

DSO: Aligning 3D Generators with Simulation Feedback for Physical Soundness

Ruining Li Chuanxia Zheng Christian Rupprecht Andrea Vedaldi
Visual Geometry Group, University of Oxford

{ruining, cxzheng, chrisr, vedaldi}@robots.ox.ac.uk

ruiningli.com/dso

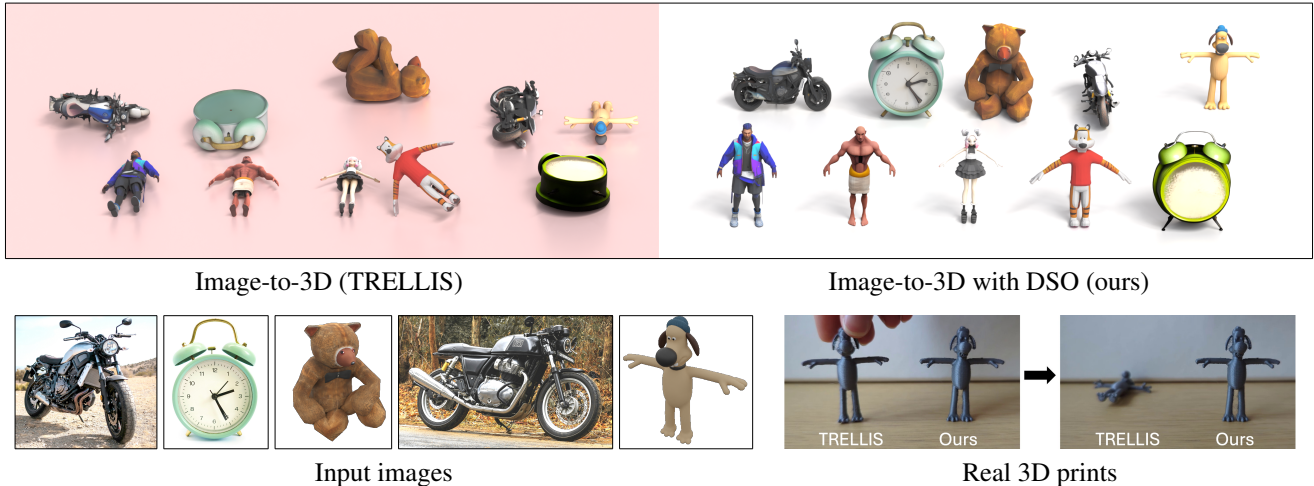


Figure 1. *Top-left*: A state-of-the-art image-to-3D model like TRELLIS often fails to reconstruct 3D objects that can stand under gravity even when prompted with images of stable objects (e.g., *bottom-left*). *Top-right*: Our method, DSO, improves the image-to-3D model via **D**irect **S**imulation **O**ptimization, significantly increasing the likelihood that generated 3D objects can stand, in physical simulation and in real-life, when 3D printed (*bottom-right*). The method incurs no additional cost at test time, and can thus generate such objects in seconds.

Abstract

Most 3D object generators prioritize aesthetic quality, often neglecting the physical constraints necessary for practical applications. One such constraint is that a 3D object should be self-supporting, i.e., remain balanced under gravity. Previous approaches to generating stable 3D objects relied on differentiable physics simulators to optimize geometry at test time, which is slow, unstable, and prone to local optima. Inspired by the literature on aligning generative models with external feedback, we propose **D**irect **S**imulation **O**ptimization (DSO). This framework leverages feedback from a (non-differentiable) simulator to increase the likelihood that the 3D generator directly outputs stable 3D objects. We construct a dataset of 3D objects labeled with stability scores obtained from the physics simulator. This dataset enables fine-tuning of the 3D generator using the stability score as an alignment metric, via direct preference optimization (DPO) or direct reward optimization (DRO)—a novel objective we introduce to align diffusion models without requiring pairwise preferences. Our experi-

ments demonstrate that the fine-tuned feed-forward generator, using either the DPO or DRO objective, is significantly faster and more likely to produce stable objects than test-time optimization. Notably, the DSO framework functions even without any ground-truth 3D objects for training, allowing the 3D generator to self-improve by automatically collecting simulation feedback on its own outputs.

1. Introduction

Given a single image of an object that is *stable under gravity*, we consider the problem of reconstructing it in 3D. Recent image-to-3D reconstructors [24, 40, 44, 45, 55, 67, 76, 85–87, 89, 95, 101, 112, 113] have focused on improving the quality of objects’ 3D geometry and appearance, but not necessarily their physical soundness. As shown in Fig. 2, when prompted with an image of a stable object, state-of-the-art generators like TRELLIS [101] and Hunyuan3D 2.0 [87] often fail to produce a stable object in 3D. The failure rate is 15% even for objects seen during *training* and

increases significantly for new objects, such as the clock and motorcycles in Fig. 1.

Stability is a common property of natural and man-made objects and is important in many applications, such as fabrication and simulation [35, 59]. It is, therefore, important to reconstruct 3D objects that satisfy this property.

Previous works on generating physically sound 3D objects [56, 61, 104] have focused on specific object categories, such as furniture. More recent methods like Atlas3D [7] and PhysComp [21] tackle a broader range of object categories. Both methods optimize a 3D model, either from scratch [7] or from the output of an off-the-shelf 3D generator [21], using *differentiable* physics-based losses that reward stability. To compute these losses, they require differentiable simulators such as [26, 52], which, despite continuous improvements, remain slower and numerically less stable than non-differentiable simulators like [53, 88]. As a result, Atlas3D and PhysComp are slow and susceptible to local optima and numerical instability.

In this paper, we aim to improve a feed-forward 3D generator so that it *directly* outputs physically stable objects without requiring test-time corrections. A naïve approach would be to use losses similar to those proposed by Atlas3D and PhysComp for feed-forward training instead of test-time optimization, but this would still require a differentiable simulator. Instead, inspired by works on aligning generative models with human preferences [71, 90], we introduce **Direct Simulation Optimization (DSO)**. This simple and effective approach fine-tunes a 3D generator by aligning it with the “preference” provided automatically by an off-the-shelf physics simulator. With this, we explore three research questions: (1) How to use this simulation preference dataset to fine-tune a 3D generator efficiently; (2) How to construct such a dataset *without* requiring ground-truth 3D data; and (3) Whether the fine-tuned generator generalizes well, outputting physically sound 3D objects from image prompts unseen during training.

Our motivation for using reward optimization is that stability, like many other physical attributes of an object, is *discrete*: either an object is stable, or it collapses under gravity. Stability does not distinguish between unstable states regardless of how close they are to becoming stable, making it difficult to optimize using techniques like gradient descent. In contrast, it is easy to determine whether an object is stable or not using a physics simulator. Hence, we reformulate the problem as a *reward-based learning task*, where we reward stable outputs and penalize unstable ones. Inspired by direct preference optimization (DPO) [71], we propose an alternative objective, direct *reward* optimization (DRO), for aligning diffusion models with external preferences. Notably, DRO does not require *pairwise* preference data for training.

Our second contribution is to show that we can derive

reward signals solely from generated data, eliminating the need to collect large datasets of stable 3D objects for training at scale. We achieve this by generating new 3D assets using the 3D generator itself. These generated 3D assets are then evaluated within a physics simulator, classifying them as stable or unstable. This process allows us to construct a fully automated self-improving pipeline, where the model is trained on its own output, assessed by a physics simulator rather than relying on a large dataset of 3D objects.

We show that, when integrated with either DPO or DRO as the objective function, our Direct Simulation Optimization framework can steer the output of the 3D generator to align with physical soundness. The final model surpasses previous approaches for physically stable 3D generation on existing evaluation benchmarks [21]. It operates in a *feed-forward* manner at test time, outperforming heavily engineered solutions like [7, 21] that perform test-time optimization, both in terms of speed and probability of generating a stable object as output. The model also generalizes well to images collected in the wild (Fig. 1).

Our experiments show that, in our setting, the proposed DRO objective achieves faster convergence and superior alignment compared to DPO, suggesting that it may be a better candidate for diffusion alignment in general. While our study focuses on stability under gravity, the reward-based approach and the self-improving optimization strategy can, in principle, be applied to any physical attributes that can be assessed via a simulator.

2. Related Work

3D generation and reconstruction. Early 3D generators used generative adversarial networks (GANs) [20] and various 3D representations such as point clouds [28, 41], voxel grids [98, 103, 115], view sets [60, 66], NeRF [4, 5, 13, 63, 75], SDF [17], and 3D Gaussian mixtures [97]. However, GANs are challenging to train on a large scale in an ‘open world’ setting. This explains why recent methods have shifted to diffusion models [23, 79], which use the same 3D representations [9, 51, 58, 62, 77, 83] while improving training stability and scalability. Other approaches train neural networks [6, 8, 27, 29, 30, 39, 82, 84, 99, 100, 106, 107] to directly regress 3D models from 2D images. Researchers have also explored scaling 3D reconstruction models [24, 85, 94] on Objaverse [11, 12], improving generalization. DreamFusion [68] and SJC [91] leverage large-scale image/video generators for 3D generation using score distillation [29, 40, 55, 68, 91, 93, 116]. The works of [18, 22, 36, 45, 47, 48, 54, 76, 86, 96, 113] fine-tune these models for generalizable 3D generation. More recently, researchers have introduced latent 3D representations [10, 101, 111] whose distributions can be effectively modeled by denoising diffusion or rectified flow [1, 42, 46]. CLAY [112] and TRELIS [101] are among the 3D genera-

tors trained in this manner, producing superior results compared to methods that rely on 2D generation.

These advances have significantly improved the quality of the geometry and appearance of generated 3D assets, but not necessarily their physical soundness. This limitation reduces their utility in downstream applications like fabrication and simulation. In contrast, we propose a 3D generation approach that explicitly optimizes physical soundness, specifically stability under gravity.

Physically-sound 3D generation. Early studies explored methods to predict physical properties from images and videos, such as mass [81], shadows [92], materials [109], occlusions [110], and support [78]. While effective in predicting specific physical parameters, these methods do not generalize directly to 3D reconstruction. Recent works like Physdiff [108], PhysGaussian [102], and PIE-NeRF [15] extend physics-based rendering [26] to NeRF [57] and 3D Gaussian Splatting [32]. These methods focus on modeling the motion of objects rather than their stability under gravity. Similar to our work, Phys-DeepSDF [56], PhyScene [104], and PhyRecon [61] incorporate explicit physical constraints in 3D reconstruction. However, these methods are limited to specific object categories, such as furniture. More related to our work, Atlas3D [7] and PhysComp [21] are not restricted to specific categories; instead, they rely on test-time optimization using carefully designed differentiable, physics-based losses. We address a similar problem but in a *feed-forward* manner using *reward-based optimization*, avoiding the need for fragile and slow physics-based losses at test time.

Preference alignment in generative models. The Direct Simulation Optimization (DSO) framework we propose can be trained using Direct Preference Optimization (DPO) [71], a technique initially developed for fine-tuning large language models. Diffusion-DPO [90] first extended DPO to vision diffusion models, enabling direct optimization of human preferences, and was further extended by [16, 43]. While various preference alignment approaches exist [2, 14, 34, 65, 69], DPO has the distinct advantage of not requiring an oracle to compute the reward signal during training and avoids the need for reward modeling. Inspired by DPO, we also propose an alternative objective named direct reward optimization (DRO), which does not require *pairwise* preference data to align the generator.

3. Method

Given a pre-trained diffusion-based 3D generator p_{ref} that takes a single image I as input and generates 3D assets $\mathbf{x}_0 \sim p_{\text{ref}}(\mathbf{x}_0|I)$, our goal is to learn a new model p_θ that produces more physically sound generations than p_{ref} . We assume access to an oracle o that, given a sample \mathbf{x}_0 , outputs $o(\mathbf{x}_0) \in \{0, 1\}$, indicating whether \mathbf{x}_0 is physically

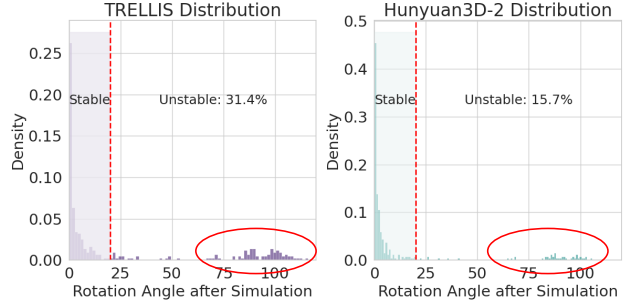


Figure 2. State-of-the-art 3D generators *cannot* robustly produce stable objects. Even when taking images of stable objects in their *training* set as input, TRELIS [101] and Hunyuan3D-2.0 [87] generate about 30% and 15% unstable assets respectively.

sound. In this paper, we focus on stability under gravity, where o is computed by a physics simulator to determine whether a 3D model \mathbf{x}_0 is self-supporting.

3.1. Challenges of Optimizing Physical Soundness

To improve the physical soundness of the generated samples, one approach is to fine-tune the model with the following objective:

$$\max_{\theta} \mathbb{E}_{I \sim \mathcal{I}, \mathbf{x}_0 \sim p_{\theta}(\mathbf{x}_0|I)} [o(\mathbf{x}_0)] - \beta \mathbb{D}_{\text{KL}} [p_{\theta}(\mathbf{x}_0|I) || p_{\text{ref}}(\mathbf{x}_0|I)], \quad (1)$$

where \mathcal{I} is the empirical distribution of a dataset of image prompts, and β is a hyperparameter trading off the two terms. The first term encourages the generated object \mathbf{x}_0 from $p_{\theta}(\mathbf{x}_0|I)$ to be physically sound, while the second term constrains the distribution to remain close to the base model to ensure that the generated geometry remains faithful to the input image I .

A key challenge in optimizing Eq. (1) is that the oracle o is *non-differentiable*. One approach to address this issue is to reframe the denoising process as a multi-step Markov decision process (MDP) [2] and optimize Eq. (1) using reinforcement learning (RL) [73, 74]. However, in our setting, evaluating o is computationally expensive due to the need to run a physical simulation and the overhead introduced by decoding latent 3D representations \mathbf{x}_0 into simulation-ready assets. The decoding process of state-of-the-art 3D generators involves querying dense 3D grid points and extracting a 3D mesh with marching cubes [87, 112], and may even require inference of another geometry generator [101]. These factors make optimization of Eq. (1) via RL computationally prohibitive.

3.2. Formulation as Reward Optimization

We aim to reformulate the objective function to be easier to optimize, specifically eliminating the need to evaluate

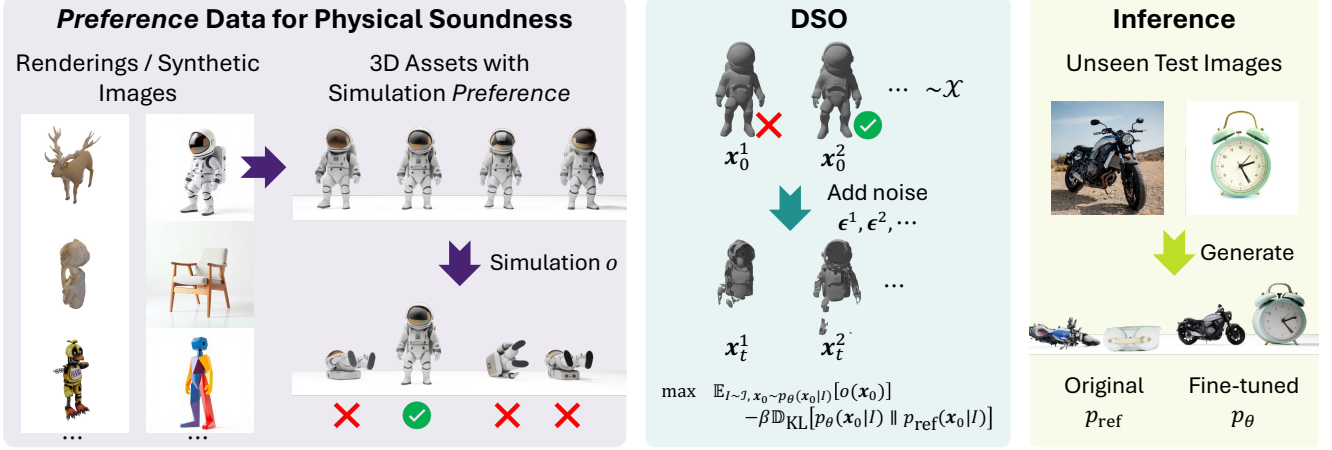


Figure 3. **Overview of Direct Simulation Optimization (DSO).** *Left:* Starting from a set of (potentially synthetic) image prompts, we task the base model p_{ref} to generate 3D models. Each model is augmented with a binary stability label through physics-based simulation (Sec. 3.3). *Middle:* Using this dataset, we fine-tune the base model by reinforcing stable samples and discouraging unstable ones. Our objective formulation enables efficient training via gradient descent without *pairwise* preferences (Sec. 3.2). *Right:* At test time, the fine-tuned model can generate self-supporting objects when conditioned on (out-of-distribution) images of stable objects captured *in the wild*.

o during training, while still preserving the intended goals of Eq. (1). This is analogous to the goal of text-to-image diffusion model alignment in Diffusion-DPO [90]. In both cases, the reward signal (*i.e.*, evaluation of o by simulation or by collecting human preferences for [90]) is hard to obtain in a scalable way during training.

Following Diffusion-DPO [90], we can re-parameterize $o(\mathbf{x}_0)$ using the optimal reverse diffusion process, modeled by $p_\theta^*(\mathbf{x}_{0:T})$, that maximizes (a lower bound of) Eq. (1):

$$o(\mathbf{x}_0) = \beta \mathbb{E}_{p_\theta(\mathbf{x}_{1:T}|\mathbf{x}_0, I)} \left[\log \frac{p_\theta^*(\mathbf{x}_{0:T}|I)}{p_{\text{ref}}(\mathbf{x}_{0:T}|I)} \right] + \beta \log Z(I), \quad (2)$$

for any $I \in \text{supp}(\mathcal{I})$, where $Z(I)$ is a normalizing term independent of p_θ . The derivation follows [90] and is detailed in Appendix A.

Direct Reward Optimization (DRO). Given an image dataset \mathcal{I} and 3D models \mathcal{X}_I corresponding to each image $I \in \mathcal{I}$, we can pre-compute $o(\mathbf{x}_0)$ for each 3D model $\mathbf{x}_0 \in \mathcal{X}_I$ to supervise p_θ using Eq. (2), via an $L1$ loss:

$$\mathcal{L} := \mathbb{E}_{I \sim \mathcal{I}, \mathbf{x}_0 \sim \mathcal{X}_I} \left[\left| o(\mathbf{x}_0) - \beta \left(\mathbb{E}_{p_\theta(\mathbf{x}_{1:T}|\mathbf{x}_0, I)} \left[\log \frac{p_\theta(\mathbf{x}_{0:T}|I)}{p_{\text{ref}}(\mathbf{x}_{0:T}|I)} \right] + \log Z(I) \right) \right| \right]. \quad (3)$$

However, despite \mathcal{L} being a function of the trainable parameters θ , it is intractable because neither $Z(I)$ nor the expectation over $\mathbf{x}_{1:T}$ can be computed efficiently.

To address this issue, we notice that the absolute value of $o(\mathbf{x}_0)$ is arbitrary, *i.e.*, we could use another oracle $o'(\mathbf{x}_0) \in \{l, u\}$ which evaluates to l for unstable \mathbf{x}_0 and u for stable

\mathbf{x}_0 , as long as $l < u$ in Eq. (1). In this setting, there exists a choice of β that leads to the same optimum p_θ^* as with the original oracle o in Eq. (1).

Since we aim to use stochastic gradient descent, which is *local* and *continuous*, to optimize \mathcal{L} , we may as well choose l and u such that, within the training compute budget, the sum of $\log Z(I)$ and the expectation over $\mathbf{x}_{1:T}$ is bounded within $(\frac{l}{\beta}, \frac{u}{\beta})$. By doing so, we can remove the absolute value in Eq. (3) and get rid of the terms independent of p_θ :

$$\arg \min \mathcal{L} = \arg \min \mathbb{E}_{I \sim \mathcal{I}, \mathbf{x}_0 \sim \mathcal{X}_I, \mathbf{x}_{1:T} \sim p_\theta(\mathbf{x}_{1:T}|\mathbf{x}_0, I)} \left[(1 - 2o(\mathbf{x}_0)) \log \frac{p_\theta(\mathbf{x}_{0:T}|I)}{p_{\text{ref}}(\mathbf{x}_{0:T}|I)} \right]. \quad (4)$$

To make sampling tractable, we approximate the reverse process $p_\theta(\mathbf{x}_{1:T}|\mathbf{x}_0, I)$ with the forward process $q(\mathbf{x}_{1:T}|\mathbf{x}_0)$, following [90]. With some algebra, this yields:

$$\mathcal{L}_{\text{DRO}} = -T \mathbb{E}_{I \sim \mathcal{I}, \mathbf{x}_0 \sim \mathcal{X}_I, t \sim \mathcal{U}(0, T), \mathbf{x}_t \sim q(\mathbf{x}_t|\mathbf{x}_0)} \left[w(t)(1 - 2o(\mathbf{x}_0)) \|\epsilon - \epsilon_\theta(\mathbf{x}_t, t)\|_2^2 \right], \quad (5)$$

where $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ is a draw from $q(\mathbf{x}_t|\mathbf{x}_0)$ and $w(t)$ is a weighting function. \mathcal{L}_{DRO} directly encourages the model to improve at denoising samples \mathbf{x}_0 with high reward (*i.e.*, $o(\mathbf{x}_0) = 1$) and to denoise less well samples \mathbf{x}_0 with low reward (*i.e.*, $o(\mathbf{x}_0) = 0$). We hence dub it direct reward optimization (*DRO*). Different from the DPO formulation [90] (which we briefly review next), fine-tuning with \mathcal{L}_{DRO} does not require *pairwise* preference data and does not query the

base model ϵ_{ref} during training, potentially applicable to more alignment settings than DPO.

Direct Preference Optimization (DPO). Alternatively, assuming \mathcal{X}_I contains both stable and unstable models, we can use the objective introduced in [90], which relies on *pairwise* preference data and minimizes a *contrastive* loss:

$$\mathcal{L}_{\text{DPO}} := -\mathbb{E}_{I \sim \mathcal{I}, (\mathbf{x}_0^w, \mathbf{x}_0^l) \sim \mathcal{X}_I^2} \left[\text{logsigmoid}(r(\mathbf{x}_0^w) - r(\mathbf{x}_0^l)) \right], \quad (6)$$

where $(\mathbf{x}_0^w, \mathbf{x}_0^l)$ is a pair of physically sound and unsound 3D models corresponding to the same image I (*i.e.*, $o(\mathbf{x}_0^w) = 1 - o(\mathbf{x}_0^l) = 1$), and r is a reward model introduced to derive the loss from the Bradley-Terry model [3]. Following the derivation in [90], this simplifies to:

$$\begin{aligned} \mathcal{L}_{\text{DPO}} = & -\mathbb{E}_{I \sim \mathcal{I}, (\mathbf{x}_0^w, \mathbf{x}_0^l) \sim \mathcal{X}_I^2, t \sim \mathcal{U}(0, T), \mathbf{x}_t^w \sim q(\mathbf{x}_t^w | \mathbf{x}_0^w), \mathbf{x}_t^l \sim q(\mathbf{x}_t^l | \mathbf{x}_0^l)} \\ & \text{logsigmoid} \left(-\beta T w(t) \left(\right. \right. \\ & \quad \left. \left. \begin{aligned} & \|\epsilon^w - \epsilon_\theta(\mathbf{x}_t^w, t)\|_2^2 - \|\epsilon^w - \epsilon_{\text{ref}}(\mathbf{x}_t^w, t)\|_2^2 \\ & - (\|\epsilon^l - \epsilon_\theta(\mathbf{x}_t^l, t)\|_2^2 - \|\epsilon^l - \epsilon_{\text{ref}}(\mathbf{x}_t^l, t)\|_2^2) \end{aligned} \right) \right), \quad (7) \end{aligned}$$

where $\epsilon^w, \epsilon^l \sim \mathcal{N}(0, \mathbf{I})$ are two independent random draws. Please refer to [90] for details.

3.3. DSO with Generated Data

We can now fine-tune the generator p_θ ¹ with Eq. (5) or Eq. (7) as the objective using stochastic gradient descent. The final cornerstone of our framework, **Direct Simulation Optimization (DSO)**, is to obtain a set of images \mathcal{I} and their corresponding 3D models $\mathcal{X}_{I \in \mathcal{I}}$. Procuring a large number of stable 3D objects for training at scale is challenging, especially if we want multiple different objects for a single image prompt as in Eq. (7). Instead, we propose a scheme that leverages the 3D models generated by the generator p_{ref} itself. As illustrated in Fig. 3, we first curate a large, diverse image dataset \mathcal{I} . These images can be either renderings of existing 3D datasets such as [11, 12], or synthetic images generated by a 2D generator such as [33, 72]. We then task the base model p_θ to create 3D models \mathcal{X}_I , taking individual images $I \in \mathcal{I}$ as input. These 3D models, subsequently augmented with physical soundness scores via physics-based simulation, are used to fine-tune the model for enhanced physical soundness, achieving self-improvement *without* relying on 3D ground truths.

¹While our presentation in Sec. 3.2 focuses on a DDPM-formulated diffusion model with discrete timesteps [23], the same approach can be readily adapted to rectified flow models [1, 42, 46] and other diffusion formulations [31, 80], as their differences primarily lie in the noise schedule and loss weighting [19].

4. Experiments

We evaluate DSO on the task of generating physically stable 3D models under gravity and compare it to prior works that use test-time optimization of physically-based losses (Sec. 4.2). We assess the ability to generate stable 3D objects while retaining the fidelity of the 3D reconstruction (as it would be trivial to make all objects stable by making them, *e.g.*, cubes). We discuss the effect of DSO on the generated geometry in Sec. 4.3 and DSO’s scaling behavior in Sec. 4.5. In Sec. 4.6, we demonstrate how DSO can be adapted to leverage exclusively synthetic 2D images instead of renderings of ground-truth 3D models.

4.1. Experiment Details

Model and data. We apply DSO to fine-tune TRELIS [101], a state-of-the-art image-to-3D generator, and measure its ability to consistently generate self-supporting 3D models before and after optimization. TRELIS contains *two* rectified flow transformers: the first generates the coarse geometry of the 3D object, and the second refines its fine-grained details. In our experiments, we fine-tune only the linear layers of the first transformer, as stability is primarily controlled by the coarse geometry. We use LoRA [25] to reduce the number of parameters to optimize. We select TRELIS because it is a state-of-the-art 3D generator and is available as open source, but our method is not specific to this model.

For the training data, we first generate a large number of 3D models with TRELIS, conditioned on Objaverse [12] renderings. We exclude objects from Objaverse with unstable ground-truth shapes and filter out low-quality ones following [101]. Additionally, we include only objects categorized by GObjaverse [70] as “Human-Shape”, “Animals”, or “Daily-Used”, as these categories often feature two-legged shapes and tall, slender structures, making them more challenging to stabilize under gravity. We render 6 images for each of the remaining 13k objects and generate 4 different models per image, yielding 312k 3D models in total. We then use the MuJoCo [88] simulator to conduct physical simulations for each model, starting from an upright pose on flat ground. We use the tilting angle at the final equilibrium state to determine stability, based on a hard cut-off of 20°: a model \mathbf{x}_0 is considered stable (*i.e.*, $o(\mathbf{x}_0) = 1$) if its tilting angle is below 20° and unstable otherwise. During training, we sample models for an image prompt uniformly at random.

Training. We use AdamW [49] to fine-tune the base model using LoRA [25] (rank 64) with a batch size of 48 on 4 NVIDIA A100 GPUs. We train *two* separate models, optimizing them using \mathcal{L}_{DRO} (Eq. (5)) for 4,000 steps and using \mathcal{L}_{DPO} (Eq. (7)) for 8,000 steps, respectively. The β in Eq. (7) is set to 500. More details can be found in Ap-

Method	Stability		Geometry	
	% Stable↑ (% Output↑)	Rot.↓	CD↓	F-Score↑
Full evaluation set (65 objects)				
TRELLIS [101]	85.1 (100)	14.14°	0.0485	73.12
Atlas3D [7]	69.4 (95.4)	32.86°	—	—
TRELLIS + DSO (w/ \mathcal{L}_{DPO})	95.1 (100)	5.42°	0.0480	73.62
TRELLIS + DSO (w/ \mathcal{L}_{DRO})	99.0 (100)	1.88°	0.0440	76.17
Partial evaluation set (11 unstable objects)				
TRELLIS [101]	54.5 (100)	39.18°	0.0529	72.48
TRELLIS + PhysComp [21]	80.3 (46.2)	18.14°	0.0698	53.73
TRELLIS + DSO (w/ \mathcal{L}_{DPO})	82.6 (100)	16.83°	0.0509	73.07
TRELLIS + DSO (w/ \mathcal{L}_{DRO})	95.5 (100)	5.58°	0.0520	73.61

Table 1. **Quantitative Results.** DSO fine-tuned models (using either \mathcal{L}_{DRO} or \mathcal{L}_{DPO}) significantly outperform baseline methods Atlas3D [7] and PhysComp [21] in both physical stability and geometric quality. Beyond improving the physical soundness of the base model TRELLIS [101], DSO also slightly improves its geometric fidelity *without* requiring ground-truth 3D supervision.

pendix B.

Evaluation. We evaluate on the dataset from [21], which consists of 100 Objaverse [12] objects from plants, animals, and characters. We exclude the 35 objects whose ground-truth shape is *not* self-supporting and render 12 images for each of the remaining objects, resulting in a final set of 65 objects and 780 images. These objects are removed from our training set.

Metrics. For quantitative results, we report the following stability measures: % *Output* counts the frequency of successfully outputting a 3D object, regardless of its stability; % *Stable* counts the percentage of stable assets among those generated; *Rotation angle* (*Rot.* in short) measures the average tilting angle of generated objects at their equilibrium state. In addition, to evaluate the mesh geometry, we report *Chamfer Distance* (*CD*) and *F-Score* (with threshold 0.05 [44, 55, 94]). Following common practices [44, 55, 94], we scale the meshes to fit within the unit cube and align the generated meshes optimally with the ground truths using Iterated Closest Point (ICP) before computing CD and F-Score.

Baselines. In addition to our base model TRELLIS [101], we consider two baseline methods designed to generate self-supporting 3D objects: *Atlas3D* [7] and *PhysComp* [21]. Atlas3D is a text-to-3D framework that combines score distillation sampling [68] with physically-based loss terms, primarily the magnitude of the object orientation change at equilibrium, computed via differentiable simulation. PhysComp takes a (volumetric) tetrahedral mesh as input and applies test-time optimization to improve its physical soundness, including its stability under gravity. This is achieved by encouraging the projection of

the center of mass to be within the convex hull of the contact points. For [7] and [21], we use their official implementations. For the text-conditioned Atlas3D, we prompt it using captions of our multi-view renderings, obtained with GPT-4V [64]. We generate *one* asset per object in the evaluation set. For PhysComp, we task it to optimize the 3D models generated by TRELLIS. Since the optimization on our hardware (24-core CPU with 668 GiB RAM in total) takes significantly longer (on average 15 minutes) than the 80 seconds reported by the authors, we only run it on an 11-object subset whose renderings lead TRELLIS to generate unstable 3D models, amounting to $11 \times 12 = 132$ runs. As the optimization time varies dramatically with mesh complexity, we set a strict time budget of 30 minutes per run.

4.2. Results and Analysis

Quantitative results. Table 1 reports the quantitative results evaluated for both baselines and our method. Notably, our DSO fine-tuned TRELLIS (using either \mathcal{L}_{DRO} or \mathcal{L}_{DPO}) outperforms all baselines on both physical stability and geometry fidelity *without* any test-time optimization.

Qualitative results. Figure 4 presents qualitative comparisons with baselines, highlighting cases where our base model TRELLIS [101] fails to generate self-supporting assets. Atlas3D [7], inheriting the limitations of SDS-based approaches [68], often suffers from over-saturation and over-smoothness (a, b, c). While incorporating physics-based stability loss, its optimization remains unreliable (a) and can introduce structural artifacts such as extraneous limbs (b, c). PhysComp [21], which refines TRELLIS outputs, does not preserve texture and can distort the original shape (a), compromising faithfulness to the input image. The method struggles to stabilize meshes in challenging scenarios (a) and frequently suffers from numerical instabilities, sometimes failing to generate outputs entirely (c). In contrast, our final model leverages the strong geometric prior of TRELLIS while significantly enhancing physical stability without introducing additional computational overhead at test time.

Analysis. We note that: (1) Differentiable simulation often suffers from numerical issues, as reflected by the lower % Output of [7] and [21], due to the need for differentiable ODE solving. DSO circumvents this requirement by framing physical soundness optimization as a reward learning task (Sec. 3.2) and augmenting 3D models with simulation feedback before training. (2) Unlike visual quality, physical stability demands high accuracy, especially in the contact region. While existing efforts to align vision generators [16, 43, 69, 90, 114] focus on enhancing visual quality, we show that alignment can also substantially improve accuracy-sensitive metrics. (3) For our task, \mathcal{L}_{DRO} proves to be a more effective objective than \mathcal{L}_{DPO} (Tab. 1) and could also be beneficial in other diffusion alignment settings, es-

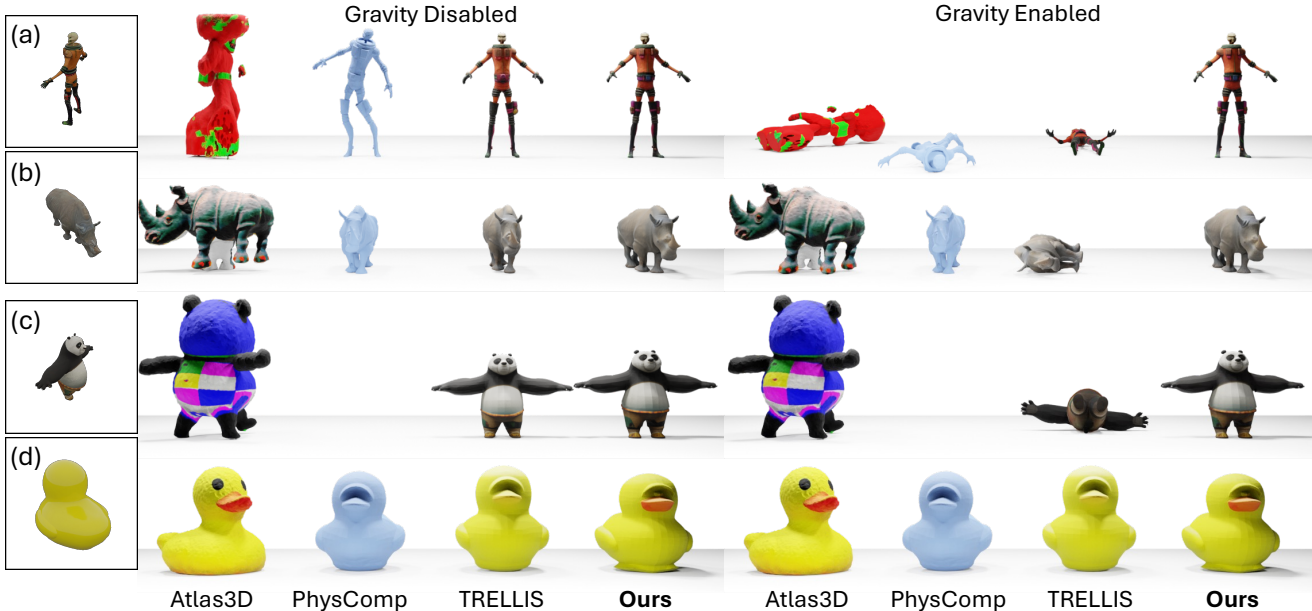


Figure 4. **Qualitative Comparison** with baseline methods. Our model can more reliably generate 3D assets that are stable under gravity and faithful to the conditioning images.

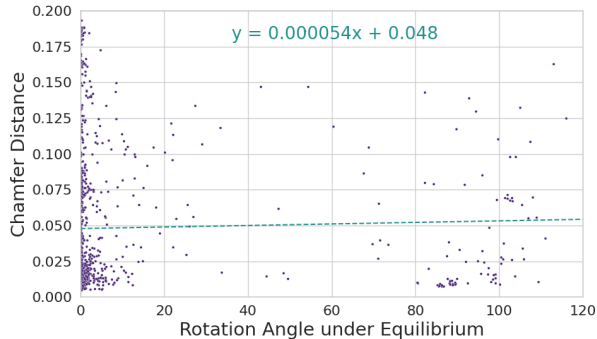


Figure 5. **(Lack of) Correlation** between geometry reconstruction quality (Chamfer Distance) and stability (rotation angle).

pecially when access to *pairwise* preference data is limited.

4.3. Physical Soundness vs. Geometry Quality

Eq. (1) and the other losses in Sec. 3 imply a potential trade-off between physical soundness and geometric quality, controlled by the parameter β . However, in Tab. 1, DSO fine-tuned on TRELIS not only enhances physical stability but also improves *geometric fidelity*. This outcome is somewhat surprising, given that TRELIS was explicitly *trained* on (at least some) objects in the evaluation with geometric losses (*e.g.*, occupancy), whereas our DSO does *not* directly supervise the base model with ground-truth geometry.

To investigate this, we generate 800 distinct 3D assets using TRELIS and analyze the relationship between their

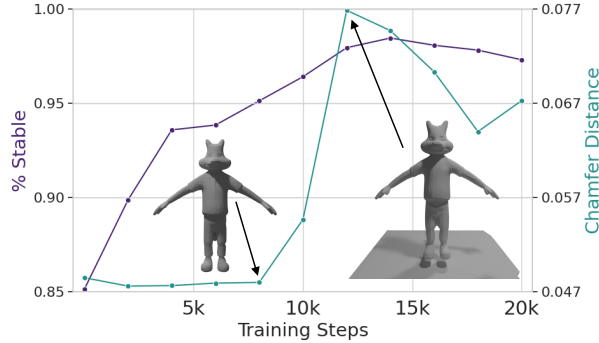
Method	Stability		Geometry	
	% Stable \uparrow	Rot. \downarrow	CD \downarrow	F-Score \uparrow
TRELIS [101]	85.1	14.14 $^\circ$	0.0485	73.12
TRELIS + SFT	89.5	10.22 $^\circ$	0.0440	76.17
TRELIS + DSO w/ \mathcal{L}_{DPO}	<u>95.1</u>	<u>5.42$^\circ$</u>	0.0480	73.62
TRELIS + DSO w/ \mathcal{L}_{DRO}	99.0	1.88$^\circ$	0.0440	76.17

Table 2. **Comparison** with Supervised Fine-tuning (SFT). SFT yields faithful geometry, but its samples are less physically stable.

geometric quality (measured by CD) and physical stability (quantified by the tilting angle at equilibrium), as shown in Fig. 5. The correlation is not statistically significant, suggesting that improving physical soundness does not need to compromise geometric quality. If anything, there is a very slight positive correlation between the two.

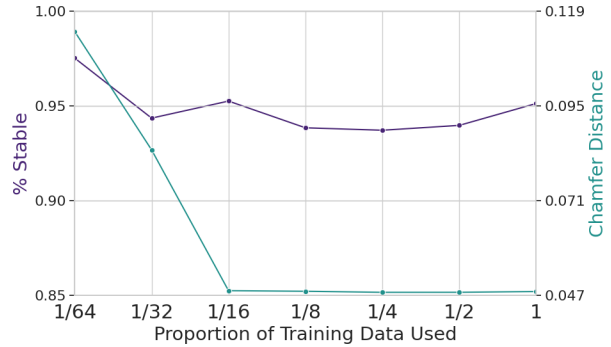
4.4. Comparison with Supervised Fine-tuning

To further assess the effectiveness of DSO, we also compare it with supervised fine-tuning (SFT) in Tab. 2. For SFT, we fine-tune TRELIS on the stable subset of our constructed dataset (*i.e.*, $\{\mathbf{x}_0 \in \mathcal{X} | o(\mathbf{x}_0) = 1\}$, consisting of 72k objects out of the 312k generated in total), using the rectified flow objective [1, 42, 46] with the same hyperparameter configuration as our main training runs for 8,000 steps. While SFT yields better geometry, its samples are less physically sound. This suggests that the model prioritizes geometry over physical plausibility, making fine-tuning 3D generators solely on physically stable objects less effective for



(a) **Scaling with training compute.** Longer training enhances the physical stability further, yet excessive training degrades geometric quality.

Figure 6. **Scaling Behaviors** of DSO with training compute (*left*) and data (*right*).



(b) **Scaling with training data.** Smaller ($\frac{1}{16}$ of the full dataset) data achieve comparable results for physical alignment.

aligning physical soundness. In contrast, by exposing the model to both stable and unstable objects, DSO encourages the model to better focus on physical properties.

4.5. Scaling Behaviors

We study how DSO scales when optimizing \mathcal{L}_{DPO} .

Scaling with training compute. Fig. 6a illustrates the progression of evaluation metrics throughout training. While longer training further enhances the physical stability measure, excessive training with DSO significantly degrades geometric quality. In particular, the model eventually “cheats” by generating a flat structure beneath the 3D asset as a base to prevent it from toppling over.

Scaling with training data. In Fig. 6b, we analyze the impact of training data size on model performance. We train 6 models with identical hyperparameters as our main training run, progressively reducing the amount of data exposed to each model. The smallest dataset used is only $\frac{1}{64}$ of the full dataset, constructed as described in Sec. 4.1. While training on extremely small datasets leads to model collapse, we find that using just $\frac{1}{16}$ of the full dataset (equivalent to 19.2k synthetic 3D models with simulation feedback) already produces results comparable to our main training run. This suggests that aligning state-of-the-art 3D generators with physical soundness requires only a modest amount of preference data. This is promising for aligning other physical properties, such as 3D scene decomposition [61, 104, 105] and part articulation [37, 38, 50], for which obtaining positive samples may be more challenging due to their rarity.

4.6. DSO without Real Data

Our training objective does *not* rely on ground-truth 3D data for supervision. Nevertheless, in our main experiments presented in Sec. 4.2, we used Objaverse renderings as prompts to construct a preference dataset. Here, we show that access to Objaverse models is *not* necessary. We substitute the renderings with object-centric synthetic images to condition the base model TRELIS to generate 3D mod-

Method	Synth.	Loss	Stability		Geometry	
			% Stable \uparrow	Rot. \downarrow	CD \downarrow	F-Score \uparrow
TRELIS [101]	—	—	85.1	14.14 $^\circ$	0.0485	73.12
TRELIS + DSO	\checkmark	\mathcal{L}_{DPO}	93.5	6.92 $^\circ$	0.0483	73.40
TRELIS + DSO	\times	\mathcal{L}_{DPO}	95.1	5.42 $^\circ$	0.0480	73.62
TRELIS + DSO	\checkmark	\mathcal{L}_{DRO}	<u>97.6</u>	<u>3.17$^\circ$</u>	<u>0.0455</u>	<u>76.05</u>
TRELIS + DSO	\times	\mathcal{L}_{DRO}	99.0	1.88$^\circ$	0.0440	76.17

Table 3. DSO can be trained solely on *synthetic* data. The resulting models achieve greater physical soundness than the base model.

els. We then evaluate the physical stability of these generated models using simulation feedback, assigning a binary preference label, which we use for DSO fine-tuning. In more detail, we task GPT-4 [64] to generate 1,000 diverse prompts of detailed object descriptions and use them to prompt FLUX [33], an open-source text-to-image model, to generate synthetic images. We then obtain a total of 64k generated 3D assets, on which we conduct physical simulation as detailed in Sec. 4.1. The performance of the model trained on this dataset is reported in Tab. 3. Despite the larger domain gap, the fine-tuned model generalizes well to the evaluation images and is more likely than the base model TRELIS to generate stable assets under gravity.

5. Conclusion

We presented DSO, a novel framework for generating physically sound 3D objects by leveraging feedback from a physics simulator. Our approach utilizes a dataset of 3D objects labeled with stability scores obtained from the simulator, potentially starting from entirely synthetic images. We fine-tune the base generator using the DPO or DRO objectives, the latter of which we introduced. The resulting *feed-forward* generator is significantly faster and more reliable at producing stable objects compared to test-time optimization methods.

Acknowledgments. This work is supported by a Toshiba Research Studentship, EPSRC SYN3D EP/Z001811/1, and

ERC-CoG UNION 101001212. We thank Minghao Guo and Bohan Wang for providing us with the evaluation set in their work [21], and Mariem Mezghanni for insightful discussions during the early stages of this project. We also thank Zeren Jiang, Minghao Chen, Jinghao Zhou, and Gabrijel Boduljak for helpful suggestions.

References

- [1] Michael Samuel Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *ICLR*, 2023. 2, 5, 7
- [2] Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. *arXiv preprint arXiv:2305.13301*, 2023. 3
- [3] Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 1952. 5
- [4] Eric R Chan, Connor Z Lin, Matthew A Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J Guibas, Jonathan Tremblay, Sameh Khamis, et al. Efficient geometry-aware 3d generative adversarial networks. In *CVPR*, 2022. 2
- [5] Eric R Chan, Marco Monteiro, Petr Kellnhöfer, Jiajun Wu, and Gordon Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *CVPR*, 2021. 2
- [6] David Charatan, Sizhe Lester Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction. In *CVPR*, 2024. 2
- [7] 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, 3, 6, 14, 15
- [8] Yuedong Chen, Haofei Xu, Chuanxia Zheng, Bohan Zhuang, Marc Pollefeys, Andreas Geiger, Tat-Jen Cham, and Jianfei Cai. Mvsplat: Efficient 3d gaussian splatting from sparse multi-view images. In *ECCV*, 2024. 2
- [9] Yuedong Chen, Chuanxia Zheng, Haofei Xu, Bohan Zhuang, Andrea Vedaldi, Tat-Jen Cham, and Jianfei Cai. Mvsplat360: Feed-forward 360 scene synthesis from sparse views. In *NeurIPS*, 2024. 2
- [10] 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 preprint arXiv:2409.12957*, 2024. 2
- [11] Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huang Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, Eli VanderBilt, Aniruddha Kembhavi, Carl Vondrick, Georgia Gkioxari, Kiana Ehsani, Ludwig Schmidt, and Ali Farhadi. Objaverse-XL: A universe of 10M+ 3D objects. In *NeurIPS*, 2023. 2, 5
- [12] 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, 5, 6
- [13] Yu Deng, Jiaolong Yang, Jianfeng Xiang, and Xin Tong. Gram: Generative radiance manifolds for 3d-aware image generation. In *CVPR*, 2022. 2
- [14] Ying Fan, Olivia Watkins, Yuqing Du, Hao Liu, Moonkyung Ryu, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, Kangwook Lee, and Kimin Lee. DPOK: Reinforcement learning for fine-tuning text-to-image diffusion models. In *NeurIPS*, 2023. 3
- [15] Yutao Feng, Yintong Shang, Xuan Li, Tianjia Shao, Chenfanfu Jiang, and Yin Yang. Pie-nerf: Physics-based interactive elastodynamics with nerf. In *CVPR*, 2024. 3
- [16] Hiroki Furuta, Heiga Zen, Dale Schuurmans, Aleksandra Faust, Yutaka Matsuo, Percy Liang, and Sherry Yang. Improving dynamic object interactions in text-to-video generation with ai feedback. *arXiv preprint arXiv:2412.02617*, 2024. 3, 6
- [17] Jun Gao, Tianchang Shen, Zian Wang, Wenzheng Chen, Kangxue Yin, Daiqing Li, Or Litany, Zan Gojcic, and Sanja Fidler. Get3d: A generative model of high quality 3d textured shapes learned from images. *NeurIPS*, 2022. 2
- [18] Ruiqi Gao, Aleksander Holynski, Philipp Henzler, Arthur Brussee, Ricardo Martin-Brualla, Pratul Srinivasan, Jonathan T Barron, and Ben Poole. Cat3d: Create anything in 3d with multi-view diffusion models. *Advances in NeurIPS*, 2024. 2
- [19] Ruiqi Gao, Emiel Hoogeboom, Jonathan Heek, Valentin De Bortoli, Kevin P. Murphy, and Tim Salimans. Diffusion meets flow matching: Two sides of the same coin. 2024. 5
- [20] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *NeurIPS*, 2014. 2
- [21] Minghao Guo, Bohan Wang, Pingchuan Ma, Tianyuan Zhang, Crystal Owens, Chuang Gan, Josh Tenenbaum, Kaiming He, and Wojciech Matusik. Physically compatible 3d object modeling from a single image. *NeurIPS*, 2024. 2, 3, 6, 9, 14
- [22] Junlin Han, Filippos Kokkinos, and Philip Torr. Vfusion3d: Learning scalable 3d generative models from video diffusion models. In *ECCV*, 2024. 2
- [23] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020. 2, 5
- [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 *The Eleventh International Conference on Learning Representations (ICLR)*, 2024. 1, 2
- [25] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *ICLR*, 2022. 5
- [26] Yuanming Hu, Luke Anderson, Tzu-Mao Li, Qi Sun, Nathan Carr, Jonathan Ragan-Kelley, and Frédo Durand. DiffTaichi: Differentiable programming for physical simulation. In *International Conference on Learning Representations (ICLR)*, 2019. 2, 3
- [27] Zixuan Huang, Varun Jampani, Anh Thai, Yuanzhen Li, Stefan Stojanov, and James M Rehg. Shapeclipper: Scalable 3d shape learning from single-view images via geo-

- metric and clip-based consistency. In *CVPR*, 2023. 2
- [28] Zitian Huang, Yikuan Yu, Jiawen Xu, Feng Ni, and Xinyi Le. Pf-net: Point fractal network for 3d point cloud completion. In *CVPR*, 2020. 2
- [29] Tomas Jakab, Ruining Li, Shangzhe Wu, Christian Rupprecht, and Andrea Vedaldi. Farm3D: Learning articulated 3d animals by distilling 2d diffusion. In *3DV*, 2024. 2
- [30] Angjoo Kanazawa, Shubham Tulsiani, Alexei A Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *ECCV*, 2018. 2
- [31] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *NeurIPS*, 2022. 5
- [32] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4), 2023. 3
- [33] Black Forest Labs. Flux. <https://github.com/black-forest-labs/flux>, 2024. 5, 8
- [34] Kimin Lee, Hao Liu, Moonkyung Ryu, Olivia Watkins, Yuqing Du, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, and Shixiang Shane Gu. Aligning text-to-image models using human feedback. *arXiv preprint arXiv:2302.12192*, 2023. 3
- [35] Chengshu Li, Ruohan Zhang, Josiah Wong, Cem Gokmen, Sanjana Srivastava, Roberto Martín-Martín, Chen Wang, Gabriel Levine, Michael Lingelbach, Jiankai Sun, et al. Behavior-1k: A benchmark for embodied ai with 1,000 everyday activities and realistic simulation. In *Conference on Robot Learning (CoRL)*. PMLR, 2023. 2
- [36] Jiahao Li, Hao Tan, Kai Zhang, Zexiang Xu, Fujun Luan, Yinghao Xu, Yicong Hong, Kalyan Sunkavalli, Greg Shakhnarovich, and Sai Bi. Instant3D: Fast text-to-3D with sparse-view generation and large reconstruction model. In *ICLR*, 2024. 2
- [37] Ruining Li, Chuanxia Zheng, Christian Rupprecht, and Andrea Vedaldi. Dragapart: Learning a part-level motion prior for articulated objects. In *ECCV*, 2024. 8
- [38] Ruining Li, Chuanxia Zheng, Christian Rupprecht, and Andrea Vedaldi. Puppet-master: Scaling interactive video generation as a motion prior for part-level dynamics. *arXiv preprint arXiv:2408.04631*, 2024. 8
- [39] Zizhang Li, Dor Litvak, Ruining Li, Yunzhi Zhang, Tomas Jakab, Christian Rupprecht, Shangzhe Wu, Andrea Vedaldi, and Jiajun Wu. Learning the 3d fauna of the web. In *CVPR*, 2024. 2
- [40] 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. 1, 2
- [41] Chen-Hsuan Lin, Chen Kong, and Simon Lucey. Learning efficient point cloud generation for dense 3d object reconstruction. In *proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 32, 2018. 2
- [42] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *ICLR*, 2023. 2, 5, 7
- [43] Jie Liu, Gongye Liu, Jiajun Liang, Ziyang Yuan, Xiaokun Liu, Mingwu Zheng, Xiele Wu, Qiulin Wang, Wenyu Qin, Menghan Xia, et al. Improving video generation with human feedback. *arXiv preprint arXiv:2501.13918*, 2025. 3, 6, 13
- [44] Minghua Liu, Chao Xu, Haian Jin, Linghao Chen, Mukund Varma T, Zexiang Xu, and Hao Su. One-2-3-45: Any single image to 3d mesh in 45 seconds without per-shape optimization. *NeurIPS*, 2023. 1, 6
- [45] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3d object. In *ICCV*, 2023. 1, 2
- [46] Xingchao Liu, Chengyue Gong, et al. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *ICLR*, 2023. 2, 5, 7
- [47] Yuan Liu, Cheng Lin, Zijiao Zeng, Xiaoxiao Long, Lingjie Liu, Taku Komura, and Wenping Wang. Syncdreamer: Generating multiview-consistent images from a single-view image. *arXiv preprint arXiv:2309.03453*, 2023. 2
- [48] Xiaoxiao Long, Yuan-Chen Guo, Cheng Lin, Yuan Liu, Zhiyang Dou, Lingjie Liu, Yuexin Ma, Song-Hai Zhang, Marc Habermann, Christian Theobalt, et al. Wonder3d: Single image to 3d using cross-domain diffusion. In *CVPR*, 2024. 2
- [49] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 5
- [50] Rundong Luo, Haoran Geng, Congyue Deng, Puhao Li, Zan Wang, Baoxiong Jia, Leonidas Guibas, and Siyuan Huang. Physpart: Physically plausible part completion for interactable objects. *arXiv preprint arXiv:2408.13724*, 2024. 8
- [51] Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation. In *CVPR*, 2021. 2
- [52] Miles Macklin. Warp: A high-performance python framework for gpu simulation and graphics. <https://github.com/nvidia/warp>, 2022. NVIDIA GPU Technology Conference (GTC). 2
- [53] Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, and Gavriel State. Isaac gym: High performance gpu-based physics simulation for robot learning. *arXiv preprint arXiv:2108.10470*, 2021. 2, 14
- [54] Luke Melas-Kyriazi, Iro Laina, Christian Rupprecht, Natalia Neverova, Andrea Vedaldi, Oran Gafni, and Filippos Kokkinos. Im-3d: Iterative multiview diffusion and reconstruction for high-quality 3d generation. In *ICLR*, 2024. 2
- [55] Luke Melas-Kyriazi, Iro Laina, Christian Rupprecht, and Andrea Vedaldi. Realfusion: 360deg reconstruction of any object from a single image. In *CVPR*, 2023. 1, 2, 6
- [56] Mariem Mezghanni, Théo Bodrito, Malika Boulkenafed, and Maks Ovsjanikov. Physical simulation layer for accurate 3d modeling. In *CVPR*, 2022. 2, 3
- [57] B Mildenhall, PP Srinivasan, M Tancik, JT Barron, R Ramamoorthi, and R Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 3
- [58] Norman Müller, Yawar Siddiqui, Lorenzo Porzi, Samuel Rota Buló, Peter Kotschieder, and Matthias Nießner. Diffrr: Rendering-guided 3d radiance field diffusion. In *CVPR*, 2023. 2

- [59] Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. Robocasa: Large-scale simulation of everyday tasks for generalist robots. *ArXiv*, 2024. 2
- [60] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. Hologan: Unsupervised learning of 3d representations from natural images. In *ICCV*, 2019. 2
- [61] Junfeng Ni, Yixin Chen, Bohan Jing, Nan Jiang, Bin Wang, Bo Dai, Puhao Li, Yixin Zhu, Song-Chun Zhu, and Siyuan Huang. Phyrecon: Physically plausible neural scene reconstruction. 2024. 2, 3, 8
- [62] Alex Nichol, Heewoo Jun, Prafulla Dhariwal, Pamela Mishkin, and Mark Chen. Point-e: A system for generating 3d point clouds from complex prompts. *arXiv preprint arXiv:2212.08751*, 2022. 2
- [63] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *CVPR*, 2021. 2
- [64] OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 6, 8
- [65] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *NeurIPS*, 2022. 3
- [66] Eunbyung Park, Jimei Yang, Ersin Yumer, Duygu Ceylan, and Alexander C Berg. Transformation-grounded image generation network for novel 3d view synthesis. In *CVPR*, 2017. 2
- [67] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. In *The Eleventh International Conference on Learning Representations (ICLR)*, 2023. 1
- [68] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. DreamFusion: Text-to-3d using 2d diffusion. In *ICLR*, 2023. 2, 6
- [69] Mihir Prabhudesai, Russell Mendonca, Zheyang Qin, Katerina Fragkiadaki, and Deepak Pathak. Video diffusion alignment via reward gradients. *arXiv preprint arXiv:2407.08737*, 2024. 3, 6
- [70] Lingteng Qiu, Guanying Chen, Xiaodong Gu, Qi Zuo, Mutian Xu, Yushuang Wu, Weihao Yuan, Zilong Dong, Liefeng Bo, and Xiaoguang Han. Richdreamer: A generalizable normal-depth diffusion model for detail richness in text-to-3d. In *CVPR*, 2024. 5
- [71] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *NeurIPS*, 2023. 2, 3
- [72] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 5
- [73] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *ICML*, 2015. 3
- [74] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 3
- [75] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. *Advances in NeurIPS*, 33, 2020. 2
- [76] Yichun Shi, Peng Wang, Jianglong Ye, Mai Long, Kejie Li, and Xiao Yang. MVDream: Multi-view diffusion for 3D generation. In *ICLR*, 2024. 1, 2
- [77] J Ryan Shue, Eric Ryan Chan, Ryan Po, Zachary Ankner, Jiajun Wu, and Gordon Wetzstein. 3d neural field generation using triplane diffusion. In *CVPR*, 2023. 2
- [78] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgb-d images. In *ECCV*. Springer, 2012. 3
- [79] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*. pmlr, 2015. 2
- [80] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *ICLR*, 2021. 5
- [81] Trevor Standley, Ozan Sener, Dawn Chen, and Silvio Savarese. image2mass: Estimating the mass of an object from its image. In *CoRL*, 2017. 3
- [82] Stanislaw Szymanowicz, Eldar Insafutdinov, Chuanxia Zheng, Dylan Campbell, João F Henriques, Christian Rupprecht, and Andrea Vedaldi. Flash3d: Feed-forward generalisable 3d scene reconstruction from a single image. In *3DV*, 2025. 2
- [83] Stanislaw Szymanowicz, Christian Rupprecht, and Andrea Vedaldi. Viewset diffusion: (0)-image-conditioned 3d generative models from 2d data. In *ICCV*, 2023. 2
- [84] Stanislaw Szymanowicz, Christian Rupprecht, and Andrea Vedaldi. Splatter image: Ultra-fast single-view 3d reconstruction. In *CVPR*, 2024. 2
- [85] Jiaxiang Tang, Zhaoxi Chen, Xiaokang Chen, Tengfei Wang, Gang Zeng, and Ziwei Liu. Lgm: Large multi-view gaussian model for high-resolution 3d content creation. In *ECCV*. Springer, 2025. 1, 2
- [86] Tencent Hunyuan3D Team. Hunyuan3d 1.0: A unified framework for text-to-3d and image-to-3d generation, 2024. 2
- [87] Tencent Hunyuan3D Team. Hunyuan3d 2.0: Scaling diffusion models for high resolution textured 3d assets generation, 2025. 1, 3
- [88] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012. 2, 5, 14
- [89] Vikram Voleti, Chun-Han Yao, Mark Boss, Adam Letts, David Pankratz, Dmitry Tochilkin, Christian Laforte, Robin Rombach, and Varun Jampani. Sv3d: Novel multi-view synthesis and 3d generation from a single image using latent video diffusion. In *ECCV*, 2024. 1
- [90] Bram Wallace, Meihua Dang, Rafael Rafailov, Linqi Zhou, Aaron Lou, Senthil Purushwalkam, Stefano Ermon, Caiming Xiong, Shafiq Joty, and Nikhil Naik. Diffusion model alignment using direct preference optimization. In *CVPR*, 2024. 2, 3, 4, 5, 6, 13

- [91] Haochen Wang, Xiaodan Du, Jiahao Li, Raymond A Yeh, and Greg Shakhnarovich. Score jacobian chaining: Lifting pretrained 2d diffusion models for 3d generation. In *CVPR*, 2023. 2
- [92] Tianyu Wang, Xiaowei Hu, Chi-Wing Fu, and Pheng-Ann Heng. Single-stage instance shadow detection with bidirectional relation learning. In *CVPR*, 2021. 3
- [93] Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. *NeurIPS*, 2023. 2
- [94] Zhengyi Wang, Yikai Wang, Yifei Chen, Chendong Xiang, Shuo Chen, Dajiang Yu, Chongxuan Li, Hang Su, and Jun Zhu. Crm: Single image to 3d textured mesh with convolutional reconstruction model. In *ECCV*, 2024. 2, 6
- [95] Daniel Watson, William Chan, Ricardo Martin Brullalla, Jonathan Ho, Andrea Tagliasacchi, and Mohammad Norouzi. Novel view synthesis with diffusion models. In *ICLR*, 2023. 1
- [96] Haohan Weng, Tianyu Yang, Jianan Wang, Yu Li, Tong Zhang, CL Chen, and Lei Zhang. Consistent123: Improve consistency for one image to 3d object synthesis. *arXiv preprint arXiv:2310.08092*, 2023. 2
- [97] Christopher Wewer, Kevin Raj, Eddy Ilg, Bernt Schiele, and Jan Eric Lenssen. latentsplat: Autoencoding variational gaussians for fast generalizable 3d reconstruction. In *ECCV*, 2024. 2
- [98] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. *NeurIPS*, 2016. 2
- [99] Shangzhe Wu, Ruining Li, Tomas Jakab, Christian Rupprecht, and Andrea Vedaldi. Magicpony: Learning articulated 3d animals in the wild. In *CVPR*, 2023. 2
- [100] Shangzhe Wu, Christian Rupprecht, and Andrea Vedaldi. Unsupervised learning of probably symmetric deformable 3d objects from images in the wild. In *CVPR*, 2020. 2
- [101] 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 preprint arXiv:2412.01506*, 2024. 1, 2, 3, 5, 6, 7, 8, 13, 14
- [102] 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. 3
- [103] Bo Yang, Hongkai Wen, Sen Wang, Ronald Clark, Andrew Markham, and Niki Trigoni. 3d object reconstruction from a single depth view with adversarial learning. In *ICCVW*, 2017. 2
- [104] Yandan Yang, Baoxiong Jia, Peiyuan Zhi, and Siyuan Huang. Physcene: Physically interactable 3d scene synthesis for embodied ai. In *CVPR*, 2024. 2, 3, 8
- [105] Kaixin Yao, Longwen Zhang, Xinhao Yan, Yan Zeng, Qixuan Zhang, Lan Xu, Wei Yang, Jiayuan Gu, and Jingyi Yu. Cast: Component-aligned 3d scene reconstruction from an rgb image. *arXiv preprint arXiv:2502.12894*, 2025. 8
- [106] Yufei Ye, Shubham Tulsiani, and Abhinav Gupta. Shelf-supervised mesh prediction in the wild. In *CVPR*, 2021. 2
- [107] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *CVPR*, 2021. 2
- [108] Ye Yuan, Jiaming Song, Umar Iqbal, Arash Vahdat, and Jan Kautz. Physdiff: Physics-guided human motion diffusion model. In *ICCV*, 2023. 3
- [109] 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 *CVPR*, 2024. 3
- [110] Guanqi Zhan, Chuanxia Zheng, Weidi Xie, and Andrew Zisserman. What does stable diffusion know about the 3d scene? In *NeurIPS*, 2024. 3
- [111] Biao Zhang, Jiapeng Tang, Matthias Niessner, and Peter Wonka. 3dshape2vecset: A 3d shape representation for neural fields and generative diffusion models. *ACM TOG*, 2023. 2
- [112] Longwen Zhang, Ziyu Wang, Qixuan Zhang, Qiwei Qiu, Anqi Pang, Haoran Jiang, Wei Yang, Lan Xu, and Jingyi Yu. Clay: A controllable large-scale generative model for creating high-quality 3d assets. *ACM TOG*, 2024. 1, 2, 3
- [113] Chuanxia Zheng and Andrea Vedaldi. Free3d: Consistent novel view synthesis without 3d representation. In *CVPR*, 2024. 1, 2
- [114] Zhenglin Zhou, Xiaobo Xia, Fan Ma, Hehe Fan, Yi Yang, and Tat-Seng Chua. Dreamdpo: Aligning text-to-3d generation with human preferences via direct preference optimization. *arXiv preprint arXiv:2502.04370*, 2025. 6
- [115] Jun-Yan Zhu, Zhoutong Zhang, Chengkai Zhang, Jiajun Wu, Antonio Torralba, Josh Tenenbaum, and Bill Freeman. Visual object networks: Image generation with disentangled 3d representations. *NeurIPS*, 2018. 2
- [116] Thomas Hanwen Zhu, Ruining Li, and Tomas Jakab. Dreamhoi: Subject-driven generation of 3d human-object interactions with diffusion priors. *arXiv preprint arXiv:2409.08278*, 2024. 2

A. Details of the Derivations

From Eq. (1) to Eq. (2). As in [90], we introduce a latent oracle O defined on the whole denoising chain $\mathbf{x}_{0:T}$, such that:

$$o(\mathbf{x}_0) = \mathbb{E}_{p_\theta(\mathbf{x}_{1:T}|\mathbf{x}_0)} [O(\mathbf{x}_{0:T})]. \quad (8)$$

Then, starting from Eq. (1), we have:

$$\begin{aligned} & \max_{\theta} \mathbb{E}_{I \sim \mathcal{I}, \mathbf{x}_0 \sim p_\theta(\mathbf{x}_0|I)} [o(\mathbf{x}_0)] - \beta \mathbb{D}_{\text{KL}} [p_\theta(\mathbf{x}_0|I) \| p_{\text{ref}}(\mathbf{x}_0|I)] \\ & \geq \max_{\theta} \mathbb{E}_{I \sim \mathcal{I}, \mathbf{x}_0 \sim p_\theta(\mathbf{x}_0|I)} [o(\mathbf{x}_0)] - \beta \mathbb{D}_{\text{KL}} [p_\theta(\mathbf{x}_{0:T}|I) \| p_{\text{ref}}(\mathbf{x}_{0:T}|I)] \\ & = \max_{\theta} \mathbb{E}_{I \sim \mathcal{I}, \mathbf{x}_{0:T} \sim p_\theta(\mathbf{x}_{0:T}|I)} [O(\mathbf{x}_{0:T})] - \beta \mathbb{D}_{\text{KL}} [p_\theta(\mathbf{x}_{0:T}|I) \| p_{\text{ref}}(\mathbf{x}_{0:T}|I)] \\ & = \beta \max_{\theta} \mathbb{E}_{I \sim \mathcal{I}, \mathbf{x}_{0:T} \sim p_\theta(\mathbf{x}_{0:T}|I)} \left[\log Z(I) - \log \frac{p_\theta(\mathbf{x}_{0:T}|I)}{p_{\text{ref}}(\mathbf{x}_{0:T}|I) \exp(O(\mathbf{x}_{0:T})/\beta)/Z(I)} \right], \end{aligned} \quad (9)$$

where $Z(I) = \sum_{\mathbf{x}_{0:T}} p_{\text{ref}}(\mathbf{x}_{0:T}|I) \exp(O(\mathbf{x}_{0:T})/\beta)$ is a normalizing factor independent of θ . Since

$$\mathbb{E}_{I \sim \mathcal{I}, \mathbf{x}_{0:T} \sim p_\theta(\mathbf{x}_{0:T}|I)} \left[\log \frac{p_\theta(\mathbf{x}_{0:T}|I)}{p_{\text{ref}}(\mathbf{x}_{0:T}|I) \exp(O(\mathbf{x}_{0:T})/\beta)/Z(I)} \right] = \mathbb{D}_{\text{KL}} [p_\theta(\mathbf{x}_{0:T}|I) \| p_{\text{ref}}(\mathbf{x}_{0:T}|I) \exp(O(\mathbf{x}_{0:T})/\beta)/Z(I)] \geq 0 \quad (10)$$

with equality if and only if the two distributions are identical, the optimal $p_\theta^*(\mathbf{x}_{0:T}|I)$ of the right-hand side of Eq. (9) has a unique closed-form solution:

$$p_\theta^*(\mathbf{x}_{0:T}|I) = p_{\text{ref}}(\mathbf{x}_{0:T}|I) \exp(O(\mathbf{x}_{0:T})/\beta)/Z(I). \quad (11)$$

Therefore,

$$O(\mathbf{x}_{0:T}) = \beta \log Z(I) + \beta \log \frac{p_\theta^*(\mathbf{x}_{0:T}|I)}{p_{\text{ref}}(\mathbf{x}_{0:T}|I)} \quad (12)$$

for any $I \in \text{supp}(\mathcal{I})$.

We can then obtain Eq. (2) by plugging Eq. (12) into Eq. (8).

From Eq. (4) to Eq. (5). Since sampling from $p_\theta(\mathbf{x}_{1:T}|\mathbf{x}_0, I)$ is intractable, we follow [90] and replace it with $q(\mathbf{x}_{1:T}|\mathbf{x}_0)$:

$$\begin{aligned} \mathcal{L}_{\text{DRO}} & := \min \mathbb{E}_{I \sim \mathcal{I}, \mathbf{x}_0 \sim \mathcal{X}_I, \mathbf{x}_{1:T} \sim q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \left[(1 - 2o(\mathbf{x}_0)) \log \frac{p_\theta(\mathbf{x}_{0:T}|I)}{p_{\text{ref}}(\mathbf{x}_{0:T}|I)} \right] \\ & = \min \mathbb{E}_{I \sim \mathcal{I}, \mathbf{x}_0 \sim \mathcal{X}_I, \mathbf{x}_{1:T} \sim q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \left[(1 - 2o(\mathbf{x}_0)) \sum_{t=1}^T \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t, I)}{p_{\text{ref}}(\mathbf{x}_{t-1}|\mathbf{x}_t, I)} \right] \\ & = \min T \mathbb{E}_{I \sim \mathcal{I}, \mathbf{x}_0 \sim \mathcal{X}_I, t \sim \mathcal{U}(0, T), \mathbf{x}_t \sim q(\mathbf{x}_t|\mathbf{x}_0), \mathbf{x}_{t-1} \sim q(\mathbf{x}_{t-1}|\mathbf{x}_0, \mathbf{x}_t)} \left[(1 - 2o(\mathbf{x}_0)) \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t, I)}{p_{\text{ref}}(\mathbf{x}_{t-1}|\mathbf{x}_t, I)} \right] \\ & = \min T \mathbb{E}_{I \sim \mathcal{I}, \mathbf{x}_0 \sim \mathcal{X}_I, t \sim \mathcal{U}(0, T), \mathbf{x}_t \sim q(\mathbf{x}_t|\mathbf{x}_0)} \left[(1 - 2o(\mathbf{x}_0)) \left(\mathbb{D}_{\text{KL}} [q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \| p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t, I)] - \mathbb{D}_{\text{KL}} [q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \| p_{\text{ref}}(\mathbf{x}_{t-1}|\mathbf{x}_t, I)] \right) \right]. \end{aligned} \quad (13)$$

Recall that for diffusion models p_θ and p_{ref} , the distributions $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$, $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t, I)$ and $p_{\text{ref}}(\mathbf{x}_{t-1}|\mathbf{x}_t, I)$ are all Gaussian. Therefore, the KL divergence on the right-hand side of Eq. (13) can be re-parameterized analytically using ϵ_θ . After some algebra, and removing all terms independent of θ , this yields Eq. (5).

B. Additional Training Details

All hyperparameters are listed in Tab. 4. We did *not* extensively tune these parameters: the LoRA parameters and the β used in \mathcal{L}_{DPO} follow [43], and the rectified flow noise level t sampling uses the distribution from TRELLIS [101].

Loss formulation	\mathcal{L}_{DRO}	\mathcal{L}_{DPO}
Optimization		
Optimizer	AdamW	AdamW
Learning rate	5×10^{-6}	5×10^{-6}
Learning rate warmup	Linear	Linear
	2,000 iterations	2,000 iterations
Weight decay	0.01	0.01
Effective batch size	48	48
Training iterations	4,000	8,000
Precision	bf16	bf16
LoRA		
Rank	64	64
α	128	128
Dropout	0	0
Miscellaneous		
Rectified flow t sampling	LogitNorm(1, 1)	LogitNorm(1, 1)
β in \mathcal{L}_{DPO}	—	500

Table 4. DSO training details and hyperparameter settings.

Method	Alarm clock		Motorcycle	
	% Stable \uparrow	Rot. \downarrow	% Stable \uparrow	Rot. \downarrow
TRELLIS [101]	67.5	14.14 $^\circ$	44.4	46.53 $^\circ$
TRELLIS + DSO	85.0	5.58$^\circ$	58.1	36.75$^\circ$

Table 5. DSO enhances the model’s ability to generate assets that remain stable under gravity from in-the-wild images of stable objects.

C. Additional Evaluation Details

For evaluation, the 3D models are generated by TRELLIS [101] and DSO fine-tuned TRELLIS using the default setting: 12 sampling steps in stage 1 with classifier-free guidance 7.5 and 12 sampling steps in stage 2 with classifier-free guidance 3. Under this setting, generating *one* model takes 10 seconds on average on an NVIDIA A100 GPU. By contrast, Atlas3D [7] takes 2 hours to generate a model using SDS and PhysComp [21] takes on average 15 minutes to optimize *one* model output by TRELLIS on our hardware.

We use MuJoCo [88] for rigid body simulation for evaluation. The 3D models are assumed to be rigid and uniform in density. We run the simulation for 10 seconds, at which almost all objects have reached the steady state.

D. Additional Results

D.1. Additional Evaluation Results

To demonstrate that the enhanced physical soundness achieved through DSO is not limited to a specific simulation environment, we report the evaluation results in Isaac Gym [53] and under perturbations in Tab. 6. For the evaluation under perturbations, we choose 4 maximum perturbation angles θ_{max} and perform 100 simulation runs with each θ_{max} where the generated 3D models are initially rotated by a random angle $\theta \in (-\theta_{\text{max}}, \theta_{\text{max}})$, following Atlas3D [7]. We then report the average stability rate of the 100 runs. In Tab. 6, TRELLIS post-trained with only MuJoCo feedback via DSO outperforms all baselines under all simulation settings, showing that the improved physical soundness generalizes well to different simulation environments.

Method	MuJoCo				Isaac Gym
	w/o perturbation	$\theta_{\text{max}} = 0.01$	$\theta_{\text{max}} = 0.02$	$\theta_{\text{max}} = 0.04$	$\theta_{\text{max}} = 0.08$ w/o perturbation
<i>Full evaluation set</i> (65 objects)					
TRELLIS [101]	85.1	84.8	84.2	82.5	77.2
Atlas3D [7]	69.4	70.3	70.2	66.3	61.8
TRELLIS + DSO (w/ \mathcal{L}_{DPO})	<u>95.1</u>	<u>94.8</u>	<u>94.1</u>	<u>92.6</u>	<u>88.0</u>
TRELLIS + DSO (w/ \mathcal{L}_{DRO})	99.0	98.8	98.6	97.2	99.6
<i>Partial evaluation set</i> (11 unstable objects)					
TRELLIS [101]	54.5	54.0	53.8	48.5	41.5
TRELLIS + PhysComp [21]	80.3	76.9	76.1	72.6	67.7
TRELLIS + DSO (w/ \mathcal{L}_{DPO})	<u>82.6</u>	<u>82.0</u>	<u>80.7</u>	<u>77.5</u>	<u>67.5</u>
TRELLIS + DSO (w/ \mathcal{L}_{DRO})	95.5	95.4	95.0	93.9	85.4

Table 6. Results evaluated under different simulation settings.



Figure 7. DSO fine-tuned TRELLIS (**ours**) is more likely to generate physically sound 3D objects when conditioned on *real-world* images of challenging categories.

D.2. Additional Comparison with Post-Processing Baselines

In Tab. 7, we compare DSO with a naive post-processing baseline that cuts the mesh flat just above the lowest vertex, following Atlas3D [7]. This method is less effective at stabilizing meshes and significantly degrades geometric quality, as reflected in the higher Chamfer distance (Tab. 7).

Method	Enforcing flat at height z				DSO (Ours)
	$z = 0.05$	$z = 0.1$	$z = 0.15$	$z = 0.2$	
% Stable	94.2	90.5	93.2	<u>95.8</u>	99.0
Chamfer Distance	<u>0.0502</u>	0.0537	0.0591	0.0662	0.0440

Table 7. **Comparison** with post-processing baselines.

D.3. Additional Results on In-the-Wild Images

To assess the generalization of DSO fine-tuned models in generating physically sound 3D objects from real-world images, we curate a set of 30 CC-licensed images for each category: stable alarm clocks and motorcycles supported by kickstands. We select these two categories because the base model, TRELLIS, struggles to generate physically stable versions of these objects. The results are reported in Tab. 5, with *randomly sampled* examples visualized in Fig. 7. As is evident, DSO enhances the model’s ability to generate assets that remain stable under gravity from in-the-wild images of stable objects.

E. Additional Discussions

A deeper analysis of DRO vs. DPO. We further analyze the similarities and differences between \mathcal{L}_{DRO} and \mathcal{L}_{DPO} . Both losses are monotonic functions of $o = \|\epsilon^w - \epsilon_\theta(\mathbf{x}_t^w, t)\|_2^2 - \|\epsilon^w - \epsilon_{\text{ref}}(\mathbf{x}_t^w, t)\|_2^2 - (\|\epsilon^l - \epsilon_\theta(\mathbf{x}_t^l, t)\|_2^2 - \|\epsilon^l - \epsilon_{\text{ref}}(\mathbf{x}_t^l, t)\|_2^2)$. In Fig. 8, we plot each loss (**left**) and its derivative with respect to o (**right**, log-scale). A key difference is that $\frac{d\mathcal{L}_{\text{DRO}}}{do}$ is constant, while $\frac{d\mathcal{L}_{\text{DPO}}}{do}$ decays exponentially as o decreases. As a result, o tends to plateau during optimization of \mathcal{L}_{DPO} . This leads to faster convergence with \mathcal{L}_{DRO} , although extended training may harm performance.

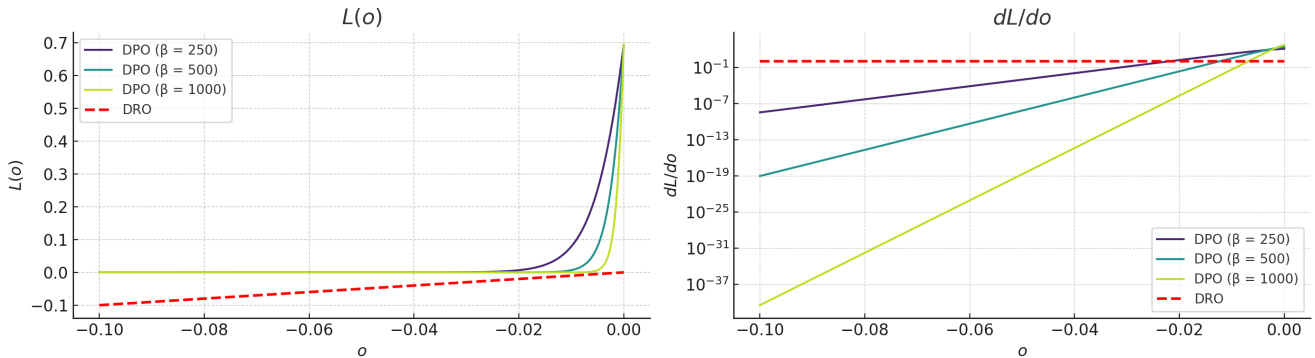


Figure 8. **Plots** of \mathcal{L}_{DRO} and \mathcal{L}_{DPO} and their derivatives.

Scaling behaviors when optimizing \mathcal{L}_{DRO} . In Sec. 4.5, we analyzed how DSO scales when optimizing \mathcal{L}_{DPO} . Here, we present the corresponding scaling behavior for \mathcal{L}_{DRO} . As shown in Tab. 8, performance peaks at 4,000 training steps, after which the geometry quality noticeably degrades—consistent with our earlier analysis. Scaling with training data follows a similar trend to that observed for \mathcal{L}_{DPO} in Fig. 6b.

Training steps	2000	3000	4000	5000
% Stable	91.5	96.9	99.0	98.7
Chamfer D.	0.0473	0.0464	0.0440	0.0853

Table 8. **Scaling behavior with training compute** of \mathcal{L}_{DRO} .

F. Limitations and Future Work

DSO’s self-improving scheme relies on the base model generating at least some positive samples, and hence may be less effective for base models where such samples are rare. DSO opens up new possibilities for integrating physical constraints into generative models, enhancing their applicability in real-world scenarios where adherence to such constraints is crucial.