Phys2Real: Fusing VLM Priors with Interactive Online Adaptation for Uncertainty-Aware Sim-to-Real Manipulation

Maggie Wang[1], Stephen Tian[1], Aiden Swann[1], Ola Shorinwa[2], Jiajun Wu[1], and Mac Schwager[1]

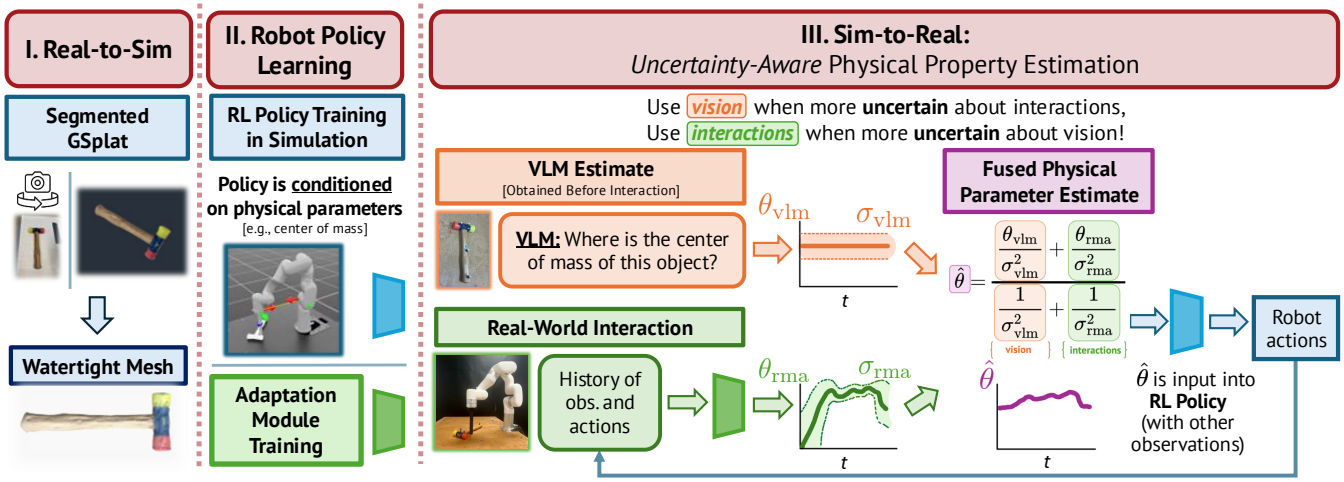


Fig. 1: Phys2Real is a real-to-sim-to-real pipeline for robotic manipulation that combines VLM-based physical parameter estimation with interaction-based adaptation through uncertainty-aware fusion. It comprises three stages: (I) *real-to-sim*: simulation-ready mesh reconstruction from segmented Gaussian Splats, (II) *policy learning*: reinforcement learning conditioned on physical parameters, e.g., center of mass, and (III) *sim-to-real transfer*: uncertainty-aware fusion of VLM priors and interaction-based estimates for online adaptation.

***Abstract*— Learning robotic manipulation policies directly in the real world can be expensive and time-consuming. While reinforcement learning (RL) policies trained in simulation present a scalable alternative, effective sim-to-real transfer remains challenging, particularly for tasks that require precise dynamics. To address this, we propose Phys2Real, a real-to-sim-to-real RL pipeline that combines vision-language model (VLM)-inferred physical parameter estimates with interactive adaptation through uncertainty-aware fusion. Our approach consists of three core components: (1) high-fidelity geometric reconstruction with 3D Gaussian splatting, (2) VLM-inferred prior distributions over physical parameters, and (3) online physical parameter estimation from interaction data. Phys2Real conditions policies on interpretable physical parameters, refining VLM predictions with online estimates via ensemble-based uncertainty quantification. On planar pushing tasks of a T-block with varying center of mass and a hammer with an off-center mass distribution, Phys2Real achieves substantial improvements over a domain randomization baseline: 100% vs 79% success rate for the bottom-weighted T-block, 57% vs 23% in the challenging top-weighted T-block, and 15% faster average task completion for hammer pushing. Ablation studies indicate that the combination of VLM and interaction information is essential for success. Project website: `https://phys2real.github.io/`**

This work is in part supported by ONR N00014-23-1-2355, ONR MURI N00014-22-1-2740, ONR MURI N00014-24-1-2748, NSF RI #2338203, and NSF FRR grant 2342246. M. Wang is supported by the NASA NSTGRO Fellowship and NSF grant 2342246. S. Tian and A. Swann are supported by NSF GRFP Grant No. DGE-1656518 and DGE-2146755, respectively.

[1]Stanford University, Stanford, CA, USA.
[2]Princeton University, Princeton, NJ, USA.

I. Introduction

Deploying robotic manipulation policies trained in simulation to the real world remains a fundamental challenge, especially for tasks requiring fine-grained physical dynamics. Robots must adapt to varying object properties such as friction, mass distribution, and compliance, which significantly affect manipulation outcomes but are difficult to model precisely. While learning from demonstrations has shown significant promise, it often lacks the physical grounding and reasoning needed to adapt to novel objects. Reinforcement learning (RL) provides a mechanism for real-time adaptation, but bridging the sim-to-real gap remains a critical obstacle.

Domain randomization (DR) is the dominant approach for sim-to-real transfer when training robotic policies with RL. By training policies across randomized simulation parameters, DR trains policies robust to real-world variations [1], but they may generalize poorly to out-of-distribution object physical properties. Even when dynamics lie within the training distribution, policies often default to averaged behaviors that may not account for object-specific variations.

We propose an alternative approach that efficiently balances robustness with performance. Specifically, we train policies to be not only *robust* to broad parameter ranges, but also *adaptive* to the specific physical properties of objects for superior performance, motivated by the question: **Can combining visual physical reasoning with interactive learning improve robot manipulation performance in**

real-world environments after training in simulation?

Humans demonstrate remarkable exploration behaviors when manipulating novel objects in unseen settings. Initial judgments are formed about an object's physical properties from visual appearance, and then these estimates are refined through interaction [2]. This integration of perception and physical reasoning enables humans to adapt their manipulation strategies to new instances of objects with potentially varying properties without extensive interaction. Inspired by this observation, our approach seeks to provide robots with a similar ability to estimate and adapt to physical properties.

We present **Phys2Real**, a framework that bridges the sim-to-real gap by combining three components:

1) **Uncertainty-aware fusion of VLM priors with interactive adaptation**: We demonstrate that VLMs can provide physical parameter estimates (e.g., center of mass) that improve manipulation performance when combined with interaction-based parameter estimation. Our work demonstrates the novel application of VLMs to physical parameter estimation for real-time low-level closed-loop control, beyond standard high-level planning.
2) **Ensemble-based uncertainty quantification**: We decompose uncertainty into epistemic and aleatoric components for interaction-based parameter estimation, and then combine these estimates with VLM priors using inverse-variance weighting. This addresses the limitations of existing adaptation methods that cannot incorporate external priors and enables adaptation during intermittent contact scenarios common in manipulation.
3) **Physically-informed digital twins**: We combine 3D Gaussian Splatting reconstructions with online physical property estimation to create digital twins that incorporate both geometric and physical information, enabling strong sim-to-real transfer compared to purely visual or adaptation-based approaches.

Our approach builds on rapid motor adaptation (RMA) [3] but adapts it for physically interpretable parameter estimation rather than learned latents, enabling direct combination with VLM priors through uncertainty weighting. The key insight is that policies can condition directly on physically interpretable parameters estimated from vision and refined through interaction, rather than learning averaged behaviors across parameter distributions as in DR. Unlike recent work that uses VLM estimates to initialize simulation parameters [4], our method updates these estimates online using real-world interaction histories during closed-loop control.

We evaluate our approach on two planar pushing tasks: (1) *T-block pushing* with varying center of mass (CoM) achieved by placing a small metal weight at different locations of a 3D printed T-block, and (2) *Hammer pushing* with an off-center mass distribution. Our results demonstrate consistent improvements: Phys2Real achieves a success rate of 100% vs. 79% for DR when the weight is at the bottom of the T-block, and 57.14% vs. 23% for DR in the more challenging configuration with weight at the top of the T-block. Furthermore, on the hammer pushing task, it achieves 15% faster average task completion.

This work demonstrates that VLMs can provide interpretable, uncertainty-calibrated physical estimates for manipulation that can be combined with interaction history. Phys2Real is a step towards more general, adaptive robotic systems that learn from perception and physical interaction.

II. Related Work

Our approach bridges sim-to-real transfer methods, policy adaptation techniques, digital twin reconstruction, and physical reasoning with foundation models. While prior work has explored these areas individually, Phys2Real uniquely combines high-fidelity 3D reconstructions with VLM-based physical parameter estimates and online adaptation to learn adaptive policies for robotic manipulation.

A. Domain Randomization and System Identification

The sim-to-real gap remains a fundamental challenge for deploying policies learned in simulation to real-world environments. One class of approaches performs domain randomization (DR) [1], [5] to address this, randomizing simulation dynamics during training to develop policies robust to a range of environmental variations. Although DR-trained policies have shown success in several domains [6]–[9], they often default to averaged behaviors that sacrifice performance for robustness, failing to adapt to object-specific variations. Our method trains simulation parameter-conditioned policies that adapt online to specific conditions, rather than an unconditioned policy trained to be robust across all conditions.

An alternative approach is to perform system identification [10] to explicitly calibrate simulation parameters to match real-world observations. However, these methods often require manual parameter tuning and yield static models that cannot adapt to varying deployment conditions. We build on their extensions to online policy adaptation methods.

B. Online Policy Adaptation

Online policy adaptation methods train universal policies conditioned on environment parameters [11] and perform online system identification for inference-time adaptation [12]. Rapid Motor Adaptation (RMA) [3], initially demonstrated for legged locomotion, trains an RL policy with an adaptation model that uses privileged information during simulation training and infers environment properties from interaction history at runtime. Related techniques have shown success across drone flight [13], prehensile manipulation [14], tool use [15], and dexterous in-hand manipulation [16].

Although RMA is effective in scenarios such as locomotion and in-hand manipulation, where the robot makes frequent contact with its environment and manipulated objects, it is challenging to deploy to general manipulation settings due to intermittent contacts that lead to uninformative histories. Our work addresses this challenge by combining VLM estimates with an ensemble of online adaptation models that provide uncertainty-aware predictions. To do this, Phys2Real conditions policies directly on physically interpretable parameters (e.g., center of mass), rather than learned latents.

C. Digital Twin Simulations and Photorealistic Rendering

Another set of methods attempts to train transferable policies in simulation by creating photorealistic simulation

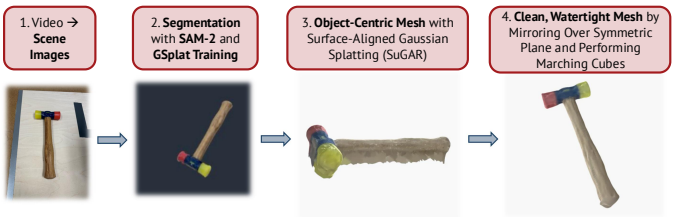


Fig. 2: **Real-to-sim mesh reconstruction pipeline.** From a video of the object, we extract image frames and segment the target object using SAM-2. We then train a GSplat and extract a surface-aligned object-centric mesh using SuGaR [33]. Finally, we generate a clean, watertight mesh, resulting in a simulation-ready asset.

environments, minimizing the sim-to-real gap. Neural Radiance Fields (NeRF) [17] and Gaussian Splatting (GSplat) [18]–[21] can reconstruct high-fidelity 3D scenes from a series of images. They can be used for simulated policy training [19], synthetic data generation [22], or as a mirrored surrogate environment [23].

Existing digital twins focus on visual fidelity, neglecting object physical properties. They usually rely on conventional physics engines with default parameters that may not match real-world dynamics. Towards addressing this, some approaches allow manual physical property specification [24] or estimate object geometry or robot physical properties from interaction data [25]–[29]. Most relevant to our work, Elhafsi et al. [4] use VLM estimates to initialize simulation parameters for simulated objects. However, our method updates these initial estimates online using real-world interaction histories while executing in closed-loop.

D. VLMs for Physical Reasoning

Recent work has demonstrated the physical reasoning capabilities of vision-language models (VLMs). PhysObjects [30] fine-tunes InstructBLIP to estimate attributes such as material properties, weight, and fragility from images. However, this work focuses primarily on using these estimates for high-level planning rather than low-level control.

Phys2Real builds on these insights by using VLMs to estimate physical parameters from images. Unlike previous work that uses VLMs primarily for high-level planning [31] or affordance prediction [32], we directly incorporate VLM-estimated physical parameters into the control policy, enabling more accurate manipulation.

III. Real-to-Sim-to-Real Policy Learning with Uncertainty-Aware Adaptation

Our method, Phys2Real, addresses the challenge of sim-to-real transfer by creating physically-informed digital twins and training adaptive manipulation policies. As illustrated in Fig. 1, our approach consists of three stages: (1) **real-to-sim** reconstruction to create geometrically accurate simulation assets, (2) **physics-parameter conditioned policy learning** in simulation with uncertainty-aware adaptation, and (3) **sim-to-real transfer** that fuses VLM priors with estimates inferred from interaction data.

A. Real-to-Sim Scene Reconstruction

For objects without known meshes, we must first obtain simulation-ready assets, ideally with minimal manual intervention. Thus, we develop a pipeline to reconstruct meshes

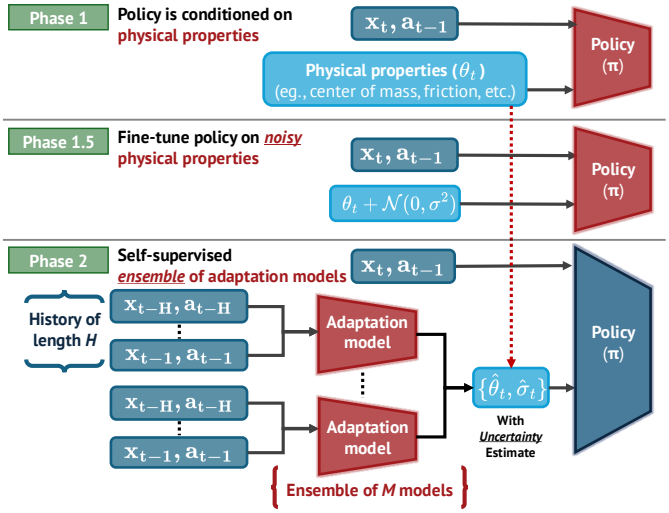


Fig. 3: **Phys2Real policy training.** The policy and adaptation models are trained in three stages, inspired by RMA [3]. **Phase 1**: the policy is conditioned on ground truth physical properties (e.g., CoM) from simulation, **Phase 1.5**: fine-tune with *noisy* physical properties to build robustness to downstream noisy estimates from the fused VLM and adaptation estimate. **Phase 2**: train an ensemble of M encoders that take in a history of observations and actions. The variance of the ensemble estimates provides an *epistemic* uncertainty (Eq. 1). Each encoder outputs a physical property estimate and an associated uncertainty, representing the model's *aleatoric* uncertainty (Eq. 2). At test time, we fuse the adaptation estimates and VLM estimates via inverse-variance weighting (Eq. 4).

directly from video frames. As shown in Fig. 2, we begin by capturing images of the target object and then segmenting them using SAM-2 [34]. We then train a 3D GSplat [18] on the object foreground images and extract a watertight mesh using SuGaR [33]. This process generates geometrically accurate assets for our simulation environment.

B. Physics-Conditioned Policy Learning

We train deep RL policies in simulation to manipulate objects. Motivated by evidence that policies with access to physical parameter information can improve performance over domain randomization approaches in locomotion [3], we explicitly condition the policy networks on physical parameter values.

To enable real-world deployments, we perform a second phase of training, during which the RL policy is frozen, and another neural network adaptation model learns to predict physical properties from historical state-action sequences. This approach is inspired by rapid motor adaptation (RMA) [3], which follows a similar two-phase learning paradigm: in the first phase, a physics encoder uses privileged information available only in simulation to produce a latent z (representing physical properties), which is then input into the RL policy. In the second phase, history information is fed into an adaptation model, which regresses on the latent z trained with the physics encoder from the first phase.

While RMA produces effective latent estimates in settings like locomotion, general manipulation tasks such as non-prehensile manipulation can often induce uninformative interaction histories, leading to poor environment parameter estimates. We propose to alleviate this by (1) making the

physics encoder uncertainty-aware, so that uninformed or poor guesses can be detected, and (2) fusing online information with VLM-informed visual estimates.

Thus, we extend RMA to *explicitly estimate physical parameters and their uncertainties*. By explicitly estimating physical parameters rather than latent vectors, we obtain interpretable outputs that are in the same representational space as VLM estimates. This is critical so that they can be fused with a linear combination weighted by respective uncertainties as described in Sec. III-C. This VLM prior is important for contact-rich, long-horizon manipulation because object physical properties may be unobservable before or during contact. A reasonable estimate of the object's physical parameters may be crucial to the policy's effectiveness.

To summarize, our policy training procedure consists of:

1) **Phase 1**: The policy is trained conditioned on ground-truth physical parameters available in simulation. This enables the policy to learn optimal behaviors for different physical configurations. Unlike standard RMA, we condition directly on interpretable physical parameters rather than learned latents, which is critical for downstream combination with VLM estimates.
2) **Phase 1.5 (optional)**: Since the policy at training time observes only ground truth physical parameter values, slightly incorrect VLM or physics encoder estimates at deployment time could lead to out-of-training-distribution states. We thus fine-tune the policy with noisy parameter estimates by applying random noise to each physical parameter during each episode. Our policies were fine-tuned with a Gaussian noise of $\sigma = 1.5\,\mathrm{cm}$.
3) **Phase 2:** We freeze the learned policy weights and train an ensemble of $M = 10$ adaptation models that learn to predict physical parameters from a history of observations and actions over a sliding window of $H = 10$. These models allow the policy to perform online adaptation during deployment. We obtain uncertainty estimates for the physical parameters using ensemble-based uncertainty quantification that captures both epistemic (model uncertainty) and aleatoric (data uncertainty) components.

As shown in Fig. 3, the actor is conditioned on object pose, robot end-effector position, and object physical properties (e.g., friction, CoM). The critic receives privileged observations, including object velocity and pose. We train policies using proximal policy optimization (PPO) [35] with 4096 parallel environments and an asymmetric actor-critic architecture in IsaacLab [36].

C. Sim-to-Real Transfer with Physical Parameter Estimation

To combat uninformative interaction histories, Phys2Real combines physical parameter estimates from online interaction with VLM-based visual reasoning. However, it is unclear how to effectively combine this information. One naïve approach is to simply condition the policy or adaptation model on the VLM estimate. But, this introduces additional training time hyperparameters, effectively increasing the state space. Instead, we use the insight that these estimates can be treated effectively as sensing information that can be fused at test time based on their *relative uncertainty*. When the uncertainty of the adaptation model is high, increased reliance can be placed on the VLM estimate, and vice versa.

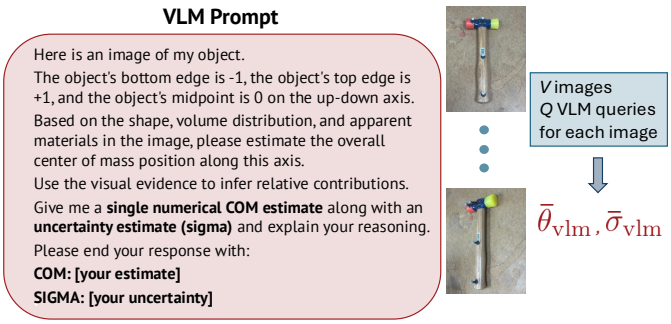


Fig. 4: **VLM priors for task-relevant physical parameters.** We query a VLM (GPT-5 [37]) to provide an estimated CoM and uncertainty given an image of the object. For each of V images, we repeat the query Q times. We then calculate the average VLM estimate $\bar{\theta}_{\mathrm{vlm}}$ along with the average uncertainty $\bar{\sigma}_{\mathrm{vlm}}$. This is fused with the RMA estimate using Eq. (4).

To instantiate this, both the VLM and adaptation model ensembles provide an estimate of the physical property and an associated uncertainty. We then fuse these estimates using inverse-variance weighting.

1) Obtaining θ_{vlm} and σ_{vlm}

We query GPT-5 [37] to estimate task-relevant physical parameters from images. For both the T-block and the hammer, we estimate the CoM along the vertical stem, which affects its rotational dynamics. As shown in Fig. 4, we capture V images from different viewpoints and query the VLM Q times for each image. We use the prompt shown in Fig. 4 to obtain physical parameter and uncertainty estimates.

The VLM estimate is computed as the aggregate mean across the prompts: $\theta_{\mathrm{vlm}} = \frac{1}{V \times Q} \sum_{i=1}^{V} \sum_{j=1}^{Q} \theta_{i,j}$.

To calculate σ_{vlm}, we take the mean of the uncertainty estimates from the VLM. We find empirically that computing the standard deviations of the estimates themselves can cause poor uncertainty estimates (i.e., the VLM may be confidently wrong). However, the model's estimate of its own uncertainty takes into account the uncertainty of an object's properties based purely on visual appearance.

2) Obtaining θ_{rma} and σ_{rma}

Given a sequence of observations $\{o_t, o_{t-1}, \ldots, o_{t-k}\}$ (where each observation includes the previous action, a_{t-1}), each ensemble member i produces an estimate θ_i of the physical parameter (e.g., CoM). The ensemble mean provides our parameter estimate, and the ensemble variance, given by

$$\sigma^2_{\mathrm{epistemic}} = \frac{1}{M} \sum (\theta_i - \theta_{\mathrm{rma}})^2. \tag{1}$$

is the epistemic uncertainty. Each ensemble member also estimates aleatoric uncertainty by outputting a mean μ_i and variance σ_i^2, trained using a Gaussian negative log-likelihood loss [39]. The ensemble's mean aleatoric uncertainty is thus

$$\sigma^2_{\mathrm{aleatoric}} = \frac{1}{M} \sum \sigma_i^2. \tag{2}$$

Our total RMA uncertainty combines both sources:

$$\sigma^2_{\mathrm{rma}} = \sigma^2_{\mathrm{epistemic}} + \sigma^2_{\mathrm{aleatoric}}. \tag{3}$$

This decomposition of uncertainties separates epistemic uncertainty, which captures model disagreement, from aleatoric uncertainty, which captures irreducible noise in observations and the environment. By combining both with a VLM prior, our method enables robust uncertainty-aware adaptation even when the robot is not in continuous contact with the object.

3) Fusing Estimates With Inverse-Variance Weighting

We combine physical property estimates from a VLM with estimates from RMA using uncertainty-based weights.

This fusion mechanism allows the system to assign greater weight to VLM physical property estimates when the RMA interaction history is uncertain, and conversely, rely more on RMA when the VLM estimates are less certain. The physically interpretable nature of θ enables direct combination of VLM priors with online adaptation, unlike approaches that use learned latent representations.

The fused estimate $\hat{\theta}$ is computed using inverse-variance weighting:

$$\hat{\theta} = \frac{\theta_{\text{vlm}}/\sigma^2_{\text{vlm}} + \theta_{\text{rma}}/\sigma^2_{\text{rma}}}{1/\sigma^2_{\text{vlm}} + 1/\sigma^2_{\text{rma}}}, \tag{4}$$

where θ_{vlm} and θ_{rma} are estimates from the VLM and the ensemble of adaptation models, and σ_{vlm} and σ_{rma} are their respective uncertainties. $\hat{\theta}$ is used to condition the policy at test time. Under the assumption that the estimates are unbiased with independent noise and known variances, this estimator is theoretically justified as the Best Linear Unbiased Estimator (BLUE), which minimizes the variance of the fused estimate.

IV. Experiments

In our experiments, we aim to understand how Phys2Real compares against baseline and privileged methods by investigating the following research questions:

Q1: Can accurate physical parameter estimation *improve policy performance* for robotic manipulation? How does Phys2Real compare with a physics-conditioned sim-to-real RL policy with privileged ground truth physical properties?

Q2: Are task-relevant physical parameters *estimated by VLMs* sufficient to improve the performance of a physics-conditioned sim-to-real RL policy?

Q3: Are task-relevant physical parameters *estimated through interaction adaptation* sufficient to improve the performance of a physics-conditioned sim-to-real RL policy?

Q4: Can Phys2Real improve policy performance when applied to *real-world objects without available object models*?

To address these questions, we evaluate Phys2Real on two non-prehensile manipulation tasks that require accurate physical modeling for successful execution:

- **T-block pushing:** To answer (**Q1**)-(**Q3**), we vary the CoM along the vertical axis of the T-block by adding a small metal weight of 143 grams. We evaluate all policies using two different configurations based on where the metal weight is placed, which alters the CoM of the object along the vertical axis: (1) **weight on the top**, where the weight is placed at $9.5\,\text{cm}$ above the geometric center of the T-block, and (2) **weight on the bottom**, where the weight is placed $6.5\,\text{cm}$ below the geometric center. This makes the ground truth CoM $6.1\,\text{cm}$ and $-0.7\,\text{cm}$ for the weights at top and bottom configurations, respectively. This task tests adaptation to varying object CoMs, which impacts the block's rotational dynamics during pushing. We use a known T-block mesh (as in DP [38]) to isolate the effects of physical parameter estimation.
- **Hammer pushing:** The CoM near the head of the hammer creates complex motion dynamics that must be accounted for during manipulation. We generate the hammer mesh through our real-to-sim reconstruction pipeline described by Fig. 2. We answer (**Q4**) with this task.

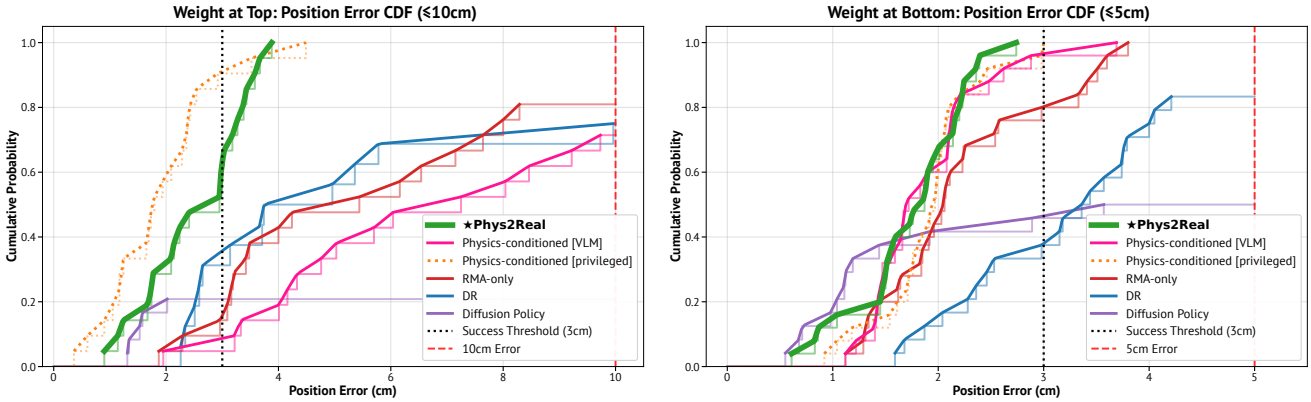
For each task, we measure four metrics: (1) success rate, (2) final position error in meters, (3) final orientation error in degrees, and (4) task completion time in seconds. We define success as achieving less than $3\,\text{cm}$ positional error and less than 20° orientation error relative to the target pose.

TABLE I: T-block pushing with the weight at the **top** (placed $9.5\,\text{cm}$ above the geometric center along the vertical axis; 21 trials per method with initial orientations varied every 45°). Arrows indicate whether higher (↑) or lower (↓) values are better, with the top-two values shown in bold. The privileged baseline is shaded in orange, and our method (Phys2Real) is highlighted in green with a star (*).

Method	Success Rate (%, ↑)	Pos. Err ± Std (cm, ↓)	Orient. Err ± Std (deg, ↓)	Time (s, ↓)
Phys2Real* [CoM $= +4.0\,\text{cm}$, $\sigma = 1.4\,\text{cm}$]	**57.14**	**2.60 ± 0.90**	**2.62 ± 1.73**	39.58
Physics-conditioned [VLM=$+4.0\,\text{cm}$]	4.76	10.10 ± 14.33	12.56 ± 32.54	40.80
Physics-conditioned [privileged=$+6.1\,\text{cm}$]	**90.48**	**1.90 ± 0.98**	**1.78 ± 1.71**	43.43
RMA-only [adaptation model] [3]	14.29	7.60 ± 8.10	4.70 ± 4.44	**37.23**
DR [$-3.5\,\text{cm}$, $+7.5\,\text{cm}$]	23.81	6.00 ± 5.78	6.77 ± 6.90	**37.00**
Diffusion Policy (DP) [38]	20.83	25.90 ± 16.89	57.38 ± 62.50	54.14

TABLE II: T-block pushing with the weight at the **bottom** (placed $6.5\,\text{cm}$ below the geometric center along the vertical axis; 24 trials per method with initial orientations varied every 45°).

Method	Success Rate (%)	Pos. Err ± Std (cm)	Orient. Err ± Std (deg)	Time (s)
Phys2Real* [CoM $= +0.76\,\text{cm}$, $\sigma = 1.55\,\text{cm}$]	**100.00**	**1.76 ± 0.54**	4.73 ± 2.68	44.28
Physics-conditioned [VLM=$+0.76\,\text{cm}$]	91.67	**1.90 ± 0.59**	**3.18 ± 1.69**	38.27
Physics-conditioned [privileged=$-0.71\,\text{cm}$]	**95.83**	1.92 ± 0.50	**2.76 ± 1.75**	**37.80**
RMA-only [adaptation model] [3]	79.17	2.23 ± 0.81	3.62 ± 2.10	45.22
DR [$-3.5\,\text{cm}$, $+7.5\,\text{cm}$]	79.17	7.14 ± 11.34	11.11 ± 13.86	**37.50**
Diffusion Policy (DP) [38]	50.00	19.34 ± 22.20	40.71 ± 52.00	38.71



(a) **CDF for weight at top configuration for position errors $\leq 10\,\mathrm{cm}$** (b) **CDF for weight at bottom configuration for position errors $\leq 5\,\mathrm{cm}$**

Fig. 5: **T-block pushing task results.** We compare the cumulative distribution functions (CDFs) of position errors at the end of rollouts from each policy, which show the percentage of experimental trials (y-axis) with position error less than or equal to a given value (x-axis). For an optimal policy, the curve should rise steeply and stay toward the left, indicating that most trials have low final error. We evaluate T-block pushing under two weight configurations: (a) weight at the top and (b) weight at the bottom. Overall, Phys2Real (green) consistently has low position error throughout the percentiles.

A. Experimental Setup

We experiment using a 6-DOF UFactory xArm robotic arm. The robot uses a cylindrical end-effector to push objects on a table. Its observations consist of object pose, robot end-effector xy positions, and estimated physical parameters. Its actions are changes in end-effector xy positions.

For real-world evaluation, we use motion capture to obtain accurate object poses. However, Phys2Real is designed to also handle visual inputs alone, and replacing motion capture with perception-based tracking is an exciting future direction.

B. Real-world Evaluation: T-block Pushing

Tables I and II show the performance of Phys2Real compared to several baselines under the two conditions (weight at top and bottom), which shift the object's CoM and result in distinct pushing dynamics and levels of difficulty for sim-to-real transfer. We also compare the performance of each method from the perspective of the cumulative distribution functions (CDF) of position errors in Figs. 5a and 5b. In these plots, the optimal policy has low error at every percentile.

1) Effect of Physical Parameter Estimation

Firstly, we would like to understand whether having an accurate estimate of the object's physical properties is beneficial for manipulation. To address **Q1**, we first compare Phys2Real against the following three methods.

The first two methods are baselines, DR and diffusion policy (DP), that are physics parameter *unaware*. For DR, the location of the weight on the T-block is randomized in simulation from the bottom to the top of the block. We use DP [38] as a strong imitation learning baseline that is also unaware of object physics parameters. We train a state-based DP using motion capture data, matching our state-based RL policies. We collect 100 demonstrations, varying the weight's location on the vertical axis of the T-block and the T-block's initial orientation in each episode, which is comparable to the domain randomization in simulation. We follow the implementation of [38].

The third method is a physics-conditioned policy with *privileged* information on the ground truth physics parameter, computed from the known T-block geometry assuming uniform density and the metal weight's measured mass. This baseline represents the oracle upper bound on performance.

When the weight is placed at the bottom of the T-block (Table II), Phys2Real achieves 100% success and the lowest position error across all methods. In comparison, DR and RMA without a VLM prior show lower success rates and higher errors. Diffusion policy also performs poorly. The results indicate that Phys2Real can match the performance of a privileged oracle by fusing vision-based priors and interactive adaptation, without using privileged information.

When we turn to the position error CDF in (Fig. 5b) for a more fine-grained understanding of position errors, we see that the Phys2Real curve rises sharply and up, indicating consistently low position error across nearly all trials. While DP has low error for around 40 percent of trials, the long tail is high. DR similarly has a long tail of high errors, showing that DP and DR easily become out of distribution, leading to high position errors. When the weight is at the top of the block, shown in Table I, all policies struggle to achieve a high success rate, except the privileged CoM-conditioned policy with a 90.48% success rate. We hypothesize that the higher CoM causes more unpredictable dynamics. Even in this case, Phys2Real is the second-best performer at 57.14%.

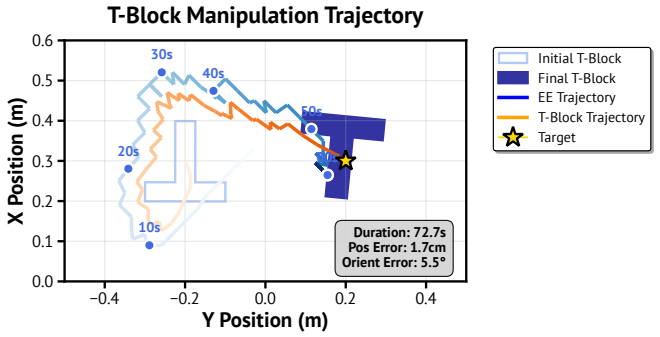
2) Impact of VLM-Only or RMA-Only Physical Parameter Estimation on Policy Performance

For **Q2**, we perform an ablation without the adaptation model, conditioning the policy only on the VLM estimate. Conversely, for **Q3**, we remove the VLM estimate and only condition the policy on the adaptation model's estimate.

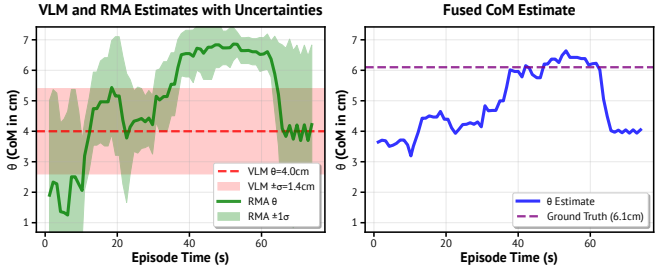
Before evaluating these ablations, we first visualize how the fused estimate evolves. Fig. 6a shows Phys2Real in a representative rollout with weight at the top, where the initial CoM estimate is uncertain (shown in Fig. 6b). As the robot interacts with the object, the uncertainty (in green) of the adaptation models decreases. The fused estimate (blue)

TABLE III: **Hammer pushing results.** Both Phys2Real and DR achieve 100% success rates across 15 trials. Here, the hammer's center of mass (CoM) and uncertainty (σ) are provided by the VLM: CoM $= 8.90\,\text{cm}$, $\sigma = 1.2\,\text{cm}$. To prevent the robot from continually pursuing small improvements, a trial is deemed successful if the hammer satisfies the success criteria for at least 2s.

Method	Success (%, ↑)	Pos. Err (cm, ↓)	Orient. Err (deg, ↓)	Time ± Std (s, ↓)
Phys2Real* [CoM $= +8.9\,\text{cm}$, $\sigma = 1.2\,\text{cm}$]	**100.00**	**1.74**	**4.68**	**77.79 ± 44.08**
DR [$-13\,\text{cm}$, $+13\,\text{cm}$]	**100.00**	**1.55**	**2.89**	**90.65 ± 42.03**



(a) **Object and end-effector trajectories.** The end-effector (blue) and T-block (orange) paths are shown as the block moves 40 cm to the right from an initial orientation of 180° (light blue outline) to a final orientation of 0° (dark blue filled shape) at the target location (gold star). The robot achieves a final position error of 1.7 cm.



(b) **Parameter estimates over time.** The VLM prior (red) provides an initial estimate ($\theta_{\text{vlm}} = 4.0\,\text{cm}$, $\sigma_{\text{vlm}} = 1.4\,\text{cm}$).

Fig. 6: **T-block with weight at the top.** Early in the task, the RMA estimate (green) is highly uncertain and far from the ground truth (∼6.0 cm). As contact continues, uncertainty decreases and the fused estimate (blue, Eq. 4) converges toward the ground truth at around 40s. After 55s, when contact ends, uncertainty rises again due to lack of new information.

converges to the ground truth of 6 cm. After interaction ends (60 s), uncertainty rises due to the lack of information about the CoM. This example illustrates how the VLM prior and interactive adaptation models are fused. We next evaluate how removing either source affects performance.

In the challenging weight at the top case, Phys2Real (57.14%) significantly outperforms the policy conditioned on just the VLM estimate (4.7%) as well as RMA-only (14.29%). This showcases that **neither the VLM estimate nor the interactive adaptation is enough to perform well individually; both sources of information are necessary for the policy to succeed**. We hypothesize that this is because the VLM CoM estimate is around 2 cm below the ground truth CoM, so the physics-conditioned on the VLM estimate policy pushes the object at more out-of-distribution locations. Similarly, without a VLM prior, the RMA-only policy pushes the object to out-of-distribution locations; because the initial parameter estimate is near the mean of the CoM distribution during Phase 2 training, it takes time for the adaptation model to adapt, and the first few initial pushes create out-of-distribution scenarios.

When the weight is placed on top (Fig. 5a), the differences become more pronounced. The privileged RMA phase 1 policy has the steepest CDF near the low-error region, while Phys2Real tracks closely behind, showing its ability to handle challenging, unstable CoM dynamics without privileged information. RMA-only and VLM-only baselines diverge further to the right, reflecting large errors and high failure rates. DP exhibits the longest tails. We hypothesize this is because during data collection, the human demonstrator can observe the weight location, which biases their actions, but this information is absent at test time.

These results show that Phys2Real consistently shifts both average performance and the full error distribution toward lower values, outperforming baselines and approaching the performance of a policy with privileged ground truth CoM information across both easy and challenging configurations.

We provide additional ablations evaluating robustness to different VLM priors in the extended version [40].

C. Real-world Evaluation: Hammer Pushing

Now that we have shown the success of Phys2Real on the T-block with a known mesh, can Phys2Real be extended to objects encountered in the real world (**Q4**)? To investigate this question, we transform real-world objects into simulation-ready assets. As shown in Fig. 2, images of the hammer are segmented with SAM-2 [34] and reconstructed into object-centric GSplats using SuGaR [33]. We mirror the GSplat across its primary axis of symmetry and apply the Marching Cubes algorithm to extract a clean, watertight mesh. While this pipeline works well for approximately symmetric objects like T-blocks and hammers, mirroring can distort the true shape and mass distribution of asymmetric objects. Extending to asymmetric objects would require alternative meshing strategies to preserve geometric fidelity.

For hammer pushing, both Phys2Real and DR achieve 100% success across 15 trials, as shown in Table III, demonstrating that the task is solvable for both methods with sufficient training. However, Phys2Real achieves faster task completion, with an average time of 77.79 s vs. 90.65 s for DR, a 14.2% improvement in task completion time, indicating that Phys2Real produces more efficient trajectories. Final position and orientation errors are comparable between the two methods.

V. Conclusion

We present Phys2Real, a real-to-sim-to-real pipeline that improves robot manipulation by combining VLM-based physical parameter estimates with interaction-based adaptation through uncertainty-aware fusion. This represents a

different paradigm from prior work that uses VLMs primarily for high-level reasoning, and from adaptive control techniques that lack foundation model priors. In real-world experiments, we show that Phys2Real outperforms DR across multiple tasks and physical configurations, achieving notable performance gains without ground-truth physical information at test time. This work demonstrates the potential of integrating visual geometry, physical understanding, and adaptive control for robotic manipulation. The shift toward adaptation through vision and interaction opens new directions that leverage foundation model capabilities while grounding them in physical interactions, enabling more adaptive and general robotic systems for novel object manipulation. Future work will explore inference-time adaptation to broader physical properties (e.g., friction, mass) in richer manipulation tasks.

ACKNOWLEDGMENTS

The authors thank John Tucker and Rohan Thakker for valuable discussions and feedback.

REFERENCES

[1] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[2] M. O. Ernst and M. S. Banks, "Humans integrate visual and haptic information in a statistically optimal fashion," *Nature*, vol. 415, pp. 429–433, 2002.

[3] A. Kumar, Z. Fu, D. Pathak, and J. Malik, "RMA: Rapid motor adaptation for legged robots," in *Robotics: Science and Systems*, 2021.

[4] A. Elhafsi, D. Morton, and M. Pavone, "Scan, materialize, simulate: A generalizable framework for physically grounded robot planning," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2026.

[5] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[6] F. Sadeghi and S. Levine, "CAD2RL: Real single-image flight without a single real image," in *Robotics: Science and Systems*, 2017.

[7] OpenAI, "Solving Rubik's cube with a robot hand," *arXiv preprint arXiv:1910.07113*, 2019.

[8] I. Exarchos, Y. Jiang, W. Yu, and C. K. Liu, "Policy transfer via kinematic domain randomization and adaptation," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[9] X. Chen, J. Hu, C. Jin, L. Li, and L. Wang, "Understanding domain randomization for sim-to-real transfer," in *International Conference on Learning Representations (ICLR)*, 2022.

[10] L. Ljung, *System identification: theory for the user*. USA: Prentice-Hall, Inc., 1986.

[11] W. Yu, J. Tan, C. K. Liu, and G. Turk, "Preparing for the unknown: Learning a universal policy with online system identification," in *Robotics: Science and Systems*, 2017.

[12] Z. Xu, J. Wu, A. Zeng, J. B. Tenenbaum, and S. Song, "DensePhysNet: Learning dense physical object representations via multi-step dynamic interactions," in *Robotics: Science and Systems*, 2019.

[13] J. Low, M. Adang, J. Yu, K. Nagami, and M. Schwager, "SOUS VIDE: Cooking visual drone navigation policies in a Gaussian splatting vacuum," *IEEE Robotics and Automation Letters*, 2025.

[14] Y. Liang, K. Ellis, and J. Henriques, "Rapid motor adaptation for robotic manipulator arms," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[15] X. Zhang, S. Liu, P. Huang, W. J. Han *et al.*, "Dynamics as prompts: In-context learning for sim-to-real system identifications," *IEEE Robotics and Automation Letters*, 2025.

[16] H. Qi, A. Kumar, R. Calandra, Y. Ma, and J. Malik, "In-hand object rotation via rapid motor adaptation," in *Conference on Robot Learning (CoRL)*, 2022.

[17] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron *et al.*, "NeRF: Representing scenes as neural radiance fields for view synthesis," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[18] B. Kerbl, G. Kopanas, T. Leimkuehler, and G. Drettakis, "3D Gaussian splatting for real-time radiance field rendering," *ACM Trans. Graph.*, 2023.

[19] M. N. Qureshi, S. Garg, F. Yandun, D. Held *et al.*, "SplatSim: Zero-shot sim2real transfer of RGB manipulation policies using Gaussian splatting," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2025.

[20] X. Li, J. Li, Z. Zhang, R. Zhang, F. Jia, T. Wang, H. Fan, K.-K. Tseng, and R. Wang, "RoboGSim: A real2sim2real robotic Gaussian splatting simulator," 2024.

[21] Y. Wu, L. Pan, W. Wu *et al.*, "RL-GSBridge: 3D Gaussian splatting based real2sim2real method for robotic manipulation learning," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2025.

[22] J. Yu, L. Fu, H. Huang, K. El-Refai *et al.*, "Real2render2real: Scaling robot data without dynamics simulation or robot hardware," 2025.

[23] J. Abou-Chakra, L. Sun, K. Rana, B. May *et al.*, "Real-is-sim: Bridging the sim-to-real gap with a dynamic digital twin for real-world robot policy evaluation," *arXiv preprint arXiv:2504.03597*, 2025.

[24] M. Torne, A. Simeonov, Z. Li, A. Chan, T. Chen, A. Gupta, and P. Agrawal, "Reconciling reality through simulation: A real-to-sim-to-real approach for robust manipulation," in *Robotics: Science and Systems*, 2024.

[25] N. Pfaff, E. Fu, J. Binagia, P. Isola, and R. Tedrake, "Scalable real2sim: Physics-aware asset generation via robotic pick-and-place setups," *arXiv preprint arXiv:2503.00370*, 2025.

[26] B. Moran, M. Comi, A. Byravan, S. Bohez, T. Erez, Z. Li, and L. Hasenclever, "Splatting physical scenes: End-to-end real-to-sim from imperfect robot data," *arXiv preprint arXiv:2506.04120*, 2025.

[27] B. Wang, X. Meng, X. Wang, Z. Zhu, A. Ye, Y. Wang, Z. Yang, C. Ni, G. Huang, and X. Wang, "EmbodieDreamer: Advancing real2sim2real transfer for policy training via embodied world modeling," *arXiv preprint arXiv:2507.05198*, 2025.

[28] 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 *Robotics: Science and Systems (RSS)*, 2025.

[29] W. Li, H. Zhao, Z. Yu, Y. Du, Q. Zou, R. Hu, and K. Xu, "PIN-WM: Learning physics-informed world models for non-prehensile manipulation," in *Robotics: Science and Systems (RSS)*, 2025.

[30] J. Gao, B. Sarkar, F. Xia, T. Xiao, J. Wu, B. Ichter, A. Majumdar, and D. Sadigh, "Physically grounded vision-language models for robotic manipulation," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

[31] C. Ning, K. Fang, and W.-C. Ma, "Prompting with the future: Open-world model predictive control with interactive digital twins," in *Robotics: Science and Systems (RSS)*, 2025.

[32] D. Guo, Y. Xiang, S. Zhao, X. Zhu, M. Tomizuka, M. Ding, and W. Zhan, "PhyGrasp: Generalizing robotic grasping with physics-informed large multimodal models," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2025.

[33] A. Guédon and V. Lepetit, "SuGaR: Surface-aligned Gaussian splatting for efficient 3D mesh reconstruction and high-quality mesh rendering," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[34] N. Ravi, V. Gabeur, Y.-T. Hu *et al.*, "SAM 2: Segment anything in images and videos," in *International Conference on Learning Representations (ICLR)*, 2025.

[35] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, arXiv:1707.06347.

[36] M. Mittal, C. Yu, Q. Yu *et al.*, "Orbit: A unified simulation framework for interactive robot learning environments," *IEEE Robotics and Automation Letters*, 2023.

[37] OpenAI, "GPT-5 Python API," https://platform.openai.com, 2025.

[38] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," *The International Journal of Robotics Research*, 2024.

[39] D. Nix and A. Weigend, "Estimating the mean and variance of the target probability distribution," in *Proceedings of IEEE International Conference on Neural Networks (ICNN)*, 1994.

[40] M. Wang, S. Tian, A. Swann, O. Shorinwa, J. Wu, and M. Schwager, "Phys2Real: Fusing vlm priors with interactive online adaptation for uncertainty-aware sim-to-real manipulation," *arXiv preprint arXiv:2510.11689*, 2025.