2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Real-to-Sim Robot Policy Evaluation with Gaussian Splatting Simulation of Soft-Body Interactions

Kaifeng Zhang[1,2*], Shuo Sha[1,2*], Hanxiao Jiang[1], Matthew Loper[2], Hyunjong Song[2], Guangyan Cai[2], Zhuo Xu[3], Xiaochen Hu[2], Changxi Zheng[1,2], Yunzhu Li[1,2]

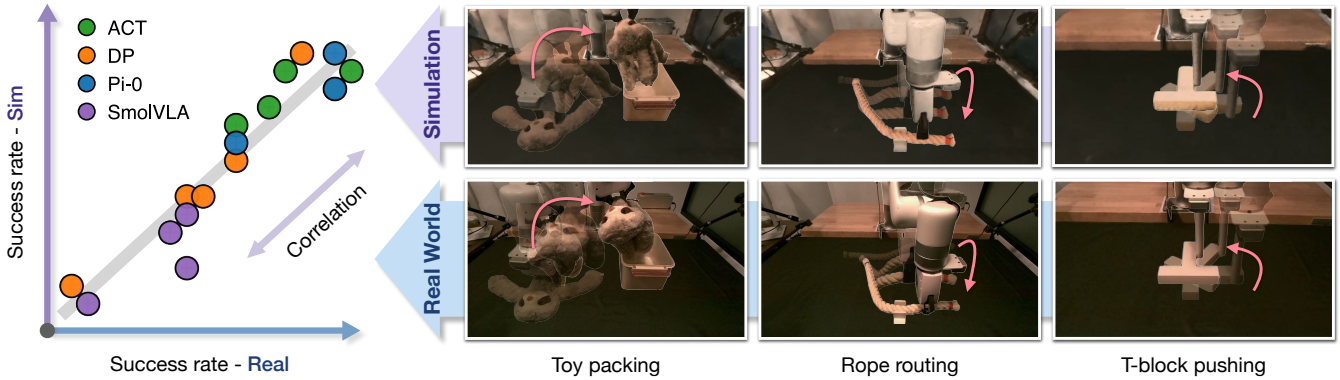

Fig. 1: **Real-to-sim policy evaluation with Gaussian Splatting simulation.** *Left:* Correlation between simulated and real-world success rates across multiple policies (ACT [1], DP [2], Pi-0 [3], SmolVLA [4]) shows that our simulation reliably predicts real-world performance. *Right:* Representative tasks used for evaluation, including plush toy packing, rope routing, and T-block pushing, are visualized in both real and simulated settings. Our framework reconstructs soft-body digital twins from real-world videos and achieves realistic appearance and motion, enabling scalable and reproducible policy assessment.

***Abstract*— Robotic manipulation policies are advancing rapidly, but their direct evaluation in the real world remains costly, time-consuming, and difficult to reproduce, particularly for tasks involving deformable objects. Simulation provides a scalable and systematic alternative, yet existing simulators often fail to capture the coupled visual and physical complexity of soft-body interactions. We present a real-to-sim policy evaluation framework that constructs soft-body digital twins from real-world videos and renders robots, objects, and environments with photorealistic fidelity using 3D Gaussian Splatting. We validate our approach on representative deformable manipulation tasks, including plush toy packing, rope routing, and T-block pushing, demonstrating that simulated rollouts correlate strongly with real-world execution performance and reveal key behavioral patterns of learned policies. Our results suggest that combining physics-informed reconstruction with high-quality rendering enables reproducible, scalable, and accurate evaluation of robotic manipulation policies. Website: `https://real2sim-eval.github.io/`**

I. Introduction

In recent years, robotic manipulation policies have advanced rapidly across a wide range of tasks [1–4]. However, their evaluation still relies heavily on real-world trials, which are slow, expensive, and difficult to reproduce. As the community shifts toward training foundation models for robotics [5–9], whose development depends on rapid iteration and large-scale benchmarking, this reliance has become a significant bottleneck.

Simulation offers a scalable and systematic alternative and is widely used for data generation and training [10–15]. Yet it is far less common as a tool for policy evaluation, primarily due to poor sim-to-real correlation: a policy that performs well in simulation often fails to translate to similar real-world success. Narrowing this gap would allow simulation to serve as a trustworthy proxy for real-world testing, greatly accelerating development cycles. This raises the central question: *how can we design simulators that are sufficiently realistic to evaluate robot policies with confidence?* To answer this question, we propose a framework for building high-fidelity simulators and investigate whether they can predict real-world policy performance reliably.

We identify two key factors for aligning simulation with reality: *appearance* and *dynamics*. On the appearance side, rendered scenes must closely match real-world observations. This is particularly challenging for policies that rely on wrist-mounted cameras, where simple green-screen compositing [16] is insufficient. We address this by leveraging 3D Gaussian Splatting (3DGS) [17], which reconstructs photorealistic scenes from a single scan and supports rendering from arbitrary viewpoints. Beyond prior uses of 3DGS for simulation [18–21], we enhance it with automatic position and color alignment and object deformation handling, which are essential for closing the appearance gap.

Dynamics present another major source of sim-to-real discrepancy. Traditional simulators rely on low-dimensional parameter tuning, which is insufficient for deformable objects with many degrees of freedom. To address this challenge, we adopt PhysTwin [22], a framework that reconstructs deformable objects as dense spring-mass systems optimized directly from object interaction videos. This approach yields efficient system identification while closely matching real-world dynamics.

[1]Columbia University [2]SceniX Inc. [3]Google DeepMind
* Equal contribution. Work partially done while interning at SceniX Inc.

We integrate these appearance and dynamics components into a unified simulator and expose it through a Gym-style interface [23]. We evaluate this framework on representative rigid- and soft-body manipulation tasks, including toy packing, rope routing, and T-block pushing, using widely adopted imitation learning algorithms: ACT [1], Diffusion Policy (DP) [2], Pi-0 [3], and SmolVLA [4]. By comparing simulated and real-world success rates and performing ablation studies, we observe a strong correlation and confirm that rendering and dynamics fidelity are both crucial to the trustworthiness of simulation-based evaluation.

In summary, our main contributions are: (1) A complete framework for evaluating robot policies in a Gaussian Splatting-based simulator using soft-body digital twins. (2) Empirical evidence that simulated rollouts strongly correlate with real-world success rates across representative tasks, using policies trained *exclusively* on real-world data (no co-training). (3) A detailed analysis of design choices that improve the reliability of simulation as a predictor of real-world performance, offering guidance for future simulation-based evaluation pipelines.

II. Related Works

A. Robot Policy Evaluation

Evaluating robot policies is essential for understanding and comparing policy behaviors. Most systems are still evaluated directly in the real world [24–29], but such evaluations are costly, time-consuming, and usually tailored to specific tasks, embodiments, and sensor setups. To enable more systematic study, prior works have introduced benchmarks, either in the real world through standardized hardware setups [30–33] or in simulation through curated assets and task suites [34–40]. Real-world benchmarks offer high fidelity but lack flexibility and scalability, while simulators often suffer from unrealistic dynamics and rendering, which limits their reliability as proxies for physical experiments. This is widely referred to as the "sim-to-real gap" [41–44]. We aim to narrow this gap by building a realistic simulator that combines high-quality rendering with faithful soft-body dynamics. Compared to SIMPLER [16] and RobotArena ∞ [45] which relies on green-screen compositing, Ctrl-World [46] which relies on video models, and Real-is-sim [18] which focuses on rigid-body simulation, our method integrates Gaussian Splatting-based rendering with soft-body digital twins derived from videos, eliminating the dependence on static cameras and providing more realistic appearance and dynamics.

B. Physical Digital Twins

Digital twins seek to realistically reconstruct and simulate real-world objects. Many existing frameworks rely on pre-specified physical parameters [47–50], which limits their ability to capture complex real-world dynamics or leverage data from human interaction. While rigid-body twin is well studied [51–54], full-order parameter identification for deformable objects remains challenging. Learning-based approaches have been proposed to capture such dynamics [55–57], but they often sacrifice physical consistency, which is critical for evaluating manipulation policies in contact-rich settings. Physics-based methods that optimize physical parameters from video observations [58–60] offer a more promising path. Among them, PhysTwin [22] reconstructs deformable objects as dense spring-mass systems directly from human-object interaction videos, achieving state-of-the-art realism and efficiency. Our work builds on PhysTwin and integrates its reconstructions with a Gaussian Splatting simulator to bridge the dynamics gap in policy evaluation.

C. Gaussian Splatting Simulators

Building simulators that closely match the real world requires high-quality rendering and accurate physics. 3D Gaussian Splatting (3DGS) [17] has recently emerged as a powerful approach for scene reconstruction, enabling photo-realistic, real-time rendering [48, 53]. Several studies have demonstrated its potential in robotics, showing that 3DGS-based rendering can improve sim-to-real transfer for vision-based policies [19, 61, 62], augment training datasets [20, 21, 63, 64], and enable real-to-sim evaluation [18, 65]. We extend this line of work by supporting soft-body interactions, incorporating PhysTwin [22] for realistic dynamics, and introducing automated position and color alignment, resulting in a complete and evaluation-ready simulator.

III. Method

A. Problem Definition

We study the policy evaluation problem: *Can a simulator reliably predict the real-world performance of visuomotor policies trained with real data?* In a typical evaluation pipeline [24, 66], multiple policies are executed across controlled initial configurations in both simulation and the real world, and performance is measured through rollout-based metrics, typically expressed as scalar scores $u \in [0,1]$. The objective is to establish a strong correlation between simulated and real-world outcomes, represented by the paired set $\{(u_{i,\text{sim}}, u_{i,\text{real}})\}_{i=1}^{N}$, where $u_{i,\text{sim}}$ and $u_{i,\text{real}}$ denote the performance of the i-th policy in simulation and reality, respectively, and N is the number of evaluated policies.

To achieve better performance correlation, one promising way is to build a simulator that yields consistent results with the real world. Formally, let $\{(s_t, o_t, a_t)\}_{t=1}^{T}$ denote the sequence of environment states s_t, robot observations o_t, and robot actions a_t over a time horizon T. A simulator for policy evaluation should contain two core components: (1) *Dynamics model:* $s_{t+1} = f(s_t, a_t)$, which predicts future states given the current state and robot actions. (2) *Appearance model:* $o_t = g(s_t)$, which renders observations in the input modality required by the policy (e.g., RGB images). Accordingly, the fidelity of simulation can be assessed along two axes: (i) the accuracy of simulated dynamics, and (ii) the realism of rendered observations.

In this work, we address both axes by jointly reducing the visual gap and the dynamics gap. We employ physics-informed reconstruction of soft-body digital twins to align simulated dynamics with real-world object behavior, and use high-resolution Gaussian Splatting as the rendering engine to

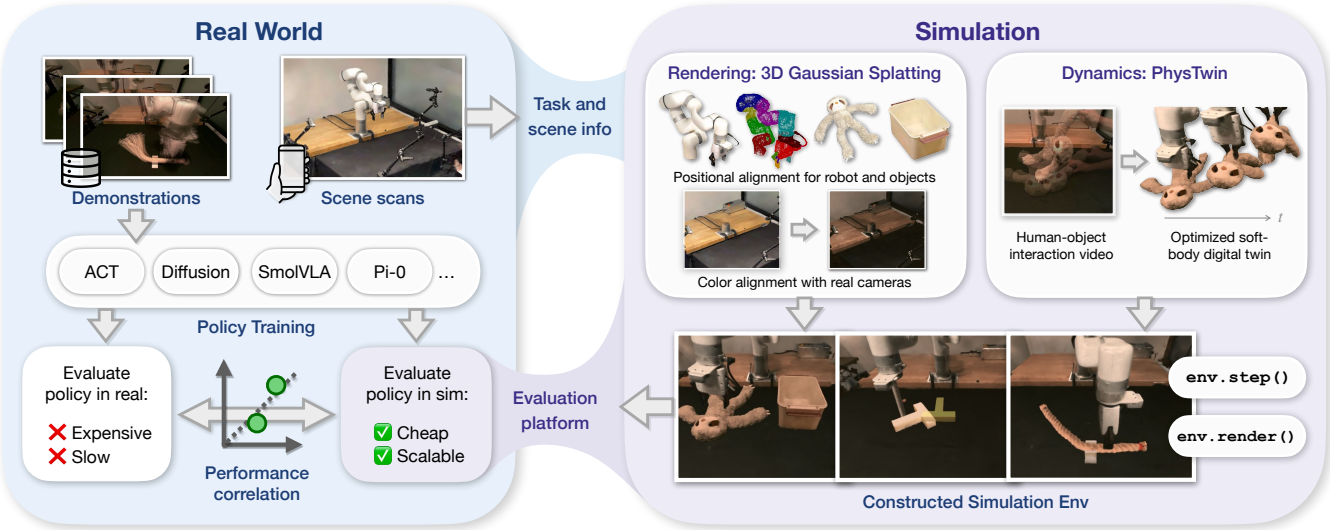

Fig. 2: **Proposed framework for real-to-sim policy evaluation.** We present a pipeline that evaluates real-world robot policies in simulation using Gaussian Splatting-based rendering and soft-body digital twins. Policies are first trained on demonstrations collected by the real robot, and a phone scan of the workspace is used to reconstruct the scene via Gaussian Splatting. The reconstruction is segmented into robot, objects, and background, then aligned in position and color to enable photorealistic rendering. For dynamics, we optimize soft-body digital twins from object interaction videos to accurately reproduce real-world behavior. The resulting simulation is exposed through a Gym-style API [23], allowing trained policies to be evaluated efficiently. Compared with real-world trials, this simulator is cheaper, reproducible, and scalable, while maintaining strong correlation with real-world performance.

generate photorealistic observations. The following sections describe these components in detail, and an overview of the full framework is shown in Figure 2.

B. Preliminary: PhysTwin

We adopt the PhysTwin [22] digital twin framework, which reconstructs and simulates deformable and rigid objects from video using a dense spring-mass system. Each object is represented as a set of mass nodes connected by springs, with springs formed between each pair of nodes within a distance threshold d. The node positions evolve according to Newtonian dynamics.

To capture the behavior of diverse real-world deformable objects with varying stiffness, friction, and other material properties, PhysTwin employs a real-to-sim pipeline that jointly optimizes a set of physical parameters, including the spring threshold d and per-spring stiffness coefficients Y. The optimization is performed from a single video of a human interacting with the object by hand: human hand keypoints are tracked and attached to the spring-mass system as kinematic control points, and system parameters are tuned to minimize the discrepancy between tracked object motions in the video and their simulated counterparts. For rigid bodies, Y is fixed to a large value to suppress deformation. We adopt this same real-to-sim process for system identification of the objects that interact with the robot (plush toy, rope, and T-block).

C. Real-to-Sim Gaussian Splatting Simulation

We now describe the construction of our Gaussian Splatting-based simulator. Our approach addresses two complementary goals: (i) closing the visual gap through GS scene reconstruction, positional alignment, and color alignment, and (ii) closing the dynamics gap through physics-based modeling and deformation handling.

1) GS Construction: We begin by acquiring the appearance of each object of interest using Scaniverse [67], an iPhone app that automatically generates GS reconstructions from video recordings. In a tabletop manipulation scene, we first scan the static robot workspace, including the robot, table, and background, then scan each experimental object individually. The resulting reconstructions are segmented into robot, objects, and background using the SuperSplat [68] interactive visualizer. This reconstruction step is required only once per task.

2) Positional Alignment: After obtaining GS reconstructions of the static background, robot, PhysTwin object, and other static objects, we align all components to the reference frames: the robot base frame and canonical object frames. PhysTwin objects and static meshes are aligned to their corresponding PhysTwin particle sets and object 3D models by applying a relative 6-DoF transformation. For the robot, we automatically compute the transformation between the reconstructed GS model and ground truth robot points (generated from its URDF) using a combination of Iterative Closest Point (ICP) [69] and RANSAC [70]. We use 2,000 points per link to ensure sufficient coverage of link geometry. Because the background GS is in the same frame as the robot GS, we apply the same transformation estimated by ICP.

To enable the simulation of the static robot GS, we associate each Gaussian kernel with its corresponding robot link through a link segmentation process. After ICP alignment, each kernel is assigned to a link by finding its nearest neighbor in the sampled robot point cloud and inheriting that point's link index. This process is applied to all links, including the gripper links, allowing us to render continuous arm motion as well as gripper opening and closing. The same procedure generalizes naturally to other robot embodiments

with available URDF models.

3) Color Alignment: A major contributor to the visual gap in GS renderings is that reconstructed scenes often lie in a different color space from the policy's training data, leading to mismatched pixel color distributions, which can affect policy performance. In our setting, GS reconstructions inherit the color characteristics of iPhone video captures, while policies are trained in the color space of the robot's cameras (e.g., Intel RealSense, which is known to introduce color shifts). To close this gap, we design a color transformation that aligns GS colors to the real camera domain.

We perform this alignment directly in RGB space. First, we render images from the scene GS at the viewpoints of the fixed real cameras, using the original Gaussian kernel colors and opacities. Next, we capture real images from the same viewpoints, forming paired data for optimization. We then solve for a transformation function f that minimizes the pixel-wise color discrepancy:

$$f^* = \arg\min_{f \in \mathscr{F}} \frac{1}{N} \sum_{i=1}^{N} \|f(p_i) - q_i\|_2, \quad p_i \in I_{GS}, \quad q_i \in I_{RS}, \quad (1)$$

where I_{GS} and I_{RS} denote GS renderings and real camera captures, N is the number of pixels, p_i and q_i are corresponding RGB values, and $\mathscr{F}$ is the function space. We parameterize $\mathscr{F}$ as the set of degree-d polynomial transformations:

$$f = \{f_i\}_{i=1}^{d}, \quad f_i \in \mathbb{R}^3, \quad (2)$$

$$f(p_i) = [f_0 \ f_1 \ \cdots \ f_d] \cdot [1 \ p_i \ \cdots \ p_i^d]^T, \quad (3)$$

which reduces the problem to a standard least-squares regression. We solve it using Iteratively Reweighted Least Squares (IRLS) [71] to improve robustness to outliers. Empirically, we find that a quadratic transform ($d = 2$) offers the best trade-off between expressivity and overfitting.

4) Physics and Deformation: With GS reconstruction and alignment mitigating the rendering gap, the physics model must accurately capture real-world dynamics. We use a custom physics engine built on NVIDIA Warp [72], extending the PhysTwin [22] spring-mass simulator to support collisions with both robot end-effectors and objects in the environment. For grasping soft-body digital twins, we avoid the common but unrealistic practice of fixing object nodes to the gripper. Instead, we model contact purely through frictional interactions between gripper fingers and the object. The gripper closing motion halts automatically once a specified total collision-force threshold is reached, yielding more realistic and stable grasps.

At each simulation step, the updated robot and environment states from the physics engine are propagated to the Gaussian kernels. For rigid bodies, including objects and robot links, kernel positions and orientations are updated using the corresponding rigid-body transformations. For deformable objects, following PhysTwin [22], we apply Linear Blend Skinning (LBS) [73] to transform each kernel based on the underlying soft-body deformation.

Overall, with GS rendering, the physics solver, and LBS-based deformation being the major computational steps, our simulator runs at 5 to 30 FPS on a single GPU, depending on the robot-object contact states. By eliminating the overhead of real-world environment resets and leveraging multi-GPU parallelization, we empirically achieve evaluation speeds several times faster than real-world execution.

D. Policy Evaluation

To evaluate visuomotor policies in our simulator, we first design tasks and perform real-world data collection and policy training. Demonstrations are collected through human teleoperation using GELLO [74], after which we scan the scene to construct the corresponding simulation environments. All policies are trained *exclusively* on real data (i.e., no co-training between simulation and reality). To improve consistency and reduce variance, we follow the practice of Kress-Gazit et al. [66] by defining a fixed set of initial object configurations for each task and performing evaluations in both simulation and the real world. In the real world, we use a real-time visualization tool that overlays simulated initial states onto live camera streams, enabling operators to accurately and consistently reproduce the starting configurations.

Policy performance u is measured in terms of binary task success rates: in the real world, success is determined by human evaluators, while in simulation, task-specific criteria are automatically computed from privileged simulation states. In this work, we evaluate the performance of several state-of-the-art imitation learning algorithms, as well as checkpoints from different training stages for each network. Notably, the simulator is readily extensible to other policy types, as we package the entire system into the widely adopted Gym environment API [23]. We are committed to open-sourcing our implementation to encourage community adoption and enable scalable, reproducible policy evaluation.

IV. Experiments

In this section, we test the performance of imitation learning policies in both the real world and our simulation environment to examine the correlation. We aim to address the following questions: (1) How strongly do the simulation and real-world performance correlate? (2) How critical are rendering and dynamics fidelity for improving this correlation? (3) What practical benefits can the correlation provide?

A. Experiment Setup

1) Tasks: We evaluate policies on three representative manipulation tasks involving both deformable and rigid objects:

- *Toy packing*: The robot picks up a plush sloth toy from the table and packs it into a small plastic box. A trial is considered successful only if the toy's arms, legs, and body are fully contained within the box, with no parts protruding.
- *Rope routing*: The robot grasps a cotton rope, lifts it, and routes it through a 3D-printed clip. Success is defined by the rope being fully threaded into the clip.
- *T-block pushing (push-T)*: A 3D-printed T-shaped block is placed on the table. Using a vertical cylindrical

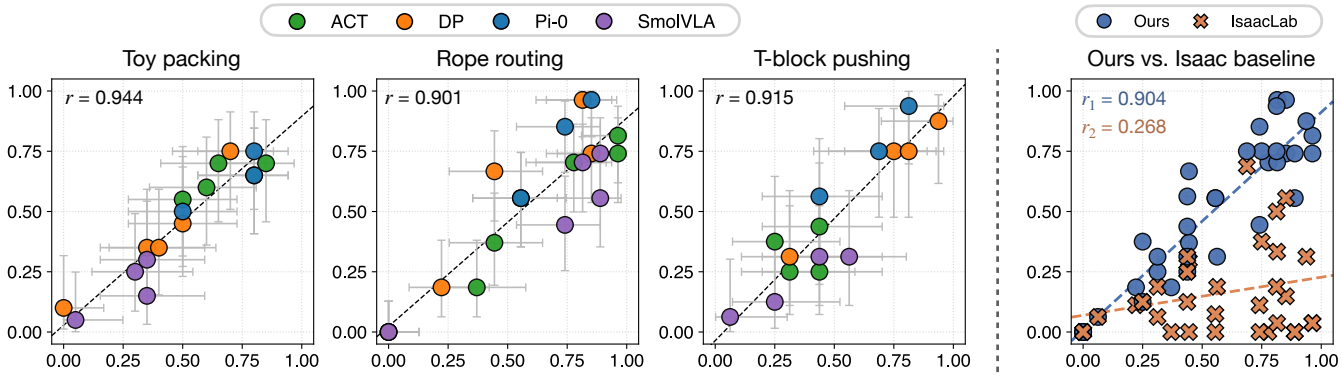

Fig. 3: **Correlation between simulation and real-world policy performance.** *Left:* Simulation success rates (y-axis) vs. real-world success rates (x-axis) for toy packing, rope routing, and T-block pushing, across multiple state-of-the-art imitation learning policies and checkpoints. The tight clustering along the diagonal indicates that, even with binary success metrics, our simulator faithfully reproduces real-world behaviors across tasks and policy robustness levels. *Right:* Compared with IsaacLab, which models rope routing and push-T tasks, our approach yields substantially stronger sim-to-real correlation, highlighting the benefit of realistic rendering and dynamics.

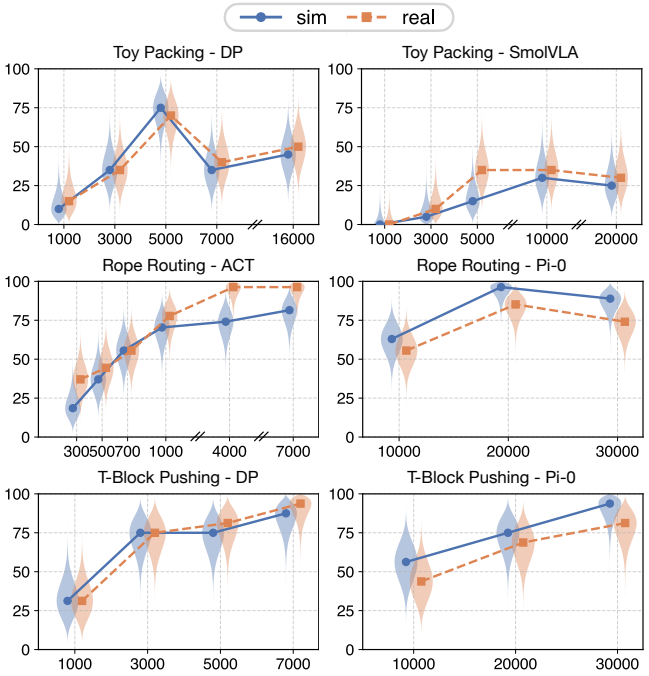

Fig. 4: **Per-policy, per-task performance across training.** x-axis: training iterations, y-axis: success rates. Simulation (blue) and real-world (orange) success rates are shown across iterations. Unlike Figure 3, which aggregates across policies, this figure shows unrolled curves for each task-policy pair. Improvements in simulation consistently correspond to improvements in the real world, establishing a positive correlation and demonstrating that our simulator can be a reliable tool for evaluating/selecting policies.

pusher, the robot must contact the block and then translate and reorient it to match a specified target pose.

Both the toy packing and rope routing tasks are challenging because the small tolerances of the box and clip require the policy to leverage visual feedback. Similarly, in push-T, the policy must infer the block's pose from images to achieve the required translation and reorientation.

2) Evaluation: To reduce variance and ensure systematic evaluation, we initialize scenes from a fixed set of configurations shared between the simulation and the real world. These initial configurations are generated in our simulator by constructing a grid over the planar position (x, y) and rotation angle θ of objects placed on the table. The grid ranges are chosen to ensure that the evaluation set provides coverage comparable to the training distribution. In the real world, objects are positioned to replicate the corresponding grid states. We use an evaluation set size of 20, 27, and 16 for toy packing, rope routing, and push-T, respectively.

We use binary success criteria for all tasks. Following [16], we quantify the alignment between simulation and real-world performance using the Mean Maximum Rank Variation (MMRV) and the Pearson correlation coefficient (r).

The number of evaluation episodes plays a critical role in the uncertainty of measured success rates [24]. To capture this variability, we report uncertainty in our results using the Clopper–Pearson confidence interval (CI). We also visualize the Bayesian posterior of policy success rates under a uniform Beta prior with violin plots.

We evaluate four state-of-the-art imitation learning policies: ACT [1], DP [2], Pi-0 [3], and SmolVLA [4]. The real-world setup consists of a single UFactory xArm 7 robot arm equipped with two calibrated Intel RealSense RGB-D cameras: a D405 mounted on the robot wrist and a D455 mounted on the table as a fixed external camera. All policies take as input images from both camera views, along with the current end-effector state. For push-T, the end-effector state includes only the 2D position (x, y); for the other tasks, it additionally includes the position, rotation, and gripper openness. Across all tasks, we collect 39-60 successful demonstrations via teleoperation using GELLO [74]. Training is performed using the open-source LeRobot [75] implementation, except for Pi-0, where we adopt the original implementation [3] for better performance.

B. Baseline

As a baseline, we use NVIDIA IsaacLab [10] as the simulation environment. Robot and environment assets are imported and aligned in position and color to match the real-world setup. IsaacLab provides a general-purpose robot simulation framework built on the PhysX physics engine, but its support for deformable objects remains limited. For ropes,

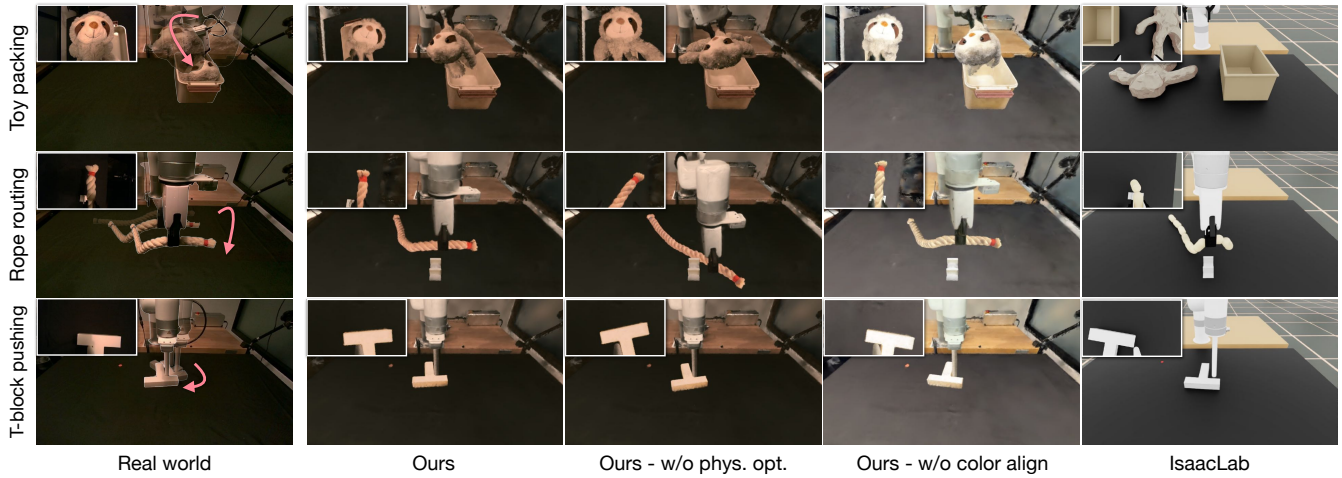

Fig. 5: **Comparison of rendering and dynamics quality.** Real-world observations (left) compared with our method, two ablations, and the IsaacLab baseline across three tasks. From right to left, visual and physical fidelity progressively improve. Without physics optimization, object dynamics deviate, causing failures such as the toy's limbs not fitting into the box or the rope slipping before routing. Without color alignment, rendered images exhibit noticeable color mismatches. The IsaacLab baseline (rightmost) shows lower realism in both rendering and dynamics compared to our approach.

we approximate deformable behavior using an articulated chain structure. However, for the plush toy, realistic grasping and deformation could not be stably simulated, making task completion infeasible; we therefore excluded this task from our quantitative comparisons.

C. Sim-and-Real Correlation

Figure 3 (left) shows the performance of all policy checkpoints in both simulation and the real world. We observe a strong correlation: policies that achieve higher success rates in reality also achieve higher success rates in our simulator, consistently across architectures and tasks. Figure 3 (right) further highlights that our simulator achieves stronger correlation than the IsaacLab baseline [10]. This is also confirmed by the quantitative results in Table I, with our simulator achieving a Pearson coefficient $r > 0.9$ for all policies. By contrast, the baseline yields only $r = 0.649$ on push-T, and an even lower $r = 0.237$ on rope routing as a result of the larger dynamics gap. The low MMRV value for the IsaacLab rope routing task arises from its consistently low success rates, which in turn produce fewer ranking violations.

D. Policy Performance Analysis

Figure 4 further illustrates per-policy, per-task performance curves across training iterations. We observe that simulation success rates generally follow the same progression as real-world success rates, further highlighting the correlation. For example, in the toy packing-DP case, both simulation and real success rates peak at iteration 5,000 and decline significantly by iteration 7,000. Similarly, in the rope routing-Pi-0 case, performance peaks around iteration 20,000. These results suggest that our simulator can be used as a practical tool for monitoring policy learning dynamics, selecting checkpoints for real-world testing, and setting approximate expectations for real-world performance.

In cases where simulation and real success rates do not overlap, such as toy packing-SmolVLA and rope routing-ACT, the simulator still captures the correct performance trend, even if the absolute success rates differ. We attribute these discrepancies to residual gaps in visual appearance and dynamics, as well as variance from the limited number of evaluation episodes (16–27 per checkpoint).

	Toy packing		Rope routing		T-block pushing	
	MMRV↓	r↑	MMRV↓	r↑	MMRV↓	r↑
IsaacLab [10]	-	-	0.270	0.237	0.196	0.649
Ours w/o color	0.200	0.805	0.343	0.714	0.354	0.529
Ours w/o phys.	0.241	0.694	0.248	0.832	**0.100**	0.905
Ours	**0.076**	**0.944**	**0.174**	**0.901**	0.108	**0.915**

TABLE I: **Quantitative comparison of correlation.** *Ours w/o color*: our method without color alignment. *Ours w/o phys.*: our method without physics optimization. Lower MMRV indicates fewer errors in ranking policy performance, while higher r reflects stronger statistical correlation. Best results are highlighted in bold.

E. Ablation Study

To measure the importance of the rendering and dynamics realism for our Gaussian Splatting simulator, we perform ablation studies on the correlation metrics MMRV and r. We provide two ablated variants of our simulation:

- *Ours w/o color alignment*: we skip the color alignment step in simulation construction and use the original GS colors in the iPhone camera space, creating a mismatch in the appearance.
- *Ours w/o physics optimization*: instead of using the fully-optimized spring stiffness Y, we use a global stiffness value shared across all springs. The global value is given by the gradient-free optimization stage in PhysTwin [22]. For push-T, we keep its rigidity and change its friction coefficients with the ground and the robot to create a mismatch in dynamics.

Figure 5 presents a visual comparison between our simulator, its ablated variants, and the baseline, using the same policy model and identical initial states. Our full method achieves the best rendering and dynamics fidelity, resulting in policy rollouts that closely match real-world outcomes.

In contrast, the w/o physics optimization variant produces inaccurate object dynamics, while the w/o color alignment variant shows clear color mismatches.

Empirically, both dynamics and appearance mismatches lead to deviations between simulated and real policy rollouts, though policies exhibit different sensitivities to each type of gap. For example, in the rope routing task, the rope fails to enter the clip when stiffness is mis-specified (*w/o physics optimization*). In the push-T task, color discrepancies alter the robot's perception, causing it to push the block differently (*w/o color alignment*).

Table I details the quantitative results. Overall, our full method achieves the highest correlation values, outperforming the ablated variants. In particular, lower MMRV values reflect more accurate policy ranking, while higher Pearson correlation coefficients (r) indicate stronger and more consistent correlations without being influenced by outlier points.

V. Conclusion

In this work, we introduced a framework for evaluating robot manipulation policies in a simulator that combines Gaussian Splatting-based rendering with real-to-sim digital twins for deformable object dynamics. By addressing both appearance and dynamics, our simulator narrows the sim-to-real gap through physics-informed reconstruction, positional and color alignment, and deformation-aware rendering.

We demonstrated the framework on representative deformable and rigid body manipulation tasks, evaluating several state-of-the-art imitation learning policies. Our experiments show that policy success rates in simulation exhibit strong correlations with real-world outcomes ($r > 0.9$). Further analysis highlights that our simulator can predict policy performance trends, enabling it to serve as a practical proxy for checkpoint selection and performance estimation. We found that both physics optimization and color alignment are critical for closing policy performance gaps.

In future work, scaling both simulation and evaluation to larger task and policy sets could provide deeper insights into the key design considerations for policy evaluation simulators. Moreover, our real-to-sim framework can be generalized to more diverse environments, supporting increasingly complex robot manipulation tasks.

Acknowledgment

This work is partially supported by the DARPA TIAMAT program (HR0011-24-9-0430), NSF Award #2409661, Toyota Research Institute, Sony Group Corporation, Samsung Research America, Google, Dalus AI, Pickle Robot, and an Amazon Research Award (Fall 2024). This article solely reflects the opinions and conclusions of its authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the sponsors. We would like to thank Wenhao Yu, Chuyuan Fu, Shivansh Patel, Ethan Lipson, Philippe Wu, and all other members of the RoboPIL lab at Columbia University and SceniX Inc. for helpful discussions and assistance throughout the project.

References

[1] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, *Learning fine-grained bimanual manipulation with low-cost hardware*, 2023. arXiv: 2304.13705 [cs.RO].

[2] C. Chi et al., "Diffusion policy: Visuomotor policy learning via action diffusion," in *RSS*, 2023.

[3] K. Black et al., π_0*: A vision-language-action flow model for general robot control*, 2024. arXiv: 2410.24164 [cs.LG].

[4] M. Shukor et al., *Smolvla: A vision-language-action model for affordable and efficient robotics*, 2025. arXiv: 2506.01844 [cs.LG].

[5] A. Brohan et al., "Rt-2: Vision-language-action models transfer web knowledge to robotic control," in *arXiv preprint arXiv:2307.15818*, 2023.

[6] M. J. Kim et al., "Openvla: An open-source vision-language-action model," *arXiv preprint arXiv:2406.09246*, 2024.

[7] P. Intelligence et al., $\pi_{0.5}$*: A vision-language-action model with open-world generalization*, 2025. arXiv: 2504.16054 [cs.LG].

[8] NVIDIA et al., "GR00T N1: An open foundation model for generalist humanoid robots," in *ArXiv Preprint*, Mar. 2025. arXiv: 2503.14734.

[9] G. R. Team et al., *Gemini robotics: Bringing ai into the physical world*, 2025. arXiv: 2503.20020 [cs.RO].

[10] NVIDIA, *NVIDIA Isaac Sim*, 2024.

[11] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *IROS*, 2012.

[12] F. Xiang et al., "SAPIEN: A simulated part-based interactive environment," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2020.

[13] C. Li et al., *Behavior-1k: A human-centered, embodied ai benchmark with 1,000 everyday activities and realistic simulation*, 2024. arXiv: 2403.09227 [cs.RO].

[14] G. Authors, *Genesis: A generative and universal physics engine for robotics and beyond*, Dec. 2024.

[15] R. Tedrake, *Drake: Model-based design and verification for robotics*, 2019.

[16] X. Li et al., "Evaluating real-world robot manipulation policies in simulation," in *CoRL*, 2024.

[17] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering," *ACM Transactions on Graphics*, vol. 42, no. 4, Jul. 2023.

[18] J. Abou-Chakra et al., *Real-is-sim: Bridging the sim-to-real gap with a dynamic digital twin*, 2025. arXiv: 2504.03597 [cs.RO].

[19] M. N. Qureshi et al., *Splatsim: Zero-shot sim2real transfer of rgb manipulation policies using gaussian splatting*, 2024. arXiv: 2409.10161 [cs.RO].

[20] X. Li et al., *Robogsim: A real2sim2real robotic gaussian splatting simulator*, 2024. arXiv: 2411.11839 [cs.RO].

[21] L. Barcellona et al., *Dream to manipulate: Compositional world models empowering robot imitation learning with imagination*, 2025. arXiv: 2412.14957 [cs.RO].

[22] H. Jiang, H.-Y. Hsu, K. Zhang, H.-N. Yu, S. Wang, and Y. Li, "Phystwin: Physics-informed reconstruction and simulation of deformable objects from videos," *ICCV*, 2025.

[23] G. Brockman et al., *Openai gym*, 2016. arXiv: 1606.01540 [cs.LG].

[24] T. L. Team et al., *A careful examination of large behavior models for multitask dexterous manipulation*, 2025. arXiv: 2507.05331 [cs.RO].

[25] J. Wang et al., *Evaluating pi0 in the wild: Strengths, problems, and the future of generalist robot policies*, 2025.

[26] A. Padalkar et al., "Open x-embodiment: Robotic learning datasets and rt-x models," *arXiv preprint arXiv:2310.08864*, 2023.

[27] A. Khazatsky et al., "Droid: A large-scale in-the-wild robot manipulation dataset," 2024.

[28] Z. Zhou et al., "Autoeval: Autonomous evaluation of generalist robot manipulation policies in the real world," *arXiv preprint arXiv:2503.24278*, 2025.

[29] P. Atreya et al., "Roboarena: Distributed real-world evaluation of generalist robot policies," *arXiv preprint arXiv:2506.18123*, 2025.

[30] B. Calli et al., "Benchmarking in manipulation research: Using the yale-cmu-berkeley object and model set," *IEEE Robotics & Automation Magazine*, vol. 22, no. 3, pp. 36–52, Sep. 2015.

[31] N. Correll et al., "Analysis and observations from the first amazon picking challenge," *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 1, pp. 172–188, 2016.

[32] G. Zhou et al., *Train offline, test online: A real robot learning benchmark*, 2023. arXiv: 2306.00942 [cs.RO].

[33] S. Dasari et al., *Rb2: Robotic manipulation benchmarking with a twist*, 2022. arXiv: 2203.08098 [cs.RO].

[34] S. Tao et al., "Maniskill3: Gpu parallelized robotics simulation and rendering for generalizable embodied ai," *RSS*, 2025.

[35] S. Srivastava et al., "Behavior: Benchmark for everyday household activities in virtual, interactive, and ecological environments," in *CoRL*, 2022, pp. 477–490.

[36] S. Nasiriany et al., "Robocasa: Large-scale simulation of everyday tasks for generalist robots," in *RSS*, 2024.

[37] Y. Zhu et al., *Robosuite: A modular simulation framework and benchmark for robot learning*, 2025. arXiv: 2009.12293 [cs.RO].

[38] X. Yang et al., *Robot policy evaluation for sim-to-real transfer: A benchmarking perspective*, 2025. arXiv: 2508.11117 [cs.RO].

[39] Y. R. Wang et al., *Roboeval: Where robotic manipulation meets structured and scalable evaluation*, 2025. arXiv: 2507.00435 [cs.RO].

[40] A. Badithela et al., "Reliable and scalable robot policy evaluation with imperfect simulators," *arXiv preprint arXiv:2510.04354*, 2025.

[41] X. B. Peng et al., "Sim-to-real transfer of robotic control with dynamics randomization," in *ICRA*, 2018, pp. 3803–3810.

[42] Y. Chebotar et al., "Closing the sim-to-real loop: Adapting simulation randomization with real world experience," in *ICRA*, IEEE, 2019, pp. 8973–8979.

[43] OpenAI et al., *Solving rubik's cube with a robot hand*, 2019. arXiv: 1910.07113 [cs.LG].

[44] D. Ho et al., *Retinagan: An object-aware approach to sim-to-real transfer*, 2021. arXiv: 2011.03148 [cs.RO].

[45] Y. Jangir et al., "Robotarena ∞: Scalable robot benchmarking via real-to-sim translation," *arXiv preprint arXiv:2510.23571*, 2025.

[46] Y. Guo, L. X. Shi, J. Chen, and C. Finn, "Ctrl-world: A controllable generative world model for robot manipulation," *arXiv preprint arXiv:2510.10125*, 2025.

[47] B. Chen et al., "Physgen3d: Crafting a miniature interactive world from a single image," in *CVPR*, 2025, pp. 6178–6189.

[48] Y. Jiang et al., "Vr-gs: A physical dynamics-aware interactive gaussian splatting system in virtual reality," in *SIGGRAPH*, 2024, pp. 1–1.

[49] T. Xie et al., "Physgaussian: Physics-integrated 3d gaussians for generative dynamics," in *CVPR*, 2024, pp. 4389–4398.

[50] R.-Z. Qiu et al., *Feature splatting: Language-driven physics-based scene synthesis and editing*, 2024. arXiv: 2404.01223 [cs.CV].

[51] B. Bianchini, M. Zhu, M. Sun, B. Jiang, C. J. Taylor, and M. Posa, "Vysics: Object reconstruction under occlusion by fusing vision and contact-rich physics," in *RSS*, Jun. 2025.

[52] W. Yang et al., *Twintrack: Bridging vision and contact physics for real-time tracking of unknown dynamic objects*, 2025. arXiv: 2505.22882 [cs.RO].

[53] J. Abou-Chakra et al., "Physically embodied gaussian splatting: A visually learnt and physically grounded 3d representation for robotics," in *CoRL*, 2024.

[54] K.-T. Yu, M. Bauza, N. Fazeli, and A. Rodriguez, "More than a million ways to be pushed. a high-fidelity experimental dataset of planar pushing," in *IROS*, IEEE, 2016, pp. 30–37.

[55] T. Pfaff, M. Fortunato, A. Sanchez-Gonzalez, and P. W. Battaglia, *Learning mesh-based simulation with graph networks*, 2021. arXiv: 2010.03409 [cs.LG].

[56] K. Zhang et al., "Adaptigraph: Material-adaptive graph-based neural dynamics for robotic manipulation," in *RSS*, 2024.

[57] K. Zhang et al., "Particle-grid neural dynamics for learning deformable object models from rgb-d videos," in *RSS*, 2025.

[58] X. Li et al., "Pac-nerf: Physics augmented continuum neural radiance fields for geometry-agnostic system identification," *arXiv preprint arXiv:2303.05512*, 2023.

[59] T. Zhang et al., "Physdreamer: Physics-based interaction with 3d objects via video generation," in *ECCV*, Springer, 2024, pp. 388–406.

[60] C. Chen et al., "Vid2sim: Generalizable, video-based reconstruction of appearance, geometry and physics for mesh-free simulation," in *CVPR*, 2025, pp. 26 545–26 555.

[61] X. Han et al., "Re^{3}sim: Generating high-fidelity simulation data via 3d-photorealistic real-to-sim for robotic manipulation," *arXiv preprint arXiv:2502.08645*, 2025.

[62] A. Escontrela et al., "Gaussgym: An open-source real-to-sim framework for learning locomotion from pixels," *arXiv preprint arXiv:2510.15352*, 2025.

[63] J. Yu et al., *Real2render2real: Scaling robot data without dynamics simulation or robot hardware*, 2025. arXiv: 2505.09601 [cs.RO].

[64] S. Yang et al., "Novel demonstration generation with gaussian splatting enables robust one-shot manipulation," *arXiv preprint arXiv:2504.13175*, 2025.

[65] G. Jiang et al., *Gsworld: Closed-loop photo-realistic simulation suite for robotic manipulation*, 2025. arXiv: 2510.20813 [cs.RO].

[66] H. Kress-Gazit et al., "Robot learning as an empirical science: Best practices for policy evaluation," *arXiv preprint arXiv:2409.09491*, 2024.

[67] Niantic, *Scaniverse*, https://scaniverse.com/.

[68] PlayCanvas and Snap Inc., *Supersplat*, https://github.com/playcanvas/supersplat, 2025.

[69] K. S. Arun et al., "Least-squares fitting of two 3-d point sets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-9, no. 5, pp. 698–700, 1987.

[70] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, Jun. 1981.

[71] P. J. Green, "Iteratively reweighted least squares for maximum likelihood estimation, and some robust and resistant alternatives," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 46, no. 2, pp. 149–170, 1984.

[72] M. Macklin, *Warp: A high-performance python framework for gpu simulation and graphics*, https://github.com/nvidia/warp, 2022.

[73] R. W. Sumner et al., "Embedded deformation for shape manipulation," vol. 26, no. 3, 80–es, Jul. 2007.

[74] P. Wu, Y. Shentu, Z. Yi, X. Lin, and P. Abbeel, "Gello: A general, low-cost, and intuitive teleoperation framework for robot manipulators," in *IROS*, 2024.

[75] R. Cadene et al., *Lerobot: State-of-the-art machine learning for real-world robotics in pytorch*, https://github.com/huggingface/lerobot, 2024.