SupPhysField: Fast and Generalizable Supervised Learning of 3D Physics from Visual Features

Anonymous Author(s)
Affiliation
Address
email

Figure 1: We introduce SupPhysField, a novel method for learning simulatable physics of 3D scenes from visual features. Trained on a curated dataset of paired 3D objects and physical material annotations, SupPhysField can predict both the discrete material types (e.g., rubber) and continuous values including Young's modulus, Poisson's ratio, and density for a variety of materials, including elastic, plastic, and granular. The predicted material parameters can then be coupled with a learned static 3D model such as Gaussian splats and a physics solver such as the Material Point Method (MPM) to produce realistic 3D simulation under physical forces such as gravity and wind.

Abstract

Inferring the physical properties of 3D scenes from visual information is a critical yet challenging task for creating interactive and realistic virtual worlds. While humans intuitively grasp material characteristics such as elasticity or stiffness, existing methods often rely on slow, per-scene optimization, limiting their generalizability and application. To address this problem, we introduce SupPhysField, a novel method that trains a generalizable neural network to predict physical properties across multiple scenes from 3D visual features purely using supervised losses. Once trained, our feed-forward network can perform fast inference of plausible material fields, which coupled with a learned static scene representation like Gaussian Splatting enables realistic physics simulation under external forces. To facilitate this research, we also collected SupPhysVerse, one of the largest known datasets of paired 3D assets and physic material annotations. Extensive evaluations demonstrate that SupPhysField is about 2.21-4.58x better and orders of magnitude faster than test-time optimization methods. By leveraging pretrained visual features like CLIP, our method can also zero-shot generalize to real-world scenes despite only ever been trained on synthetic data. https://neurips-2025-20627.github.io/

1 Introduction

Advances in learning-based scene reconstruction with Neural Radiance Fields [25] and Gaussian Splatting [16] have made it possible to recreate photorealistic 3D geometry and appearance from sparse camera views, with broad applications from immersive content creation to robotics and simulation. However, these approaches focus exclusively on visual appearance—capturing the geometry and colors of a scene while remaining blind to its underlying physical properties.

Yet the world is not merely a static collection of shapes and textures. Objects bend, fold, bounce, and deform according to their material composition and the forces acting upon them. Consequently, there has been a growing body of work that aims to integrate physics into 3D scene modeling [27, 24, 20, 10, 9, 36, 28, 11, 23, 37, 5]. Current approaches for acquiring the material properties of the scene generally fall into two categories, each with significant limitations. Some works such as [36, 11] require users to manually specify material parameters for the entire scene based on domain knowledge. This manual approach is limited in its application as it places a heavy burden on the user and lacks fine-grained detail. Another line of work aims to automate the material discovery process via test-time optimization. Works including [14, 20, 40, 13, 23, 39] leverage differentiable physics solvers, iteratively optimizing material fields by comparing simulated outcomes against ground-truth observations or realism scores from video generative models. However, predicting physical parameters for hundreds of thousands of particles from sparse signals (i.e., a single rendering or distillation scalar loss) is an extremely slow and difficult optimization process, often taking hours on a single scene. Furthermore, this heavy per-scene memorization does not generalize: for each new scene, the incredibly slow optimization has to be run from scratch again.

In this paper, we propose a new framework, SupPhysField, which unifies geometry, appearance, and physics learning via direct supervised learning. Our approach is inspired by how humans intuitively understand physics: when we see a tree swaying in the wind, we do not memorize the stiffness values for each specific coordinate (x, y, z) – instead, we learn that objects with tree-like visual features behave in certain ways when forces are applied. This physical understanding from visual cues allows us to anticipate the motion of a different tree or even other vegetation like grass, in an entirely new context. Thus, our insight is to leverage rich 3D visual features such as those distilled from CLIP [29] to predict physical materials in a direct supervised and feed-forward way. Once trained, our model can associate visual patterns (e.g., "if it looks like vegetation") with physical behaviors (e.g., "it should have material properties similar to a tree"), enabling fast inference and generalization across scenes. To facilitate this research, we have curated and labeled SupPhysVerse, a dataset of 1624 paired 3D objects and annotated materials spanning 10 semantic classes. To our knowledge, this is the largest open-source dataset of paired 3D assets and physical material labels. Trained on SupPhysVerse, our feed-forward network can predict material fields that are 2.21-4.58x better and orders of magnitude faster than test-time optimization methods. By leveraging pretrained visual features, SupPhysField can also zero-shot generalize to real-world scenes despite only ever being trained on synthetic data.

Our contributions include:

1. **Novel Framework for 3D Physics Prediction**: We introduce SupPhysField, a unified framework that predicts discrete material types and continuous physical parameters (Youngs modulus, Poissons ratio, density) directly from visual features using supervised learning.
2. **SupPhysVerse Dataset**: We curate and release SupPhysVerse, the largest open-source dataset of 3D objects with physical material annotations (1624 objects, 10 semantic classes).

3. **Fast and Generalizable Inference**: By leveraging pretrained visual features from CLIP and a feed-forward 3D U-Net, SUPPHYSFIELD performs inference orders of magnitude faster than prior test-time optimization approaches, achieving a 2.21-4.58x improvement in realism scores as evaluated by a state-of-the-art vision-language model.
4. **Zero-Shot Generalization to Real Scenes**: Despite being trained solely on synthetic data, SUPPHYSFIELD generalizes to real-world scenes, showing how visual feature distillation can effectively bridge the sim-to-real gap.
5. **Seamless Integration with MPM Solvers**: The predicted material fields can be directly coupled with Gaussian splatting models for realistic physics simulations under applied forces such as wind and gravity, enabling interactive and visually plausible 3D scene animations.

2 Related Work

2D World Models Some early works [3, 2] learn to predict material labels on 2D images. Recently, learning forward dynamics from 2D video frames has also been explored extensively. For instance, Google's Genie [26] trains a next-frame prediction model conditioned on latent actions derived from user inputs, capturing intuitive 2D physics in an unsupervised manner. While these methods achieve impressive 2D generation and control, they do not explicitly model 3D geometry or a physically grounded world. Other works such as [6, 21] also explore generating or editing images based on learned real-world dynamics. While these methods achieve impressive results in 2D visual synthesis and can imply motion dynamics, they typically do not explicitly model 3D geometry, and only encode physics implicitly via next-frame prediction rather than through explicit material parameters, nor do they infer physically grounded material properties decoupled from appearances. These can lead to problems such as a lack of object permanence or implausible interactions. In contrast, SUPPHYSFIELD directly operates in 3D, predicting explicit physical parameters (e.g., Young's modulus, density) for 3D objects, enabling their integration into 3D physics simulators or neural networks [33] for realistic interaction.

Manual Assignment or Assignment of Physics using LLMs A number of recent methods have explored combining learned 3D scene representations (e.g., Gaussian splatting) with a physics solver where material parameters are assigned manually or through high-level heuristics. This often involves users specifying material types for the scene [36, 1] or using scripted object-to-material dictionaries [28] or large language and vision-language models [12, 4, 37, 19, 35, 22] to guide the assignment.

Test-time material optimization using videos Other works explore more automatic and principled ways to infer material properties using rendered videos. Some techniques [14, 20, 40, 15, 38] optimize material parameters by comparing simulated deformations against ground-truth observations, often requiring ground-truth multi-view videos of objects or ground-truth particle positions under known forces. More recent approaches [13, 23, 39] use video diffusion models as priors to optimize physics via a motion distillation loss. Notably, these approaches suffer from extremely slow per-scene optimization, often taking hours on a single scene, and do not generalize to new scenes. In stark contrast, SUPPHYSFIELD employs a feed-forward neural network that, once trained, predicts physical parameters in seconds, and can generalize to unseen scenes. A recent work Vid2Sim [5] also aims to learn a generalizable material prediction network across scenes. This was done by encoding a front-view video of the object in motion with a foundation video transformer [32] and learning to regress these motion priors into physical parameters. Unlike Vid2Sim, SUPPHYSFIELD does not require videos, relying instead on visual features from static images.

3 Method

Our central thesis is that 3D visual appearance provides sufficient information to recover an object's physical parameters. Texture, shading, and shape features captured from multiple calibrated images correlate with physical quantities such as Young's modulus and Poisson's ratio. By learning a mapping from these visual features to material properties, we can augment a volumetric reconstruction model (e.g., Gaussian splatting) with a point-wise material estimate, without requiring force response observations. In Sec. 3.1, we detail our framework, leveraging rich visual priors from CLIP to predict a material field, which can be used by a physics solver to animate objects responding to

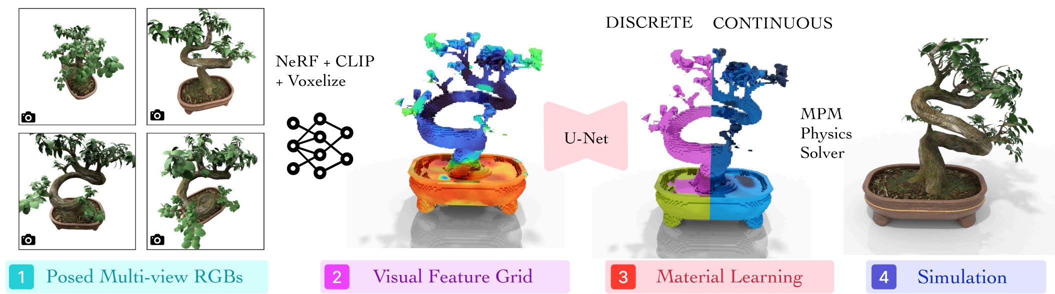

Figure 2: **Method Overview**. From posed multi-view RGB images of a static scene, SUPPHYSFIELD first reconstructs a 3D model with NeRF and distilled CLIP features [30]. Then, we voxelize the features into a regular $N \times N \times N \times D$ grid where N is the grid size and D is the CLIP feature dimension. A U-Net neural network [8] is trained to map the feature grid to the material field $\hat{\mathcal{M}}_G$ which consists of a discrete material model ID and continuous Young's modulus, Poisson's ratio, and density value for each voxel. Coupled with a separately trained Gaussian splatting model, $\hat{\mathcal{M}}_G$ can be used to simulate physics with a physics solver such as MPM.

external forces. To train this model, we curated SUPPHYSVERSE, a large dataset of paired 3D assets and material annotations, as detailed in Sec. 3.2. Figure 2 gives an overview of our method.

3.1 SUPPHYSFIELD Physics Learning

Problem Formulation Formally, the goal is to learn a mapping:

$$f_\theta : (\mathcal{I}, \Pi) \longrightarrow \hat{\mathcal{M}} \tag{1}$$

that turns some calibrated RGB images of the static scene $\mathcal{I} = \{I_k\}_{k=1}^{K}$ and their joint camera specification Π into a continuous three-dimensional *material field*. For every point $\mathbf{p} \in \mathbb{R}^3$ within the scene bounds, the field returns

$$\hat{\mathcal{M}}(\mathbf{p}) \;=\; \Big(\hat{\ell}(\mathbf{p}),\; \hat{E}(\mathbf{p}),\; \hat{\nu}(\mathbf{p}),\; \hat{d}(\mathbf{p})\Big) \;,$$

where $\hat{\ell} : \mathbb{R}^3 \to \{1, \ldots, L\}$ is the discrete material class and $\hat{E}, \hat{\nu}, d : \mathbb{R}^3 \to \mathbb{R}$ are the continuous Young's modulus, Poisson's ratio, and density value respectively. Recall that the discrete material class, also known as the constitutive law, in Material Point Method is a combination of the choices of an expert-defined hyperelastic energy function $\mathcal{E}$ and return mapping $\mathcal{P}$ (Sec. A.1). Learning a point-mapping like this provides a fine-grained material segmentation where for every spatial location we assign both a semantic material label and the physical parameters that characterise that material. Learning the mapping in Eqn. (1) directly from 2D images to 3D materials is clearly not simple neither sample efficient. Instead, we leverage a distilled feature field which has rich visual priors to represent the intermediate mapping between 2D images and 3D visual featutes, and then a separate U-Net architecture to compute the mapping between 3D visual features and physical materials. We describe these components below.

3D Visual Feature Distillation Recent work on distilled feature fields has shown that dense 2D visual feature embeddings extracted from foundation models, such as CLIP, based on images can be lifted into 3D, yielding a volumetric representation that is both geometrically accurate and rich in terms of visual and semantic priors [30]. These works have used distilled features to better understand 3D scenes for robotics manipulation tasks. To our knowledge, this idea has not been applied to material prediction, despite the promise in using semantically rich 3D feature volumes to encode cues about an objects composition and stiffness. Here we augment the classical NeRF representation [25] to predict a view-independent feature vector in addition to color and density, i.e.,

$$F_\theta : (\mathbf{x}, \mathbf{d}) \;\longmapsto\; \big(\mathbf{f}(\mathbf{x}),\; c(\mathbf{x}, \mathbf{d}),\; \sigma(\mathbf{x})\big) \;,$$

where $c \in \mathbb{R}^3$, and $\sigma \in \mathbb{R}_{\geq 0}$ are the standard color and radiance from NeRF and the extra output $\mathbf{f} \in \mathbb{R}^d$ is a high-dimensional descriptor capturing visual semantics (e.g., object identity or other attributes), which we assume to be view-independent. We can render both the color and feature channels into any camera view via the standard volume rendering procedure. Concretely, for a camera ray $r(t) = \mathbf{o} + t\mathbf{d}$ passing through a pixel p, the accumulated color $C(p)$ and feature vector $F(p)$ are given by integrals along the ray:

$$C(p) = \int_{t_n}^{t_f} T(t), \sigma(r(t)), c(r(t), \mathbf{d})\, dt \qquad F(p) = \int_{t_n}^{t_f} T(t), \sigma(r(t)), f(r(t))\, dt \;, \tag{2}$$

where $T(t) = \exp\big(-\int_{t_n}^{t} \sigma(r(s))\,ds\big)$ is the accumulated transmittance from the ray origin to depth t. At each training iteration, a batch of rays is sampled from the input views. For each ray r (pixel p), we enforce that the rendered color $C(p)$ matches the ground-truth pixel RGB $C^*(p)$, while the rendered feature $F(p)$ matches the corresponding CLIP-based feature vector $F^*(p)$ extracted from the image. The loss of the network is:

$$\mathcal{L} = \sum_p \big\|C(p) - C^*(p)\big\|_2^2 + \lambda_{\text{feat}} \sum_p \big\|F(p) - F^*(p)\big\|_2^2 \ ;$$

the first term enforces color fidelity, while the second aligns the rendered volumetric CLIP features with the dense 2D features extracted from the training images.

From a trained distilled feature field F_θ, we obtain a regular feature grid F_G of dimension $N \times N \times N \times D$ grid, where $N = 64$ is the grid size and $D = 768$ is the CLIP feature dimension. This is done via voxelization using known scene bounds. For our synthetic dataset, we center and normalize all objects within a unit cube.

Material Grid Learning Our material learning network f_M consists of a feature projector f_P and a U-Net f_U. As the CLIP features are very high-dimensional which can cause memory issues on GPUs, we learn a feature projector network f_P, which consists of three layers of 3D convolution mapping CLIP features $\mathbb{R}^{768}$ to a low-dimensional manifold $\mathbb{R}^{64}$. We then use the U-Net architecture f_U from OpenAI's Guided Diffusion codebase [8] with 2D convolution replaced by 3D kernels to learn the mapping from the projected feature grid F_G to a material grid $\hat{\mathcal{M}}_G(\mathbf{p})$, which is a voxelized version of the material field $\hat{\mathcal{M}}(\mathbf{p})$. The feature projector f_P and U-Net f_U are jointly trained end-to-end via a cross entropy and mean-squared error loss to both predict the discrete material classification and the continuous values including Young's modulus, Poisson's ratio and density.

We found that our voxel grids are very sparse with around 98% of the voxels being background. Naively trained, the material network f_M would learn to always predict background. Thus, we also separately compute an occupancy mask grid $\mathbb{M} \in \mathbb{R}^N \times \mathbb{R}^N \times \mathbb{R}^N$, constructed by filtering out all voxels whose NeRF densities fall below a threshold $\alpha = 0.01$. The supervised losses—cross entropy and mean squared errors—are only enforced on the occupied voxels. Concretely, the masked supervised loss consists of a discrete cross entropy and continuous mean-squared error loss:

$$\begin{aligned}\mathcal{L}_{\text{sup}} = \frac{1}{N_{occ}} \sum_{\mathbf{p}\in\mathcal{G}} \mathbb{M}(\mathbf{p}) \Big[\lambda \cdot \mathrm{CE}(\hat{\ell}(\mathbf{p}), \ell^{GT}(\mathbf{p})) + (\hat{E}(\mathbf{p}) - E^{GT}(\mathbf{p}))^2 \\ + (\hat{\nu}(\mathbf{p}) - \nu^{GT}(\mathbf{p}))^2 + (\hat{d}(\mathbf{p}) - d^{GT}(\mathbf{p}))^2\Big] \ ,\end{aligned} \tag{3}$$

where $N_{occ} = \sum_{\mathbf{p}\in\mathcal{G}} \mathbb{M}(\mathbf{p})$ is the total number of occupied voxels in the grid, $\hat{\ell}(\mathbf{p})$ and $\ell^{GT}(\mathbf{p})$ are the predicted material class logits and the ground-truth, CE is the cross entropy loss, λ is a loss balancing factor, and E, ν, d are the Young's modulus, Poisson's ratio and density values, respectively. The material network f_G is trained on 12 NVIDIA RTX A6000 GPUs, each with a batch size of 4, in one day using the Adam optimizer [18].

Physics Simulation We use the Material Point Method (MPM) to simulate physics. The MPM solver (Sec. A.1.2) takes a point cloud of initial particle poses along with predicted material properties, and the external force specification, and simulates the particles' transformations and deformations. Although it is possible to sample particles from a NeRF model (e.g., via Poisson disk sampling [9]), we have found that it is easier to use a Gaussian Splatting model (Sec. A.1.1) as each Gaussian can naturally be thought of as a MPM particle [36]. Thus, we separately learn a Gaussian splatting model from posed multi-view RGB images. We then transfer the material properties from our predicted material grid into the Gaussian splatting model via nearest neighbor interpolation.

3.2 SUPPHYSVERSE Dataset

We collect one of the largest and highest quality known datasets of diverse objects with annotated physical materials. Our dataset (Fig. 3) covers 10 semantic classes, ranging from organic matter (trees, shrubs, grass, flowers) and granular media (sand, snow and mud) to hollow containers (soda-cans, metal crates), and toys (rubber ducks, sport balls). The dataset is sourced from Objaverse [7], the largest open-source dataset of 3D assets. Since Objaverse objects do not have physical parameter annotations, we develop an automatic multi-stage labeling pipeline leveraging foundation vision-language models i.e., Gemini-2.5-Pro [31]. More details is given in Appendix A.2.

Figure 3: **SupPhysVerse Dataset Overview.** We collect 1624 high-quality single-object assets, spanning 10 semantic classes (a), and 6 constitutive material types (b). The dataset is annotated with detailed physical properties including spatially varying discrete material types (b), Young's modulus (c), Poisson's ratio (d), and mass density (e). The left figure shows representative examples from the dataset: organic matter (*tree, shrubs, grass, flowers*), deformable toys (*rubber ducks*), sports equipment (*sport balls*), granular media (*sand, snow & mud*), and hollow containers (*soda cans, metal crates*).

4 Experiments

Dataset We train SUPPHYSFIELD on a random 90% split of the SUPPHYSVERSE dataset. We evaluate on 38 synthetic scenes from the test set of SUPPHYSVERSE, and three real-world scene from the NeRF [25] and LERF [17] datasets.

Simulation Details We use the material point method (MPM) implementation from PhysGaussian [36] as the physics solver. The solver takes a gaussian splatting model augmented with physics where each Gaussian particle also has a discrete material model ID, and continuous Young's modulus, Poisson's ratio, and density values. Each simulation is run for around 50 to 125 frames on a single Nvidia RTX A6000 GPU. External forces such as gravity and wind are applied to the static scenes as boundary conditions to create physics animations.

Baselines We evaluate SUPPHYSFIELD against two recent test-time optimization methods: DreamPhysics [13] and OmniPhysGS [23], and a LLM method – NeRF2Physics [37]. DreamPhysics optimizes a Young's modulus field, requiring users to specify other values including material ID, Poisson's ratio, and density. OmniPhysGS, on the other hand, selects a hyperelastic energy density function and a return mapping model, which, in combination, specifies a material ID for each point in the field, requiring other physics parameters to be manually specified. Both methods rely on a user prompt such as "a tree swing in the wind" and a generative video diffusion model to optimize a motion distillation loss. SUPPHYSFIELD, in contrast, infers all discrete and continuous parameters jointly (Fig. 5). NeRF2Physics first captions the scene and query a LLM for all plausible material types (e.g., "metal") along with the associated continuous values. Then, the material semantic names are associated with 3D points in the CLIP feature field, and physical properties are thus assigned via weighted similarities. This method is similar to our dataset labeling in principle with some notable difference as detailed in Appendix A.2, allowing SUPPHYSVERSE to have much more high-quality labels. SUPPHYSFIELD thus produces much less noisy predictions (Fig. 6).

Evaluation Metrics We utilize a state-of-the-art vision-language model, Gemini-2.5-Pro [31], from Google as a judge. The model is prompted to compare the rendered candidate animations generated using physics parameters predicted by different baselines, and score those videos on a scale from 0 to 5, where a higher score is better. We also measure the reconstruction quality using

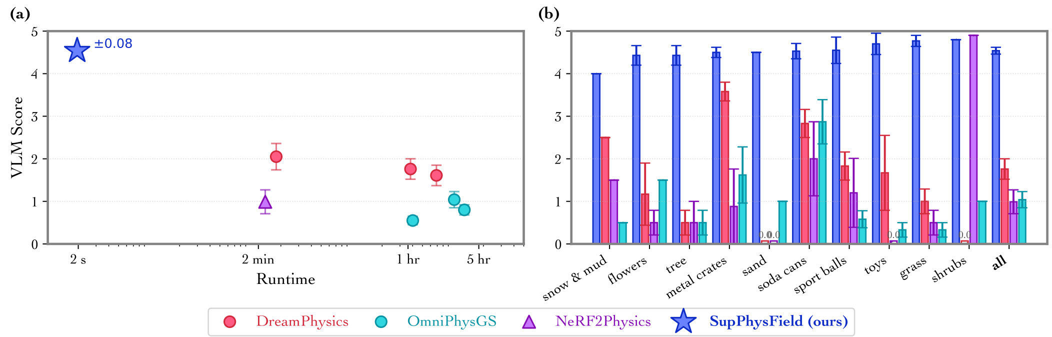

Figure 4: **Main VLM Results.** (a) **VLM score versus wall-clock time:** SUPPHYSFIELD is three orders of magnitude faster than previous works while achieving 2.21-4.58x improvement in realism. Test-time optimization methods are run with varying numbers of epochs i.e., $1, 25, 50$ for DreamPhysics and $1, 2, 5$ for OmniPhysGS while inference methods are only run once. (b) **Per-class VLM score:** Our method leads on every object class. Standard errors are also included.

Table 1: **Main Quantitative Results.** We report the average reconstruction quality (PSNR, SSIM) against the reference videos in SUPPHYSVERSE, the VLM scores, and five other metrics our method optimizes including material accuracy and continuous errors over E, ν, ρ. Standard errors and 95% CI are also included, and best values are **bolded**. SUPPHYSFIELD-CLIP is by far the best method across all metrics, achieving 2.21-4.58x improvement in VLM score and 3.6-30.3% gains in PSNR and SSIM. Our CLIP variant is also notably more accurate than RGB and occupancy features as measured by material class accuracy and average continuous MSE on the test set. While our method simultaneously reovers all physical properties, some prior works only predict a subset, hence “-".

Method	**PSNR** ↑	**SSIM** ↑	**VLM** ↑	**Mat. Acc.** ↑	**Avg. Cont. MSE** ↓	E **err** ↓	ν **err** ↓	ρ **err** ↓
DreamPhysics [13]								
1 epoch	19.398 ±1.090	0.880 ±0.020	2.05 ±0.31	-	-	2.393 ±0.123	-	-
25 epochs	19.078 ±0.939	0.881 ±0.019	1.76 ±0.24	-	-	1.419 ±0.097	-	-
50 epochs	19.189 ±0.980	0.880 ±0.020	1.61 ±0.24	-	-	1.387 ±0.097	-	-
OmniPhysGS [23]								
1 epoch	17.907 ±0.359	0.882 ±0.007	0.55 ±0.10	0.072 ±0.0511	-	-	-	-
2 epochs	17.889 ±0.372	0.882 ±0.007	1.04 ±0.19	0.109 ±0.0704	-	-	-	-
5 epochs	17.842 ±0.354	0.883 ±0.007	0.80 ±0.12	0.104 ±0.0681	-	-	-	-
NeRF2Physics [37]	18.517 ±0.644	0.886 ±0.013	0.99 ±0.28	0.274 ±0.001	0.858 ±0.109	1.115 ±0.165	0.462 ±0.106	0.997 ±0.162
SUPPHYSFIELD								
Occupancy	17.887 ±1.524	0.866 ±0.027	1.76 ±0.41	0.686 ±0.054	0.175 ±0.021	0.138 ±0.027	0.177 ±0.027	0.209 ±0.032
RGB	18.652 ±2.031	0.861 ±0.035	2.53 ±0.46	0.641 ±0.066	0.197 ±0.023	0.144 ±0.026	0.191 ±0.028	0.256 ±0.035
CLIP (ours)	**23.256 ±2.456**	**0.918 ±0.023**	**4.54 ±0.08**	**0.809 ±0.043**	**0.105 ±0.013**	**0.072 ±0.016**	**0.118 ±0.015**	**0.125 ±0.020**

PSNR and SSIM metric against the reference videos in the SUPPHYSVERSE dataset. Other metrics our method optimizes including class accuracy and continuous errors over E, ν, ρ are also computed.

4.1 Synthetic Scene Experiments

Figure 4 (a) plots Gemini score versus runtime. SUPPHYSFIELD achieves a VLM score of $\mathbf{4.54 \pm 0.08}$ – a **2.21-4.58x** improvement over all baselines – while reducing inference time from minutes or hours to **2 s**. A per-class breakdown in Fig. 4 (b) shows our lead in all classes. In Table 1, our model improves perceptual metrics such as PSNR and SSIM by $3.6 - 30.3\%$ and VLM scores by $2.21 - 4.58$x over prior works. Figure 5 qualitatively visualizes the physical properties predicted by our network, showing SUPPHYSFIELD's ability to cleanly and accurately recover discrete and continuous parameters across a diverse sets of objects and continuous value spectrum. Figure 6 visualises four representative scenes, comparing SUPPHYSFIELD against prior works. DreamPhysics leaves stiff artifacts due to missegmentation or overly high predicted E values, OmniPhysGS collapses under force, and NeRF2Physics introduces high-frequency noise, whereas SUPPHYSFIELD generates smooth, class-consistent motion and segment boundaries.

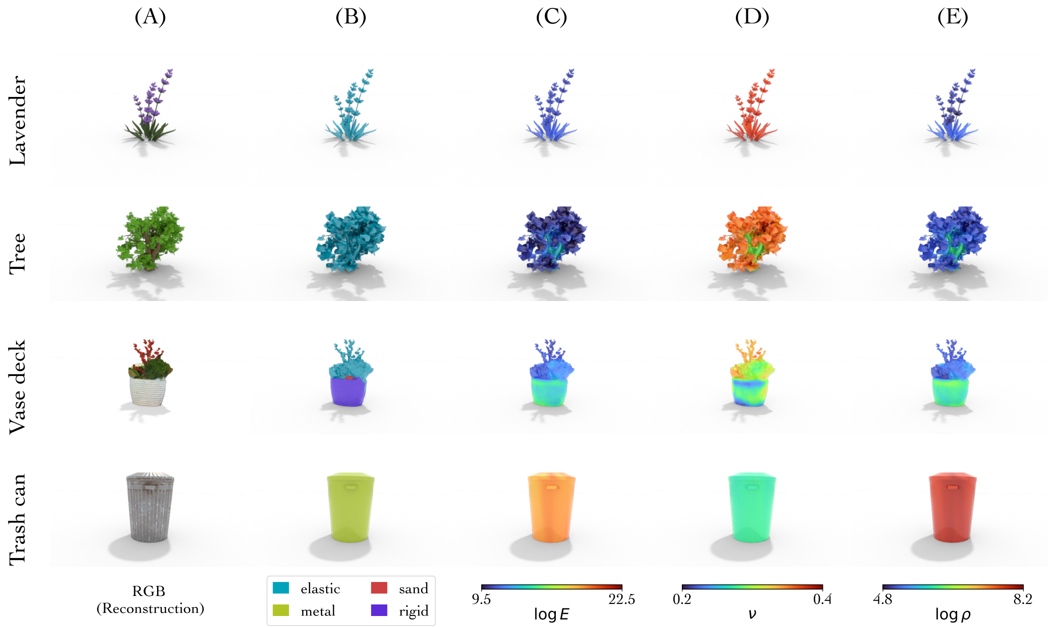

Figure 5: **SUPPHYSFIELD Prediction Visualization.** SUPPHYSFIELD simultaneously recovers discrete material class (B), continuous Young's modulus (C), Poisson's ratio (D), and mass density (E) with a high degree of accuracy. For example, the model correctly labels foliage as elastic and the metal can as rigid, while recovering realistic stiffness and density gradients within each object.

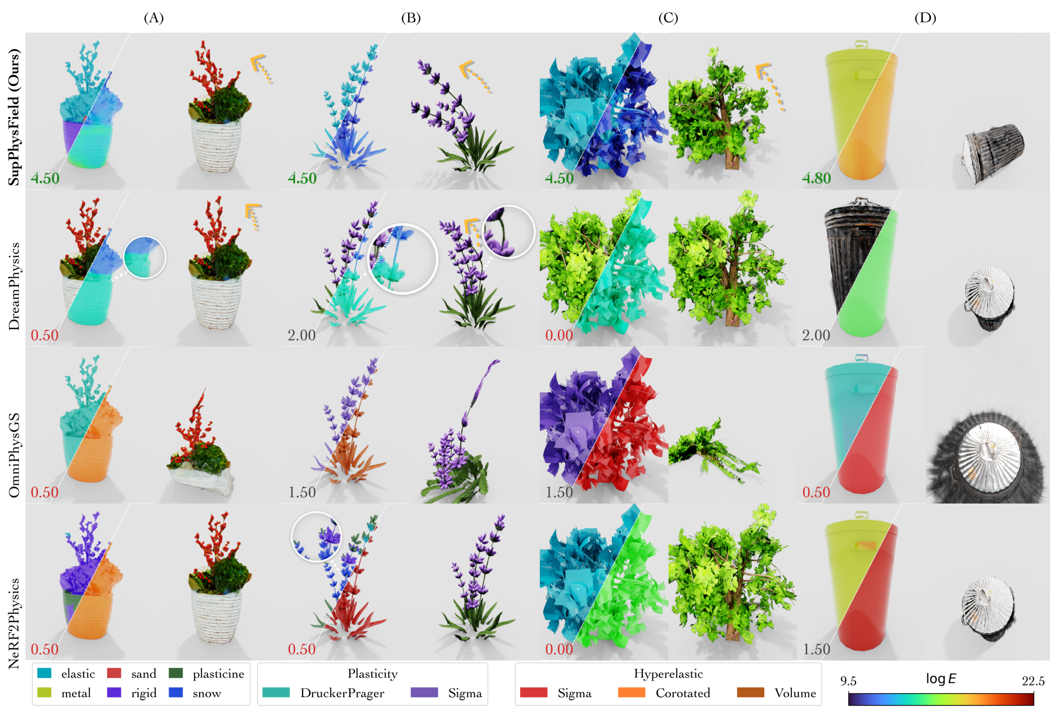

Figure 6: **Qualitative comparison on synthetic scenes.** Best Gemini score per scene is highlighted in **Green** while low scores are in **Red**. We visualized the predicted material class and E predictions (left, right respectively) for SUPPHYSFIELD and Nerf2Physics, E for DreamPhysics (right), and the plasticity and hyperelastic function classes predicted by OmniPhysGS. SUPPHYSFIELD produces stable, physically plausible motion while DreamPhysics remains overly stiff due to inaccurate fine-grained E prediction or too high E (e.g., see tree (C)), OmniPhysGS collapses under load due to unrealistic combination of plasticity and hyperelastic functions, and NeRF2Physics exhibits noisy artifacts. Please https://neurips-2025-20627.github.io/ for the videos.

Figure 7: **SupPhysField's Zero-shot Real-scene Generalization.** Trained only on synthetic SupPhysVerse, SupPhysField can predict plausible physic properties, enabling realistic MPM simulation of real scenes. Here, we visualize the material types (left) and Young's modulus (right) prediction in the first frame, and subsequent frames impacted by a wind force. Please see the videos in our website https://neurips-2025-20627.github.io/.

4.2 Zero-shot Generalization to Real-World Scenes

Without any real-scene supervision, SupPhysField can zero-shot generalize as shown in Fig. 7. Our method correctly assigns rigid vase bases and flexible leaves, yielding realistic motion that closely matches human expectation. No other baseline generalises under this setting.

4.3 SupPhysField's Feature Type Ablation

Replacing CLIP with RGB or occupancy features drops VLM score by 40-60 % and nearly doubles parameter MSE (Table 1, rows Occupancy and RGB). The material class prediction also dramatically drops across most classes as shown in Fig. 9. Figure 8 shows the failure modes for real scenes, highlighting RGB and occupancy's struggle to generalize to unseen data as compared to CLIP.

5 Conclusion and Limitations

We presented SupPhysField, a framework that jointly reconstructs geometry, appearance, and explicit physical material fields from posed RGB images. By distilling rich CLIP features into 3D and training a feed-forward 3D U-Net with per-voxel material supervision on our new SupPhysVerse dataset, SupPhysField avoids the expensive test-time optimization required by prior work. Once trained, it produces full material fields in a few seconds, improving Gemini realism scores by 14.5% to 51.8% over DreamPhysics and OmniPhysGS while reducing inference time by three orders of magnitude. SupPhysField leverages CLIP's strong visual priors, which enables zero-shot transfer to real scenes, even though it is only trained on synthetic data. The method enables realistic, physically plausible 3D scene animation with off-the-shelf MPM solvers.

Limitations We take the first step towards learning a supervised model for physical material prediction. Like prior art, our work focuses on single object interaction leaving multi-object scenes

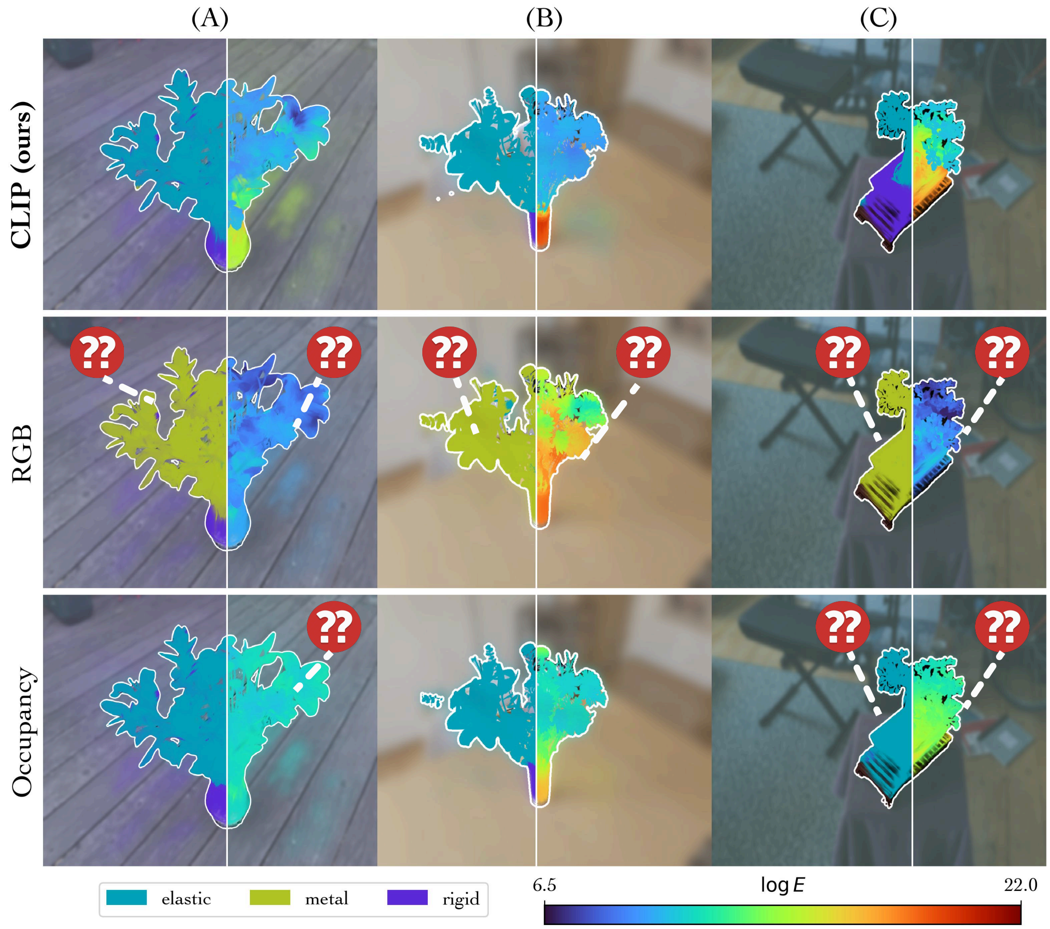

Figure 8: **SUPPHYSFIELD's Feature Type Ablation on Real Scenes.** Replacing CLIP features with RGB or occupancy severely degrades the material prediction. Incorrect predictions such as leave mislaballed as metal or Young's modulus being uniform within an object are marked with question marks. This highlights the power of pretrained visual features in bridging the sim2real gap.

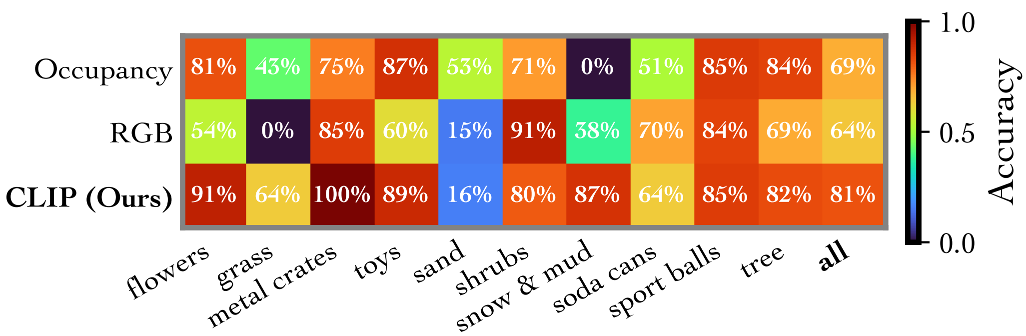

Figure 9: **SUPPHYSFIELD Ablation's Per-class Accuracy on synthetic scenes**. CLIP features generalizes in synthetic scenes, outperfoming RGB and occupancy on 9/10 classes.

for future investigation. Another limitation is that while our UNet predict a point estimate for each voxel, materials in the real-world contain uncertainty that visual information alone cannot resolve (e.g., a tree can be stiff or flexible). A promising extension is to learn a distribution of materials (e.g., using diffusion) instead.

References

[1] Jad Abou-Chakra, Krishan Rana, Feras Dayoub, and Niko Suenderhauf. Physically embodied gaussian splatting: A realtime correctable world model for robotics. In *8th Annual Conference on Robot Learning*, 2024. URL https://openreview.net/forum?id=AEq0onGrN2.

[2] Daniel M Bear, Elias Wang, Damian Mrowca, Felix J Binder, Hsiao-Yu Fish Tung, RT Pramod, Cameron Holdaway, Sirui Tao, Kevin Smith, Fan-Yun Sun, et al. Physion: Evaluating physical prediction from vision in humans and machines. *arXiv preprint arXiv:2106.08261*, 2021.

[3] Sean Bell, Paul Upchurch, Noah Snavely, and Kavita Bala. Material recognition in the wild with the materials in context database. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3479–3487, 2015.

[4] 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. *arXiv preprint arXiv:2503.20746*, 2025.

[5] Chuhao Chen, Zhiyang Dou, Chen Wang, Yiming Huang, Anjun Chen, Qiao Feng, Jiatao Gu, and Lingjie Liu. Vid2sim: Generalizable, video-based reconstruction of appearance, geometry and physics for mesh-free simulation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

[6] Xi Chen, Zhifei Zhang, He Zhang, Yuqian Zhou, Soo Ye Kim, Qing Liu, Yijun Li, Jianming Zhang, Nanxuan Zhao, Yilin Wang, et al. Unireal: Universal image generation and editing via learning real-world dynamics. *arXiv preprint arXiv:2412.07774*, 2024.

[7] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects, 2022. URL https://arxiv.org/abs/2212.08051.

[8] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[9] Yutao Feng, Yintong Shang, Xuan Li, Tianjia Shao, Chenfanfu Jiang, and Yin Yang. Pie-nerf: Physics-based interactive elastodynamics with nerf, 2023.

[10] Michael Fischer, Iliyan Georgiev, Thibault Groueix, Vladimir G Kim, Tobias Ritschel, and Valentin Deschaintre. Sama: Material-aware 3d selection and segmentation. *arXiv preprint arXiv:2411.19322*, 2024.

[11] Minghao Guo, Bohan Wang, Pingchuan Ma, Tianyuan Zhang, Crystal Elaine Owens, Chuang Gan, Joshua B. Tenenbaum, Kaiming He, and Wojciech Matusik. Physically compatible 3d object modeling from a single image. *arXiv preprint arXiv:2405.20510*, 2024.

[12] Hao-Yu Hsu, Zhi-Hao Lin, Albert Zhai, Hongchi Xia, and Shenlong Wang. Autovfx: Physically realistic video editing from natural language instructions. *arXiv preprint arXiv:2411.02394*, 2024.

[13] Tianyu Huang, Yihan Zeng, Hui Li, Wangmeng Zuo, and Rynson WH Lau. Dreamphysics: Learning physical properties of dynamic 3d gaussians with video diffusion priors. *arXiv preprint arXiv:2406.01476*, 2024.

[14] Krishna Murthy Jatavallabhula, Miles Macklin, Florian Golemo, Vikram Voleti, Linda Petrini, Martin Weiss, Breandan Considine, Jerome Parent-Levesque, Kevin Xie, Kenny Erleben, Liam Paull, Florian Shkurti, Derek Nowrouzezahrai, and Sanja Fidler. gradsim: Differentiable simulation for system identification and visuomotor control. *International Conference on Learning Representations (ICLR)*, 2021. URL https://openreview.net/forum?id=c_E8kFWfhp0.

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

[16] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023.

[17] Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. Lerf: Language embedded radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19729–19739, 2023.

[18] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[19] Long Le, Jason Xie, William Liang, Hung-Ju Wang, Yue Yang, Yecheng Jason Ma, Kyle Vedder, Arjun Krishna, Dinesh Jayaraman, and Eric Eaton. Articulate-anything: Automatic modeling of articulated objects via a vision-language foundation model. *arXiv preprint arXiv:2410.13882*, 2024.

[20] 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 *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=tVkrbkz42vc.

[21] Zhengqi Li, Richard Tucker, Noah Snavely, and Aleksander Holynski. Generative image dynamics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24142–24153, 2024.

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

[23] Yuchen Lin, Chenguo Lin, Jianjin Xu, and Yadong MU. OmniphysGS: 3d constitutive gaussians for general physics-based dynamics generation. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=9HZtP6I5lv.

[24] 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 *International Conference on Machine Learning*, pages 23279–23300. PMLR, 2023.

[25] 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. *Communications of the ACM*, 65(1):99–106, 2021.

[26] Jack Parker-Holder, Philip Ball, Jake Bruce, Vibhavari Dasagi, Kristian Holsheimer, Christos Kaplanis, Alexandre Moufarek, Guy Scully, Jeremy Shar, Jimmy Shi, Stephen Spencer, Jessica Yung, Michael Dennis, Sultan Kenjeyev, Shangbang Long, Vlad Mnih, Harris Chan, Maxime Gazeau, Bonnie Li, Fabio Pardo, Luyu Wang, Lei Zhang, Frederic Besse, Tim Harley, Anna Mitenkova, Jane Wang, Jeff Clune, Demis Hassabis, Raia Hadsell, Adrian Bolton, Satinder Singh, and Tim Rocktäschel. Genie 2: A large-scale foundation world model. 2024. URL https://deepmind.google/discover/blog/genie-2-a-large-scale-foundation-world-model/.

[27] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural Radiance Fields for Dynamic Scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[28] Ri-Zhao Qiu, Ge Yang, Weijia Zeng, and Xiaolong Wang. Feature splatting: Language-driven physics-based scene synthesis and editing. *arXiv preprint arXiv:2404.01223*, 2024.

[29] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[30] William Shen, Ge Yang, Alan Yu, Jansen Wong, Leslie Pack Kaelbling, and Phillip Isola. Distilled feature fields enable few-shot language-guided manipulation, 2023. URL https://arxiv.org/abs/2308.07931.

[31] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[32] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. *Advances in neural information processing systems*, 35:10078–10093, 2022.

[33] Chen Wang, Chuhao Chen, Yiming Huang, Zhiyang Dou, Yuan Liu, Jiatao Gu, and Lingjie Liu. Physctrl: Generative physics for controllable and physics-grounded video generation. In *arXiv preprint*, 2025.

[34] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in neural information processing systems*, 33:5776–5788, 2020.

[35] Hongchi Xia, Zhi-Hao Lin, Wei-Chiu Ma, and Shenlong Wang. Video2game: Real-time, interactive, realistic and browser-compatible environment from a single video, 2024.

[36] Tianyi Xie, Zeshun Zong, Yuxing Qiu, Xuan Li, Yutao Feng, Yin Yang, and Chenfanfu Jiang. Physgaussian: Physics-integrated 3d gaussians for generative dynamics. *arXiv preprint arXiv:2311.12198*, 2023.

[37] Albert J Zhai, Yuan Shen, Emily Y Chen, Gloria X Wang, Xinlei Wang, Sheng Wang, Kaiyu Guan, and Shenlong Wang. Physical property understanding from language-embedded feature fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 28296–28305, 2024.

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

[39] 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 *European Conference on Computer Vision*. Springer, 2024.

[40] Licheng Zhong, Hong-Xing Yu, Jiajun Wu, and Yunzhu Li. Reconstruction and simulation of elastic objects with spring-mass 3d gaussians. *European Conference on Computer Vision (ECCV)*, 2024.

A Appendix

A.1 Preliminaries

This section briefly reviews foundational concepts in 3D scene representation and physics modeling relevant to our work.

A.1.1 Learned Scene Representation

Reconstructing 3D scenes from 2D images is commonly achieved by learning a parameterized representation, F_θ, optimized to render novel views that match observed images $\{I^{(i)}\}_{i=1}^{M}$ given camera parameters $\{\pi^{(i)}\}_{i=1}^{M}$. This typically involves minimizing a photometric loss:

$$\min_\theta \sum_{i=1}^{M} \left\| \hat{I}^{(i)}(\theta) - I^{(i)} \right\|_2^2 \quad ,$$

where $\hat{I}^{(i)}(\theta)$ is the image rendered from viewpoint i. Two prominent representations are Neural Radiance Fields (NeRF) and Gaussian Splatting (GS) models.

Neural Radiance Fields (NeRF) [25] model a scene as a continuous function $F_\theta : (\mathbf{x}, \mathbf{d}) \mapsto (c, \sigma)$, mapping a 3D location $\mathbf{x}$ and viewing direction $\mathbf{d}$ to an emitted color c and volume density σ. Images are synthesized using volume rendering, integrating color and density along camera rays. This process' differentiability allows for end-to-end optimization from images.

Gaussian Splatting (GS) [16] represents scenes as a collection of 3D Gaussian primitives, each defined by a center μ_i, covariance Σ_i, color $\mathbf{c}_i$, and opacity α_i. These Gaussians are projected onto the image plane and blended using alpha compositing to render views.

In our work, the principles of neural scene representation, particularly NeRF-like architectures, are leveraged not only for visual reconstruction but also for creating dense 3D visual feature fields. As detailed in Sec. 3.1, we utilize a NeRF-based model to distill 2D image features (e.g., from CLIP) into a volumetric 3D feature grid. This 3D feature representation, F_G, then serves as the primary input to our physics prediction network. For subsequent physics simulation, GS offers a convenient particle-based representation.

A.1.2 Material Point Method (MPM) for Physics Simulation

To simulate how objects move and deform under applied forces, a physics engine requires knowledge of their material properties. These properties are typically defined within the framework of continuum mechanics, which describes the behavior of materials at a macroscopic level. The fundamental equations of motion (conservation of mass and momentum) are:

$$\rho \frac{D\mathbf{v}}{Dt} \;=\; \nabla \cdot \boldsymbol{\sigma} \;+\; \mathbf{f}^{\text{ext}} \qquad\qquad \nabla \cdot \mathbf{v} \;=\; 0 \;, \tag{4}$$

where ρ is mass density, $\mathbf{v}$ the velocity field, $\boldsymbol{\sigma}$ the Cauchy stress tensor, and $\mathbf{f}^{\text{ext}}$ any external force (e.g. gravity or user interactions). The material-specific *constitutive laws* define how $\boldsymbol{\sigma}$ depends on the local deformation gradient $\mathbf{F}$. For elastic materials, stress depends purely on the recoverable strain; for plastic materials, a yield condition enforces partial flow once strain exceeds a threshold.

Constitutive Laws and Parameters Most continuum simulations separate the constitutive model into two core components:

$$\begin{aligned} &\mathcal{E}_\mu : \mathbf{F}^e \;\mapsto\; \mathbf{P}, \\ &\mathcal{P}_\mu : \mathbf{F}^{\,e,\text{trial}} \;\mapsto\; \mathbf{F}^{\,e,\text{new}} \quad , \end{aligned} \tag{5}$$

where $\mathbf{F}^e$ is the *elastic* portion of the deformation gradient, $\mathbf{P}$ is the (First) Piola–Kirchhoff stress, and μ represents the set of material parameters (e.g. Youngs modulus E, Poisson's ratio ν, yield stress). The *elastic law* $\mathcal{E}_\mu$ computes stress from the current elastic deformation, while the *return-mapping* $\mathcal{P}_\mu$ projects any trial elastic update $\mathbf{F}^{\,e,\text{trial}}$ onto the feasible yield surface if plastic flow is triggered. Typically, the constitutive laws i.e., $\mathcal{E}_\mu$ and $\mathcal{P}_\mu$ are hand-designed by domain experts. The choice of $\mathcal{E}$ and $\mathcal{P}$ jointly define a class of material (e.g., rubber). Within a material class, additional continuous parameters μ including Young's modulus, Poisson's ratio and density can be specified for a more granular control of the material properties (e.g., stiffness of rubber). In our work, SUPPHYSFIELD jointly predicts the discrete material model and the continuous material parameters.

A.2 SupPhysVerse Dataset Details

We heavily curate the dataset to a set of 1624 objects after a multi-stage filter that removes multi-object scenes, missing textures, duplicated assets, and objects whose material labeling is either ambiguous or physically implausible.

First, we define some object class (e.g., “tree") and some alternative query terms (e.g., “ficus, fern, evergreen etc"). We then use a sentence transformer model [34] to compute the cosine similarity between the search terms and the name of each Objaverse object. We select $k = 500$ objects with the highest similarity score for each class, creating an initial candidate pool. However, since Objaverse objects vary greatly in asset quality, lighting conditions, and some scenes contain multiple objects which are not suitable for our material learning, an additional filtering step is needed. The Gemini VLM is prompted to filter out low-quality or unsuitable scenes. A distilled NeRF model is fitted to each object. Then, the VLM is provided five multi-view RGB images of an object, and prompted to provide a list of the object's semantic parts along with associated material class and ranges for continuous values (e.g., see Fig. 10). The ranges such as $E \in \{1e4, 1e5\}$ allow us to simulate a wider range of dynamics from flexible to more rigid trees. The VLM is also prompted to specify a list of constraints such as to ensure that the leaf's density is lower than the trunk's. We then sample the continuous values from the VLM's specified ranges subject to the constraint via rejection sampling. The semantic parts (e.g., “pot") are used with the CLIP distilled feature field to compute a 3D semantic segmentation of the object into parts, and the sampled material properties are applied uniformly to all points within a part. This ground-truth material and feature fields are then voxelized into regular grids for use in supervised learning by the SupPhysField framework.

```
{ "pot": {"density": [400, 600], "E": [1e8, 2e8], "nu": [0.2, 0.4], "material_id": 6},
  "trunk": {"density": [300, 500], "E": [5e5, 1e7], "nu": [0.3, 0.45], "material_id": 0},
  "leaf": {"density": [100, 300], "E": [1e4, 1e5], "nu": [0.35, 0.48], "material_id": 0},
  "constraints" : "assert leaf_{density} < trunk_{density}, ...",}
```

Figure 10: An example of a material annotation by Gemini VLM for the SupPhysVerse dataset.

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 abstract and introduction state the claims and contributions of this paper.

   Guidelines:
   - The answer NA 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 NA 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: We discuss the limitations in the last section.

   Guidelines:
   - The answer NA 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: [NA]

   Justification: N/A

   Guidelines:
   - The answer NA 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: The paper discusses all implementation details necessary for reproduction. We will also release the training data, code, and checkpoints.

   Guidelines:

   - The answer NA 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: We will release the training data, code, and checkpoints.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://nips.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://nips.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, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide implementation details.

   Guidelines:

   - The answer NA 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: [Yes]

   Justification: We include standard error bars along with the mean scores.

   Guidelines:

   - The answer NA 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: We provide details on our hardware setup and training duration.

   Guidelines:

   - The answer NA 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 conform to the NeurIPS Code of Ethics.

Guidelines:

- The answer NA 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: [NA]

Justification: The authors have not ascertained a path towards misuse using this technology.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA 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., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The authors have not ascertained a path towards misuse using this technology.

Guidelines:

- The answer NA 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: The creators of the original dataset and models are properly cited.

Guidelines:

- The answer NA 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 discuss our dataset at length in Sec. 3.2.

    Guidelines:
    - The answer NA 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: [NA]

    Justification: The paper does not involve human subjects.

    Guidelines:
    - The answer NA 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: [NA]

    Justification: The paper does not involve human subjects.

    Guidelines:
    - The answer NA 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 rigorousness, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: The paper discuss the use of LLMs as it is critical to the paper's approach.

Guidelines:

- The answer NA 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 (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.