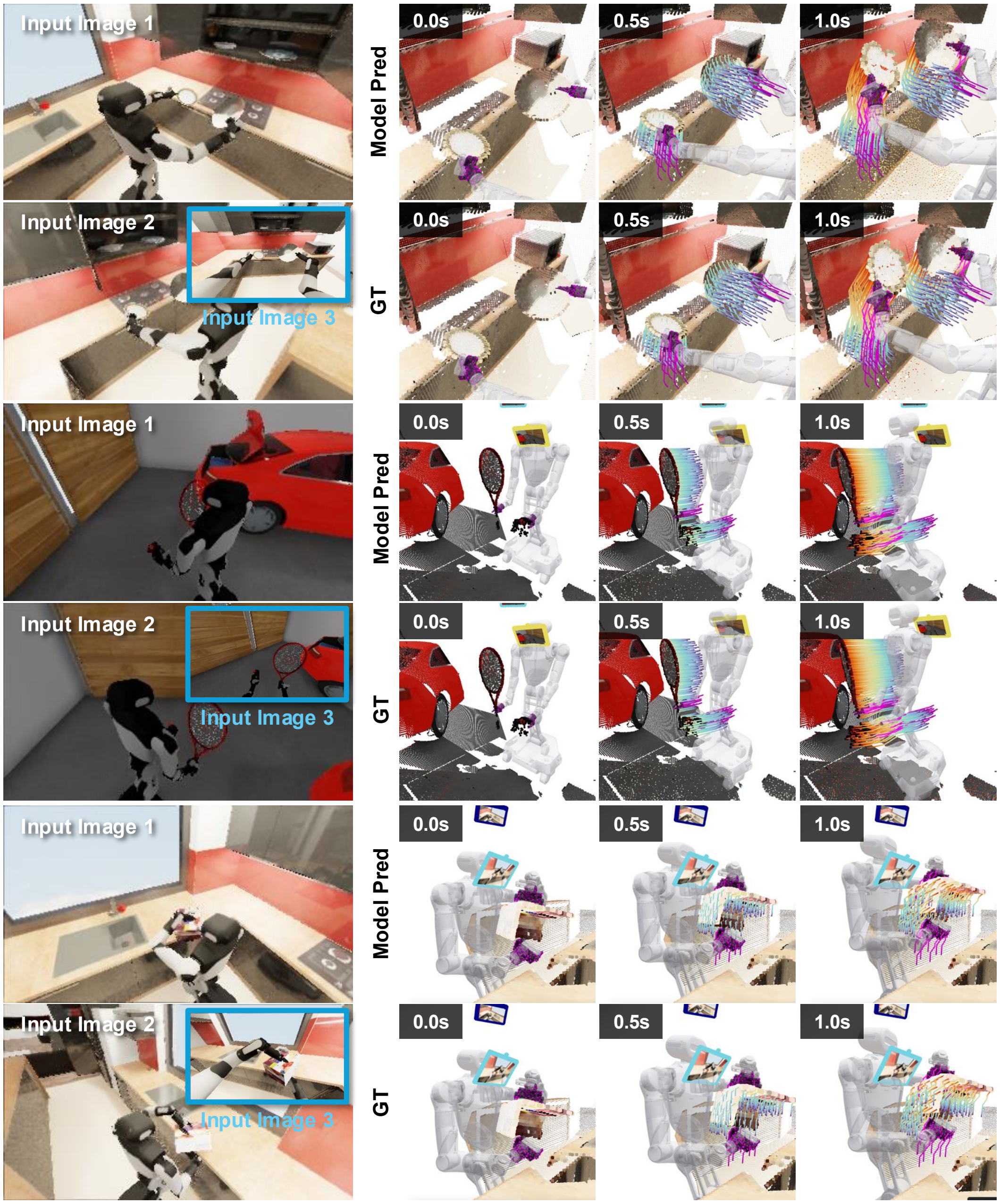

Figure 21: **BEHAVIOR-1K Unseen Rollouts**, including bimanual manipulation, whole-body behaviors, and implicit shape completion.