SOPHY: Generating Simulation-Ready Objects with PHYsical Materials

Junyi Cao[1*] Evangelos Kalogerakis[1,2]
[1]UMass Amherst [2]TU Crete

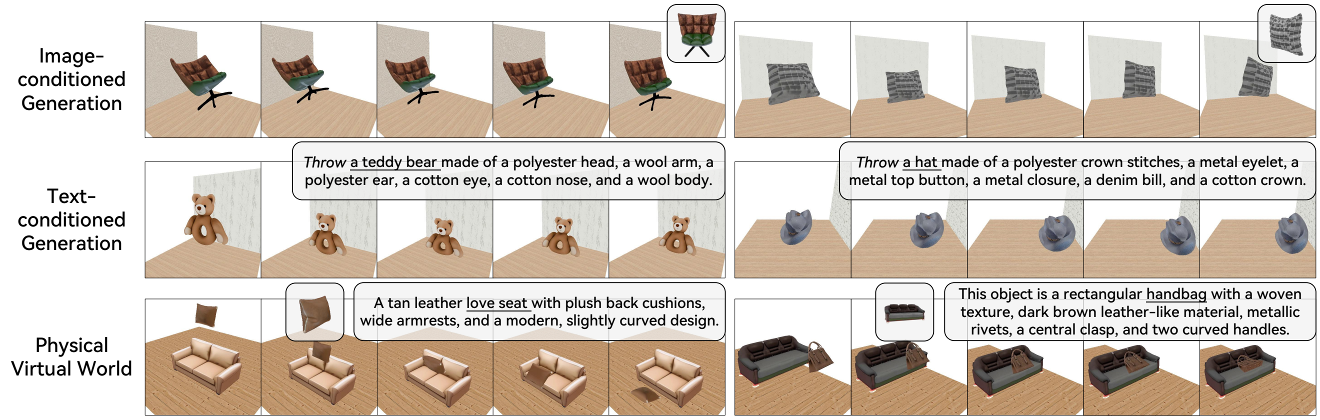

Figure 1. SOPHY is a physics-aware generative model designed for creating 3D objects that possess plausible geometries, textures, and physical properties, making them ready for simulation. (*Top*) Our method can generate these assets from a single image; (*middle*) or from a text prompt (conditions are shown as insets in the top right corner of each generated object). (*Bottom*) We further showcase SOPHY's potential for constructing a physically accurate virtual world by combining various image and text conditions to generate diverse objects.

Abstract

We present SOPHY, a generative model for 3D physics-aware shape synthesis. Unlike existing 3D generative models that focus solely on static geometry or 4D models that produce physics-agnostic animations, our method jointly synthesizes shape, texture, and material properties related to physics-grounded dynamics, making the generated objects ready for simulations and interactive, dynamic environments. To train our model, we introduce a dataset of 3D objects annotated with detailed physical material attributes, along with an efficient pipeline for material annotation. Our method enables applications such as text-driven generation of interactive, physics-aware 3D objects and single-image reconstruction of physically plausible shapes. Furthermore, our experiments show that jointly modeling shape and material properties enhances the realism and fidelity of the generated shapes, improving performance on both generative geometry and physical plausibility. Project page: https://xjay18.github.io/SOPHY.

1. Introduction

Generating high-quality 3D assets for use in interactive virtual environments or for manufacturing functional 3D objects is a fundamental challenge in generative AI and digital content creation. Unfortunately, current generative models of 3D shapes [15, 67, 97] are limited to generating only static geometry and textures. Although there is significant effort in building 4D generative models for both geometry and motion synthesis [42, 73, 85], current methods are limited to fixed animations and are incapable of producing motions resulting from complex, dynamic object interactions.

In this work, we introduce a diffusion-based generative model for physics-aware 3D object synthesis, capable of generating 3D assets with detailed shape, texture, and, most importantly, physical material attributes governing kinematic object deformations (*e.g.*, elastic deformations, plastic softening) and frictional interactions. Our approach produces physics-aware, interactive 3D objects, which are helpful for downstream applications such as physically based simulation, robotic interaction, and manufacturing. There are various challenges to developing such a generative approach. First, current 3D and 4D datasets do not contain physical material attributes for objects, making it hard to train, or even fine-tune such physics-aware generative models. Second, it is unclear how to jointly model the interplay of shape and material in 3D asset generation. Clearly, material attributes are not independent of geometric shapes, as repeatedly observed in prior works of physical material recognition from shapes [1, 40, 44, 69]. For example, a chair's backrest made of thin rods or strips is usually

[*] This work was done during an internship at TU Crete.

metal, while a thicker, block-like backrest is often made of fabric or soft padding.

Our work aims to address the above challenges. First, we introduce a semi-automatic annotation pipeline, combining Vision Language Model (VLM)'s guidance with iterative expert (mechanical engineer) feedback for more efficient object annotation with material properties. Second, we introduce a generative model that builds upon recent advancements in latent diffusion models, yet incorporates novel insights on representing 3D objects as compact latent codes capturing shape geometry, albedo color, and material properties related to elasticity and plasticity, as well as capturing their interdependence through cross-attention blocks. Additionally, we integrate a texture enhancement module as a post-processing step that refines the generated albedo color, resulting in a high-quality texture map for improved appearance. Overall, we observe that modeling shape and material properties together enhances the realism and fidelity of the geometry in the generated shapes. Our contributions are:

- We present a semi-automatic pipeline to obtain annotations describing object physical behavior, and a new dataset of 3K objects and 15K parts annotated with detailed physical material properties.
- We introduce an autoencoder for representing physics-aware 3D objects into compact latents, along with a diffusion-based model that jointly synthesizes geometry, texture, and physical materials.
- We showcase applications of our framework for synthesizing simulation-ready objects based on texts or images, as well as physics-aware virtual worlds filled with objects generated by our method.

2. Related Work

3D asset generation. In recent years, we have witnessed an explosion of 3D generative models capable of generating detailed geometry and texture. The advances have been driven by expressive neural representations, *e.g.*, NeRFs [54, 61], Gaussian splats [31], signed distance fields [14, 59], occupancy fields [53], and numerous other representations for capturing shape or/and texture [2, 11, 12, 21, 24, 25, 29, 43, 56, 60, 66, 88, 89]. We also refer readers to the recent surveys of 3D generative model [39, 82]. Unfortunately, most 3D generative models overlook the physical properties of objects, limiting their use in interactive and simulation environments. To address this gap, recent research has begun incorporating physical priors and constraints into the generative pipeline. For example, DiffuseBot [81] evaluates generated 3D robot models based on their simulation performance and refines the sampling distribution to favor more successful designs. Atlas3D [13] and Phys-Comp [23] enforce static equilibrium constraints during shape optimization, ensuring that generated objects remain stable under gravity. Despite these advances, all these prior works assume uniform or limited material properties, restricting their ability to generate diverse and physically intricate 3D assets. In contrast, our method jointly optimizes 3D shape, albedo color, and material properties with a network that captures their natural dependencies.

Material-annotated 3D datasets. The advances in 3D generative models have also been led by the development of influential 3D datasets such as ShapeNet [8] and Objaverse [17]. More recent datasets, including ABO [16], Matsynth [78], and BlenderVault [46], incorporate surface texture information to train models for 3D generation and more plausible, photorealistic rendering. Despite this progress, a critical gap remains in the availability of detailed physical material information essential for accurately modeling material properties for physics-based simulations. While datasets such as ShapeNet-Mat [44], 3DCoMPaT [40] and its extensions [1, 69] attempt to address this limitation by providing part-level material labels, their annotations are coarse and unsuitable for direct use in physical simulators. In a concurrent work, PhysX-3D [7] released a dataset of 3D objects with part-level physical annotations. While it includes detailed material parameters, the vast majority of parts exhibit high stiffness values, consistent with rigid-body modeling rather than deformable or soft-body behavior. In contrast, our dataset encompasses objects whose parts span a much broader range of deformability, including both rigid and highly deformable components.

4D content generation. Existing 4D generation approaches [36, 37, 42, 72, 83, 85] primarily rely on data-driven pipelines using image or video diffusion models, such as DreamGaussian4D [64], which utilizes Stable Video Diffusion [5] for animating 3D Gaussian splats (3DGS) [31] from a single image, and STAG4D [92], which generates multi-view images via text-to-video modules. However, these methods rely on pre-trained generative models that lack physical understanding, often resulting in physically unrealistic motions [3]. Another line of research [20, 47, 71] focuses on part-level motion based on articulation, yet can accommodate only rigid motion of parts. To address more general physics-aware dynamics, [87, 95] integrate the Material Point Method (MPM) [28] to animate 3DGS. Building on these efforts, several recent studies [10, 45, 50, 74] have focused on physical dynamics generation from a single image. They typically follow a two-stage pipeline: generating 3D shapes first, then applying homogeneous physical attributes via manual selection or VLM estimation. This separation can lead to inaccuracies because it fails to account for the material heterogeneity and the tightly coupled nature of geometry observed in the real world. In contrast, our approach uses a learning-based framework to simultaneously model geometries, colors, and materials, enhancing both physical realism and geometric coherence in the generated 3D objects.

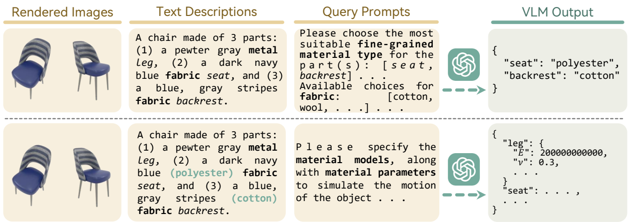

Figure 2. **VLM-guided material annotation of shape parts.**

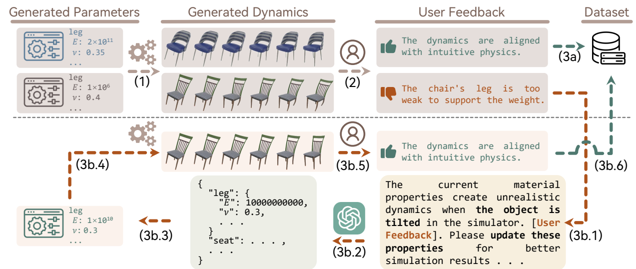

Figure 3. **Expert verification of material annotations.**

3. Material Annotation

We first describe the procedure for creating a dataset of 3D objects annotated with detailed physical material properties. Specifically, we annotate shapes with the material parameters used in the Material Point Method (MPM) [26, 28, 70], a popular simulator known for its effectiveness in handling complex simulations of behaviors and inertia effects inherent in solids [6, 38, 87], including but not limited to elastic and plastic deformation of solids and frictional interactions. The material parameters include: (a) Young's modulus, (b) Poisson's ratio, (c) yield stress, (d) friction angle, (e) density, and (f) material behavior type (also known as material model), including pure elastic deformation [84], plastic deformation with softening (damage) [79], plastic deformation without softening [80], and granular deformation [33].

Unfortunately, no datasets of 3D objects exist with such detailed material properties. The most related dataset is 3DCoMPaT [40] along with its subsequent versions [1, 69], which only provide coarse material labeling, such as "plastic," "fabric," and "metal". While these labels offer some insight, they are too broad for precise simulations in physics engines. For example, rigid PVC is much stiffer than flexible LDPE, despite both being classified as "plastic". Expert knowledge is typically required to accurately determine material properties, making it difficult for annotators without a strong physics background and experience with commonly used material types in objects to provide precise labels. Hiring experts to label detailed properties for each component of every 3D object in a large database would be highly cumbersome. Thus, we devised a semi-automatic pipeline that combines material property annotation by VLMs, followed by iterative expert feedback and verification.

3.1. VLM-guided Material Property Proposals

VLMs have recently become prominent in physical reasoning [9, 34, 45, 48, 96, 98] due to their extensive knowledge bases built on multi-modal data. Thus, we leveraged their zero-shot reasoning ability to give an initial estimation of material properties for input 3D shapes. We started by choosing 12 shape categories from 3DCoMPaT200 [1], which contained a variety of material compositions and behaviors capable of elastic or plastic deformation, such as bags, pillows, and chairs. We skipped categories where objects behave rigidly (*e.g.*, cabinets, faucets). As shown in Figure 2(top), we provided a popular VLM (ChatGPT-4o [57]) with (a) two automatically generated, rendered views of each textured object in the selected categories, (b) a textual description of the shape, including its object category, part tag, coarse material label, and color, as provided by 3DCoMPaT200, (c) a list of available fine-grained material categories *e.g.*, for "plastic", we include sub-types such as polypropylene, rigid or flexible PVC, and so on. We provide this fine-grained list in Section 10.1. Our text prompt asks the VLM to identify the most plausible fine-grained material category per part. Then for each part, in a second round, we further prompted the VLM to provide the most likely material models and parameters (*e.g.*, Young's modulus, Poisson's ratio, yield stress, and friction angle) based on a similar textual description, including this time the fine-grained material category as shown in Figure 2(bottom). This two-round prompt provided better material property prediction based on our expert verification.

3.2. Expert Verification and Feedback

Although VLMs often offer reasonable initial estimates of material properties, they are not always reliable. To address this issue, we develop a pipeline that generates simulation videos of objects based on the material properties provided by the VLM, then ask experts with mechanical engineering backgrounds to assess their physical plausibility.

Test scenarios. We created five test scenarios to simulate object dynamics: (1) *Drop*ping an object from a certain height; (2) *Throw*ing an object in a certain direction; (3) *Tilt*ing an object; (4) *Drag*ging an object; (5) Applying a short-term, time-variant (*e.g.*, *wind*) force. We obtain object motions under these scenarios using a particle-based simulator [99] that we extended to model heterogeneous materials per particle, since our dataset often contains objects whose components are made of different materials. Then, we render object sequences with a high-performance renderer [31, 35] and obtain the simulation videos. After that, we enlisted five mechanical engineering graduates to evaluate the physical plausibility of each video and instructed them to reach a consensus on their judgment. As shown in Figure 3(2)-(3a), for simulated objects deemed as realistic by this group, we store their material properties in our dataset. For objects deemed to have unrealistic motion, we

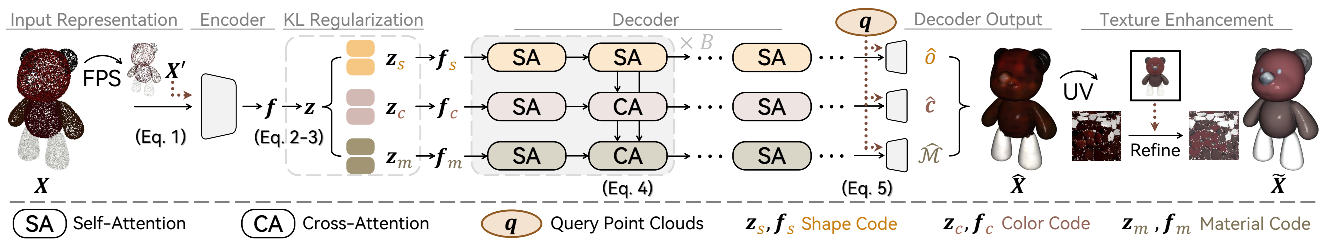

Figure 4. **Our pipeline for compressing objects to compact shape, texture, and material codes.**

ask the group to provide feedback to the VLM, specifying which part of the object seems to move in an implausible manner and why. This allows us to re-query VLMs for material parameters, as demonstrated in Figure 3(3b.1)-(3b.5), and generate new simulation videos automatically based on the updated parameters. This process is iterative until satisfactory simulation results are achieved.

3.3. Dataset Summary

We obtained $3,004$ verified 3D models in 12 object categories originating from the 3DCoMPaT200 [1] dataset. Every shape part is labeled with physical material properties and detailed material categories, resulting in a total of $15,575$ labeled parts. We split our dataset into training, validation, and test sets, following 3DCoMPaT200– the total number of samples in each split is 2462, 180, and 362, respectively. More data statistics are shown in Section 9.

4. Generative Model

Our proposed generative model, SOPHY, is based on Stable Diffusion [65], which generates data by denoising a compressed feature space (latent space). To extract the latent space, we process shapes alongside color and material parameters through a variational autoencoder (Section 4.1). A latent diffusion model is then trained to jointly capture the distribution of geometries, colors, and materials (Section 4.2), recognizing their interdependencies—certain shapes are closely linked to specific materials and colors (*e.g.*, thin cantilever-shaped chair bases typically use metal and feature a metallic gray finish). Modeling these properties independently of each other would yield unlikely or impossible materials for sampled geometries. After training our diffusion model, it can be sampled to generate new latent codes, which are then decoded to obtain shapes, compatible colors, and physical material properties.

4.1. Shape, Texture and Material Autoencoder

Our autoencoder (Figure 4) compresses input objects into compact latent codes that encode shape, color (albedo texture), and material properties. We discuss it below in detail.

Input representation. We represent an object as a dense surface point cloud $\boldsymbol{P} = \{\boldsymbol{p}_j\}_{j=1}^N$, where $\boldsymbol{p}_j$ is a 3D point position ($N = 2048$ in our implementation). The input to our autoencoder is a color- and material-augmented point cloud $\boldsymbol{X} = \{\boldsymbol{x}_j\}_{j=1}^N$, where each entry $\boldsymbol{x}_j$ contains the following per-point information: (a) 3D position $\boldsymbol{p}_j$, (b) RGB color $\boldsymbol{c}_j$, (c) a material property vector $\boldsymbol{m}_j$ with the following 9 entries: (i) logarithm of Young's modulus E_j (a scalar), (ii) Poisson's ratio ν_j (a scalar), (iii) logarithm of yield stress σ_j (a scalar), (iv) friction angle ϕ_j (a scalar), (v) density ρ_j (a scalar), and (vi) a 4-dimensional learnable embedding $\boldsymbol{\mu}_j$ of material behavior type. We represent material information by point rather than by part to account for cases where a semantic part has varied materials.

Encoder. To aggregate the per-point information from the augmented point cloud, we design a set-to-set network inspired by 3DShape2VecSet [94] as our encoder. We first sub-sample a smaller point cloud $\boldsymbol{P}'$ with $M = 512$ fewer points from the original point cloud through furthest point sampling (FPS) [62], and augment it with color and material information to obtain $\boldsymbol{X}' = \{\boldsymbol{x}'_i\}_{i=1}^M$. Then, we use cross attention to produce the object's latent representation:

$$\{\boldsymbol{f}_i\}_{i=1}^M = \mathrm{CrossAttn}\left(g(\{\boldsymbol{x}'_i\}_{i=1}^M), g(\{\boldsymbol{x}_j\}_{j=1}^N)\right), \quad (1)$$

where g is a linear layer projecting the augmented point cloud into the embedding space $\mathbb{R}^C$ ($C = 512$ in our implementation) and $\{\boldsymbol{f}_i\}_{i=1}^M$ is a set of latent codes.

KL regularization. The above representation is quite high-dimensional. We thus seek to compress it towards a more compact latent code. Following Stable Diffusion [65] and 3DShape2VecSet [94], we adopt a variational autoencoder (VAE) regularized with the KL-divergence to achieve this effect. We first use two fully connected (FC) layers to project each latent code $\boldsymbol{f}_i$ to mean and variance:

$$\boldsymbol{f}_i^\mu = \mathrm{FC}_\mu(\boldsymbol{f}_i), \ \boldsymbol{f}_i^\sigma = \mathrm{FC}_\sigma(\boldsymbol{f}_i), \quad (2)$$

where $\boldsymbol{f}_i^\mu, \boldsymbol{f}_i^\sigma \in \mathbb{R}^{C_0}$ and $C_0 \ll C$ ($C_0 = 24$ in our implementation). Then, we use the reparameterization sampling and obtain a smaller latent code $\boldsymbol{z}_i \in \mathbb{R}^{C_0}$:

$$\boldsymbol{z}_i = \boldsymbol{f}_i^\mu + \epsilon \cdot \boldsymbol{f}_i^\sigma, \ \epsilon \sim \mathcal{N}(0, 1), \quad (3)$$

which enables us to train diffusion models on a lower-dimensional space later. Finally, we project $\boldsymbol{z}_i$ back into the original embedding space $\mathbb{R}^C$ with another FC layer.

Decoder. One possible decoder design follows the approach of 3DShape2VecSet [94], where latent codes are first processed through a series of self-attention blocks, then for a query point $\boldsymbol{q} \in \mathbb{R}^3$, its occupancy is determined by interpolating the latent codes based on the query position and transforming the result through a fully connected layer. This design could be naturally extended to also decode color and material information per query point. However, we observed worse performance using this approach, as color and material information are not meaningful for non-occupied query points (*i.e.*, points outside the shape volume).

Empirically, we achieved better results with an alternative design that decodes geometry into occupancy values first, followed by color and material properties for occupied query points. Performance was further improved when we explicitly split the latent codes $\boldsymbol{z}_i$ into three sub-codes: the shape code $\boldsymbol{z}_{i,s}$ for occupancy decoding, the color code $\boldsymbol{z}_{i,c}$ for texture decoding, and the material code $\boldsymbol{z}_{i,m}$ for material property decoding. Each has a shape of $M \times \frac{C_0}{3}$. To decode these components, we introduce three dedicated branches, each responsible for processing one of the sub-codes. We also apply cross-attention layers to the color and material branches, enabling them to attend to information from previously decoded latents. This design models the natural dependency of color on geometry and material properties on both geometry and color, which is inspired by the workflow commonly used in 3D asset creation, where shape is defined first, followed by texture and physical attributes. The cross-attention in our decoder is formulated as follows:

$$\begin{aligned}\{\boldsymbol{f}_{i,s}^{(l)}\} = \mathrm{SA}\big(\{\boldsymbol{f}_{i,s}^{(l-1)}\}\big),\ \{\boldsymbol{f}_{i,c}^{(l)}\} = \mathrm{CA}\big(\{\boldsymbol{f}_{i,c}^{(l-1)}\}, \{\boldsymbol{f}_{i,s}^{(l-1)}\}\big),\\ \{\boldsymbol{f}_{i,m}^{(l)}\} = \mathrm{CA}\big(\{\boldsymbol{f}_{i,m}^{(l-1)}\}, \{[\boldsymbol{f}_{i,s}^{(l-1)}, \boldsymbol{f}_{i,c}^{(l-1)}]\}\big),\end{aligned} \tag{4}$$

where SA, CA denotes self- and cross-attention, respectively. l is the layer index, and $[\cdot,\cdot]$ is concatenation.

Decoder output. Given a query point $\boldsymbol{q} \in \mathbb{R}^3$, the occupancy values are decoded by interpolating the shape subcodes from the last layer (layer L) and processing them through an FC block:

$$O(\boldsymbol{q}){=}\mathrm{FC}\Big(\sum_{i}^{M}\mathrm{Softmax}\big(\frac{q(\boldsymbol{f}_{\boldsymbol{q}})\cdot k(\boldsymbol{f}_{i,s}^{(L)})}{\sqrt{C}}\big)v(\boldsymbol{f}_{i,s}^{(L)})\Big), \tag{5}$$

where $\boldsymbol{f}_{\boldsymbol{q}}$ is a feature vector obtained by processing the query point through our encoder described in Equation (1). The functions $q(\cdot), k(\cdot), v(\cdot)$ are the query-key-value linear transformations used in attention [77]. By disentangling the latent codes, we ensure that occupancy is determined solely based on the shape sub-codes relevant to this task.

Color and material properties are decoded similarly, each using its own FC block and interpolation over the corresponding color and material sub-codes. For color prediction, normalization is applied post-FC block to ensure output ranges from $[0, 1]$. Young's modulus and yield stress are predicted on a logarithmic scale since they are always positive and can often be quite large. The friction angle, ranging from $[0, \pi/2]$, uses a sigmoid activation followed by scaling. The categorical material behavior type is predicted with a softmax activation. Material properties are stored only for query points identified as occupied, and color values are predicted only for points on the mesh surface from marching cubes [51].

Autoencoder training. While training our variational autoencoder, we jointly optimize a combination of loss functions involving 3D occupancies, colors, and material properties. Specifically, we minimize a weighted sum of the following losses: (a) occupancy loss $\mathcal{L}_o$ (binary cross-entropy), (b) color loss $\mathcal{L}_c$ (ℓ_1 norm), (c) Young's modulus loss $\mathcal{L}_E$ (ℓ_1 norm applied to logarithmic values), (d) Poisson's ratio loss $\mathcal{L}_\nu$ (ℓ_1 norm), (e) yield stress loss $\mathcal{L}_\sigma$ (ℓ_1 norm applied to logarithmic values), (f) friction angle loss $\mathcal{L}_\phi$ (ℓ_1 norm), (g) density loss $\mathcal{L}_\rho$ (ℓ_1 norm), (h) material model loss $\mathcal{L}_M$ (cross-entropy), (i) a regularization loss $\mathcal{L}_r$ imposing a KL penalty towards a standard normal distribution on the latent codes, as typically used in VAEs [32]. We note that the color and material losses are computed only for training query points on or inside the object's surface. For off-surface training points, we assign color and material properties by copying them from their nearest surface points. The combined loss function is:

$$\mathcal{L} = \mathcal{L}_o + \sum_{\omega} \lambda_\omega \mathcal{L}_\omega, \quad \omega = \{c, E, \nu, \sigma, \phi, \rho, M, r\}, \tag{6}$$

where λ_ω are weight parameters for balancing different terms. To further enhance training, we leverage the pretrained 3DShape2VecSet model [94], which was trained on ShapeNet [8] (a larger dataset containing 55K shapes) using occupancy supervision alone. Specifically, we initialize all network layers shared with [94], including the cross-attention weights on point positions in our encoder, the self-attention weights on shape sub-codes, and the occupancy decoder weights, using their pretrained values. This provides a small boost compared to training our model from scratch on our smaller 3K-shape dataset. We also note that we adopt the latent set representation [94] in our autoencoder due to its simplicity and efficiency in compressing 3D objects. Our main insight is the joint modeling of shape, texture, and physical properties, and this concept is applicable beyond this specific representation.

Texture enhancement. The decoded object $\hat{\boldsymbol{X}}$ exhibits a rather flat and blurry visual appearance due to insufficient texture resolution inherited by the vertex-based color representation (see Figure 4). Thus, we refine its texture based on renderings of the input shape (in the auto-encoding setting) or conditioned signals (in the generative setting). We first generate a UV map, unwrap the object, and then bake its per-vertex color information into a 2D texture using the

UV layout. Then, we refine the 2D texture using texture diffusion models [91, 97] to improve visual quality. This process typically takes 1 minute per object and serves as an optional post-processing step for our autoencoder.

4.2. Diffusion

Generating a simulation-ready object involves executing the reverse process of a diffusion that progressively transforms Gaussian noise in the latent space of objects into target latent codes: $\boldsymbol{Z} = \{\boldsymbol{z}_i\}_{i=1}^{M}$. Sampling is performed by solving the stochastic differential equations from the EDM diffusion pipeline [30]. Once a latent code is sampled, we pass it through our trained decoder to extract occupancy values on a dense $\mathbb{R}^3$ grid. These values are then converted into a mesh using marching cubes, then color and material properties are decoded at densely sampled mesh points.

Training. To train the diffusion model, we use the loss:

$$\mathcal{L}_{\text{edm}} = \mathbb{E}_{\hat{\boldsymbol{Z}}\sim p_{data}} \mathbb{E}_{\boldsymbol{N}\sim\mathcal{N}(\boldsymbol{0},\sigma_t^2\boldsymbol{I})} \| D(\hat{\boldsymbol{Z}} + \boldsymbol{N}, \sigma_t, \boldsymbol{c}) - \hat{\boldsymbol{Z}} \|_2^2, \tag{7}$$

where D is the denoising network, $\hat{\boldsymbol{Z}}$ are training latent codes, $\boldsymbol{N}$ are added Gaussian noises, σ_t denotes the noise level, and $\boldsymbol{c}$ is a signal for conditioning. We considered two input conditions to our denoiser: (a) an RGB image of an object, where $\boldsymbol{c}$ represents here the extracted features from a pre-trained image encoder (“DINOv2-ViT-B/14” [58]), (b) a text prompt, where $\boldsymbol{c}$ represents features from a pre-trained text encoder (“CLIP-ViT-L/14” [63]). Details on the denoiser structure and conditional signals are provided in Section 8.1 and Section 8.2, respectively. *Our code and dataset are available on our project page.*

5. Experiments

We evaluate SOPHY on its autoencoder effectiveness (Section 5.1), its generative capabilities for image-conditioned generation (Section 5.2) and text-conditioned generation (Section 5.3) of simulation-ready 3D objects. All our comparisons were performed in the same test split of our dataset, described in Section 3.3.

5.1. Auto-encoding Evaluation

In terms of auto-encoding evaluation, our goal is to check how well we are able to recover a test 3D shape, along with its color and material properties, given its input representation. This evaluation is common in 3D latent-based generative models [94], since it is imperative for the autoencoder to capture latent spaces that can generalize to novel inputs.

Competing methods. We stress that no existing generative model matches our setting of joint shape and physical material synthesis in a feedforward manner; thus, we compare with variant designs for our autoencoder:

(a) baseline, or in short “*B. Dec.*”, is a model that excludes color and material properties from the generation process *i.e.*, it generates a 3D shape, then predicts color conditioned on the shape through a decoder, and then the material through another decoder. This choice aims to evaluate the benefit of incorporating the physical materials in the generation process, *i.e.*, whether it is simply better to generate the 3D shape first, then guess its most likely texture and material discriminatively. For this baseline, we use 3DShape2VecSet as the generative model, trained on the same split as our method. We use a texture decoder, which decodes its latent shape code with a set of self-attention blocks to per-point colors. We use one more decoder to decode the latent shape code to material properties, including a cross-attention block for conditioning on colors. The number of total parameters remains comparable to ours.

(b) w/o CA is a degraded variant of our proposed SOPHY. It discards the cross-attention blocks used in the color and material decoder branch. This variant is equivalent to decoding our latent sub-codes with three non-interacting branches for occupancy, color, and material predictions.

(c) Fused is another variant of SOPHY that uses a unified representation for the latent code, without separating it into the shape, color, and material sub-codes. The decoder infers all the properties conditioned on this single latent code.

Metric	B. Dec.	w/o CA	Fused	SOPHY
M.B. Acc(%) $\uparrow$	71.04	92.77	93.23	**93.55**
MAE-log(E) $\downarrow$	1.18	0.50	0.47	**0.45**
MAE-ν($\times10^{-2}$) $\downarrow$	4.10	3.06	**2.98**	3.06
MAE-log(σ) $\downarrow$	1.07	0.41	0.32	**0.29**
MAE-ϕ ($\times10^{-2}$) $\downarrow$	5.45	1.89	**1.22**	1.28
MAE-ρ $\downarrow$	0.16	0.14	0.13	**0.09**
Sim-CD ($\times10^{-3}$) $\downarrow$	53.84	17.47	10.05	**8.72**
MAE-c($\times10^{-2}$) $\downarrow$	8.75	8.47	8.35	**8.13**
IoU(%) $\uparrow$	90.69	90.75	90.66	**90.89**
CD($\times10^{-4}$) $\downarrow$	3.51	3.34	3.30	**3.02**
F-Score(%) $\uparrow$	93.65	93.77	93.79	**93.85**

Table 1. **Quantitative comparison in the auto-encoding setting.** The upper, middle, and lower sections display the average metrics for predictions of material, color, and shape, respectively.

Metrics. We report the classification accuracy of the material behavior type (M.B. Acc.) and the mean absolute error (MAE) of color and all our material parameters. In addition, we report Chamfer distance (Sim-CD) between densely sampled points of the predicted shapes and ground-truth ones under the deformed states computed by our simulator along the whole simulation trajectory for the dropping test scenario (Section 3.2). Finally, we consider purely geometric measures for the rest state of the reconstructed test shape compared to the ground truth. We use the standard metrics of Chamfer Distance (CD), volumetric Intersection-over-Union (IoU), and F-score, as used in prior arts [94]. We average all values for each object in our test set.

Results. As shown in Table 1, compared to the baseline model, jointly modeling shape, color, and material in a shared embedding space largely improves material metrics,

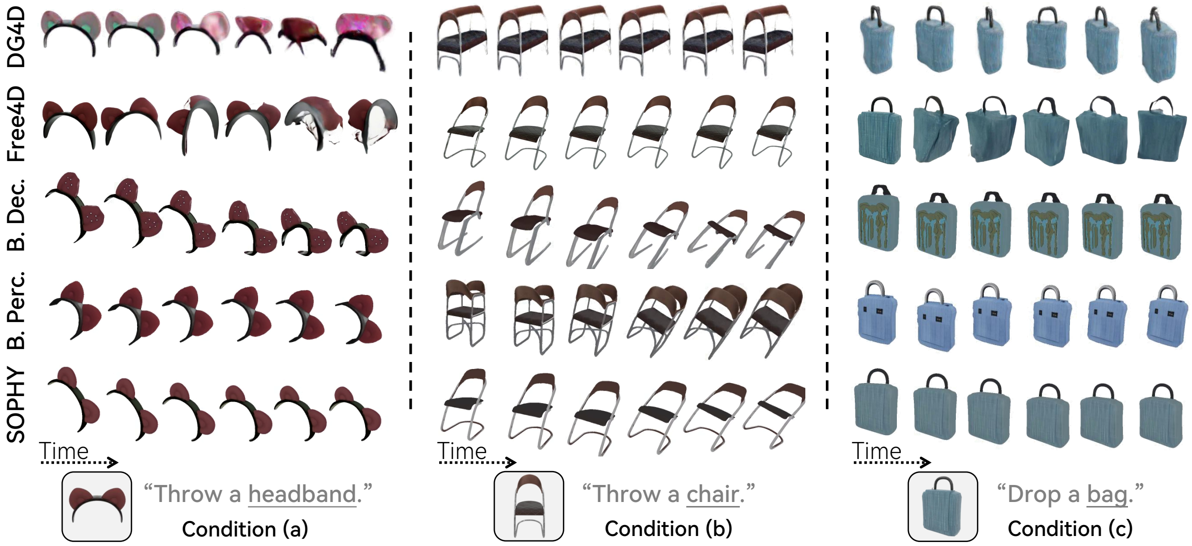

Figure 5. **Qualitative comparison in the image-to-4D setting.** The gray texts next to the conditioned images describe the desired dynamics. Note that the objects fall onto a ground plane, which is not shown here for clarity.

with modest gains in color and geometry evaluation metrics. Notably, the material type accuracy is improved by more than 20%, while a factor of $5\times$ reduces Sim-CD, which supports our hypothesis that geometry, color, and material attributes are strongly correlated and should be jointly modeled in the generative process. Compared to the "w/o CA", our full model has $2\times$ lower Chamfer distance during simulation, and still slightly better performance in all other metrics. Finally, we observe that the "Fused" variant has a tiny edge over our model in Poisson's ratio and friction angle prediction. Yet, it performs significantly worse in terms of Sim-CD, and worse in all other measures. We suspect it is due to an uneven network capacity allocation across occupancy, color, and material within a single fused latent code.

5.2. Image-to-4D

We now discuss applying SOPHY to generate 4D dynamics given a single input RGB image. Specifically, given a 3D object with physics material properties generated by our method conditioned on the image, we plug it into a virtual 3D environment with other objects or primitives, *e.g.*, ground planes, walls, and so on, and animate it based on its material properties and interactions with the environment. To evaluate in this setting, we render a 2D image from each object of our test split, provide it to our trained diffusion model for generation, then simulate the generated object using the test scenarios of Section 3.2 to create 4D scenes.

Competing methods. We first compare SOPHY with conventional image-to-4D methods, which generate deforming 3DGS over time without any explicit physics-based representation. We examine DreamGaussian4D (DG4D) [64], a pioneering work in this field, along with Free4D [49], which is one of the state-of-the-art methods that distills pre-trained foundation models for consistent 4D generation. We also compare with our "baseline" discussed in the previous section. In addition, we implement another physics-aware baseline inspired by recent work [10, 45], which adopts a two-stage framework for our task (see Section 2), and we call it "*B. Perc*". Specifically, this alternative baseline uses the perceptual models [52, 57] for material property estimation given an off-the-shelf 3D generation model [86]. After obtaining the material properties, we use the same simulator to create dynamics as SOPHY does. More details about this baseline can be found in Section 8.4.

Metric	DG4D	Free4D	B. Dec.	B. Perc.	SOPHY
VideoPhy ↑	-0.28	-0.21	-0.15	0.21	**0.43**
CLIP ↑	0.71	0.74	0.73	0.74	**0.77**
Time (min) ↓	10	73	**5**	7	**5**

Table 2. **Image-to-4D.** "B. Dec." and "B. Perc." denote the decoder-based and perception-based baselines, respectively.

Metrics. Given rendered images of the 4D scenes generated by any of the above compared methods, we leverage VideoPhy [3], a state-of-the-art VLM trained based on human annotations, to evaluate whether the generated videos align with real-world physics. The method produces a per-scene score that is uncalibrated for different scenarios. Thus, following [74], we apply z-score normalization to calibrate the scores across all methods for each scene and report the average values. A positive z-score indicates a method performs better than average, while negative means worse. Additionally, we use the CLIP [63] score to evaluate the consistency between the generated results and the input conditions. We also report the average time to create a video for an object starting from an image.

Results. In Table 2, SOPHY outperforms other methods, demonstrating the highest z-score for alignment with real-world physics based on VideoPhy. For CLIP scores, our method is better aligned with the conditioned sig-

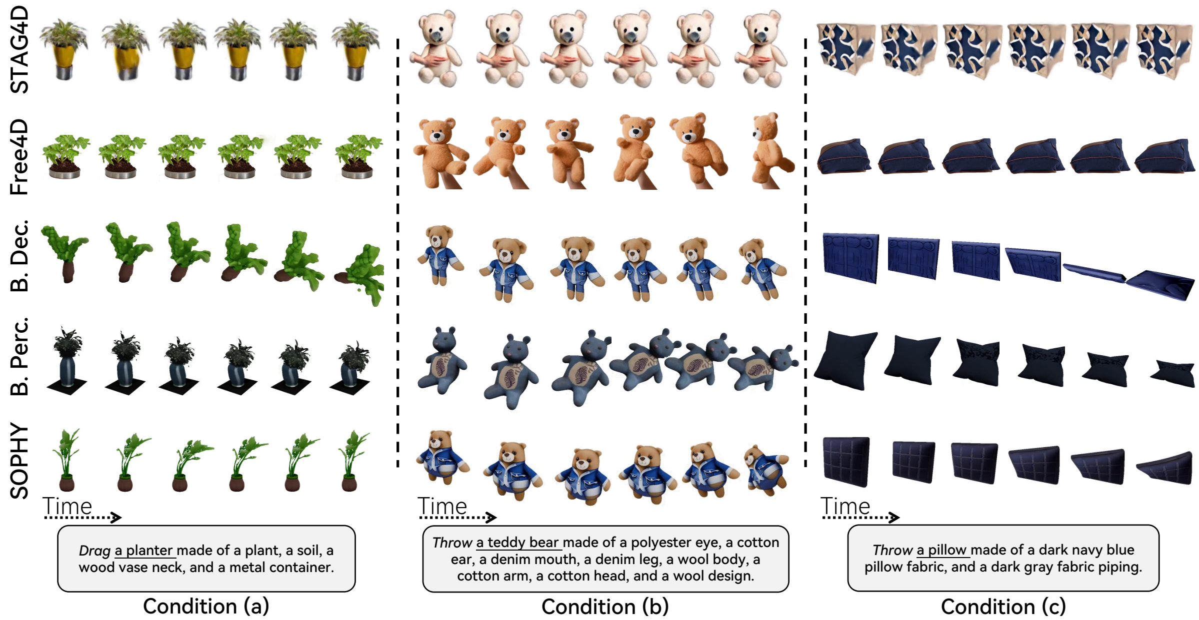

Figure 6. **Qualitative comparison in the text-to-4D setting.** The text prompts used as input conditions are shown below. Note that the ground plane is not shown here as well.

Metric	STAG4D	Free4D	B. Dec.	B. Perc.	SOPHY
VideoPhy ↑	-1.23	0.07	-0.18	0.58	**0.76**
CLIP ↑	0.13	0.17	0.15	**0.18**	**0.18**
Time (min) ↓	66	73	**5**	7	**5**

Table 3. **Text-to-4D.** "B. Dec." and "B. Perc." denote the decoder-based and perception-based baselines, respectively.

nals. Qualitative results in Figure 5 reveal that DG4D and Free4D struggle with generating plausible animations, such as fluctuating headband sizes, due to neglecting physical constraints. Both baselines, "B. Dec." and "B. Perc.", generate geometries and physical materials independently. Thus, they often create physically implausible objects (*e.g.*, strange chairs in Figure 5(b)) or predict inappropriate materials for shapes (*e.g.*, too stiff materials for a bag that looked like it was made of fabric in Figure 5(c)). In contrast, SOPHY produces more realistic dynamics, aligning closely with the given images.

5.3. Text-to-4D

In this subsection, we evaluate SOPHY for text-to-4D. The evaluation is the same as before, with the only difference being the input condition. We test on prompts that describe the object category, part tags, and fine-grained material categories for our test objects, following the prompts of Section 3.1. We also include the test scenario in the prompt.

Competing works & Metrics. We compare our method with STAG4D [92], which generates deforming 3DGS driven by text prompts. We also consider Free4D [49] and the two baselines in the previous section here. The evaluation metrics are the same as in the image-to-4D setting.

Results. In Table 3, SOPHY achieves the highest performance in VideoPhy and CLIP scores, demonstrating better alignment with real-world physics and given text prompts. Visual results in Figure 6 show that STAG4D fails to produce noticeable deformations, even for soft objects like teddy bears. Although Free4D can occasionally generate dynamic content as shown in Figure 6(b), the content mismatches the conditions. Since the two baselines exclude material properties from the generation process, they fail to produce appropriate physical attributes aligned with the text prompts. For instance, in Figure 6(a), "B. Dec." generates a very *stiff* plant while "B. Perc." creates a *soft* metal container. Compared to them, our method produces more geometrically coherent and physically plausible results.

6. Conclusion

We presented a new generative model of 3D objects incorporating geometry, color, and physical material properties. Our experiments show significant benefits of the approach, including generating physically plausible 4D videos from images and texts. Future work could expand our model to synthesize entire scenes and other phenomena, such as fluids [41, 55, 76] or gases [19]. Additionally, using more accurate simulators, like finite elements [4, 68, 75], could enhance our model since it generates the complete volume of objects. Our annotated dataset is only a first step towards more physics-aware 3D datasets – enlarging it with more objects, articulation structures [22, 90], and additional material properties would be a fruitful direction.

Acknowledgments. This work has received funding from the European Research Council (ERC) under the Horizon research and innovation programme (No. 101124742).

References

[1] Mahmoud Ahmed, Xiang Li, Arpit Prajapati, and Mohamed Elhoseiny. 3DCoMPaT200: Language-grounded compositional understanding of parts and materials of 3D shapes. In *NeurIPS*, 2024. 1, 2, 3, 4

[2] Maciej Bala, Yin Cui, Yifan Ding, Yunhao Ge, Zekun Hao, Jon Hasselgren, Jacob Huffman, Jingyi Jin, JP Lewis, Zhaoshuo Li, et al. Edify 3D: Scalable high-quality 3D asset generation. *arXiv:2411.07135*, 2024. 2

[3] Hritik Bansal, Zongyu Lin, Tianyi Xie, Zeshun Zong, Michal Yarom, Yonatan Bitton, Chenfanfu Jiang, Yizhou Sun, Kai-Wei Chang, and Aditya Grover. VideoPhy: Evaluating physical commonsense for video generation. *arXiv:2406.03520*, 2024. 2, 7

[4] Jernej Barbič, Fun Shing Sin, and Daniel Schroeder. Vega FEM Library, 2012. http://www.jernejbarbic.com/vega. 8

[5] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv:2311.15127*, 2023. 2

[6] Junyi Cao, Shanyan Guan, Yanhao Ge, Wei Li, Xiaokang Yang, and Chao Ma. NeuMA: Neural material adaptor for visual grounding of intrinsic dynamics. In *NeurIPS*, 2024. 3

[7] Ziang Cao, Zhaoxi Chen, Liang Pan, and Ziwei Liu. Physx-3D: Physical-grounded 3D asset generation. In *NeurIPS*, 2025. 2

[8] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. ShapeNet: An information-rich 3D model repository. *arXiv:1512.03012*, 2015. 2, 5

[9] Annie S Chen, Alec M Lessing, Andy Tang, Govind Chada, Laura Smith, Sergey Levine, and Chelsea Finn. Commonsense reasoning for legged robot adaptation with vision-language models. *arXiv:2407.02666*, 2024. 3

[10] 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:2503.20746*, 2025. 2, 7

[11] Yiwen Chen, Tong He, Di Huang, Weicai Ye, Sijin Chen, Jiaxiang Tang, Xin Chen, Zhongang Cai, Lei Yang, Gang Yu, et al. MeshAnything: Artist-created mesh generation with autoregressive transformers. *arXiv:2406.10163*, 2024. 2

[12] Yiwen Chen, Yikai Wang, Yihao Luo, Zhengyi Wang, Zilong Chen, Jun Zhu, Chi Zhang, and Guosheng Lin. Meshanything V2: Artist-created mesh generation with adjacent mesh tokenization. *arXiv:2408.02555*, 2024. 2

[13] Yunuo Chen, Tianyi Xie, Zeshun Zong, Xuan Li, Feng Gao, Yin Yang, Ying Nian Wu, and Chenfanfu Jiang. Atlas3D: Physically constrained self-supporting text-to-3D for simulation and fabrication. In *NeurIPS*, 2024. 2

[14] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *CVPR*, 2019. 2

[15] Zhaoxi Chen, Jiaxiang Tang, Yuhao Dong, Ziang Cao, Fangzhou Hong, Yushi Lan, Tengfei Wang, Haozhe Xie, Tong Wu, Shunsuke Saito, et al. 3DTopia-XL: Scaling high-quality 3D asset generation via primitive diffusion. *arXiv:2409.12957*, 2024. 1

[16] Jasmine Collins, Shubham Goel, Kenan Deng, Achleshwar Luthra, Leon Xu, Erhan Gundogdu, Xi Zhang, Tomas F Yago Vicente, Thomas Dideriksen, Himanshu Arora, et al. ABO: Dataset and benchmarks for real-world 3D object understanding. In *CVPR*, 2022. 2

[17] 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. In *CVPR*, 2023. 2, 1

[18] Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Hickman, Krista Reymann, Thomas B McHugh, and Vincent Vanhoucke. Google Scanned Objects: A high-quality dataset of 3D scanned household items. In *ICRA*, 2022. 1, 3

[19] Bryan E Feldman, James F O'brien, and Bryan M Klingner. Animating gases with hybrid meshes. *ACM TOG*, 24(3): 904–909, 2005. 8

[20] Mingju Gao, Yike Pan, Huan-ang Gao, Zongzheng Zhang, Wenyi Li, Hao Dong, Hao Tang, Li Yi, and Hao Zhao. PartRM: Modeling part-level dynamics with large cross-state reconstruction model. In *CVPR*, 2025. 2

[21] Yiangos Georgiou, Marios Loizou, Melinos Averkiou, and Evangelos Kalogerakis. Im2SurfTex: Surface texture generation via neural backprojection of multi-view images. *arXiv:2502.14006*, 2025. 2

[22] Pradyumn Goyal, Dmitry Petrov, Sheldon Andrews, Yizhak Ben-Shabat, Hsueh-Ti Derek Liu, and Evangelos Kalogerakis. Geopard: Geometric pretraining for articulation prediction in 3D shapes. In *ICCV*, 2025. 8

[23] 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. In *NeurIPS*, 2024. 2

[24] Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. LRM: Large reconstruction model for single image to 3D. In *ICLR*, 2024. 2

[25] Jingyu Hu, Ka-Hei Hui, Zhengzhe Liu, Ruihui Li, and Chi-Wing Fu. Neural wavelet-domain diffusion for 3D shape generation, inversion, and manipulation. *ACM TOG*, 43(2): 1–18, 2024. 2

[26] 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 TOG*, 2018. 3

[27] Jingwei Huang, Yichao Zhou, and Leonidas Guibas. ManifoldPlus: A robust and scalable watertight manifold surface generation method for triangle soups. *arXiv:2005.11621*, 2020. 1

[28] Chenfanfu Jiang, Craig Schroeder, Joseph Teran, Alexey Stomakhin, and Andrew Selle. The material point method for simulating continuum materials. In *SIGGRAPH 2016 courses*. ACM, 2016. 2, 3

[29] Heewoo Jun and Alex Nichol. Shap-E: Generating conditional 3D implicit functions. *arXiv:2305.02463*, 2023. 2

[30] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *NeurIPS*, 2022. 6, 4

[31] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkuehler, and George Drettakis. 3D gaussian splatting for real-time radiance field rendering. *ACM TOG*, 42(4):1–14, 2023. 2, 3

[32] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014. 5

[33] Gergely Klár, Theodore Gast, Andre Pradhana, Chuyuan Fu, Craig Schroeder, Chenfanfu Jiang, and Joseph Teran. Drucker-prager elastoplasticity for sand animation. *ACM TOG*, 35(4):1–12, 2016. 3

[34] Wenqiang Lai, Tianwei Zhang, Tin Lun Lam, and Yuan Gao. Vision-language model-based physical reasoning for robot liquid perception. In *IROS*, 2024. 3

[35] Samuli Laine, Janne Hellsten, Tero Karras, Yeongho Seol, Jaakko Lehtinen, and Timo Aila. Modular primitives for high-performance differentiable rendering. *ACM TOG*, 39 (6), 2020. 3

[36] Yao-Chih Lee, Yi-Ting Chen, Andrew Wang, Ting-Hsuan Liao, Brandon Y Feng, and Jia-Bin Huang. Vivid-Dream: Generating 3D scene with ambient dynamics. *arXiv:2405.20334*, 2024. 2

[37] Renjie Li, Panwang Pan, Bangbang Yang, Dejia Xu, Shijie Zhou, Xuanyang Zhang, Zeming Li, Achuta Kadambi, Zhangyang Wang, Zhengzhong Tu, et al. 4K4DGen: Panoramic 4D generation at 4K resolution. *arXiv:2406.13527*, 2024. 2

[38] 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 *ICLR*, 2022. 3

[39] Xiaoyu Li, Qi Zhang, Di Kang, Weihao Cheng, Yiming Gao, Jingbo Zhang, Zhihao Liang, Jing Liao, Yan-Pei Cao, and Ying Shan. Advances in 3D generation: A survey. *arXiv:2401.17807*, 2024. 2

[40] Yuchen Li, Ujjwal Upadhyay, Habib Slim, Ahmed Abdelreheem, Arpit Prajapati, Suhail Pothigara, Peter Wonka, and Mohamed Elhoseiny. 3DCoMPaT: Composition of materials on parts of 3D things. In *ECCV*, 2022. 1, 2, 3

[41] Yifei Li, Yuchen Sun, Pingchuan Ma, Eftychios Sifakis, Tao Du, Bo Zhu, and Wojciech Matusik. NeuralFluid: Neural fluidic system design and control with differentiable simulation. In *NeurIPS*, 2024. 8

[42] Hanwen Liang, Yuyang Yin, Dejia Xu, Zhangyang Wang, Konstantinos N Plataniotis, Yao Zhao, Yunchao Wei, et al. Diffusion4D: Fast spatial-temporal consistent 4D generation via video diffusion models. In *NeurIPS*, 2024. 1, 2

[43] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3D: High-resolution text-to-3D content creation. In *CVPR*, 2023. 2

[44] Hubert Lin, Melinos Averkiou, Evangelos Kalogerakis, Balazs Kovacs, Siddhant Ranade, Vladimir Kim, Siddhartha Chaudhuri, and Kavita Bala. Learning material-aware local descriptors for 3D shapes. In *3DV*, 2018. 1, 2

[45] Jiajing Lin, Zhenzhong Wang, Shu Jiang, Yongjie Hou, and Min Jiang. Phys4DGen: A physics-driven framework for controllable and efficient 4D content generation from a single image. *arXiv:2411.16800*, 2024. 2, 3, 7

[46] Yehonathan Litman, Or Patashnik, Kangle Deng, Aviral Agrawal, Rushikesh Zawar, Fernando De la Torre, and Shubham Tulsiani. Materialfusion: Enhancing inverse rendering with material diffusion priors. In *3DV*, 2025. 2

[47] Isabella Liu, Zhan Xu, Wang Yifan, Hao Tan, Zexiang Xu, Xiaolong Wang, Hao Su, and Zifan Shi. RigAnything: Template-free autoregressive rigging for diverse 3D assets. *ACM TOG*, 44(4), 2025. 2

[48] Shaowei Liu, Zhongzheng Ren, Saurabh Gupta, and Shenlong Wang. PhysGen: Rigid-body physics-grounded image-to-video generation. In *ECCV*, 2024. 3

[49] Tianqi Liu, Zihao Huang, Zhaoxi Chen, Guangcong Wang, Shoukang Hu, Liao Shen, Huiqiang Sun, Zhiguo Cao, Wei Li, and Ziwei Liu. Free4D: Tuning-free 4D scene generation with spatial-temporal consistency. In *ICCV*, 2025. 7, 8

[50] 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 *CVPR*, 2025. 2

[51] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3D surface construction algorithm. *ACM SIGGRAPH Computer Graphics*, 21(4):163–169, 1987. 5

[52] Ziqi Ma, Yisong Yue, and Georgia Gkioxari. Find any part in 3D. *arXiv:2411.13550*, 2024. 7, 2

[53] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3D reconstruction in function space. In *CVPR*, 2019. 2

[54] 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 *ECCV*, 2020. 2, 1

[55] Matthias Müller, David Charypar, and Markus Gross. Particle-based fluid simulation for interactive applications. In *ACM SIGGRAPH/Eurographics symposium on Computer Animation*, 2003. 8

[56] Michael Oechsle, Lars Mescheder, Michael Niemeyer, Thilo Strauss, and Andreas Geiger. Texture fields: Learning texture representations in function space. In *ICCV*, 2019. 2

[57] OpenAI. GPT-4o system card. *arXiv:2410.21276*, 2024. 3, 7, 1, 2

[58] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. 6

[59] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *CVPR*, 2019. 2

[60] Dmitry Petrov, Pradyumn Goyal, Vikas Thamizharasan, Vladimir Kim, Matheus Gadelha, Melinos Averkiou, Siddhartha Chaudhuri, and Evangelos Kalogerakis. GEM3D: Generative medial abstractions for 3D shape synthesis. In *SIGGRAPH*, 2024. 2

[61] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. DreamFusion: Text-to-3D using 2D Diffusion. In *ICLR*, 2023. 2

[62] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J. Guibas. PointNet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*, 2017. 4

[63] 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 *ICML*, 2021. 6, 7

[64] Jiawei Ren, Liang Pan, Jiaxiang Tang, Chi Zhang, Ang Cao, Gang Zeng, and Ziwei Liu. DreamGaussian4D: Generative 4D gaussian splatting. *arXiv:2312.17142*, 2023. 2, 7

[65] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 4

[66] Tianchang Shen, Jun Gao, Kangxue Yin, Ming-Yu Liu, and Sanja Fidler. Deep marching tetrahedra: a hybrid representation for high-resolution 3D shape synthesis. In *NeurIPS*, 2021. 2, 8

[67] Yawar Siddiqui, Tom Monnier, Filippos Kokkinos, Mahendra Kariya, Yanir Kleiman, Emilien Garreau, Oran Gafni, Natalia Neverova, Andrea Vedaldi, Roman Shapovalov, et al. Meta 3D AssetGen: Text-to-mesh generation with high-quality geometry, texture, and PBR materials. In *NeurIPS*, 2024. 1, 8

[68] Eftychios Sifakis and Jernej Barbic. FEM simulation of 3D deformable solids: a practitioner's guide to theory, discretization and model reduction. In *SIGGRAPH 2012 courses*. ACM, 2012. 8

[69] Habib Slim, Xiang Li, Yuchen Li, Mahmoud Ahmed, Mohamed Ayman, Ujjwal Upadhyay, Ahmed Abdelreheem, Arpit Prajapati, Suhail Pothigara, Peter Wonka, et al. 3DCoMPaT++: An improved large-scale 3D vision dataset for compositional recognition. *arXiv:2310.18511*, 2023. 1, 2, 3

[70] Deborah Sulsky, Zhen Chen, and Howard L Schreyer. A particle method for history-dependent materials. *Computer methods in applied mechanics and engineering*, 118(1-2): 179–196, 1994. 3

[71] Mingze Sun, Shiwei Mao, Keyi Chen, Yurun Chen, Shunlin Lu, Jingbo Wang, Junting Dong, and Ruqi Huang. ARMO: Autoregressive rigging for multi-category objects. *arXiv preprint arXiv:2503.20663*, 2025. 2

[72] Qi Sun, Zhiyang Guo, Ziyu Wan, Jing Nathan Yan, Shengming Yin, Wengang Zhou, Jing Liao, and Houqiang Li. EG4D: Explicit generation of 4D object without score distillation. In *ICLR*, 2025. 2

[73] Wenqiang Sun, Shuo Chen, Fangfu Liu, Zilong Chen, Yueqi Duan, Jun Zhang, and Yikai Wang. DimensionX: Create any 3D and 4D scenes from a single image with controllable video diffusion. *arXiv:2411.04928*, 2024. 1

[74] Xiyang Tan, Ying Jiang, Xuan Li, Zeshun Zong, Tianyi Xie, Yin Yang, and Chenfanfu Jiang. PhysMotion: Physics-grounded dynamics from a single image. *arXiv:2411.17189*, 2024. 2, 7

[75] Ty Trusty, Danny Kaufman, and David IW Levin. Mixed variational finite elements for implicit simulation of deformables. In *SIGGRAPH Asia*, 2022. 8

[76] Benjamin Ummenhofer, Lukas Prantl, Nils Thuerey, and Vladlen Koltun. Lagrangian fluid simulation with continuous convolutions. In *ICLR*, 2020. 8

[77] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 5

[78] Giuseppe Vecchio and Valentin Deschaintre. MatSynth: A modern PBR materials dataset. In *CVPR*, 2024. 2

[79] Stephanie Wang. *A Material Point Method for Elastoplasticity with Ductile Fracture and Frictional Contact*. University of California, Los Angeles, 2020. 3

[80] Stephanie Wang, Mengyuan Ding, Theodore F Gast, Leyi Zhu, Steven Gagniere, Chenfanfu Jiang, and Joseph M Teran. Simulation and visualization of ductile fracture with the material point method. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 2(2):1–20, 2019. 3

[81] Tsun-Hsuan Johnson Wang, Juntian Zheng, Pingchuan Ma, Yilun Du, Byungchul Kim, Andrew Spielberg, Josh Tenenbaum, Chuang Gan, and Daniela Rus. DiffuseBot: Breeding soft robots with physics-augmented generative diffusion models. In *NeurIPS*, 2023. 2

[82] Zhen Wang, Dongyuan Li, and Renhe Jiang. Diffusion models in 3D vision: A survey. *arXiv:2410.04738*, 2024. 2

[83] Daniel Watson, Saurabh Saxena, Lala Li, Andrea Tagliasacchi, and David J Fleet. Controlling space and time with diffusion models. In *ICLR*, 2024. 2

[84] Alan Wineman. Some results for generalized neo-hookean elastic materials. *International Journal of Non-Linear Mechanics*, 40(2-3):271–279, 2005. 3

[85] Rundi Wu, Ruiqi Gao, Ben Poole, Alex Trevithick, Changxi Zheng, Jonathan T Barron, and Aleksander Holynski. Cat4D: Create anything in 4D with multi-view video diffusion models. *arXiv:2411.18613*, 2024. 1, 2

[86] 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. *arXiv:2412.01506*, 2024. 7, 2, 8

[87] Tianyi Xie, Zeshun Zong, Yuxing Qiu, Xuan Li, Yutao Feng, Yin Yang, and Chenfanfu Jiang. PhysGaussian: Physics-integrated 3D gaussians for generative dynamics. In *CVPR*, 2024. 2, 3

[88] Jiale Xu, Weihao Cheng, Yiming Gao, Xintao Wang, Shenghua Gao, and Ying Shan. InstantMesh: Efficient 3D mesh generation from a single image with sparse-view large reconstruction models. *arXiv:2404.07191*, 2024. 2

[89] Yinghao Xu, Zifan Shi, Wang Yifan, Hansheng Chen, Ceyuan Yang, Sida Peng, Yujun Shen, and Gordon Wetzstein. GRM: Large gaussian reconstruction model for efficient 3D reconstruction and generation. In *ECCV*, 2024. 2

[90] Zhan Xu, Yang Zhou, Evangelos Kalogerakis, Chris Landreth, and Karan Singh. Rignet: Neural rigging for articulated characters. *ACM Trans. on Graphics*, 39, 2020. 8

[91] Xianfang Zeng, Xin Chen, Zhongqi Qi, Wen Liu, Zibo Zhao, Zhibin Wang, Bin Fu, Yong Liu, and Gang Yu. Paint3D: Paint anything 3D with lighting-less texture diffusion models. In *CVPR*, 2024. 6

[92] Yifei Zeng, Yanqin Jiang, Siyu Zhu, Yuanxun Lu, Youtian Lin, Hao Zhu, Weiming Hu, Xun Cao, and Yao Yao. STAG4D: Spatial-temporal anchored generative 4D gaussians. In *ECCV*, 2024. 2, 8

[93] Biao Zhang and Peter Wonka. LaGeM: A large geometry model for 3D representation learning and diffusion. In *ICLR*, 2025. 3

[94] Biao Zhang, Jiapeng Tang, Matthias Niessner, and Peter Wonka. 3DShape2VecSet: A 3D shape representation for neural fields and generative diffusion models. *ACM TOG*, 42 (4):1–16, 2023. 4, 5, 6, 1

[95] 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 *ECCV*, 2024. 2

[96] Haoyu Zhao, Hao Wang, Xingyue Zhao, Hongqiu Wang, Zhiyu Wu, Chengjiang Long, and Hua Zou. Automated 3D physical simulation of open-world scene with gaussian splatting. *arXiv:2411.12789*, 2024. 3

[97] Zibo Zhao, Zeqiang Lai, Qingxiang Lin, Yunfei Zhao, Haolin Liu, Shuhui Yang, Yifei Feng, Mingxin Yang, Sheng Zhang, Xianghui Yang, et al. Hunyuan3D 2.0: Scaling diffusion models for high resolution textured 3D assets generation. *arXiv:2501.12202*, 2025. 1, 6

[98] Haoyu Zhen, Xiaowen Qiu, Peihao Chen, Jincheng Yang, Xin Yan, Yilun Du, Yining Hong, and Chuang Gan. 3D-VLA: A 3D vision-language-action generative world model. In *ICML*, 2024. 3

[99] Zeshun Zong, Xuan Li, Minchen Li, Maurizio M Chiaramonte, Wojciech Matusik, Eitan Grinspun, Kevin Carlberg, Chenfanfu Jiang, and Peter Yichen Chen. Neural stress fields for reduced-order elastoplasticity and fracture. In *SIGGRAPH Asia*, 2023. 3