DeformMaster: An Interactive Physics-Neural World Model for Deformable Objects from Videos

Can Li Nankai University
Zhoujian Li Zhejiang University
Ren Li Southern University of Science and Technology
Jie Gu Rightly Robotics, A4x
Lei Lei Shanghai Innovation Institute, University of Science and Technology of China
Jingmin Chen Rightly Robotics, A4x
Lei Sun✉ Nankai University

Abstract

World models for deformable objects should recover not only geometry and appearance, but also underlying physical dynamics, interaction grounding, and material behavior. Learning such a model from real videos is challenging because deformable linear, planar, and volumetric objects evolve under high-dimensional deformation, noisy interactions, and complex material response. The model must infer a physical state from visual observations, roll it forward under new interactions, and render the resulting dynamics with high visual fidelity. We therefore present DeformMaster, a video-derived interactive physics-neural world model that turns real interaction videos into an online interactive model of deformable objects within a unified dynamics-and-appearance framework. DeformMaster preserves structured physical rollout while using a neural residual to compensate for unmodeled effects, grounds sparse hand motion as distributed compliant actuators for hand-continuum interaction, represents material response with spatially varying constitutive experts, and drives high-fidelity 4D appearance from the predicted physical evolution. Experiments on real-world deformable-object sequences demonstrate DeformMaster's ability to roll out future dynamics and render dynamic appearance, outperforming state-of-the-art baselines while supporting novel action rollout, material-parameter variation, and dynamic novel-view synthesis. Code and video demos are available at the project page: https://can-lee.github.io/deformmaster-web/.

1 Introduction

World models should encompass not only scene geometry, appearance, and temporal motion, but also the underlying physical attributes, governing dynamics, and causal interactions. Such models are especially important for embodied AI, where an agent must predict how the world will change under its own actions rather than merely reconstruct observations. Deformable objects make this goal particularly challenging: linear, planar, and volumetric objects evolve in high-dimensional state spaces, and their evolution is dictated by distributed strain, complex material response, self-contact, and external forces. A useful deformable-object world model must therefore infer the underlying physical state, support online interaction by rolling it forward under novel actions, and render its evolving appearance from novel views.

Existing methods have made some progress in reconstructing or generating dynamic deformable scenes. Neural and Gaussian representations [Mildenhall et al., 2020, Kerbl et al., 2023] can recover

✉ Corresponding author: Lei Sun (sunl@nankai.edu.cn). Can Li (canli@mail.nankai.edu.cn) and Lei Sun are with the College of Artificial Intelligence, Nankai University. Can Li, Zhoujian Li, and Lei Lei interned at Rightly Robotics, A4x.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

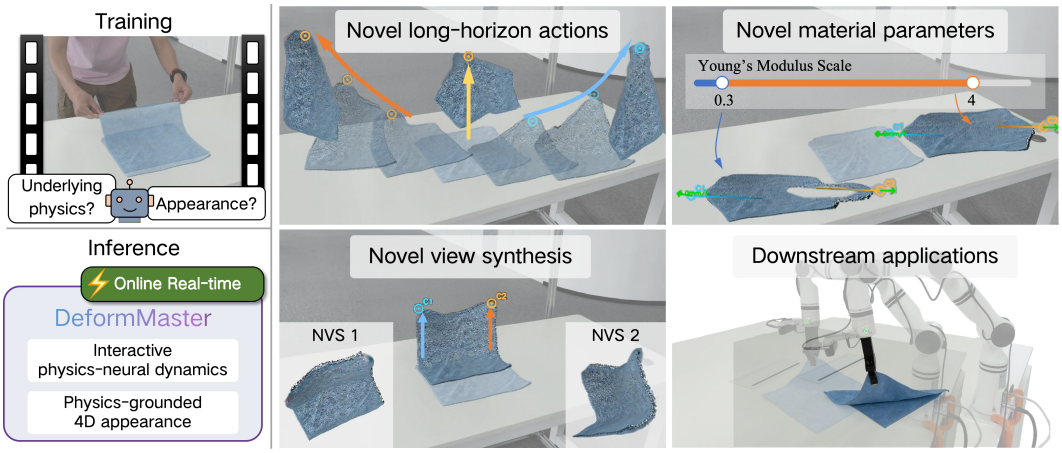

Figure 1: DeformMaster turns a phone-captured monocular video of deformable objects into an online interactive world model. By recovering both underlying physics and high-fidelity appearance, it supports long-horizon rollouts under novel actions, material-parameter variation, and novel-view synthesis, facilitating downstream embodied applications. (https://can-lee.github.io/deformmaster-web/)

high-quality appearance from observations, and physics-aware reconstruction methods further fit physics engines for deformable objects from visual observations [Li et al., 2023, Cai et al., 2024, Zhong et al., 2024, Jiang et al., 2025]. However, parameter identification within an idealized physics model can still struggle to account for real-world phenomena beyond the model assumptions. Recent works further exploit video diffusion either as a dynamic prior to supervise physics fitting [Zhang et al., 2024b, Liu et al., 2025] or as a generative engine for 4D synthesis [Lu et al., 2026]. However, video generation models [Yang et al., 2025] primarily imagine how the world looks; they often lack a reliable understanding of action-conditioned dynamics, making them difficult to control through explicit interactions and prone to hallucinated dynamics that do not match the real physical scene.

Several lines of work attempt to introduce physical controllability, but each leaves a critical gap. Physical digital twins such as PhysTwin [Jiang et al., 2025] and Spring-Gaus [Zhong et al., 2024] couple simulation substrates with Gaussian appearance, while EMPM [Chen et al., 2026] fits differentiable MPM for deformable object manipulation; yet fixed physics substrates and pure parameter fitting can hinder the model's ability to generalize to diverse real-world videos. Learned dynamics models based on particle-graph or particle-grid networks improve flexibility [Sanchez-Gonzalez et al., 2020, Zhang et al., 2025a,b, 2024a]. However, fully learned transitions often suffer from heavy data dependency, remain tied to the training distribution, and drift significantly during long-horizon novel-action rollouts. Hybrid physics-generative systems use physics to carry action semantics into 4D content [Li et al., 2025b, Zhan et al., 2026, Liu et al., 2026], but they primarily target generation rather than learning object dynamics from observed real-world interactions.

Our key insight is that real-world deformable-object world modeling should jointly recover underlying physical dynamics, interaction grounding, complex material behavior, and appearance tied to the evolving physical state. This requires more than fitting a simulator or learning visual motion alone: the model must remain stable under long-horizon interaction, absorb deviations from idealized physics, translate noisy observed contacts into effective actuation, and keep rendered appearance consistent with the predicted physical evolution. To address these requirements, we propose DeformMaster, an instance-specific interactive physics-neural world model that turns real interaction videos into an online queryable representation for novel action rollout, novel material-parameter variation, and dynamic novel-view synthesis, as illustrated in Figure 1. We summarize the contributions as follows:

1. We propose Physics-Neural Particle-Grid Dynamics (PNPGD) that augments differentiable physics with a neural residual, preserving physics-guided rollouts while compensating for unmodeled real-world effects.
2. We propose Distributed Compliant Actuators (DCA), which turn noisy sparse hand tracks into compliant, spatially distributed actuation for stable and effective hand-continuum interaction.

3. We introduce a Mixture of Constitutive Experts (MoCE) that blends canonical material laws with spatially varying weights to capture heterogeneous material response.
4. We develop DeformMaster, a video-derived deformable-object world model that pairs interactive physics-neural dynamics with physics-grounded high-fidelity 4D appearance.

The rest of the paper is organized as follows. Section 2 reviews related works. Section 3 and Section 4 present our proposed DeformMaster and its evaluation. Section 5 concludes the paper.

2 Related Work

Reconstruction and physics simulation of deformable objects. Physics-based reconstruction methods aim to recover simulatable representations of deformable objects from visual observations. PAC-NeRF [Li et al., 2023], GIC [Cai et al., 2024], PhysGaussian [Xie et al., 2024], OmniPhysGS [Lin et al., 2025], PhysSplat [Zhao et al., 2025a], PhysGM [Lv et al., 2026], NGFF [Li et al., 2026], and PhysGen3D [Chen et al., 2025] integrate physics simulation with reconstructed 3D representations, while PhysDreamer [Zhang et al., 2024b], PhysFlow [Liu et al., 2025], and Phys4D [Lu et al., 2026] use video generation models as motion priors or generative backbones to synthesize plausible 4D dynamics. These works validate physics priors for reconstruction, but generative models [Yang et al., 2025, Zhang et al., 2025c] remain difficult to control through physical interaction inputs, and physical digital twins can be limited by fixed substrates, pure parameter fitting, or brittle action grounding. Closest to our setting, PhysTwin and its extensions [Jiang et al., 2025, Zhang et al., 2026] and Spring-Gaus [Zhong et al., 2024] couple physics substrates with Gaussian splatting from videos, and EMPM [Chen et al., 2026] fits differentiable MPM for manipulation. NCLaw [Ma et al., 2023], MASIV [Zhao et al., 2025b], and NeuMA [Cao et al., 2024] learn or adapt neural constitutive models, and VisionLaw [Lin et al., 2026b] infers symbolic constitutive laws from visual observations. We instead use physics-neural dynamics as the core transition model, together with compliant distributed actuation and heterogeneous constitutive modeling, to turn reconstruction into interactive modeling.

Neural dynamics of deformable objects. Learning-based simulators replace analytical dynamics with neural transition models [Ai et al., 2025]. Particle graph [Sanchez-Gonzalez et al., 2020] and particle-grid networks [Zhang et al., 2025a] model ropes, cloths, and volumetric objects; GS-Dynamics [Zhang et al., 2025b] couples Gaussian tracking with graph dynamics, and AdaptiGraph [Zhang et al., 2024a] conditions graph dynamics on physical-property estimates. These methods are flexible, but purely learned transitions often need substantial data, remain tied to the training distribution, and drift under long rollouts or novel actions. We instead use a physics-guided rollout for stronger generalization, with neural dynamics acting as a residual correction for real-world mismatch.

Hybrid physics-generative world models. Recent systems also use physics to support action-conditioned 4D world prediction. Building on the static-scene precursor WonderWorld [Yu et al., 2025], the Wonder series couples physics solvers with video generation for interactive content creation through WonderPlay [Li et al., 2025b], PerpetualWonder [Zhan et al., 2026], and RealWonder [Liu et al., 2026]. Force Prompting [Gillman et al., 2025] and Goal Force [Gillman et al., 2026] further fine-tune video diffusion models on synthetic physics primitives to absorb force control signals. These methods show that physics can guide video generation, but require large-scale generative-model training. We instead learn from real videos the underlying physics in a data-efficient way.

3 DeformMaster

We seek an instance-specific world model of deformable objects that (i) rolls out stable dynamics aligned with real-world observations, (ii) grounds noisy hand tracks for effective hand-continuum interaction, (iii) captures complex material response, and (iv) renders high-fidelity appearance grounded in physics. The model is optimized separately for each observed interaction sequence. To this end, as illustrated in Figure 2, DeformMaster pairs interactive physics-neural dynamics (Section 3.2) with physics-grounded appearance (Section 3.3) through four components: PNPGD for dynamics rollout, DCA for interaction, MoCE for material response, and Gaussian Splatting for rendering. Physics preliminaries are provided in Appendix A.3.

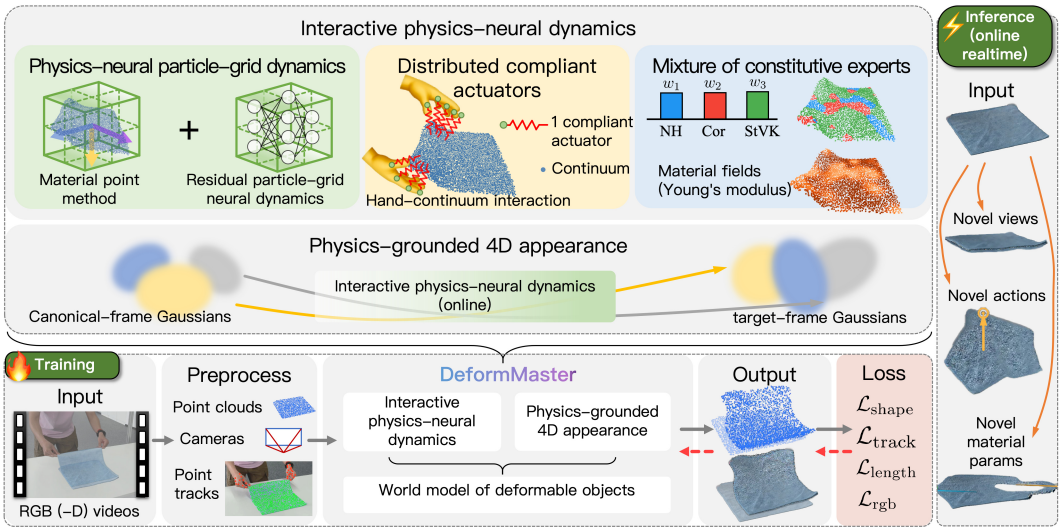

Figure 2: Overview of DeformMaster. From interaction videos, DeformMaster unifies interactive physics-neural dynamics and physics-grounded 4D appearance in a single deformable-object world model. The dynamics module integrates Physics-Neural Particle-Grid Dynamics, Distributed Compliant Actuators, and a Mixture of Constitutive Experts to support stable rollout, robust hand-continuum interaction, and heterogeneous material response. Within the dynamics module, actuators provide accelerations, and constitutive experts provide stresses for the material point method update. Gaussian splats are driven by the dynamics branch. Training optimizes dynamics and appearance using geometric, tracking, and image supervision. Inference supports novel view, action, and material-parameter conditions online.

3.1 Problem Formulation

State. We represent the deformable state as $s_t = (s_t^{\text{mat}}, s_t^{\text{app}})$, where s_t^{mat} denotes the material-particle state and s_t^{app} the appearance-particle state. **Actions.** The action a_t consists of observed hand or actuator anchor positions and velocities. **Observations.** Supervision comes from monocular RGB or multi-view RGB-D videos with extracted point clouds, camera poses, and dense 3D tracks. **Learning objective.** We learn (θ, ϕ, ψ) for the joint dynamics-appearance model:

$$s_{t+1}^{\text{mat}} = \mathcal{F}_{\theta,\phi}(s_t^{\text{mat}}, a_t), \qquad \hat{I}_{t+1} = \mathcal{G}_\psi(s_{t+1}^{\text{app}}) = \mathcal{G}_\psi\big(\mathcal{B}_\psi(s_{t+1}^{\text{mat}})\big), \tag{1}$$

where $\mathcal{F}_{\theta,\phi}$ is the hybrid physics-neural rollout operator for material dynamics; $\mathcal{B}_\psi$ bridges material particles to appearance particles; and $\mathcal{G}_\psi$ renders that state into images. Training matches both rolled-out material states and rendered frames to observations.

3.2 Interactive Physics-Neural Dynamics

Physics-Neural Particle-Grid Dynamics (PNPGD). Physics-based simulators provide structured, stable priors for dynamics, but real observations exhibit systematic effects that no idealized model can fully express. We therefore pair an explicit physics block with a neural residual that absorbs the unmodeled mismatch.

We decompose deformation dynamics $\mathcal{F}_{\theta,\phi}$ into a physics block $\mathcal{P}_\theta$ and a residual block $\mathcal{R}_\phi$:

$$\mathcal{F}_{\theta,\phi} = \mathcal{P}_\theta \oplus \mathcal{R}_\phi. \tag{2}$$

Concretely, $\mathcal{P}_\theta$ first advances the state over one frame with differentiable MPM [Hu et al., 2018] under material parameters θ, producing a tentative next state (with s_t denoting s_t^{mat} for brevity):

$$\tilde{s}_{t+1} = \mathcal{P}_\theta^{\text{MPM}}\big(s_t, a_t\big), \tag{3}$$

where $s_t = (\mathbf{x}_p^t, \mathbf{v}_p^t, \mathbf{F}_p^t, \mathbf{C}_p^t)$ is the MPM particle state consisting of per-particle position, velocity, deformation gradient, and affine matrix; a_t denotes the action (detailed in DCA below). Second, after the MPM rollout over one frame, a residual block $\mathcal{R}_\phi$ predicts a neural velocity correction $\Delta\mathbf{v}_p$,

$$\Delta\mathbf{v}_p = \mathcal{R}_\phi\big(\tilde{s}_{t+1}, s_t, h_t\big), \tag{4}$$

where $h_t = \{(\mathbf{x}_p^{t-i}, \mathbf{v}_p^{t-i})\}_{i=1}^{H}$ is a short kinematic history. The final residual-corrected state s_{t+1} is then given by

$$\mathbf{v}_p^{t+1} \;=\; \tilde{\mathbf{v}}_p^{t+1} + \Delta\mathbf{v}_p, \qquad \mathbf{x}_p^{t+1} \;=\; \tilde{\mathbf{x}}_p^{t+1} + \Delta\mathbf{v}_p\,\Delta t, \tag{5}$$

where the residual updates only particle positions and velocities, while $\mathbf{F}_p^{t+1}$ and $\mathbf{C}_p^{t+1}$ are inherited from $\tilde{s}_{t+1}$. By design $\|\Delta\mathbf{v}_p\|$ is bounded, so $\mathcal{R}_\phi$ acts as a perturbation at the frame level rather than a free state predictor.

To compose cleanly with MPM, residual block $\mathcal{R}_\phi$ uses a similar particle-grid representation. We reformulate Particle-Grid Neural Dynamics [Zhang et al., 2025a] as this residual block: a PointNet encoder [Qi et al., 2017] produces a per-particle latent feature

$$\mathbf{f}_p \;=\; E_\phi\big(\tilde{\mathbf{x}}_p^{t+1},\, \tilde{\mathbf{v}}_p^{t+1},\, \tilde{\mathbf{x}}_p^{t+1} - \mathbf{x}_p^{t},\, h_t\big), \tag{6}$$

that summarises the post-MPM state and history h_t; a coordinate-conditioned MLP decoder with Fourier positional encoding [Mildenhall et al., 2020] predicts a bounded grid-node correction $\mathbf{u}_g = \alpha\,\tanh(\cdot)$, which is then mapped back to particles by the same B-spline weights MPM uses for P2G/G2P transfers, yielding $\Delta\mathbf{v}_p$. The residual architecture mirrors MPM's particle-grid hybrid through an Eulerian grid representation and particle-grid transfer. Details of the MPM configuration and neural residual architecture are provided in Sections A.3 and A.4.

Distributed Compliant Actuator (DCA). Hand-continuum interaction is often the fragile part of real-to-sim deformable modeling. Vision-derived hand or actuator tracks are noisy and sparse, so hard pointwise constraints surface two failure modes: tracking noise is injected as velocity spikes, and point loads deform only a tiny neighborhood. DCA addresses these failure modes with *compliance*, which turns hard constraints into compliant actuator-particle couplings absorbing high-frequency noise, and *distribution*, which spreads actuation over a finite contact patch to drive bulk motion in a soft continuum.

Concretely, DCA applies compliant actuator-to-particle couplings over a local actuator neighborhood, producing the actuator-induced acceleration $\mathbf{a}_p$ on particle p:

$$\mathbf{a}_p \;=\; n_p^{-1/2} \sum\nolimits_{c\in\mathcal{N}(p)} \Big[k_p\big((\mathbf{x}_c - \mathbf{x}_p) - \mathbf{o}_{p,c}^0\big) + k_d\big(\mathbf{v}_c - \mathbf{v}_p\big)\Big], \tag{7}$$

where k_p, k_d are stiffness and damping (DCA gains), $\mathbf{x}_p, \mathbf{v}_p$ are the particle state, $\mathbf{x}_c, \mathbf{v}_c$ are actuator-anchor state, $\mathbf{o}_{p,c}^0$ is the initial rest offset, $\mathcal{N}(p)$ is the local actuator neighborhood, $n_p = |\mathcal{N}(p)|$. The $n_p^{-1/2}$ factor normalizes over multiple anchors to avoid force stacking.

Mixture of Constitutive Experts (MoCE). Real deformable objects rarely conform to a single idealized constitutive law. Their response is shaped by material composition, processing history, and scene-specific deformation patterns. We therefore represent stress with a finite mixture of constitutive experts whose weights can adapt across continuum regions. We model the first Piola–Kirchhoff stress as a spatially varying mixture of constitutive experts:

$$\mathbf{P}_{\text{mix}}(\mathbf{F}_p; E_p, \nu_p) \;=\; \sum\nolimits_k w_{k,p}\,\mathbf{P}_k(\mathbf{F}_p; E_p, \nu_p), \tag{8}$$

where $\mathbf{P}_k$ is the stress map of expert $k \in \{\text{NH, Cor, StVK}\}$, $\mathbf{F}_p$ is the deformation gradient of particle p, and E_p, ν_p denote learnable Young's modulus and Poisson's ratio. The mixture weights $w_{k,p}$ are spatially varying, implemented through patch-level expert logits and interpolated to particles as detailed in Section A.5.

3.3 Physics-Grounded 4D Appearance

Online interaction requires high-fidelity rendering without re-optimizing a dynamic appearance model after every new action. We therefore keep rollout on a compact material-particle state and use it to drive Gaussian appearance. Given the particle trajectory predicted by $\mathcal{F}_{\theta,\phi}$, the bridge $\mathcal{B}_\psi$ deforms Gaussians using LBS [Sumner et al., 2007, Huang et al., 2024]. The update is incremental: each frame applies the particle motion from the previous frame to the current one, rather than re-skinning from the canonical state. This keeps rendering efficient and aligned with physical motion without learning a separate dynamic reconstruction model.

Table 1: Overall future-prediction comparison on twenty real-world PhysTwin deformation sequences. We report dynamics accuracy and rendered appearance quality averaged over unseen test frames. DeformMaster achieves the strongest overall performance, improving dynamics and appearance fidelity while remaining comparable to PhysTwin on Track error.

Method	Future dynamics			Future appearance		
	IoU ↑	Chamfer ↓	Track ↓	PSNR ↑	SSIM ↑	LPIPS ↓
PhysTwin [Jiang et al., 2025]	0.734	0.012	**0.023**	25.16	0.935	0.061
Spring-Gaus [Zhong et al., 2024]	0.464	0.062	0.094	22.49	0.924	0.113
GS-Dynamics [Zhang et al., 2025b]	0.498	0.041	0.070	22.54	0.924	0.097
DeformMaster (ours)	**0.748**	**0.011**	0.024	**25.41**	**0.936**	**0.061**

Table 2: Per-category dynamics prediction on the PhysTwin sequences, grouped into deformable linear, planar, and volumetric objects. DeformMaster improves on linear and volumetric objects, while the remaining Track gap is concentrated in planar object sequences.

Method	Linear (n=3)			Planar (n=9)			Volumetric (n=8)		
	IoU ↑	Chamfer ↓	Track ↓	IoU ↑	Chamfer ↓	Track ↓	IoU ↑	Chamfer ↓	Track ↓
PhysTwin [Jiang et al., 2025]	0.658	0.007	0.013	0.738	0.013	**0.028**	0.748	0.013	0.021
DeformMaster (ours)	**0.721**	**0.005**	**0.010**	**0.748**	**0.013**	0.032	**0.756**	**0.012**	**0.020**

3.4 Training Scheme

We train DeformMaster with a combined dynamics-and-appearance objective:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{dynamics}} + \mathcal{L}_{\text{appearance}} = \lambda_{\text{track}}\,\mathcal{L}_{\text{track}} + \lambda_{\text{shape}}\,\mathcal{L}_{\text{shape}} + \lambda_{\text{len}}\,\mathcal{L}_{\text{len}} + \lambda_{\text{rgb}}\,\mathcal{L}_{\text{rgb}}, \tag{9}$$

where $\mathcal{L}_{\text{dynamics}}$ contains track, shape, and length terms: $\mathcal{L}_{\text{track}}$ aligns 3D point trajectories, $\mathcal{L}_{\text{shape}}$ aligns shape with Chamfer distance, and $\mathcal{L}_{\text{len}}$ preserves local lengths. $\mathcal{L}_{\text{appearance}}$ is $\mathcal{L}_{\text{rgb}}$, a photometric loss on rendered frames. Optimization proceeds in multiple stages. Stages 1–2 alternate between dynamics training and DCA gain selection: we first learn the material fields and neural residual with default DCA gains under dynamics loss, use the warm-started dynamics to select gains with CMA-ES [Hansen, 2006], and then continue learning the material fields and neural residual with the updated gains. Stage 3 first optimizes Gaussian splats with appearance loss and then uses the total loss (*i.e.*, adding RGB loss) to refine the dynamics branch.

4 Experiments

4.1 Setup

We organize the evaluation around the contributions of DeformMaster. **Implementation.** Implementation details (preprocessing, MPM, neural residual, MoCE, and training) are deferred to Section A. **Dataset.** We evaluate on 20 real deformable-object sequences from PhysTwin [Jiang et al., 2025], spanning deformable linear (n=3, ropes), planar (n=9, cloths and package), and volumetric (n=8, soft-bodied toys) objects, all captured with calibrated three-view RGB-D videos at 30 fps. For future prediction, we train on the first 70% of each sequence and evaluate on the remaining 30%. **Baselines.** We compare the full system against PhysTwin [Jiang et al., 2025], Spring-Gaus [Zhong et al., 2024], and GS-Dynamics [Zhang et al., 2025b], and ablate PNPGD, DCA, MoCE, and RGB-guided dynamics refinement (Sections 4.2 and 4.3). **Metrics.** We evaluate future rollout with dynamics metrics (Chamfer distance, Track error, and mask IoU) and appearance metrics (PSNR, SSIM, and LPIPS). **Online playground.** Our method supports online dynamics rollout at over 15 fps; the online playground is shown in Section A.1, with additional interactive results provided on the project page. Code for the playground is publicly available at https://can-lee.github.io/deformmaster-web/.

4.2 Main Results

Table 1 reports future-frame prediction results on the 20 PhysTwin sequences. Our DeformMaster achieves the strongest overall performance, improving mask IoU and Chamfer distance over all



Figure 3: Qualitative comparison on two representative deformation sequences. Columns progress from the learning stage to future prediction, and each case compares ground truth (GT), DeformMaster (Ours), and PhysTwin. Yellow arrows indicate the applied actions. Blue and red dashed boxes mark corresponding regions in our prediction and PhysTwin, respectively. DeformMaster better preserves the future object configuration and yields stronger overlap with GT in the highlighted regions.

baselines and producing the best rendered appearance. Compared with PhysTwin, our method slightly improves Chamfer (0.011 vs. 0.012) and IoU (0.748 vs. 0.734), while maintaining a comparable Track error (0.024 vs. 0.023). The advantage over learning-based Gaussian dynamics baselines is more pronounced: our method reduces Chamfer by more than $3\times$ and Track error by at least $2.9\times$ compared with Spring-Gaus and GS-Dynamics. Since the appearance metrics are computed by deforming the same first-frame Gaussian representation with the predicted trajectory, the improvements in PSNR, SSIM, and LPIPS mainly reflect better long-horizon rollout rather than renderer-specific tuning.

Table 2 further breaks down the dynamics results by object type. The gains are most pronounced on linear objects (ropes), where our DeformMaster improves IoU from 0.658 to 0.721 and reduces Chamfer and Track error from $0.007/0.013$ to $0.005/0.010$. On volumetric objects (soft-bodied toys), our method also improves all three dynamics metrics, indicating that the particle-grid-based transition model benefits objects with substantial three-dimensional deformation. Planar objects are the most challenging case: our method slightly improves IoU and matches Chamfer, but its Track error is higher than PhysTwin (0.032 vs. 0.028). This suggests that the small overall Track gap in Table 1 is mainly driven by planar sequences, where single-layer cloth-like motion is less naturally matched to the particle-grid continuum simulator.

Figure 3 visualizes two representative deformable volumetric objects from the PhysTwin sequences. During the learning stage, both methods capture the dynamics and appearance well. In future prediction, our method better preserves the object configuration under upward pulling and gravity, and remains more closely aligned with the ground truth in the highlighted regions. These results qualitatively echo the design goals: stable long-horizon rollouts, compensation for idealized-physics mismatch, effective actuation from noisy contacts, and appearance tied to the predicted physical state. In contrast, PhysTwin tends to under-deform or drift away from the observed shape, leading to weaker overlap in the same regions.

Table 3: Quantitative dynamics and appearance prediction under novel actions. We use object dynamics learned from one interaction sequence to predict responses to unseen action trajectories from another sequence of the same object, without retraining or adaptation.

Method	IoU ↑	Chamfer ↓	Track ↓	PSNR ↑	SSIM ↑	LPIPS ↓
PhysTwin [Jiang et al., 2025]	0.5837	0.01954	0.03355	23.40	0.9308	0.0741
DeformMaster (ours)	**0.6326**	**0.01882**	**0.03299**	**23.62**	**0.9311**	**0.0721**

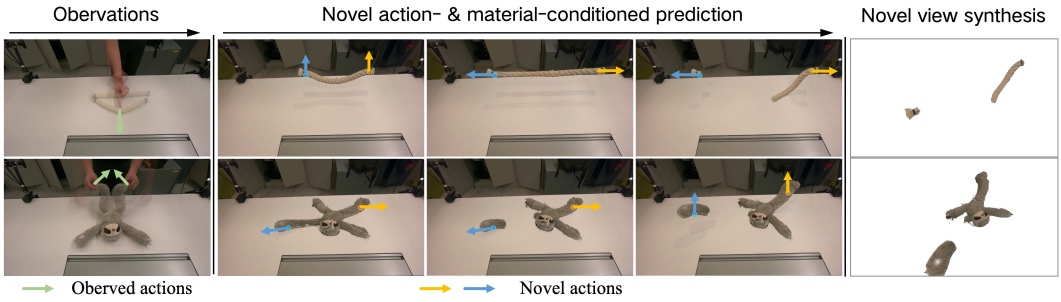

Figure 4: Generalization to novel action-, material-, and view-conditioned prediction with our DeformMaster. DeformMaster supports rollouts under novel actions and material scales. Scaling the material fields to $0.3\times$ their learned values produces fracture behavior, while the final column renders the fracture states from a novel view, demonstrating interactive ability beyond the observations and baselines.

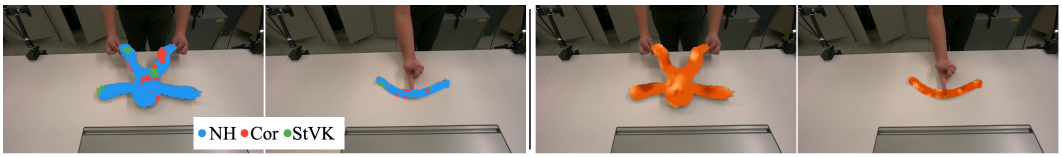

Figure 5: Qualitative visualization of MoCE and material fields. For readability, the left two panels visualize the MoCE mixture by showing only the largest-weight constitutive expert at each particle. The right two panels visualize the material field as the spatial distribution of Young's modulus. The results qualitatively show that our method captures spatially varying material behavior for deformable objects, rather than assuming a single homogeneous constitutive response.

To evaluate generalization to unseen actions, we use object dynamics learned from one interaction sequence to predict responses to new action trajectories from another sequence of the same object. For both methods, all learned parameters remain fixed, and the target sequence provides only the initial geometry and controller trajectory, with no target-action training or adaptation. Predictions are evaluated against target observations on the future-test split. Table 3 shows that DeformMaster improves all six dynamics and appearance metrics over PhysTwin, demonstrating its ability to predict responses to unseen actions without retraining. Evaluation details are provided in Section A.8. We also evaluate view generalization by rendering predicted future dynamics from unseen viewpoints. As shown in Table 6, DeformMaster achieves higher PSNR and SSIM and lower LPIPS than PhysTwin.

Figure 4 shows that our model can be queried under novel action, material parameters, and view conditions. Starting from the same observed object, our method performs rollouts under novel conditions by changing the actuation direction and material parameter scale, and then renders the predicted state from a novel camera view. Notably, scaling the material fields to $0.3\times$ their recovered values produces material fracture, illustrating a discontinuous behavior that is difficult for PhysTwin's fixed topological connectivity to express.

Figure 5 visualizes MoCE and material fields on representative deformable objects. In MoCE, each particle uses a mixture over constitutive experts; for readability, the visualization shows only the dominant expert with the largest mixture weight at each particle. The displayed expert maps and the corresponding Young's modulus fields are spatially non-uniform, indicating that the optimized

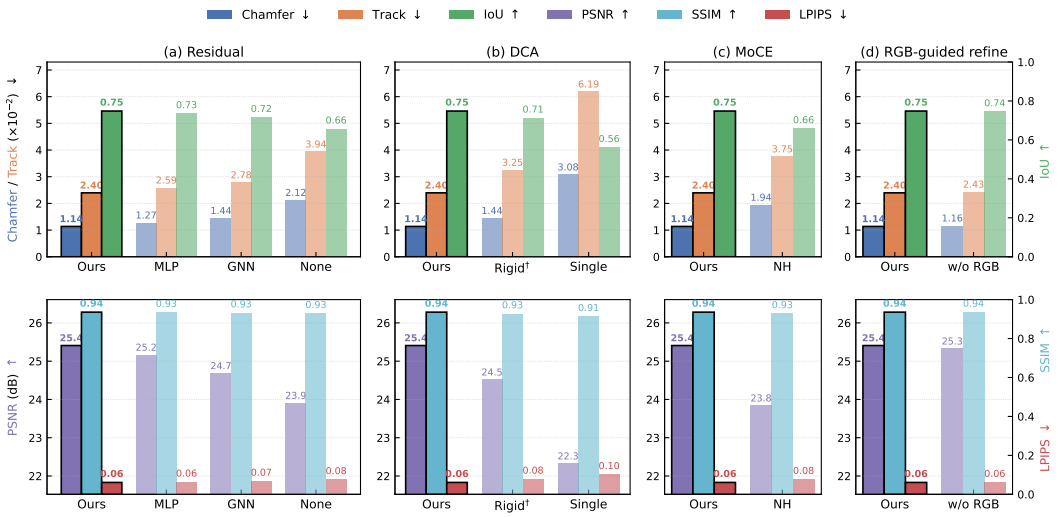

Figure 6: Ablation summary. Columns (a)–(d) compare four design choices, with dynamics metrics in the top row (Chamfer/Track in 10^{-2}, IoU) and appearance metrics in the bottom row (PSNR, SSIM, LPIPS). **(a) Residual**: the neural particle-grid residual performs best, while MLP/GNN residuals and removing the residual (None, *i.e.*, MPM only) degrade both dynamics rollout and appearance. **(b) DCA**: removing compliance (Rigid†, reported on $11/20$ successfully trained sequences) or collapsing the distributed actuator to a single one (Single) both reduce performance, showing the need for distributed compliant actuation. **(c) MoCE**: replacing the mixture with a single Neo-Hookean expert worsens both dynamics and appearance. **(d) RGB-guided refinement**: fine-tuning with RGB loss in training stage 3 improves dynamics metrics and rendered appearance over the dynamics-loss-only variant (w/o RGB). Overall, each component contributes to the final system; "Ours" is highlighted with a solid edge, and full numerical results are in Section A.9.

material response adapts across object regions. Together, these visualizations show that our model captures region-dependent material behavior from video observations, rather than reducing each deformable object to a single homogeneous constitutive response.

4.3 Ablation Study

PNPGD: Residual Matters, and Its Architecture Matters. Figure 6(a) ablates PNPGD by varying the residual branch. We compare the proposed particle-grid residual with three alternatives: removing the residual entirely, using an MLP residual that predicts independent per-particle corrections, and using a GNN residual in the style of particle-based neural simulators [Sanchez-Gonzalez et al., 2020]. The MPM-only variant without residual degrades both dynamics and appearance, confirming that the pure physics simulator provides a useful prior but cannot by itself absorb the systematic mismatch present in real videos. Among residual variants, both the MLP and GNN underperform the proposed design, showing that the particle-grid residual architecture is more effective. Together, these results support the motivation in Section 3.2: a neural residual is needed to bridge the gap between idealized physics and real-world observations, and the particle-grid residual architecture matters. Full numbers are reported in Tables 7 and 8.

DCA: Both Compliance and Distribution Matter. Figure 6(b) ablates the two design choices in DCA. The rigid variant removes compliance by replacing compliant actuator-particle coupling with hard constraints, while the single-actuator variant keeps compliance but collapses each distributed actuator to one actuator. The rigid variant diverges on $9/20$ sequences, including all 8 volumetric cases, which is consistent with the motivation that noisy vision-derived tracks should not be injected as hard pointwise constraints. The single-actuator variant is numerically stable but reduces accuracy, showing that pointwise forcing is insufficient to drive bulk deformation in a soft continuum. These results verify the two-part DCA design in Section 3.2: compliance stabilizes action grounding under noisy tracks, and distribution improves force transmission over contact regions of deformable objects. Full numbers are reported in Tables 9 and 10.

MoCE vs. Single Expert. Figure 6(c) ablates the constitutive model by replacing MoCE with a single Neo-Hookean expert while keeping the rest of the pipeline unchanged. This removes the spatially varying constitutive experts and forces all particles to share the same canonical constitutive law. The single-expert variant worsens both rollout accuracy and rendered appearance, indicating that real deformation sequences contain heterogeneous material responses that cannot be fully represented by a global constitutive choice. In contrast, MoCE represents the stress response as a spatially varying mixture of canonical constitutive experts, allowing the model to adapt across continuum regions while remaining grounded in analytic constitutive laws. Full numbers are reported in Tables 11 and 12.

Effect of RGB-Guided Refinement. Figure 6(d) evaluates RGB-guided dynamics refinement by ablating the RGB loss in training stage 3. We compare whether to use the total loss (*i.e.*, adding RGB loss) to refine the dynamics branch. Because the rendered appearance is driven by the predicted material-particle trajectory (Equation (1)), the RGB loss provides supervision to the underlying dynamics rather than only to image-space appearance. Adding this loss yields consistent overall gains, improving both dynamics and rendered appearance metrics. Per-category results in Table 14 show that the dynamics gains mainly come from deformable planar and volumetric objects, while linear objects remain nearly unchanged. These results show that RGB supervision in stage 3 contributes to rollout refinement. Full numbers are reported in Tables 13 and 14.

4.4 Runtime, Memory, and Scalability

To assess the computational cost of online interaction, we profile DeformMaster on a single NVIDIA A100 GPU. Dynamics rollout for the particle-heavy "double_lift_sloth" case takes 62.674 ms/frame (15.99 FPS; Table 15). Including appearance computation brings the estimated total latency to 75.612 ms/frame (13.23 FPS). MPM and the neural residual account for 82.5% of the total latency, versus 2.18% for Gaussian rendering. Separately measured peak memory is 2.263 GB for dynamics and 1.859 GB for appearance.

Table 16 reports dynamics-only scaling. Increasing substeps from 21 to 84 raises median latency from 42.184 to 111.333 ms/frame; total rollout time increases with the horizon. Increasing particles from 9,133 to 22,132 raises latency and memory, while grid resolution has little effect over the tested 32–64 range. Peak dynamics memory stays within approximately 2.3 GB.

Preprocessing takes minutes, while training requires approximately 2.5 hours of dynamics learning, DCA gain selection, and approximately 2–3 hours of appearance learning and RGB-guided refinement, with peak training memory around 20 GB (Table 17). These one-time per-instance costs are separate from online simulation and rendering.

5 Conclusion

We proposed DeformMaster, a video-derived interactive physics-neural world model for deformable objects. From real interaction videos, DeformMaster couples interactive physics-neural dynamics with physics-grounded appearance, enabling controllable rollout and high-fidelity rendering. This design reflects the central contributions: robust physics-neural dynamics, stable grounding of real interactions, heterogeneous material modeling, and 4D appearance synthesis driven by the underlying physics. Experiments on multi-category real sequences show the strongest overall future dynamics and appearance, with clear gains over baselines. Ablations further confirm that each component contributes to rollout accuracy and appearance quality. Limitations are discussed in Section A.11. Promising directions include extending DeformMaster to modeling under partial observations [Li et al., 2025a], contact-rich scenarios and fluid dynamics, and integrating language-conditioned goal prediction for robotic manipulation [Wu et al., 2025].

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 62173192 and No. 62373365). This work was completed during an internship at Rightly Robotics, A4x. We thank Ren Li and Jie Gu for insightful discussions, Zhoujian Li for assistance, and Hanxiao Jiang for his excellent work PhysTwin and generous guidance.

References

Bo Ai, Stephen Tian, Haochen Shi, Yixuan Wang, Tobias Pfaff, Cheston Tan, Henrik I. Christensen, Hao Su, Jiajun Wu, and Yunzhu Li. A review of learning-based dynamics models for robotic manipulation. *Science Robotics*, 10(106):eadt1497, 2025. doi: 10.1126/scirobotics.adt1497.

Junhao Cai, Yuji Yang, Weihao Yuan, Yisheng He, Zilong Dong, Liefeng Bo, Hui Cheng, and Qifeng Chen. GIC: Gaussian-informed continuum for physical property identification and simulation. In *Advances in Neural Information Processing Systems*, volume 37, pages 75035–75063, 2024. doi: 10.52202/079017-2388.

Junyi Cao, Shanyan Guan, Yanhao Ge, Wei Li, Xiaokang Yang, and Chao Ma. Neuma: Neural material adaptor for visual grounding of intrinsic dynamics. In *Advances in Neural Information Processing Systems*, volume 37, pages 65643–65669, 2024. doi: 10.52202/079017-2097.

Boyuan Chen, Hanxiao Jiang, Shaowei Liu, Saurabh Gupta, Yunzhu Li, Hao Zhao, and Shenlong Wang. PhysGen3D: Crafting a miniature interactive world from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6178–6189, 2025. doi: 10.1109/CVPR52734.2025.00579.

Yunuo Chen, Yafei Hu, Lingfeng Sun, Tushar Kusnur, Laura Herlant, and Chenfanfu Jiang. EMPM: Embodied MPM for modeling and simulation of deformable objects. *IEEE Robotics and Automation Letters*, 11(4):4179–4186, 2026. doi: 10.1109/LRA.2026.3664610.

Nate Gillman, Charles Herrmann, Michael Freeman, Daksh Aggarwal, Evan Luo, Deqing Sun, and Chen Sun. Force prompting: Video generation models can learn and generalize physics-based control signals. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pages 103174–103201, 2025. doi: 10.52202/085713-3448.

Nate Gillman, Yinghua Zhou, Zitian Tang, Evan Luo, Arjan Chakravarthy, Daksh Aggarwal, Michael Freeman, and Chen Sun. Goal force: Teaching video models to accomplish physics-conditioned goals. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20077–20087, 2026.

Nikolaus Hansen. The CMA evolution strategy: A comparing review. In Jose A. Lozano, Pedro Larrañaga, Iñaki Inza, and Endika Bengoetxea, editors, *Towards a New Evolutionary Computation: Advances in the Estimation of Distribution Algorithms*, volume 192 of *Studies in Fuzziness and Soft Computing*, pages 75–102. Springer, Berlin, Heidelberg, 2006. doi: 10.1007/3-540-32494-1_4.

Yuanming Hu, Yu Fang, Ziheng Ge, Ziyin Qu, Yixin Zhu, Andre Pradhana, and Chenfanfu Jiang. A moving least squares material point method with displacement discontinuity and two-way rigid body coupling. *ACM Transactions on Graphics*, 37(4):1–14, 2018. doi: 10.1145/3197517.3201293.

Yi-Hua Huang, Yang-Tian Sun, Ziyi Yang, Xiaoyang Lyu, Yan-Pei Cao, and Xiaojuan Qi. SC-GS: Sparse-controlled gaussian splatting for editable dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4220–4230, 2024. doi: 10.1109/CVPR52733.2024.00404.

Hanxiao Jiang, Hao-Yu Hsu, Kaifeng Zhang, Hsin-Ni Yu, Shenlong Wang, and Yunzhu Li. PhysTwin: Physics-informed reconstruction and simulation of deformable objects from videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7219–7230, 2025. doi: 10.1109/ICCV51701.2025.00678.

Nikita Karaev, Yuri Makarov, Jianyuan Wang, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. CoTracker3: Simpler and better point tracking by pseudo-labelling real videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6013–6022, 2025. doi: 10.1109/ICCV51701.2025.00568.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian Splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4):1–14, 2023. doi: 10.1145/3592433.

Can Li, Jingyang Liu, and Lei Sun. Learning-based keypoints detection with topological order on deformable linear objects from incomplete point clouds. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 650–656, 2025a. doi: 10.1109/iros60139.2025.11246954.

Shiqian Li, Ruihong Shen, Junfeng Ni, Chang Pan, Chi Zhang, and Yixin Zhu. Learning physics-grounded 4D dynamics with neural gaussian force fields. In *International Conference on Learning Representations*, volume 2026, pages 81165–81207, 2026.

Xuan Li, Yi-Ling Qiao, Peter Yichen Chen, Krishna Murthy Jatavallabhula, Ming Lin, Chenfanfu Jiang, and Chuang Gan. PAC-NeRF: Physics augmented continuum neural radiance fields for geometry-agnostic system identification. In *International Conference on Learning Representations (ICLR)*, 2023.

Zizhang Li, Hong-Xing Yu, Wei Liu, Yin Yang, Charles Herrmann, Gordon Wetzstein, and Jiajun Wu. WonderPlay: Dynamic 3D scene generation from a single image and actions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9080–9090, 2025b. doi: 10.1109/ICCV51701.2025.00849.

Haotong Lin, Sili Chen, Jun Hao Liew, Donny Y. Chen, Zhenyu Li, Yang Zhao, Sida Peng, Hengkai Guo, Xiaowei Zhou, Guang Shi, Jiashi Feng, and Bingyi Kang. Depth Anything 3: Recovering the visual space from any views. In *International Conference on Learning Representations*, volume 2026, pages 141261–141285, 2026a.

Jiajing Lin, Shu Jiang, Qingyuan Zeng, Zhenzhong Wang, and Min Jiang. Visionlaw: Inferring interpretable intrinsic dynamics from visual observations via bilevel optimization. In *International Conference on Learning Representations*, volume 2026, pages 96128–96165, 2026b.

Yuchen Lin, Chenguo Lin, Jianjin Xu, and Yadong Mu. OmniPhysGS: 3D constitutive gaussians for general physics-based dynamics generation. In *International Conference on Learning Representations*, volume 2025, pages 70160–70187, 2025.

Wei Liu, Ziyu Chen, Zizhang Li, Yue Wang, Hong-Xing Yu, and Jiajun Wu. RealWonder: Real-time physical action-conditioned video generation. *arXiv preprint arXiv:2603.05449*, 2026. doi: 10.48550/arXiv.2603.05449.

Zhuoman Liu, Weicai Ye, Yan Luximon, Pengfei Wan, and Di Zhang. Unleashing the potential of multi-modal foundation models and video diffusion for 4D dynamic physical scene simulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11016–11025, 2025. doi: 10.1109/CVPR52734.2025.01029.

Haoran Lu, Shang Wu, Songling Liu, Jianshu Zhang, Maojiang Su, Guo Ye, Chenwei Xu, Lie Lu, Pranav Maneriker, Fan Du, Zhaoran Wang, and Han Liu. Phys4D: Fine-grained physics-consistent 4D modeling from video diffusion. *arXiv preprint arXiv:2603.03485*, 2026. doi: 10.48550/arXiv.2603.03485.

Chunji Lv, Zequn Chen, Donglin Di, Weinan Zhang, Hao Li, Wei Chen, Yinjie Lei, and Changsheng Li. PhysGM: Large physical gaussian model for feed-forward 4D synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 29855–29865, 2026.

Pingchuan Ma, Peter Yichen Chen, Bolei Deng, Joshua B. Tenenbaum, Tao Du, Chuang Gan, and Wojciech Matusik. Learning neural constitutive laws from motion observations for generalizable PDE dynamics. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 23279–23300, 2023.

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *Computer Vision – ECCV 2020*, pages 405–421, 2020. doi: 10.1007/978-3-030-58452-8_24.

Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. PointNet: Deep learning on point sets for 3D classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 77–85, 2017. doi: 10.1109/CVPR.2017.16.

Tianhe Ren, Shilong Liu, Ailing Zeng, Jing Lin, Kunchang Li, He Cao, Jiayu Chen, Xinyu Huang, Yukang Chen, Feng Yan, Zhaoyang Zeng, Hao Zhang, Feng Li, Jie Yang, Hongyang Li, Qing Jiang, and Lei Zhang. Grounded sam: Assembling open-world models for diverse visual tasks, 2024.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 8459–8468. PMLR, 2020.

Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperGlue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4937–4946, 2020. doi: 10.1109/CVPR42600.2020.00499.

Robert W. Sumner, Johannes Schmid, and Mark Pauly. Embedded deformation for shape manipulation. *ACM Transactions on Graphics*, 26(3):80, 2007. doi: 10.1145/1276377.1276478.

Jianyuan Wang, Minghao Chen, Shangzhan Zhang, Nikita Karaev, Johannes Schönberger, Patrick Labatut, Piotr Bojanowski, David Novotny, Andrea Vedaldi, and Christian Rupprecht. VGGT-Ω. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21486–21499, June 2026.

Ruicheng Wang, Sicheng Xu, Yue Dong, Yu Deng, Jianfeng Xiang, Zelong Lv, Guangzhong Sun, Xin Tong, and Jiaolong Yang. MoGe-2: Accurate monocular geometry with metric scale and sharp details. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pages 35928–35959, 2025. doi: 10.52202/085713-1207.

Jianyang Wu, Jie Gu, Xiaokang Ma, Fangzhou Qiu, Chu Tang, and Jingmin Chen. Stimulating imagination: Towards general-purpose "something something placement". In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 701–707, 2025. doi: 10.1109/IROS60139.2025.11245844.

Jianfeng Xiang, Zelong Lv, Sicheng Xu, Yu Deng, Ruicheng Wang, Bowen Zhang, Dong Chen, Xin Tong, and Jiaolong Yang. Structured 3D latents for scalable and versatile 3D generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21469–21480, 2025. doi: 10.1109/CVPR52734.2025.02000.

Tianyi Xie, Zeshun Zong, Yuxing Qiu, Xuan Li, Yutao Feng, Yin Yang, and Chenfanfu Jiang. PhysGaussian: Physics-integrated 3D gaussians for generative dynamics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4389–4398, 2024. doi: 10.1109/CVPR52733.2024.00420.

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

Hong-Xing Yu, Haoyi Duan, Charles Herrmann, William T. Freeman, and Jiajun Wu. WonderWorld: Interactive 3D scene generation from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5916–5926, 2025. doi: 10.1109/CVPR52734.2025.00555.

Jiahao Zhan, Zizhang Li, Hong-Xing Yu, and Jiajun Wu. PerpetualWonder: Long-horizon action-conditioned 4D scene generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 25370–25380, 2026.

Kaifeng Zhang, Baoyu Li, Kris Hauser, and Yunzhu Li. AdaptiGraph: Material-adaptive graph-based neural dynamics for robotic manipulation. In *Proceedings of Robotics: Science and Systems*, 2024a. doi: 10.15607/RSS.2024.XX.010.

Kaifeng Zhang, Baoyu Li, Kris Hauser, and Yunzhu Li. Particle-grid neural dynamics for learning deformable object models from RGB-D videos. In *Proceedings of Robotics: Science and Systems*, 2025a. doi: 10.15607/RSS.2025.XXI.036.

Kaifeng Zhang, Shuo Sha, Hanxiao Jiang, Matthew Loper, Hyunjong Song, Guangyan Cai, Zhuo Xu, Xiaochen Hu, Changxi Zheng, and Yunzhu Li. Real-to-sim robot policy evaluation with Gaussian splatting simulation of soft-body interactions. In *2026 IEEE International Conference on Robotics and Automation (ICRA)*, pages 22356–22363, 2026. doi: 10.1109/icra57385.2026.11696757.

Mingtong Zhang, Kaifeng Zhang, and Yunzhu Li. Dynamic 3D gaussian tracking for graph-based neural dynamics modeling. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pages 1851–1862. PMLR, 2025b.

Tianyuan Zhang, Hong-Xing Yu, Rundi Wu, Brandon Y. Feng, Changxi Zheng, Noah Snavely, Jiajun Wu, and William T. Freeman. PhysDreamer: Physics-based interaction with 3D objects via video generation. In *Computer Vision – ECCV 2024*, pages 388–406, 2024b. doi: 10.1007/978-3-031-72627-9_22.

Zhenghao Zhang, Junchao Liao, Menghao Li, Zuozhuo Dai, Bingxue Qiu, Siyu Zhu, Long Qin, and Weizhi Wang. Tora: Trajectory-oriented diffusion transformer for video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2063–2073, 2025c. doi: 10.1109/CVPR52734.2025.00198.

Haoyu Zhao, Hao Wang, Xingyue Zhao, Hao Fei, Hongqiu Wang, Chengjiang Long, and Hua Zou. PhysSplat: Efficient physics simulation for 3D scenes via MLLM-guided gaussian splatting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5242–5252, 2025a. doi: 10.1109/ICCV51701.2025.00498.

Yizhou Zhao, Haoyu Chen, Chunjiang Liu, Zhenyang Li, Charles Herrmann, Junhwa Hur, Yinxiao Li, Ming-Hsuan Yang, Bhiksha Raj, and Min Xu. Toward material-agnostic system identification from videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5944–5956, October 2025b.

Licheng Zhong, Hong-Xing Yu, Jiajun Wu, and Yunzhu Li. Reconstruction and simulation of elastic objects with spring-mass 3D gaussians. In *Computer Vision – ECCV 2024*, pages 407–423, 2024. doi: 10.1007/978-3-031-72627-9_23.

A Appendix

A.1 Online Interactive Playground

We provide an online interactive playground for DeformMaster, where users can select interactive points, adjust material parameters, manipulate the deformable object through keyboard inputs, and inspect synchronized novel-view renderings during rollout. Figure 7 shows the playground interface, including the central interactive view, two novel-view renderers, controller status, material parameter adjustment, and keyboard control panels.

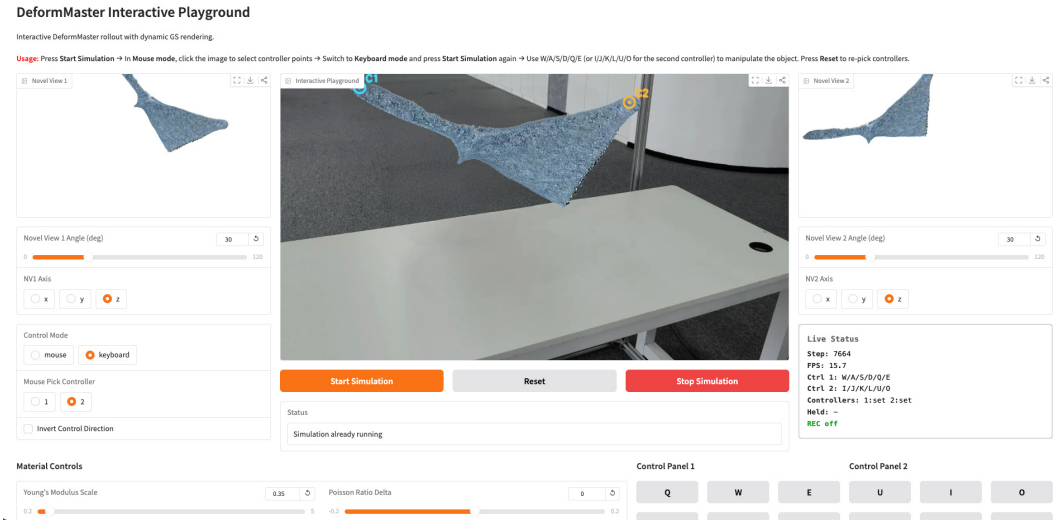

Figure 7: Online interactive playground for DeformMaster. The interface supports interaction-point selection, keyboard-based object manipulation, material-parameter adjustment, and synchronized novel-view rendering during online rollout. The live status panel reports the online rollout speed, showing real-time interaction at over 15 fps in this example.

A.2 Downstream Applications

The interactive world model DeformMaster enables a range of embodied downstream tasks. As shown in Figure 8, its interaction capability can be used to synthesize additional deformable-object data, while the model supports model-predictive-control-based robotic manipulation. The resulting 4D representation also provides convenient visualization and novel-view image synthesis.

A.3 Physics Preliminaries

MPM [Hu et al., 2018] represents a deformable object with particles carrying position, velocity, and deformation information. Each substep transfers particle quantities to a background grid, updates grid velocities under internal and external forces, and transfers the updated motion back to particles.

A constitutive law maps deformation to stress, determining how the material resists deformation. In DeformMaster, MoCE combines constitutive experts with spatially varying weights to supply stresses for the MPM update (Equation (8)).

Tracked hand or actuator positions and velocities define moving anchors. DCA couples these anchors to nearby object particles, converting relative position and velocity errors into distributed accelerations that drive the simulated motion (Equation (7)).

Table 4 lists the MPM substrate parameters used throughout training and rollout. Volumetric object interior particle filling follows PhysGaussian [Xie et al., 2024]. One video-frame transition applies $N_{\text{sub}} = \Delta t/\delta t$ MPM substeps. Per-category numeric overrides are noted; values without an override are shared.

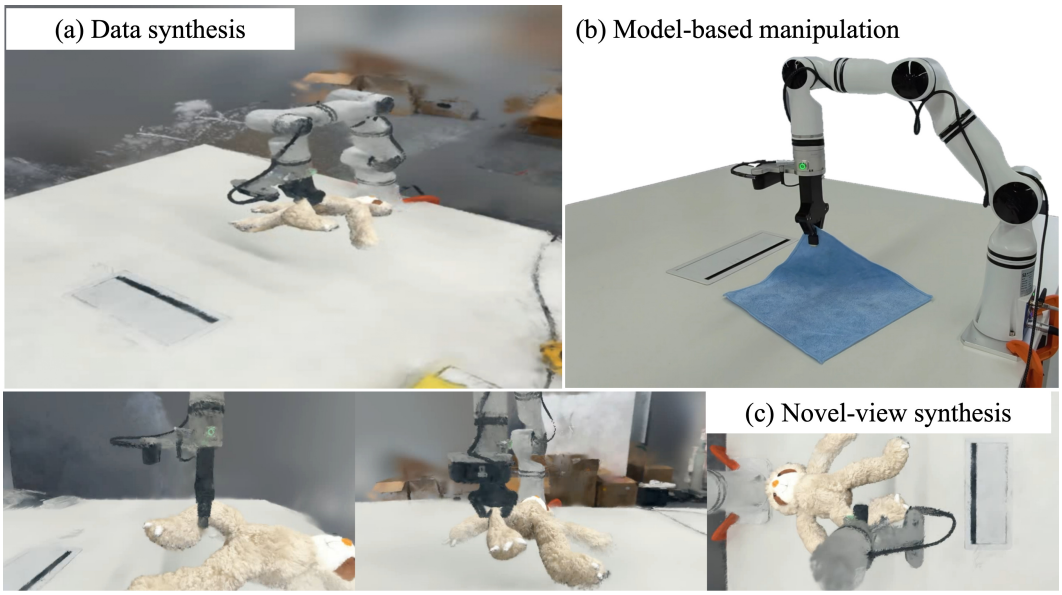

Figure 8: Applications of DeformMaster. (a) DeformMaster leverages interactive manipulation to synthesize additional deformable-object data beyond the observed video. (b) DeformMaster enables model predictive control for deformable-object manipulation. (c) It supports convenient visualization and dynamic novel-view image synthesis.

A.4 Neural Residual Architecture

The main text describes neural particle-grid residual and gives the encoder equation in Equation (6). Here we provide the decoder and particle-grid interpolation equations that complete the neural residual $\mathcal{R}_\phi$.

The encoder E_ϕ is a PointNet [Qi et al., 2017] whose inputs are the post-MPM position $\tilde{\mathbf{x}}_p^{t+1}$ and velocity $\tilde{\mathbf{v}}_p^{t+1}$ of particle p (components of $\tilde{s}_{t+1}$), the displacement $\tilde{\mathbf{x}}_p^{t+1} - \mathbf{x}_p^t$ accumulated by MPM during the current frame, and the kinematic history h_t of Equation (4). The decoder maps each grid node coordinate $\mathbf{x}_g$ to a node-level correction $\mathbf{u}_g$ using a coordinate-conditioned MLP,

$$\mathbf{u}_g \;=\; \alpha \cdot \tanh\big(D_\phi(\mathbf{f}_p,\, \gamma(\mathbf{x}_g))\big), \tag{10}$$

where D_ϕ is the MLP, $\gamma(\cdot)$ is Fourier positional encoding of the node coordinate, and α is a scalar that bounds the magnitude of the per-particle correction. Finally, the per-particle correction is the B-spline-weighted sum over the surrounding grid nodes,

$$\Delta\mathbf{v}_p \;=\; \sum_{g \,\in\, \mathcal{N}_p} w_{p,g}\, \mathbf{u}_g, \tag{11}$$

where $\mathcal{N}_p$ is the set of grid nodes supporting particle p under MPM's B-spline interpolation, and $w_{p,g}$ are the same interpolation weights used by MPM for P2G/G2P transfers. This mirrors MPM's particle-grid hybrid at the level of grid discretization and particle-grid transfer: the encoder extracts particle features from the point set, while the decoder predicts local node-wise residuals on each particle's supporting grid neighborhood before interpolating them back to particles. Concrete network widths and training hyperparameters are listed in Table 5.

A.5 MoCE Parameterization

The main text defines the constitutive mixture at the particle level in Equation (8). The mixture is taken over three canonical hyperelastic experts: **Neo-Hookean (NH)**, **fixed Corotated (Cor)**, and **St. Venant–Kirchhoff (StVK)**, each producing a first Piola–Kirchhoff stress $\mathbf{P}_k(\mathbf{F}_p; E_p, \nu_p)$ from the deformation gradient under shared Young's modulus and Poisson's ratio. In the implementation, the learnable physical parameters are stored on a persistent set of material patches and interpolated to particles before each MPM rollout.

Table 4: MPM solver configuration. *linear / planar / volumetric* columns indicate per-category overrides where they differ from the default.

Group	Symbol / name	Value
Discretisation		
Maximum particles per scene	$N_{\max}$	10^5
Background grid	G	32^3
Kernel	cubic B-spline	27-node support
Substep	δt	8×10^{-4} s
Frame interval	Δt	$1/30$ s
Substeps per frame	$N_{\text{sub}} = \Delta t / \delta t$	42
Forces & boundary		
Gravity	$\mathbf{g}$	$(0,\ 0,\ -9.8)$ m/s^2
Particle damping	-	20.0 (planar/linear), 5.0 (volumetric)
Grid velocity damping	multiplicative per substep	0.999
Position clip	$[\mathbf{x}_{\min}, \mathbf{x}_{\max}]$	$[0.05,\ 0.95]^3$ (in shifted unit cube)
Floor margin	-	0.0 (planar/linear), 0.05 (volumetric)
Constitutive (per-particle, log-sigmoid bounded)		
Young's modulus bounds	E	$[10^4, 10^6]$ Pa (planar) $[10^5, 10^7]$ Pa (volumetric)
Poisson ratio bounds	ν	$[0,\ 0.35]$ (planar) $[0,\ 0.45]$ (volumetric)
SVD clamp	on $\mathbf{F}$ singular values	$[1/2.0,\ 2.0]$
Active experts	MoCE (Section A.5)	{NH, Cor., StVK}
Differentiation		
Backend	NVIDIA Warp	-
Truncated-BPTT window	last-W substeps recorded	$W = 20$
Per-parameter clip	on physics gradients	1.0
Global gradient clip	on physics parameters	10.0

Patch centers are sampled once from the first-frame geometry using farthest-point sampling. Each simulation particle p is assigned its three nearest patch anchors $\mathcal{A}(p)$, and the normalized inverse-distance weights

$$\beta_{p,c} = \frac{1/d(\mathbf{x}_p^0, \mathbf{a}_c)}{\sum_{c' \in \mathcal{A}(p)} 1/d(\mathbf{x}_p^0, \mathbf{a}_{c'})}, \qquad c \in \mathcal{A}(p), \tag{12}$$

are kept fixed so that patch parameters remain tied to the same material regions during optimization. The expert weights are represented by patch logits $\ell_{k,c}$ and converted to patch probabilities by

$$q_{k,c} = \mathrm{softmax}_k(\ell_{k,c}). \tag{13}$$

The particle-level mixture weights used in Equation (8) are then

$$w_{k,p} = \sum_{c \in \mathcal{A}(p)} \beta_{p,c}\, q_{k,c}. \tag{14}$$

The same patch-to-particle interpolation is applied to the bounded Young's modulus and Poisson's ratio parameters before converting them to Lamé parameters for the MPM stress computation.

A.6 Video Preprocessing Pipeline

Each input clip is processed by a fixed cascade of foundation models that converts raw RGB(-D) frames into the data structures consumed by the trainer (object masks, dense 3D point tracks for the object and the actuator, a first-frame canonical mesh, and calibrated camera intrinsics/extrinsics). The cascade is run once offline per clip and shared across all training stages.

Object segmentation and tracking. Object masks are produced by Grounded-SAM-2 [Ren et al., 2024]: a text prompt naming the target object (e.g. "cloth", "rope", "stuffed toy") is fed to Grounding DINO for open-vocabulary detection on the first frame, the resulting bounding box is passed to

Table 5: Neural Residual configuration. Values are shared across material categories unless otherwise noted; MPM solver settings (substep, grid, etc.) are listed separately in Table 4.

Group	Symbol / name	Value
Inputs		
Per-particle channels	$9 + 6H$	27 (with H=3)
ground fields	pos, vel, frame displacement	9
history	H frames of (pos, vel)	$6H$
Centring	subtract particle-cloud centroid	yes
PointNet encoder (Lagrangian)		
Conv1D widths	$(9{+}6H) \to 64 \to 128 \to 64$	-
Normalisation	GroupNorm	-
Output	per-particle feature	64-dim
Neural-field decoder (Eulerian)		
Query nodes / particle	MPM B-spline support	27 (cubic kernel)
Decoder grid	co-located with MPM grid	see Table 4
Positional encoding	Fourier bands L	6 ($\dim 15$)
MLP	layers $\times$ width	4×128
Skip connection	every 4 layers	yes
Output activation	$\tanh \cdot \alpha$	$\alpha = 1.0$ m/s
Particle interpolation	B-spline weights (P2G/G2P)	27-node
Residual cadence	once per frame (post all substeps)	-
Optimisation		
Optimiser	Adam	β_1=0.9, β_2=0.999
Learning rate	η_ϕ	5×10^{-3}
Gradient norm clip	-	5.0
Residual regulariser	$\lambda_{\text{reg}} \|\Delta\mathbf{v}\|^2$	$\lambda_{\text{reg}} = 10^{-3}$
Differentiability	MPM via Warp autodiff	end-to-end
Truncated-BPTT window	MPM tape length (in substeps)	see Table 4

SAM 2 to obtain a high-fidelity initial mask, and SAM 2's video-mode tracker propagates the mask through all T frames. The same pipeline is run with prompt "hand" to obtain a per-frame controller mask, used both to exclude hand pixels from the photometric loss and to anchor actuator positions.

Dense 2D–3D tracking. Within the object mask we sample a dense set of query points on the first frame (farthest-point sampling on the masked region) and propagate them across time with CoTracker3 [Karaev et al., 2025], whose pseudo-label-trained design is robust to occlusion and partial out-of-view. The resulting 2D tracks are unprojected to 3D using the per-frame depth from the next step, producing the dense $(T, N, 3)$ point trajectory that drives the Chamfer/Track losses in $\mathcal{L}_{\text{dynamics}}$.

Monocular geometry (uncalibrated single-camera setting). For monocular sequences we estimate per-frame depth and camera pose with VGGT-Omega [Wang et al., 2026] or Depth Anything 3 [Lin et al., 2026a]. We resolve global scale by fitting a per-clip scale factor that aligns its predictions with the metric point map produced by MoGe-2 [Wang et al., 2025] on the masked object. We apply the estimated scale consistently to the predicted depth maps and camera translations, aligning the reconstruction with MoGe-2's metric-scale geometric prior.

Shape generative prior. To initialize particles in regions occluded at t=0, we use TRELLIS [Xiang et al., 2025] to generate a canonical textured mesh from the segmented first frame. The mesh is aligned to the observed first-frame point cloud by rendering candidate mesh views, selecting the best match to the real RGB crop with SuperGlue [Sarlin et al., 2020] keypoint matching, solving the 6-DoF pose with PnP, and then refining it with an ARAP regularizer. The aligned mesh is volumetrically sampled together to populate the MPM particles.

A.7 Training Details

Stage 1–2 alternating optimization. The interactive physics-neural dynamics is implemented with a mixed NVIDIA Warp/PyTorch backend: the MPM simulator runs in Warp, while the neural residual

is implemented in PyTorch. The MPM step is differentiable end-to-end via NVIDIA Warp's autodiff backend, so the residual is optimised jointly with the physics parameters rather than in two stages.

Stages 1 and 2 are coupled: DCA gains (k_p, k_d) and material fields are mutually dependent through the simulator rollout. We therefore alternate between gradient-based dynamics training and gradient-free gain selection. We first optimize the material fields and neural residual with default gains, then run CMA-ES gain selection using the warm-started dynamics, and finally continue optimizing the material fields and neural residual with the selected gains.

We add a small ℓ_2 penalty $\lambda_{\text{reg}}\|\Delta\mathbf{v}\|^2$ to the training loss to discourage unnecessary correction; the value of λ_{reg} is given in Table 5, and the truncated-BPTT horizon over which gradients are propagated is set by the MPM tape window in Table 4.

RGB-guided refinement in Training Stage 3. Stage 3 has two steps. We first optimize a first-frame Gaussian Splatting representation $\mathcal{G}_0$ with appearance supervision, then freeze $\mathcal{G}_0$ and resume from the Stage 2 dynamics checkpoint for 30 additional iterations. During this refinement step, $\mathcal{G}_0$ is deformed by linear blend skinning along the predicted particle trajectory $\mathbf{x}_{1:t}$, rasterised through each training-view camera, and compared to the recorded frame using a masked ℓ_1 RGB loss. Object pixels retain their captured RGB, background pixels are set to the renderer background to penalize Gaussian bleed, and hand pixels are excluded because the renderer contains no hand model. The refinement loss keeps the dynamics terms and adds RGB supervision, $\mathcal{L}_{\text{stage3}} = \mathcal{L}_{\text{dynamics}} + \lambda_{\text{rgb}}\mathcal{L}_{\text{rgb}}$ with $\lambda_{\text{rgb}} = 10^{-2}$. Because $\mathcal{G}_0$ is deformed from the predicted particle trajectory, the photometric gradient flows through LBS back to particle motion and the neural residual, acting as an RGB-guided constraint on the dynamics rather than an update to the frozen Gaussians.

A.8 Novel-Action and Dynamic Novel-View Evaluation

We conduct a controlled same-object cross-action evaluation, using object dynamics learned from one interaction sequence to predict responses to unseen action trajectories from another sequence of the same object, with all learned parameters fixed. The unseen actions involve different contacts, directions, magnitudes, and trajectories. Five same-object groups contain 15 interaction sequences. For example, the zebra group contains "single_lift_zebra", "double_lift_zebra", and "double_stretch_zebra"; transferring from "single_lift_zebra" to "double_lift_zebra" and transferring in the reverse direction count as two separate evaluations. Evaluating all ordered pairs of distinct sequences within the five object groups yields 32 directed source-to-target evaluations per method. The source checkpoint and learned parameters, including the material parameters, DCA gains, and physics-neural dynamics parameters for DeformMaster, remain fixed, and the solver settings are unchanged. The target sequence supplies only the initial geometry and controller trajectory. No target-action training, adaptation, or parameter selection is used; target observations serve only as ground truth on the future-test split.

We also evaluate dynamic novel-view synthesis by rendering predicted future dynamics from unseen viewpoints. Table 6 reports appearance metrics on the future-test frames.

Table 6: Quantitative dynamic novel-view synthesis on future-test frames from unseen views.

Method	PSNR ↑	SSIM ↑	LPIPS ↓
PhysTwin [Jiang et al., 2025]	28.6606	0.9693	0.0341
DeformMaster (ours)	**28.8056**	**0.9703**	**0.0319**

A.9 Ablation Details

Tables 7 to 14 provide the full numerical ablations for the residual branch, DCA, MoCE, and Stage 3 RGB-guided dynamics refinement. Each ablation is reported in two views: an *overall* table with all six paper metrics (Tables 7, 9, 11 and 13) and a *per-category* dynamics table grouped by Linear / Planar / Volumetric (Tables 8, 10, 12 and 14).

A.10 Runtime, Memory, and Scalability

Tables 15 to 17 provide the detailed measurements supporting Section 4.4. We profile "double_lift_sloth" on one NVIDIA A100-SXM4-80GB, using 22,132 physical particles, 102,613 Gaussians, a 32^3 grid, 42 MPM substeps per frame, and 848×480 rendering as the default configuration.

For the component breakdown in Table 15, we use five warmup frames, 50 measured frames, three repetitions, and synchronized timing. Dynamics and appearance are timed independently; the end-to-end latency is their derived sum, and peak GPU memory is measured separately for each. The dynamics-only sweeps in Table 16 report medians over nine measured repeats, varying one factor at a time. Latency is reported per frame except for the horizon sweep, which reports total rollout time. Table 17 lists preprocessing and one-time training costs separately from online inference.

A.11 Limitations

DeformMaster still has several limitations. First, particle-grid simulation is less specialized for very thin planar objects, such as single-layer cloth, than spring-mass systems with fixed topological links. This is reflected by the remaining gap to PhysTwin on planar object Track (Section 4.2), where explicit spring-mass connectivity provides a strong inductive bias for cloth-like bending and stretching. Second, interaction modeling relies on observed hand motion. When the hand or actuator is severely occluded in the input video, the recovered interaction signal can be incomplete, which in turn degrades the action-conditioned rollout. Third, runtime depends on the number of physics substeps used during simulation. Reducing the substep count of the MPM solver improves real-time performance, whereas high-accuracy deformation simulation often requires more substeps; this creates a trade-off between fidelity and runtime. Finally, the learned material fields and MoCE weights are effective parameters, not uniquely identified physical constants. The model remains instance-specific and does not support zero-shot generalization to arbitrary new objects.

Table 7: Residual ablation: overall test metrics on the 20-case PhysTwin benchmark. Chamfer and Track in units of 10^{-2}.

Variant	Future dynamics			Future appearance		
	IoU ↑	Chamfer ↓	Track ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Ours (PNPGD)	**0.748**	**1.14**	**2.40**	**25.41**	**0.936**	**0.061**
MLP	0.735	1.27	2.59	25.16	0.935	0.063
GNN	0.715	1.44	2.78	24.68	0.932	0.068
No residual	0.656	2.12	3.94	23.90	0.929	0.076

Table 8: Ablation on the residual branch $\mathcal{R}_\phi$: per-category dynamics on the PhysTwin benchmark. Chamfer and Track in units of 10^{-2}. **Ours**: neural particle-grid. **MLP**: per-particle correction with no spatial structure. **GNN**: 3-round KNN graph network. **No residual**: pure MPM rollout.

Variant	Linear (n=3)			Planar (n=9)			Volumetric (n=8)		
	IoU ↑	Chamfer ↓	Track ↓	IoU ↑	Chamfer ↓	Track ↓	IoU ↑	Chamfer ↓	Track ↓
Ours	0.721	**0.50**	**1.01**	**0.748**	**1.30**	**3.19**	**0.756**	**1.19**	**2.02**
MLP	**0.727**	0.52	1.04	0.733	1.45	3.43	0.739	1.35	2.21
GNN	0.700	0.57	1.19	0.719	1.66	3.63	0.714	1.52	2.42
No residual	0.628	0.78	1.48	0.695	2.08	4.52	0.614	2.67	4.20

Table 9: DCA ablation: overall test metrics. Chamfer and Track in units of 10^{-2}. ‡ rigid is averaged over the $11/20$ sequences that survived training (the 8 Volumetric and 1 Linear that diverged are excluded).

Variant	Future dynamics			Future appearance		
	IoU ↑	Chamfer ↓	Track ↓	PSNR ↑	SSIM ↑	LPIPS ↓
DCA (ours)	**0.748**	**1.14**	**2.40**	**25.41**	**0.936**	**0.061**
rigid‡	0.713	1.44	3.25	24.51	0.927	0.078
single actuator	0.561	3.08	6.19	22.32	0.914	0.102

Table 10: DCA ablation: per-category dynamics. Chamfer and Track in units of 10^{-2}. **rigid** (no compliance): replace compliant coupling with hard Dirichlet boundary. **single actuator** (no distribution): keep compliance but actuate each particle through a single actuator instead of a distributed contact patch. "NaN" marks the 8 Volumetric sequences that diverged near iteration 0 under hard pointwise constraints; † rigid completes only $2/3$ Linear sequences.

Variant	Linear (n=3)			Planar (n=9)			Volumetric (n=8)		
	IoU ↑	Chamfer ↓	Track ↓	IoU ↑	Chamfer ↓	Track ↓	IoU ↑	Chamfer ↓	Track ↓
DCA (ours)	**0.721**	**0.50**	**1.01**	**0.748**	**1.30**	**3.19**	**0.756**	**1.19**	**2.02**
rigid	$0.675^{\dagger}$	$0.51^{\dagger}$	$1.14^{\dagger}$	0.718	1.65	3.72	NaN	NaN	NaN
single actuator	0.510	0.92	2.40	0.591	2.74	5.69	0.534	4.26	8.16

Table 11: MoCE ablation: overall test metrics. Chamfer and Track in units of 10^{-2}.

Variant	Future dynamics			Future appearance		
	IoU ↑	Chamfer ↓	Track ↓	PSNR ↑	SSIM ↑	LPIPS ↓
MoCE (ours)	**0.748**	**1.14**	**2.40**	**25.41**	**0.936**	**0.061**
Single Neo-Hookean	0.658	1.94	3.75	23.84	0.928	0.077

Table 12: Constitutive ablation: replacing the MoCE mixture with a single canonical Neo-Hookean expert. Per-category dynamics on the PhysTwin benchmark; Chamfer and Track in units of 10^{-2}. (The MoCE ablation keeps the neural residual active. Since the residual can absorb part of the mismatch caused by a simplified constitutive model, this comparison may underestimate the contribution of MoCE.)

Variant	Linear (n=3)			Planar (n=9)			Volumetric (n=8)		
	IoU ↑	Chamfer ↓	Track ↓	IoU ↑	Chamfer ↓	Track ↓	IoU ↑	Chamfer ↓	Track ↓
MoCE (ours)	**0.721**	**0.50**	**1.01**	**0.748**	**1.30**	**3.19**	**0.756**	**1.19**	**2.02**
Single Neo-Hookean	0.642	0.69	1.48	0.693	2.04	4.65	0.617	2.30	3.59

Table 13: Stage 3 RGB-guided refinement: overall test metrics. Chamfer and Track in units of 10^{-2}.

Variant	Future dynamics			Future appearance		
	IoU ↑	Chamfer ↓	Track ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Dynamics loss only	0.745	1.16	2.43	25.32	0.935	0.062
RGB-guided refinement (ours)	**0.748**	**1.14**	**2.40**	**25.41**	**0.936**	**0.061**

Table 14: Ablation of RGB-guided dynamics refinement in Stage 3. Per-category dynamics on the PhysTwin benchmark; Chamfer and Track in units of 10^{-2}.

Variant	Linear (n=3)			Planar (n=9)			Volumetric (n=8)		
	IoU ↑	Chamfer ↓	Track ↓	IoU ↑	Chamfer ↓	Track ↓	IoU ↑	Chamfer ↓	Track ↓
Dynamics loss only	**0.725**	**0.50**	**1.01**	0.743	1.33	3.22	0.752	1.22	2.08
RGB-guided refine	0.721	**0.50**	**1.01**	**0.748**	**1.30**	**3.19**	**0.756**	**1.19**	**2.02**

Table 15: Online inference breakdown with 22,132 physical particles, 102,613 Gaussians, a 32^3 grid, 42 MPM substeps per frame, and 848×480 rendering. The derived end-to-end latency is the sum of independently timed dynamics and appearance components.

Component	Mean latency (ms)	Share
Dynamics: MPM substeps	43.026	56.90%
Dynamics: Neural residual	19.359	25.60%
Other dynamics overhead	0.289	0.38%
Particle-to-Gaussian deformation	11.289	14.93%
Gaussian rendering	1.649	2.18%
Derived end-to-end	75.612	100%

Table 16: Dynamics-only runtime and peak GPU memory scaling on "double_lift_sloth"; medians over nine measured repeats. Bold entries denote the default configuration. Each sweep varies only the indicated factor while holding the others fixed.

Sweep	Setting	Median latency	GPU memory (GB)
Particles	6,660	50.267 ms/frame	≈1.1
	9,133	48.656 ms/frame	≈1.3
	22,132	**62.443** ms/frame	**≈2.3**
Grid resolution	**32**	**63.303** ms/frame	≈2.3
	48	61.212 ms/frame	≈2.3
	64	59.888 ms/frame	≈2.3
Substeps	21	42.184 ms/frame	≈2.3
	42	**63.303** ms/frame	≈2.3
	84	111.333 ms/frame	≈2.3
Rollout horizon	10 frames	0.580 s/rollout	≈2.3
	30 frames	1.852 s/rollout	≈2.3
	50 frames	**3.165** s/rollout	≈2.3

Table 17: Preprocessing and one-time training costs.

Stage	Time	Peak GPU memory
Preprocessing: depth/camera/geometry	1 min 45.85 s	19.45 GB
Preprocessing: segmentation/point tracking	6 min 22.29 s	10.94 GB
Training stage: dynamics learning	≈2.5 h	≈20 GB
Training stage: DCA gain selection	≈16.6 min/scene	≈2.30 GB
Training stage: appearance learning + RGB-guided refinement	≈2–3 h	≈20 GB

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The four contributions stated in the abstract and the introduction (PNPGD, DCA, MoCE, and the DeformMaster system) correspond directly to the methods in Sections 3.2 to 3.3, and are individually evaluated by the ablations in Section 4.3. Quantitative comparisons are reported on the PhysTwin benchmark, and qualitative generalization to custom data beyond this benchmark is demonstrated in Figure 1.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section A.11 in the appendix discusses three concrete limitations: (i) particle-grid simulation provides a weaker inductive bias than spring-mass systems for very thin single-layer cloth, which is reflected in the remaining gap to PhysTwin on planar Track; (ii) the recovered interaction signal degrades when the hand or actuator is severely occluded in the input video, which in turn degrades action-conditioned rollouts; (iii) the number of physics substeps creates a trade-off between simulation fidelity and runtime speed.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best

judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper does not contain formal theorems or proofs; it proposes an empirical physics-neural world model and evaluates it on a real-video benchmark.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section 3 describes the full pipeline (PNPGD, DCA, MoCE, and physics-grounded 4D appearance), and Sections A.4 to A.7 provide the neural-residual architecture, MoCE parameterization, video preprocessing pipeline, and training schedule with key hyperparameters needed to reproduce the main results.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.

(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Code and model checkpoints are publicly available through the project page linked in the abstract. Evaluation data comes from the public PhysTwin dataset. The main paper and appendix (Sections A.4 to A.7 and 3) provide implementation details for reproducing the results.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 4.1 states the benchmark and category split (linear / planar / volumetric) and the train/test temporal split. Section A.7 details the staged optimization schedule (Stages 1–2 alternating dynamics training and DCA gain selection, followed by Stage 3 appearance learning and RGB-guided dynamics refinement), optimizer, learning rates, and key hyperparameters; Sections A.3 to A.5 list architectural and parameterization choices.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: Each scene requires a multi-stage training pipeline that takes hours of GPU wallclock time, so per-method results are not repeated across multiple random seeds. To convey variability beyond a single aggregate, we report per-category breakdowns over linear / planar / volumetric objects in Table 2 and present per-component ablations in Sections A.9 and 4.3.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: All training and inference are run on a single NVIDIA A100 GPU. Per scene, Training stage 1+2 dynamics takes approximately 2–3 hours of wallclock time and Stage 3 appearance learning and RGB-guided refinement adds a comparable amount; peak GPU memory stays within $\sim$20 GB, well within the 80 GB capacity of one A100. Compute details are summarized in Section A.10.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: We have reviewed the NeurIPS Code of Ethics and the research conforms to it. The work uses only public RGB-D / RGB videos of inanimate deformable objects, does not involve human subjects, and does not release high-risk assets.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: DeformMaster offers a video-derived physics-neural world model for deformable objects, with potential downstream applications including 4D content creation, simulation, and robotic manipulation. We do not identify specific negative societal impacts beyond those generic to vision and physics-simulation research.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper does not release pretrained language models, image generators, or scraped datasets that pose a high risk for misuse.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.

- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All third-party assets used in this work are properly cited in Sections A.6, 2 and 4, including the PhysTwin benchmark, Grounded-SAM-2, CoTracker3, Depth Anything 3, MoGe-2, TRELLIS, 3D Gaussian Splatting, and the MLS-MPM solver. We use these assets in accordance with their respective licenses and terms of use.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We release code and model checkpoints through the project page. The accompanying README documents environment setup, checkpoint download and loading, and usage of the online playground.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The work does not involve crowdsourcing or research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The work does not involve human subjects, so IRB approval is not applicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The core methodology of DeformMaster—physics-neural particle-grid dynamics, distributed compliant actuators, and the mixture of constitutive experts—does not involve LLMs as any important, original, or non-standard component.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.