THE NEXT STEP IN 3D GENERATION POST-TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Post-training has become an effective approach for aligning generative models with human preferences and task-specific objectives, yet its application to 3D generation remains limited. Existing methods primarily target text-to-3D generation, support only specific model architectures, and rely on insufficiently justified reward and rendering designs, often yielding marginal improvements at substantial training cost. To address these limitations, we introduce **Next3DGen**, an unified framework that takes the next step in 3D generation post-training by supporting both text-conditioned and image-conditioned generation post-training across single-stage and multi-stage models. We systematically investigate recent efficient RL algorithms and develop a principled reward pipeline based on explicit criteria for 3D generation quality. For image-to-3D generation, Next3DGen applies complementary rewards to reference-aligned and its antipodal view, improving unobserved regions while preserving input alignment. For multi-stage generators, we propose tree-structured rollouts that jointly optimize geometry and texture through stage-specific credit assignment. Experiments on text-to-3D and image-to-3D benchmarks show that Next3DGen consistently improves generation quality, condition alignment, and 3D consistency over pretrained models, substantially outperforms existing post-training methods, while maintaining training efficiency.

1 INTRODUCTION

Recent advances (Xiang et al., 2025b; Yan et al., 2026) enable high-quality 3D asset creation from text or image, with applications in simulation and extended reality. However, pretrained 3D generators are not explicitly optimized for the qualities required by users and downstream tasks. Since supervised fine-tuning requires costly data and struggles to capture subjective preferences, reward-based post-training offers a promising approach to aligning 3D generators with desired qualities.

Post-training 3D generators, however, poses challenges not addressed by 2D methods (Liu et al., 2025a; Zheng et al., 2026), as a 3D asset must be evaluated holistically for semantic alignment, visual quality, and geometric plausibility. These properties are typically assessed by applying reward models to multi-view renderings, making optimization sensitive to reward selection, viewpoint configuration, and reward aggregation. Sparse views may fail to capture the object's frontal view, while naive averaging can encourage "reward hacking" (Ba et al., 2026), such as duplicated or inconsistent structures. Effective 3D post-training therefore requires not only an efficient optimization algorithm, but also a reward pipeline tailored to the properties of 3D generation.

Despite these requirements, existing approaches leave several important gaps. Nabla-R2D3 (Liu et al., 2025b) and AR3D-R1 (Tang et al., 2025) investigate RL post-training for text-to-3D generation. Although they demonstrate the feasibility of reward-based 3D post-training, both are restricted to text-conditioning and specific model families, excluding image-to-3D generation and modern multi-stage geometry-to-texture models (Xiang et al., 2025b; Hunyuan3D et al., 2025). Their reward pipelines are also either overly simplistic, relying solely on 2D rendering-based rewards, or unnecessarily complex, combining semantically overlapping signals. Consequently, the benefits of complementary reward signals in improving 3D generation remains underexplored.

These limitations raise three fundamental questions. First, how should rewards and viewpoints be designed to improve text-conditioned generation without encouraging view-specific artifacts, while enabling image-conditioned post-training without compromising input alignment? Second, which post-training algorithm delivers strong reward improvements while maintaining high training effi-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

ciency for modern 3D generators? Third, how can post-training be extended from a single generator to a multi-stage pipeline with appropriate credit assignment across geometry and texture stages?

To answer these questions, we introduce Next3DGen, a unified RL post-training framework supporting text- and image-conditioned generation across single- and multi-stage architectures. First, we develop simple, principled, and effective reward pipelines tailored to each conditioning setting. For text-to-3D, we evaluate aesthetics, prompt alignment, and semantic plausibility while examining reward models, rendering configurations, reward combinations, and view aggregation. For image-to-3D, we apply an alignment reward to the reference-aligned view and quality rewards to its antipodal view, separating alignment with the input image from the improvement of unseen regions. Second, we systematically evaluate recent efficient RL algorithms and identify a method that achieves both strong reward improvements and high training efficiency. Finally, we introduce tree-structured rollouts for multi-stage generation, in which each geometry candidate branches into multiple texture variants. Texture rollouts are compared only when they share the same geometry, while geometry rewards are aggregated from their texture children, enabling reliable stage-wise credit assignment.

Our contributions are summarized as follows:

- We propose Next3DGen, a unified RL post-training framework that supports both text-to-3D and image-to-3D generation across single- and multi-stage architectures.
- We develop an efficient, well-grounded reward pipeline tailored to text-to-3D and image-to-3D generation, with complementary reward and viewpoint designs for each setting.
- We present a tree-structured training strategy for multi-stage generators that jointly optimizes geometry and texture models while enabling stage-specific credit assignment.
- We systematically evaluate recent efficient RL algorithms for post-training large 3D models. Extensive experiments show that Next3DGen improves overall generation quality over pretrained models and existing post-training frameworks while remaining training-efficient.

2 Related Work

3D Generative Models. Recent work has developed feed-forward 3D foundation models, progressing from latent diffusion over point clouds and neural fields (Nichol et al., 2022; Zeng et al., 2022) to large-scale 3D DiTs that generate fully textured assets (Hunyuan3D et al., 2025; Zhao et al., 2026; Xiang et al., 2025a). TRELLIS (Xiang et al., 2025b) generates sparse structure and structured latent features in two stages, whereas UniLat3D (Wu et al., 2025) unifies them within a single latent space and flow model. Building on UniLat3D, TIGON (Cen et al., 2026) supports text, image, and joint text-image conditioning. More recently, TripoSplat (Yan et al., 2026) introduces adaptive 3D Gaussian representations for image-to-3D generation. Building on these architectural advances, we focus on improving pretrained 3D generators through reward-driven post-training.

Diffusion Model Post-training. Reward-based post-training aligns diffusion models with human preferences through gradient- or RL-based optimization. Gradient-based methods directly differentiate rewards but incur substantial memory overhead (Wu et al., 2024; Clark et al., 2024; Liu et al., 2025c). RL-based approaches began with DDPO (Black et al., 2024), which formulates denoising as a Markov Decision Process, and were later extended to flow-matching models by FlowGRPO (Liu et al., 2025a). DiffusionNFT (Zheng et al., 2026) and FDFO (McAllister et al., 2026) further improve efficiency through forward-process optimization and paired-rollout finite differences, respectively. However, their efficacy for modern large-scale 3D generators remains underexplored.

Post-Training for 3D Generation. Motivated by the success of diffusion post-training in 2D generation, recent work has extended reward-based alignment to native 3D generators. Such adaptation is important because text-conditioned models often lack fine-grained visual quality, while image-conditioned models remain sensitive to the input viewpoint and may generate implausible unseen regions (Cen et al., 2026). Nabla-R2D3 (Liu et al., 2025b) uses gradient-based reward optimization, whereas AR3D-R1 (Tang et al., 2025) introduces hierarchical GRPO. However, both are limited to text conditioning and specific model families: Nabla-R2D3 has substantial memory overhead and does not readily support multi-stage pipelines, while AR3D-R1 targets only autoregressive models. Both also provide modest gains over their pretrained generators. Our unified framework addresses these limitations across single- and multi-stage architectures, improving text-to-3D quality and alignment while preserving input fidelity and refining unseen regions in image-to-3D generation.

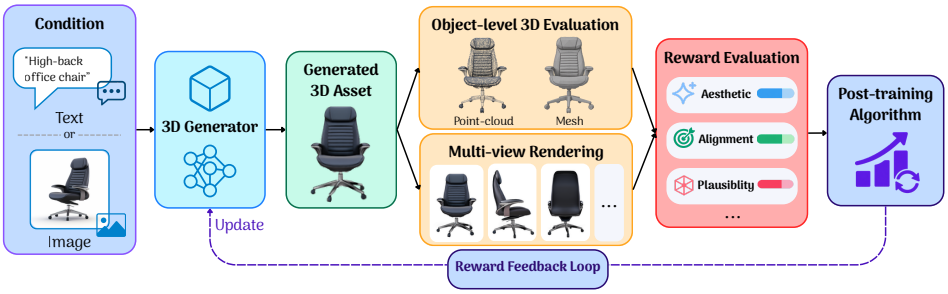

Figure 1: Overview of reward-based post-training for 3D generation.

3 Reward-Based 3D Post-Training

Reward-based post-training aims to improve a pretrained 3D generator through evaluative feedback. As illustrated in Fig. 1, the generator produces a 3D asset conditioned on text or an image. The asset is evaluated as a complete 3D object and through multi-view renderings to obtain reward signals. These signals guide the post-training algorithm in updating the generator, forming a feedback loop.

Existing methods differ mainly in their reward and rendering designs. Nabla-R2D3 (Liu et al., 2025b) does not systematically define 3D generation quality criteria and post-trains with either aesthetic or normal-consistency rewards. Its four approximately orthogonal views also provide limited elevation coverage and may miss informative viewpoints. It also uses mean aggregation, which may encourage "reward hacking" and produce visual artifacts. AR3D-R1 (Tang et al., 2025) defines richer quality criteria but combines several semantically related signals with stage-wise weights, resulting in a complex pipeline that is challenging to transfer across model families. Its reward-model selection and view aggregation are also insufficiently studied, and its six-view configuration may overlook important regions. Moreover, both methods neglect image-conditioned 3D generation.

4 Reward Design

As reward evaluation is central to our 3D post-training framework, we first investigate how rewards and viewpoints should be designed for text- and image-conditioned 3D generation post-training. We then tailor the reward signals, evaluation views, and 3D representations to the requirements of each conditioning setting. Figure 2 summarizes the reward pipelines of Next3DGen.

4.1 Text-to-3D Reward

For text-to-3D generation, we define three complementary rewards targeting distinct aspects: *Aesthetic*, *Alignment*, and *Plausibility*, which jointly promote visually appealing, semantically aligned, and geometrically consistent outputs.

Given n rollouts, let p_i, O_i, and $\{I_{i,v}\}_{v\in\mathcal{V}}$ denote the prompt, generated object, and rendered images of the i-th rollout, respectively, where $i \in \{1,\dots,n\}$ and $\mathcal{V}$ is the set of viewpoints. We use a 2D preference model to evaluate the Aesthetic Reward for each prompt–image pair $(p_i, I_{i,v})$ and a vision-language model (VLM) to evaluate the Alignment Reward using the prompt: *"Does this image match the caption p_i? Answer Yes or No."*

Different views of a 3D object naturally have different reward upper bounds, as frontal views often contain more distinctive appearance and semantic features than side or back views. Averaging rewards across views may therefore encourage "reward hacking" by pressuring the model to improve inherently lower-scoring views (Ba et al., 2026). For instance, the model may duplicate these features across views, resulting in "Janus" artifacts (Hong et al., 2023). We instead apply max aggregation to Aesthetic and Alignment Rewards, assessing each object through its best-scoring view:

$$R_i^{\text{aesth}} = \max_{v\in\mathcal{V}} q_{\text{pref}}(p_i, I_{i,v}), \qquad R_i^{\text{align}} = \max_{v\in\mathcal{V}} a_{\text{VLM}}(p_i, I_{i,v}), \tag{1}$$

where $\mathcal{V}$ denotes the set of camera viewpoints. For VLM model, given the final-token logits z_Y and z_N for `Yes` and `No`, respectively, we compute $a_{\text{VLM}} = 100\,\sigma(z_Y - z_N)$.

Since view-based rewards do not assess the complete 3D structure, we complement them with a Plausibility Reward that evaluates geometric plausibility and consistency from a point-cloud rep-

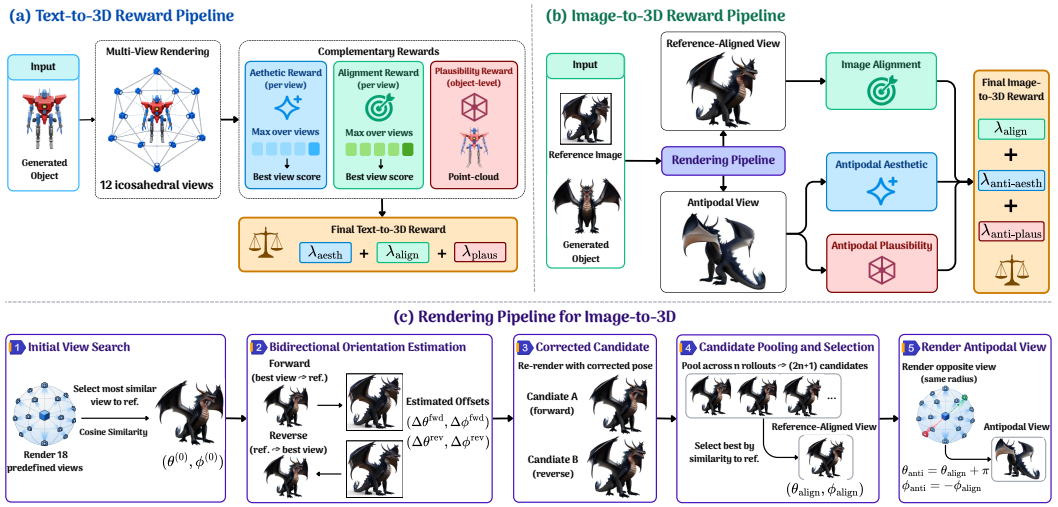

Figure 2: **Overview of Next3DGen's reward and rendering pipelines.** (a) Text-to-3D rewards combine multi-view aesthetics and alignment with point-cloud plausibility. (b) Image-to-3D rewards evaluate reference-view alignment and antipodal-view quality using the rendering pipeline in (c). (c) Rendering pipeline for image-to-3D reward evaluation with camera correction.

resentation. Specifically, we employ a 3D multimodal model to evaluate a normalized point cloud $P(O_i)$ containing 8192 points sampled from O_i, with the prompt: *"Is this object a plausible shape of p_i? Answer Yes or No."* Using a logit-based scoring function, we define:

$$R_i^{\text{plaus}} = s_{\text{3D-MLLM}}\big(p_i, P(O_i)\big). \tag{2}$$

The final text-to-3D reward is the weighted sum:

$$R_i = \lambda_{\text{aesth}} R_i^{\text{aesth}} + \lambda_{\text{align}} R_i^{\text{align}} + \lambda_{\text{plaus}} R_i^{\text{plaus}}. \tag{3}$$

Unless otherwise specified, we instantiate the 2D preference model, vision-language model and 3D multimodal model with HPSv3 (Ma et al., 2025), Qwen2.5-VL (Bai et al., 2025) and PointLLM (Xu et al., 2024), respectively.

4.2 Image-to-3D Reward

For image-to-3D generation, we prioritize improving alignment with the observed reference view while also ensuring that the unobserved side is visually appealing and semantically plausible. We therefore define three rewards: *Image Alignment*, *Antipodal Aesthetic*, and *Antipodal Plausibility*. Image Alignment Reward is applied to the reference-aligned view, while the antipodal rewards are applied only to its antipodal view. This spatial separation helps prevent preference rewards from compromising alignment with the input image.

Given a reference image I_i^{ref} and its reference-aligned rendering I_i^{align}, we employ an image encoder to measure image alignment as the cosine similarity between their embeddings. In our implementation, we use DreamSim (Sundaram et al., 2024) as an image encoder:

$$R_i^{\text{alignment}} = \left\langle E_{\text{D}}\left(I_i^{\text{ref}}\right), E_{\text{D}}\left(I_i^{\text{align}}\right) \right\rangle, \tag{4}$$

For the antipodal view I_i^{anti}, we employ a 2D preference model to evaluate visual quality using the prompt *"Opposite view of a p_i"*. Since optimizing aesthetics alone may produce implausible geometry, we complement this reward with a VLM that assesses plausibility from the ordered image pair $(I_i^{\text{ref}}, I_i^{\text{anti}})$ using the prompt: *"Image 1 is a reference view of a p_i. Image 2 is its candidate antipodal view. Is Image 2 plausible given Image 1? Answer Yes or No."* We define:

$$R_i^{\text{anti-aesth}} = q_{\text{pref}}\left(p_i, I_i^{\text{anti}}\right), \qquad R_i^{\text{anti-plaus}} = a_{\text{VLM}}\left(p_i, I_i^{\text{ref}}, I_i^{\text{anti}}\right). \tag{5}$$

The final image-to-3D reward is the weighted sum, with λ_{img} chosen to prioritize input alignment:

$$R_i = \lambda_{\text{alignment}} R_i^{\text{alignment}} + \lambda_{\text{anti-aesth}} R_i^{\text{anti-aesth}} + \lambda_{\text{anti-plaus}} R_i^{\text{anti-plaus}}. \tag{6}$$

Together, these rewards preserve fidelity to the reference view while improving the appearance and plausibility of the unobserved side.

4.3 Rendering Pipeline

We render each generated 3D asset from multiple viewpoints for view-based reward evaluation. For text-to-3D generation, we follow the T3Bench (He et al., 2024) camera configuration, placing 12 cameras at the vertices of an icosahedron to provide broad elevation coverage. We fix the camera radius and field of view so that the object remains centered and fully visible.

For image-to-3D generation, our reward design requires identifying the rendered view that best aligns with the reference image, allowing separate rewards to be applied to the reference-aligned and unobserved views. However, modern 3D generative models typically operate in a canonical space in which the reference camera pose is unknown. We therefore search for the reference-aligned view within a fixed object-centered coordinate system, placing cameras on a sphere of radius r. For azimuth θ and elevation ϕ, the camera center is:

$$(x, y, z) = r\,(\cos\phi\cos\theta, \cos\phi\sin\theta, \sin\phi)\,. \tag{7}$$

Because objects in the reference images are predominantly upright, we use a look-at camera with a fixed up direction to constrain the in-plane roll. We then render 18 predefined views, comprising 12 icosahedral and six additional axis-aligned views. We then select the view with the highest embedding similarity to the reference image as the initial pose estimate $(\theta_i^{(0)}, \phi_i^{(0)})$.

Although the initial pose $(\theta_i^{(0)}, \phi_i^{(0)})$ is close to the reference pose, a residual mismatch may remain. We therefore use OriAny.V2 (Wang et al., 2026) to estimate pose corrections bidirectionally. The forward query treats the selected rendering as the source and the reference image as the target, while the reverse query swaps their roles. We denote the resulting offsets in the common coordinate system by $(\Delta\theta_i^{\mathrm{fwd}}, \Delta\phi_i^{\mathrm{fwd}})$ and $(\Delta\theta_i^{\mathrm{rev}}, \Delta\phi_i^{\mathrm{rev}})$, respectively. The two corrected pose candidates are:

$$(\theta_{i,A}, \phi_{i,A}) = (\theta_i^{(0)} + \Delta\theta_i^{\mathrm{fwd}}, \phi_i^{(0)} + \Delta\phi_i^{\mathrm{fwd}}), \tag{8}$$

$$(\theta_{i,B}, \phi_{i,B}) = (\theta_i^{(0)} - \Delta\theta_i^{\mathrm{rev}}, \phi_i^{(0)} - \Delta\phi_i^{\mathrm{rev}}). \tag{9}$$

For n rollouts, we pool the resulting $2n$ candidates and render every object from all candidate poses. Let $I_i^{(0)}$ denote the initial rendering for rollout i, and let $I_i^{(k)}$ denote its rendering from the k-th pooled candidate pose, where $k \in \mathcal{C}$ and $|\mathcal{C}| = 2n$. The selected reference-aligned pose is:

$$k_i^* = \arg\max_{k\in\{0\}\cup\mathcal{C}} \left\langle E_{\mathrm{D}}\left(I_i^{\mathrm{ref}}\right), E_{\mathrm{D}}\left(I_i^{(k)}\right)\right\rangle, \qquad (\theta_{\mathrm{align},i}, \phi_{\mathrm{align},i}) = (\theta_i^{(k_i^*)}, \phi_i^{(k_i^*)}). \tag{10}$$

Together, bidirectional inference mitigates the estimator's sensitivity to input order, while shared candidate pooling enables fair comparisons across rollouts and reduces bias from any individual orientation estimate. At the same radius, we render its antipodal view as follows:

$$\theta_{\mathrm{anti},i} = \theta_{\mathrm{align},i} + \pi \pmod{2\pi}, \qquad \phi_{\mathrm{anti},i} = -\phi_{\mathrm{align},i}. \tag{11}$$

5 Multi-stage Adaptation

Modern 3D generation typically involves multiple interdependent stages, such as geometry synthesis followed by texture generation, while rewards are evaluated only on the final asset, making post-training challenging. A straightforward approach is to post-train one stage while keeping the others fixed, but this produces suboptimal results. Optimizing geometry alone may improve shape quality but cannot address appearance-related requirements, potentially causing the model to exploit the objective through implausible structural changes that compensate for fixed textures, as we show in Sec. 7.2. Conversely, optimizing only the texture model produces limited variation because all rollouts share the same geometry, resulting in small reward differences and weak optimization signals. We therefore jointly post-train all stages using a shared rollout tree, with rewards evaluated on the final 3D assets. Further details are provided in Sec. B.

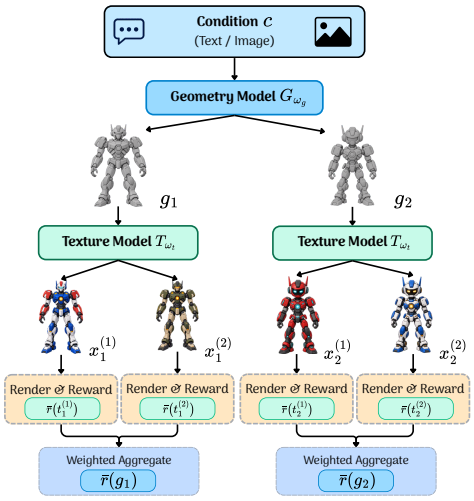

Figure 3: Binary rollouts for a two-stage generator.

Table 1: Comparison of RL algorithms for text-conditioned TIGON post-training without CFG.

Method	CLIP ↑		PickScore ↑		HPSv3++ ↑		Uni3D ↑
	Max	Mean	Max	Mean	Max	Mean	
Base Model	0.332	0.309	21.099	20.633	0.429	-0.142	0.183
Nabla-GFlowNet	0.318	0.293	20.871	20.403	-0.111	-0.562	0.169
DiffusionNFT	0.326	0.299	20.899	20.434	0.174	-0.315	0.176
Flow-GRPO	0.335	0.312	21.363	20.890	1.524	0.608	0.184
FDFO	**0.342**	**0.319**	**21.520**	**21.116**	**3.353**	**2.133**	**0.193**

6 Experiments

6.1 Experimental Setup

Base Models. For most experiments, we use TIGON (Cen et al., 2026), a single-stage model supporting both text and image conditioning, to compare RL algorithms and ablate reward designs within a unified framework. For multi-stage post-training, we adopt TRELLIS (Xiang et al., 2025b), a two-stage model supporting both modalities. We further post-train TripoSplat (Yan et al., 2026), a strong image-to-3D model, to assess the generalizability of our method to a more recent architecture.

Implementation details. We sample 6,273 Objaverse-XL captions (Deitke et al., 2023), and generate a corresponding image for each using Hunyuan-DiT (Huang et al., 2025). We evaluate RL algorithms, reward designs, and ablations on 100 post-training prompts, while comparisons with base models and other 3D generators use 100 prompts held out from post-training. For image-to-3D evaluation, we post-train on 5,646 image-caption pairs and reserve the remaining 627 for evaluation. We post-train each model for 100 epochs and report the final-epoch results.

Text-conditioned Metrics. For each generated object, we render eight views along a fixed camera orbit centered at the origin. We evaluate each view against the input prompt using CLIP similarity (Radford et al., 2021), PickScore (Kirstain et al., 2023), and HPSv3++ (Liu et al., 2026), and report both the mean and maximum scores across views, while Uni3D (Zhou et al., 2023) is employed to assess object-level alignment with the text prompt.

Image-conditioned Metrics. Following the rendering setup in Sec. 4.3, we extract reference-aligned and antipodal views. We measure the CLIP (Radford et al., 2021), DreamSim (Sundaram et al., 2024), and DINOv3 (Siméoni et al., 2025) similarities between the reference-aligned view and the input image, and evaluate the antipodal view using HPSv3++ (Liu et al., 2026). We also measure object-level image alignment with Uni3D (Zhou et al., 2023).

6.2 Exploration of RL Algorithm

We compare four representative post-training algorithms in Tab. 1: Nabla-GFlowNet (Liu et al., 2025c), DiffusionNFT (Zheng et al., 2026), Flow-GRPO (Liu et al., 2025a), and FDFO (McAllister et al., 2026). We evaluate all four methods on text-to-3D post-training under the same per-epoch sampling budget and train each for 100 epochs.

Our results show that FDFO achieves the largest improvement over the pretrained baseline within the same number of training epochs, demonstrating the highest training efficiency. Its group size of two is substantially smaller than the 24–48 samples used by Flow-GRPO and DiffusionNFT, allowing FDFO to cover more prompts and obtain broader training signals under the same per-epoch sampling budget. In contrast, these larger group sizes reduce prompt diversity, while Nabla-GFlowNet is slower and more memory-intensive because each selected denoising timestep requires differentiating the reward through the entire TIGON pipeline, including the flow model, 3D decoder, differentiable rasterizer, and reward models. As FDFO achieves the strongest improvement within the fixed training budget while remaining memory-efficient, we adopt it for all subsequent experiments.

6.3 Generation Results

We compare each post-trained model with its corresponding base model to isolate the improvements introduced by our method, alongside comparisons with existing 3D generation methods. All models are evaluated with and without classifier-free guidance (Ho & Salimans, 2022) (CFG). For methods that produce both Gaussian and mesh outputs, we use decoded Gaussians for evaluation.

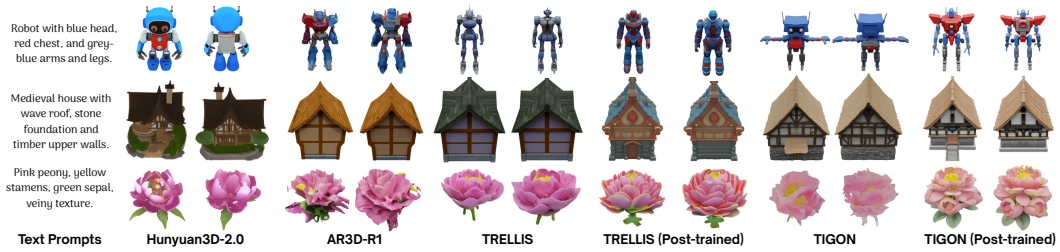

Figure 4: Qualitative comparison of text-to-3D generation models with CFG.

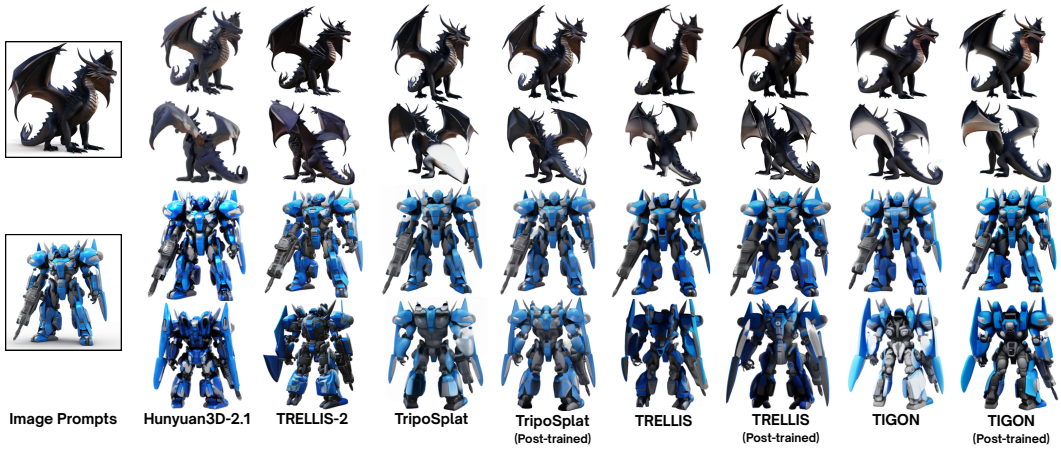

Figure 5: Comparison of image-to-3D generation models with CFG at 512×512 resolution.

Text-Conditioned Generation. We compare our post-trained TRELLIS (Xiang et al., 2025b) and TIGON (Cen et al., 2026) models with their pretrained counterparts, as well as Hunyuan3D-2.0 (Zhao et al., 2026) and AR3D-R1 (Tang et al., 2025). Hunyuan3D-2.0 serves as a strong pretrained text-to-3D baseline, while AR3D-R1 represents an existing RL-post-trained model. As shown in Table 2 and Fig. 4, our method consistently improves both visual quality and prompt alignment of the base models, achieving an unprecedented level of quality in text-to-3D generation.

Image-Conditioned Generation. We evaluate our post-trained TRELLIS (Xiang et al., 2025b), TIGON (Cen et al., 2026), and TripoSplat (Yan et al., 2026) models against their pretrained counterparts and two strong image-to-3D baselines, Hunyuan3D-2.1 (Hunyuan3D et al., 2025) and TRELLIS.2 (Xiang et al., 2025a). Table 3 shows that our framework consistently improves the base models, achieving state-of-the-art performance on most metrics. As illustrated in Fig. 5, post-training also substantially improves visual quality while remaining competitive with strong existing models.

Comparison of Reward Designs. We compare our reward design with those adopted by Nabla-R2D3 (Liu et al., 2025b) and AR3D-R1 (Tang et al., 2025), using TIGON as the shared base model. To isolate the effect of reward design, all variants use the same FDFO algorithm, training budget, and evaluation protocol without CFG. Table 4 and Fig. 6 show our reward design achieves superior aesthetic quality and condition alignment while avoiding multi-face artifacts, demonstrating greater robustness to reward hacking and yielding more plausible and consistent results.

7 Ablation Studies

We ablate two key design choices: reward components and multi-stage post-training. All experiments use the training and evaluation data described in Sec. 6.1. For the reward ablation, we post-train TIGON (Cen et al., 2026) for 50 epochs and report the final-epoch results. For the multi-stage ablation, we post-train TRELLIS (Xiang et al., 2025b) for 100 epochs and report the final-epoch results. More ablation studies are provided in Sec. F.

7.1 Reward Design Ablation

Text-Conditioned Rewards. In Sec. 4.1, we design three rewards that intuitively target distinct aspects of 3D generation quality. To verify this, we evaluate them individually and in combination.

Table 2: Quantitative comparison of text-to-3D generation models with and without CFG. (†: requires an additional image-generation model; *: post-trained using our framework.)

Model	CFG	CLIP ↑		PickScore ↑		HPSv3++ ↑		Uni3D ↑
		Max	Mean	Max	Mean	Max	Mean	
Hunyuan3D-2.0†	✗	0.322	0.283	20.713	20.043	-0.042	-0.693	0.173
	✓	0.324	0.299	20.954	20.308	0.680	-0.307	0.177
AR3D-R1	✗	0.265	0.235	20.105	19.653	-0.905	-1.048	0.141
	✓	0.296	0.263	20.502	20.040	-0.631	-0.879	0.156
TRELLIS	✗	0.271	0.238	20.072	19.604	-0.607	-0.851	0.146
	✓	0.329	0.305	21.092	20.652	0.802	0.096	0.166
TRELLIS*	✗	0.300	0.276	20.707	20.306	0.989	0.212	0.166
	✓	0.333	0.310	21.145	20.734	1.297	0.480	0.171
TIGON	✗	0.315	0.287	20.788	20.299	-0.119	-0.543	0.162
	✓	0.332	0.306	21.165	20.658	0.581	-0.118	0.177
TIGON*	✗	0.331	0.309	21.217	20.816	2.278	1.219	0.178
	✓	**0.342**	**0.318**	**21.379**	**20.949**	**2.476**	**1.400**	**0.183**

Table 3: Quantitative comparison of image-to-3D generation models with and without CFG. (†: evaluated using shaded images of PBR meshes; *: post-trained using our framework.)

Model	CFG	CLIP ↑	DreamSim ↑	DINOv3 ↑	HPSv3++ AV ↑	Uni3D ↑
Hunyuan3D-2.1†	✗	0.408	0.389	0.123	-1.235	0.143
	✓	0.830	0.805	0.801	0.775	0.230
TRELLIS.2†	✗	0.797	0.774	0.728	0.873	0.234
	✓	0.858	0.831	0.827	1.654	**0.295**
TRELLIS	✗	0.865	0.855	0.858	0.884	0.250
	✓	0.897	0.894	0.898	1.009	0.256
TRELLIS*	✗	0.875	0.871	0.876	1.200	0.255
	✓	0.901	0.901	0.904	1.213	0.256
TIGON	✗	0.871	0.857	0.854	0.790	0.242
	✓	0.905	0.899	0.906	1.082	0.249
TIGON*	✗	0.911	0.907	0.910	1.456	0.262
	✓	0.914	0.908	0.914	1.529	0.264
TripoSplat	✗	0.788	0.779	0.752	-0.213	0.187
	✓	0.919	0.923	0.925	0.843	0.224
TripoSplat*	✗	0.917	0.923	0.926	1.398	0.225
	✓	**0.929**	**0.938**	**0.938**	**1.667**	0.234

Table 4: Comparison of reward pipelines for text-conditioned TIGON post-training without CFG.

Reward Pipeline	CLIP ↑		PickScore ↑		HPSv3++ ↑		Uni3D ↑
	Max	Mean	Max	Mean	Max	Mean	
Base Model	0.332	0.309	21.099	20.633	0.429	-0.142	0.183
Nabla-R2D3	0.324	0.300	21.077	20.636	0.677	-0.049	0.168
AR3D-R1	0.329	0.305	21.262	20.973	1.598	0.629	0.183
Next3DGen (Ours)	**0.342**	**0.319**	**21.520**	**21.116**	**3.353**	**2.133**	**0.193**

Table 5 shows that these rewards complement one another in improving generation quality. Aesthetic Reward primarily improves visual metrics, whereas Alignment Reward improves alignment-related metrics. Although Plausibility Reward provides limited improvements on its own, combining it with the other two rewards enables the model to achieve the best performance across all metrics, outperforming their combination without it.

Image-Conditioned Rewards. In Sec. 4.2, we prioritize alignment with the input image while ensuring that unseen views remain visually appealing and semantically plausible. We therefore evaluate the Image Alignment Reward alone and in combination with either or both the Antipodal Aesthetic and Antipodal Plausibility Rewards. Table 6 shows that the Image Alignment Reward ac-



Figure 6: Comparison of reward pipelines for text-conditioned TIGON post-training without CFG.



Figure 7: Stage-1-only vs. joint two-stage TRELLIS post-training with CFG.

Table 5: Text-conditioned reward ablation using max view aggregation. AE, AL, and PL denote Aesthetic, Alignment, and Plausibility Rewards.

Reward	CLIP ↑	HPSv3++ ↑	Uni3D ↑
Base Model	0.332	0.429	0.183
AE	0.332	2.592	0.185
AL	0.345	1.577	**0.189**
PL	0.334	1.411	0.179
AE + AL	0.332	2.401	0.185
AE + AL + PL	**0.348**	**2.645**	**0.189**

Table 6: Image-conditioned reward ablation. IA, AA, and AP denote Image Alignment, Antipodal Aesthetic, and Antipodal Plausibility Rewards.

Reward	DINOv3 ↑	HPSv3++ AV ↑	Uni3D ↑
Base Model	0.854	0.790	0.242
IA	0.901	1.224	0.255
IA + AA	0.891	1.317	0.255
IA + AP	0.902	1.264	0.258
IA + AA + AP	**0.905**	**1.395**	**0.259**

Table 7: Stage-wise post-training ablation for text-conditioned TRELLIS without CFG using max view aggregation.

Training Stage	CLIP ↑	HPSv3++ ↑	Uni3D ↑
Base Model	0.298	-0.203	0.164
Stage 1 Only	0.310	0.690	0.171
Both Stages	**0.322**	**1.753**	**0.182**

Table 8: Stage-wise post-training ablation for image-conditioned TRELLIS without CFG.

Training Stage	DINOv3 ↑	HPSv3++ AV ↑	Uni3D ↑
Base Model	0.858	0.884	0.250
Stage 1 Only	0.865	1.096	0.252
Both Stages	**0.876**	**1.200**	**0.255**

counts for most of the improvement over the base model, while combining all three rewards achieves the best performance across all metrics, confirming their complementary effects.

7.2 Individual-Stage vs. Joint Post-Training

We compare post-training the first stage alone with jointly post-training both stages. We omit second-stage-only results because fixed geometry yields insufficient rollout variation and weak signals for effective optimization. As shown in Table 7 and Fig. 7, first-stage-only post-training improves the base model but can introduce implausible structural changes to compensate for the fixed subsequent stage. In contrast, joint post-training enables coordinated refinement across stages, yielding more plausible results and consistently better performance across all metrics.

8 Conclusion

We introduced Next3DGen, a unified framework for reward-based post-training of text- and image-conditioned 3D generators across single- and multi-stage architectures. Next3DGen combines conditioning-specific reward and rendering pipelines with an efficient post-training algorithm to improve aesthetic quality, condition alignment, and 3D plausibility. For multi-stage generators, our tree-structured rollouts enable joint optimization with stage-specific credit assignment. Extensive experiments across multiple architectures demonstrate consistent improvements over pretrained models and existing post-training approaches. We hope Next3DGen provides a general foundation for aligning future 3D generative models with diverse user preferences and task-specific objectives.

AI USE STATEMENT

In this work, we used generative AI tools to polish author-written text, create small visual elements for figures, and assist with portions of code development. All manuscript content was written before language polishing, and all AI-assisted text, visualizations, and code were reviewed, edited, and verified by the authors. We did not use generative AI for research ideation, literature review, experimental design, data analysis, or drawing scientific conclusions; these tasks were performed by the authors. We take full responsibility for the final content of this work, including all text, claims, code, and visual artifacts produced with the assistance of generative AI.

REFERENCES

Ying Ba, Tianyu Zhang, Mohan Zhou, Yalong Bai, Wenyi Mo, Guiwei Zhang, Bing Su, and Ji-Rong Wen. Pareto-guided optimal transport for multi-reward alignment, 2026. URL https://arxiv.org/abs/2605.13155.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025. URL https://arxiv.org/abs/2502.13923.

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning, 2024. URL https://arxiv.org/abs/2305.13301.

Jiazhong Cen, Jiemin Fang, Sikuang Li, Guanjun Wu, Chen Yang, Taoran Yi, Zanwei Zhou, Zhikuan Bao, Lingxi Xie, Wei Shen, and Qi Tian. Text-image conditioned 3d generation, 2026. URL https://arxiv.org/abs/2603.21295.

Kevin Clark, Paul Vicol, Kevin Swersky, and David J Fleet. Directly fine-tuning diffusion models on differentiable rewards, 2024. URL https://arxiv.org/abs/2309.17400.

Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong 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, 2023. URL https://arxiv.org/abs/2307.05663.

Yuze He, Yushi Bai, Matthieu Lin, Wang Zhao, Yubin Hu, Jenny Sheng, Ran Yi, Juanzi Li, and Yong-Jin Liu. T^3bench: Benchmarking current progress in text-to-3d generation, 2024. URL https://arxiv.org/abs/2310.02977.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance, 2022. URL https://arxiv.org/abs/2207.12598.

Susung Hong, Donghoon Ahn, and Seungryong Kim. Debiasing scores and prompts of 2d diffusion for view-consistent text-to-3d generation, 2023. URL https://arxiv.org/abs/2303.15413.

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, 2021. URL https://arxiv.org/abs/2106.09685.

Minbin Huang, Yanxin Long, Xinchi Deng, Ruihang Chu, Jiangfeng Xiong, Xiaodan Liang, Hong Cheng, Qinglin Lu, and Wei Liu. Dialoggen: Multi-modal interactive dialogue system for multi-turn text-to-image generation, 2025. URL https://arxiv.org/abs/2403.08857.

Team Hunyuan3D, Shuhui Yang, Mingxin Yang, Yifei Feng, Xin Huang, Sheng Zhang, Zebin He, Di Luo, Haolin Liu, Yunfei Zhao, Qingxiang Lin, Zeqiang Lai, Xianghui Yang, Huiwen Shi, Zibo Zhao, Bowen Zhang, Hongyu Yan, Lifu Wang, Sicong Liu, Jihong Zhang, Meng Chen, Liang Dong, Yiwen Jia, Yulin Cai, Jiaao Yu, Yixuan Tang, Dongyuan Guo, Junlin Yu, Hao Zhang, Zheng Ye, Peng He, Runzhou Wu, Shida Wei, Chao Zhang, Yonghao Tan, Yifu Sun, Lin Niu, Shirui

Huang, Bojian Zheng, Shu Liu, Shilin Chen, Xiang Yuan, Xiaofeng Yang, Kai Liu, Jianchen Zhu, Peng Chen, Tian Liu, Di Wang, Yuhong Liu, Linus, Jie Jiang, Jingwei Huang, and Chunchao Guo. Hunyuan3d 2.1: From images to high-fidelity 3d assets with production-ready pbr material, 2025. URL https://arxiv.org/abs/2506.15442.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL https://arxiv.org/abs/1412.6980.

Yuval Kirstain, Adam Polyak, Uriel Singer, Shahbuland Matiana, Joe Penna, and Omer Levy. Pick-a-pic: An open dataset of user preferences for text-to-image generation, 2023. URL https://arxiv.org/abs/2305.01569.

Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Wanli Ouyang. Flow-grpo: Training flow matching models via online rl, 2025a. URL https://arxiv.org/abs/2505.05470.

Qingming Liu, Zhen Liu, Dinghuai Zhang, and Kui Jia. Nabla-r2d3: Effective and efficient 3d diffusion alignment with 2d rewards, 2025b. URL https://arxiv.org/abs/2506.15684.

Yijun Liu, Jie Huang, Zeyue Xue, Yuming Li, Ruizhe He, Haoran Li, Shijia Ge, and Siming Fu. Hpsv3++: Scaling reward models across the full spectrum of diffusion model capabilities, 2026. URL https://arxiv.org/abs/2606.14657.

Zhen Liu, Tim Z Xiao, , Weiyang Liu, Yoshua Bengio, and Dinghuai Zhang. Efficient diversity-preserving diffusion alignment via gradient-informed gflownets. In *ICLR*, 2025c.

Yuhang Ma, Yunhao Shui, Xiaoshi Wu, Keqiang Sun, and Hongsheng Li. Hpsv3: Towards wide-spectrum human preference score, 2025. URL https://arxiv.org/abs/2508.03789.

David McAllister, Miika Aittala, Tero Karras, Janne Hellsten, Angjoo Kanazawa, Timo Aila, and Samuli Laine. Finite difference flow optimization for rl post-training of text-to-image models, 2026. URL https://arxiv.org/abs/2603.12893.

Alex Nichol, Heewoo Jun, Prafulla Dhariwal, Pamela Mishkin, and Mark Chen. Point-e: A system for generating 3d point clouds from complex prompts, 2022. URL https://arxiv.org/abs/2212.08751.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. URL https://arxiv.org/abs/2103.00020.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. Dinov3, 2025. URL https://arxiv.org/abs/2508.10104.

Shobhita Sundaram, Stephanie Fu, Lukas Muttenthaler, Netanel Y. Tamir, Lucy Chai, Simon Kornblith, Trevor Darrell, and Phillip Isola. When does perceptual alignment benefit vision representations?, 2024. URL https://arxiv.org/abs/2410.10817.

Yiwen Tang, Zoey Guo, Kaixin Zhu, Ray Zhang, Qizhi Chen, Dongzhi Jiang, Junli Liu, Bohan Zeng, Haoming Song, Delin Qu, Tianyi Bai, Dan Xu, Wentao Zhang, and Bin Zhao. Are we ready for rl in text-to-3d generation? a progressive investigation, 2025. URL https://arxiv.org/abs/2512.10949.

Zehan Wang, Ziang Zhang, Jiayang Xu, Jialei Wang, Tianyu Pang, Chao Du, HengShuang Zhao, and Zhou Zhao. Orient anything v2: Unifying orientation and rotation understanding, 2026. URL https://arxiv.org/abs/2601.05573.

Guanjun Wu, Jiemin Fang, Chen Yang, Sikuang Li, Taoran Yi, Jia Lu, Zanwei Zhou, Jiazhong Cen, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Xinggang Wang, and Qi Tian. Unilat3d: Geometry-appearance unified latents for single-stage 3d generation, 2025. URL https://arxiv.org/abs/2509.25079.

Xiaoshi Wu, Yiming Hao, Keqiang Sun, Yixiong Chen, Feng Zhu, Rui Zhao, and Hongsheng Li. Human preference score v2: A solid benchmark for evaluating human preferences of text-to-image synthesis, 2023. URL https://arxiv.org/abs/2306.09341.

Xiaoshi Wu, Yiming Hao, Manyuan Zhang, Keqiang Sun, Zhaoyang Huang, Guanglu Song, Yu Liu, and Hongsheng Li. Deep reward supervisions for tuning text-to-image diffusion models, 2024. URL https://arxiv.org/abs/2405.00760.

Jianfeng Xiang, Xiaoxue Chen, Sicheng Xu, Ruicheng Wang, Zelong Lv, Yu Deng, Hongyuan Zhu, Yue Dong, Hao Zhao, Nicholas Jing Yuan, and Jiaolong Yang. Native and compact structured latents for 3d generation, 2025a. URL https://arxiv.org/abs/2512.14692.

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, 2025b. URL https://arxiv.org/abs/2412.01506.

Runsen Xu, Xiaolong Wang, Tai Wang, Yilun Chen, Jiangmiao Pang, and Dahua Lin. Pointllm: Empowering large language models to understand point clouds, 2024. URL https://arxiv.org/abs/2308.16911.

Runjie Yan, Yan-Pei Cao, Peng Wang, Ding Liang, and Yuan-Chen Guo. Generative 3d gaussians with learned density control, 2026. URL https://arxiv.org/abs/2605.16355.

Xiaohui Zeng, Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, and Karsten Kreis. Lion: Latent point diffusion models for 3d shape generation, 2022. URL https://arxiv.org/abs/2210.06978.

Zibo Zhao, Zeqiang Lai, Qingxiang Lin, Yunfei Zhao, Haolin Liu, Shuhui Yang, Yifei Feng, Mingxin Yang, Sheng Zhang, Xianghui Yang, Huiwen Shi, Sicong Liu, Junta Wu, Yihang Lian, Fan Yang, Ruining Tang, Zebin He, Xinzhou Wang, Jian Liu, Xuhui Zuo, Zhuo Chen, Biwen Lei, Haohan Weng, Jing Xu, Yiling Zhu, Xinhai Liu, Lixin Xu, Changrong Hu, Shaoxiong Yang, Song Zhang, Yang Liu, Tianyu Huang, Lifu Wang, Jihong Zhang, Meng Chen, Liang Dong, Yiwen Jia, Yulin Cai, Jiaao Yu, Yixuan Tang, Hao Zhang, Zheng Ye, Peng He, Runzhou Wu, Chao Zhang, Yonghao Tan, Jie Xiao, Yangyu Tao, Jianchen Zhu, Jinbao Xue, Kai Liu, Chongqing Zhao, Xinming Wu, Zhichao Hu, Lei Qin, Jianbing Peng, Zhan Li, Minghui Chen, Xipeng Zhang, Lin Niu, Paige Wang, Yingkai Wang, Haozhao Kuang, Zhongyi Fan, Xu Zheng, Weihao Zhuang, YingPing He, Tian Liu, Yong Yang, Di Wang, Yuhong Liu, Jie Jiang, Jingwei Huang, and Chunchao Guo. Hunyuan3d 2.0: Scaling diffusion models for high resolution textured 3d assets generation, 2026. URL https://arxiv.org/abs/2501.12202.

Kaiwen Zheng, Huayu Chen, Haotian Ye, Haoxiang Wang, Qinsheng Zhang, Kai Jiang, Hang Su, Stefano Ermon, Jun Zhu, and Ming-Yu Liu. Diffusionnft: Online diffusion reinforcement with forward process, 2026. URL https://arxiv.org/abs/2509.16117.

Junsheng Zhou, Jinsheng Wang, Baorui Ma, Yu-Shen Liu, Tiejun Huang, and Xinlong Wang. Uni3d: Exploring unified 3d representation at scale, 2023. URL https://arxiv.org/abs/2310.06773.

A PRELIMINARIES

RL Post-Training Algorithms. Let c denote a prompt and $x_0 \sim p_\omega(x_0 \mid c)$ be the final denoised latent generated by the diffusion model parameterized by ω. This latent is mapped to the final output y via a decoder D, such that $y = D(x_0)$. Given a (potentially black-box) reward function $\mathcal{R}$, the post-training objective is to maximize the expected reward:

$$\max_{\omega} \mathbb{E}_{c,x_0\sim p_\omega(x|c)}\big[\mathcal{R}(D(x_0), c)\big]. \tag{12}$$

FDFO (McAllister et al., 2026) compares paired rollouts indexed by $b \in \{1, 2\}$. The rewards identify a winner w and loser ℓ branch, defining a non-negative gap $\Delta\mathcal{R} = \mathcal{R}(y^{(w)}, c) - \mathcal{R}(y^{(\ell)}, c) \geq 0$. FDFO broadcasts this terminal preference to local cached timesteps t_i by computing a normalized displacement $\overline{\Delta x} = (x_0^{(w)} - x_0^{(\ell)})/(\|x_0^{(w)} - x_0^{(\ell)}\|_2^2 + \varepsilon)$. This constructs a target velocity: $v^{(b)}_{\text{target},k} = v^{(b)}_{\text{ref},k} - s^{(b)}\overline{\Delta x}$, where $v^{(b)}_{\text{ref},k}$ is the rollout model's velocity and $s^{(b)} = 1$ for the winner branch and -1 for the loser. The model $v^{(b)}_{\omega,k}$ is updated by applying a trust-region penalty to a proxy ratio $\rho^{(b,k)}_\omega$:

$$\rho^{(b,k)}_\omega = \exp\left(\left\|v^{(b)}_{\text{target},k} - v^{(b)}_{\text{ref},k}\right\|_2^2 - \left\|v^{(b)}_{\text{target},k} - v^{(b)}_{\omega,k}\right\|_2^2\right), \tag{13}$$

$$\mathcal{L}_{\text{FDFO}}(\omega) = -\Delta\mathcal{R}\,\rho^{(b,k)}_\omega - \frac{\Delta\mathcal{R}}{2\epsilon_{\text{SPO}}}\left(\rho^{(b,k)}_\omega - 1\right)^2. \tag{14}$$

where $\epsilon_{\text{SPO}} > 0$ is a trust-region penalty hyperparameter.

Alternatively, to apply Policy Gradient methods like **FlowGRPO** (Liu et al., 2025a), the reverse process must be framed as a stochastic Markov Decision Process (MDP). FlowGRPO injects stochasticity into the deterministic flow via an SDE under velocity parameterization v_ω:

$$dx_t = \underbrace{\left[v_\omega(x_t, t) + \frac{g_t^2}{2t}\big(x_t + (1-t)v_\omega(x_t, t)\big)\right]}_{f_\omega(x_t,t)} dt + g_t dw_t, \tag{15}$$

where $g_t = a\sqrt{t(1-t)}$ controls the injected noise. Euler discretization yields likelihood-tractable Gaussian transition kernels:

$$\pi_\omega(x_{t-\Delta t}|x_t) = \mathcal{N}\Big(x_t + f_\omega(x_t, t)\Delta t,\ g_t^2\Delta t I\Big), \tag{16}$$

which enables direct step-by-step optimization using standard policy gradients.

B MULTI-STAGE ADAPTATION

Hierarchical rollout construction. We first consider a two-stage generator comprising a geometry model G_{ω_g} and a texture model T_{ω_t}. Figure 3 illustrates its hierarchical rollout tree using groups of size two. Given a condition c, the geometry model generates N_g candidate geometries:

$$g_i \sim G_{\omega_g}(\cdot \mid c), \qquad i \in \{1, \ldots, N_g\}. \tag{17}$$

For each geometry candidate g_i, the texture model generates N_t textured variants:

$$t_i^{(j)} \sim T_{\omega_t}(\cdot \mid c, g_i), \qquad j \in \{1, \ldots, N_t\}. \tag{18}$$

The resulting $N_g N_t$ final assets form a hierarchical rollout tree. The first level captures geometry variation, while the second captures texture variation conditioned on each geometry. The final assets and their terminal rewards are defined as:

$$O_i^{(j)} = \big(g_i, t_i^{(j)}\big), \qquad r_i^{(j)} = R\big(O_i^{(j)}, c\big). \tag{19}$$

Stage-wise credit assignment. Given the hierarchical rollouts above, each texture variant $t_i^{(j)}$ directly inherits the terminal reward of its corresponding final asset:

$$\bar{r}\big(t_i^{(j)}\big) = r_i^{(j)}. \tag{20}$$

To assign credit to the preceding geometry stage, we aggregate the rewards of the N_t texture variants generated from each geometry g_i:

$$\bar{r}(g_i) = \sum_{j=1}^{N_t} w_i^{(j)} \bar{r}\big(t_i^{(j)}\big), \tag{21}$$

where the weights are obtained by normalizing the stored log-probabilities of the texture rollouts:

$$w_i^{(j)} = \frac{\exp\Big(\log \pi_{\theta_{\text{old}}}\big(t_i^{(j)} \mid c, g_i\big)\Big)}{\displaystyle\sum_{i=1}^{N_t} \exp\Big(\log \pi_{\theta_{\text{old}}}\big(t_i^{(j)} \mid c, g_i\big)\Big)}, \qquad \sum_{j=1}^{N_t} w_i^{(j)} = 1. \tag{22}$$

Importantly, RL optimization compares only sibling rollouts that share the same parent. At the texture stage, the comparison group associated with geometry g_i is:

$$\mathcal{T}_i = \Big\{ t_i^{(j)} \Big\}_{j=1}^{N_t}, \tag{23}$$

This restriction is essential because texture rollouts conditioned on different geometries may have mismatched latent shapes, making their trajectories incompatible for direct comparison under DPO-style objectives. Comparing only sibling rollouts ensures compatible latent representations and enables fair comparisons among candidates that share the same condition and all preceding-stage decisions. For instance, FDFO forms paired texture-stage update directions only between siblings:

$$\Delta r_i^{(1,2)} = r_i^{(1)} - r_i^{(2)}, \qquad t_i^{(1)}, t_i^{(2)} \in \mathcal{T}_i, \tag{24}$$

In general, terminal rewards are recursively propagated to preceding stages through a normalized, probability-weighted sum over each parent's immediate children. Accordingly, the rollout tree for an L-stage generator has depth L, and the number of children per node is determined by the group size of the underlying RL algorithm.

C More Implementation Details

C.1 Training Details

We provide additional implementation details for each base model, including model-specific sampling configurations, optimization hyperparameters, FDFO hyperparameters, and reward weights.

Model Configuration. We apply LoRA (Hu et al., 2021) to all linear layers of TIGON (Cen et al., 2026), TRELLIS (Xiang et al., 2025b), and TripoSplat (Yan et al., 2026), using rank 16, scaling factor 32, zero dropout, and bf16 precision. TIGON uses 35 sampling steps with a churn weight of 0.0025. TRELLIS uses 25 sampling steps per stage, for a total of 50 steps, with a churn weight of 0.004, while TripoSplat uses 20 sampling steps with a churn weight of 0.005. All models are trained with a CFG (Ho & Salimans, 2022) scale of 1.0 and a KL-divergence weight of 0.04.

Optimization. We use the Adam optimizer (Kingma & Ba, 2017) with a learning rate of 1×10^{-4}, $\beta_1 = 0.9$, $\beta_2 = 0.999$, weight decay of 1×10^{-4}, and $\epsilon = 1 \times 10^{-8}$. We set the clipping range to 0.03 and clip the gradient norm to 1.0. Each epoch contains 128 rollout pairs and 4 gradient-update steps.

Rendering Configuration. For reward evaluation, we render each view at a resolution of 512×512 using a camera radius of 2 and a field of view of $40°$.

Reward Weights. We select reward weights based on the scales of reward differences produced by the corresponding reward models. For text-to-3D post-training, we use:

$$\lambda_{\text{aesth}} = 1.0, \qquad \lambda_{\text{align}} = 0.1, \qquad \lambda_{\text{plaus}} = 0.1.$$

For image-to-3D post-training, we assign a larger weight to the Image Alignment Reward to prioritize fidelity to the input image, while accounting for differences in reward-difference scales:

$$\lambda_{\text{alignment}} = 10.0, \qquad \lambda_{\text{anti-aesth}} = 0.05, \qquad \lambda_{\text{anti-plaus}} = 0.005.$$

C.2 Computational Cost

FDFO Training Cost. For all models, we adjust the per-GPU batch size to maintain approximately 60–70 GB of VRAM usage, resulting in a batch size of 2 per GPU. We train each model for 100 epochs on 8 NVIDIA A100 80 GB GPUs, requiring approximately 2 days for TripoSplat, 3 days for TIGON, and 4 days for TRELLIS. The difference in training time primarily arises from the number of sampling steps required by each model.

RL Baseline Training Cost. In Sec. 6.2, we train TIGON with FlowGRPO (Liu et al., 2025a) and DiffusionNFT (Zheng et al., 2026) using a group size of 32 and 8 prompts per epoch, yielding 256 samples per epoch to match the sampling budget of FDFO (McAllister et al., 2026). Both methods use a batch size of 2 per GPU and require approximately 3 days for 100 epochs. Under the same sampling budget and batch size, training with Nabla-GFlowNet (Liu et al., 2025c) requires approximately 6 days for 100 epochs.

D More Experiment Details

We provide additional experimental details, including the rendering protocols for text-to-3D and image-to-3D evaluation, Uni3D inference procedure, and inference configurations of each baseline.

D.1 Evaluation Protocol

Text-to-3D Evaluation. As described in Sec. 6.1, we render eight views with yaw angles uniformly spaced at $45°$. We use a fixed pitch of $30°$, a camera radius of 2, a field of view of $40°$, and a resolution of 512×512.

Image-to-3D Evaluation. We follow the rendering pipeline described in Sec. 4.3, using a camera radius of 2, a field of view of $40°$, and a resolution of 512×512.

Uni3D Evaluation. To compute the Uni3D score (Zhou et al., 2023), we uniformly sample 10,000 points from each generated 3D asset, encode the resulting point cloud, and compare its representation with the corresponding text or image CLIP embedding (Radford et al., 2021).

D.2 Baseline Configurations

We evaluate each baseline using its default inference configuration. For evaluations without classifier-free guidance (CFG), We set the guidance scales of all modules within each 3D generator to 1.0, while leaving those of external modules, such as text-to-image models and texture-baking modules, unchanged.

Text-to-3D Baselines. For Hunyuan3D-2.0 (Zhao et al., 2026), we run Hunyuan-DiT (Huang et al., 2025) for text-to-image generation using 25 steps and a CFG scale of 5.0. Shape generation uses 50 steps with a CFG scale of 5.0. Texture generation uses 50 delight steps with a CFG scale of 1.5, followed by 30 multiview painting steps with a guidance scale of 2.0. For AR3D-R1 (Tang et al., 2025) and TRELLIS (Xiang et al., 2025b), both generation stages use 25 steps with a CFG scale of 7.5. TIGON uses 35 steps with a CFG scale of 3.0.

Image-to-3D Baselines. Hunyuan3D-2.1 (Hunyuan3D et al., 2025) uses 50 diffusion steps with a CFG scale of 5.0 for shape generation. For texture generation, its multiview diffusion stage uses 15 steps with a CFG scale of 3.0. TRELLIS.2 (Xiang et al., 2025a) uses 12 steps for each of its sparse-structure, shape-SLat, and texture-SLat stages, with guidance scales of 7.5, 7.5, and 1.0, respectively. For TRELLIS (Xiang et al., 2025b), both generation stages use 25 steps with a CFG scale of 5.0. TIGON uses 35 steps with a CFG scale of 3.0, while TripoSplat (Yan et al., 2026) uses 20 steps with a CFG scale of 3.0.

E More Experiments

Because the evaluation prompts in Tab. 2 were held out from post-training, we additionally evaluate on prompts used during post-training to assess the reward optimization capability of our framework. As shown in Tab. 9, post-training consistently improves all evaluated metrics across both TRELLIS

Table 9: Quantitative comparison of text-to-3D generation models with and without CFG used during post-training. (†: requires an additional image-generation model; *: post-trained using our framework.)

Model	CFG	CLIP ↑		PickScore ↑		HPSv3++ ↑		Uni3D ↑
		Max	Mean	Max	Mean	Max	Mean	
Hunyuan3D-2.0†	✗	0.323	0.289	21.017	20.345	0.409	-0.474	0.175
	✓	0.329	0.299	21.209	20.614	1.113	-0.044	0.182
AR3D-R1	✗	0.288	0.259	20.583	20.117	-0.414	-0.763	0.158
	✓	0.317	0.288	20.940	20.488	-0.070	-0.521	0.172
TRELLIS	✗	0.298	0.271	20.637	20.181	-0.203	-0.654	0.164
	✓	0.338	0.315	21.472	21.074	1.307	0.437	0.181
TRELLIS*	✗	0.322	0.300	21.119	20.703	1.753	0.808	0.182
	✓	<u>0.344</u>	<u>0.321</u>	21.501	21.103	1.950	0.971	0.185
TIGON	✗	0.332	0.309	21.099	20.633	0.429	-0.142	0.183
	✓	0.343	0.319	21.391	20.912	1.217	0.341	0.186
TIGON*	✗	0.342	0.319	<u>21.520</u>	<u>21.116</u>	<u>3.353</u>	**2.133**	<u>0.193</u>
	✓	**0.348**	**0.326**	**21.619**	**21.217**	**3.386**	<u>2.129</u>	**0.195**

Table 10: Ablation of reward model selection for text-conditioned TIGON post-training without CFG.

Reward Model	CLIP ↑		PickScore ↑		HPSv3++ ↑		Uni3D ↑
	Max	Mean	Max	Mean	Max	Mean	
Base Model	0.298	0.271	20.637	20.181	-0.203	-0.654	0.164
PickScore	**0.349**	**0.327**	**21.564**	**21.133**	2.223	1.158	0.189
HPSv2.1	0.337	0.313	21.451	21.003	2.543	1.403	0.181
HPSv3++	0.340	0.316	21.500	21.048	2.884	1.730	0.187
HPSv3	<u>0.342</u>	<u>0.319</u>	<u>21.520</u>	<u>21.116</u>	**3.353**	**2.133**	**0.193**

and TIGON. These consistent improvements across both exposed and held-out prompts demonstrate that our framework effectively optimizes 3D generators toward the desired reward objectives while enhancing qualities valued by users and downstream applications.

F More Ablation Studies

We conduct additional ablations on three key design choices in our framework: reward model selection, rendering configuration, and view aggregation. For these ablations, we post-train TIGON (Cen et al., 2026) for 50 epochs and report the final-epoch results. These experiments provide further insights into effective reward-based post-training for 3D generation. We also examine the effect of reward-based post-training on each 3D representation.

F.1 Reward Model Selection

We ablate the Aesthetic Reward model selection for text-to-3D post-training by comparing PickScore (Kirstain et al., 2023), HPSv2.1 (Wu et al., 2023), HPSv3++ (Liu et al., 2026), and HPSv3 (Ma et al., 2025). As shown in Table 10, PickScore achieves the highest CLIP and PickScore scores. However, these metrics may favor closely related reward models: PickScore is evaluated by the same model and may also benefit on CLIP because it builds upon CLIP (Radford et al., 2021). We therefore complement the quantitative results with the qualitative comparison in Fig. 8. HPSv3 produces more visually detailed assets and delivers the strongest overall performance, including the highest HPSv3++ and Uni3D scores. Based on both quantitative and qualitative evidence, we adopt HPSv3 as the default model for Aesthetic Reward, while retaining compatibility with alternative reward models.



Figure 8: **Qualitative comparison of Aesthetic Reward model selection for text-to-3D post-training without CFG.** HPSv3 produces more visually detailed and consistent 3D assets than the alternative reward models.

Table 11: Ablation of similarity models for reference-aligned view selection and Image Alignment Reward in image-conditioned TIGON post-training without CFG.

Rendering Configuration	CLIP ↑	DreamSim ↑	DINOv3 ↑	Uni3D ↑
Base Model	0.865	0.855	0.858	0.250
DINOv3	<u>0.904</u>	<u>0.895</u>	**0.904**	<u>0.254</u>
DreamSim	**0.905**	**0.898**	<u>0.901</u>	**0.255**

In contrast, we find that model selection for the other reward components, including VLMs, image encoders, and 3D multimodal language models, has a limited impact on final performance. For example, Tab. 11 shows that DreamSim and DINOv3 achieve comparable results. We therefore choose the models used in our current setup based primarily on their availability and implementation practicality, while our framework remains compatible with alternative models.

F.2 Rendering Configuration

We examine how rendering configurations affect reward-based post-training in both text-conditioned and image-conditioned settings. For text-to-3D generation, we study the effect of view coverage, while for image-to-3D generation, we evaluate the importance of camera correction.

For text-conditioned post-training, we compare our 12-icosahedron-view setup with the six uniformly spaced orbit views used by AR3D-R1 (Tang et al., 2025), keeping the reward design fixed. Table 12 shows that our setup performs better across all metrics, benefiting from broader elevation and azimuth coverage that reduces the risk of missing frontal views under extreme orientations. For image-conditioned post-training, we evaluate our camera correction technique against a variant using the same rendered views and Image Alignment Reward but treating the initially most similar view as the reference-aligned view. Table 13 demonstrates the effectiveness of camera correction.

F.3 View Aggregation

As discussed in Sec. 4.1, different views naturally have different reward upper bounds, and averaging rewards across views may encourage reward hacking and Janus artifacts. We therefore compare mean aggregation, max aggregation, and an alternating Max-Mean strategy that switches between the two across successive epochs.

Table 12: Ablation of rendering setups for text-conditioned TIGON post-training without CFG.

Rendering Setup	CLIP ↑		PickScore ↑		HPSv3++ ↑		Uni3D ↑
	Max	Mean	Max	Mean	Max	Mean	
Base Model	0.298	0.271	20.637	20.181	-0.203	-0.654	0.164
AR3D-R1 Views	0.339	0.317	21.523	21.069	2.267	1.147	0.184
12 Icosahedron Views (Ours)	**0.348**	**0.326**	**21.592**	**21.145**	**2.645**	**1.432**	**0.189**

Table 13: Ablation of camera correction for image-conditioned TIGON post-training without CFG. Without camera correction, we select the maximum DreamSim score over 18 rendered views.

Rendering Configuration	CLIP ↑	DreamSim ↑	DINOv3 ↑	Uni3D ↑
Base Model	0.865	0.855	0.858	0.250
w/o Camera Correction	0.893	0.885	0.887	0.251
w/ Camera Correction (Ours)	**0.905**	**0.898**	**0.901**	**0.255**

Table 14: Ablation of view aggregation strategies for text-conditioned TIGON post-training without CFG.

View Aggregation	CLIP ↑		PickScore ↑		HPSv3++ ↑		Uni3D ↑
	Max	Mean	Max	Mean	Max	Mean	
Base Model	0.298	0.271	20.637	20.181	-0.203	-0.654	0.164
Mean	0.230	0.201	19.857	19.495	0.087	-0.339	0.102
Max-Mean	0.328	0.306	21.319	20.895	2.319	1.264	0.176
Max	**0.348**	**0.326**	**21.592**	**21.145**	**2.645**	**1.432**	**0.188**

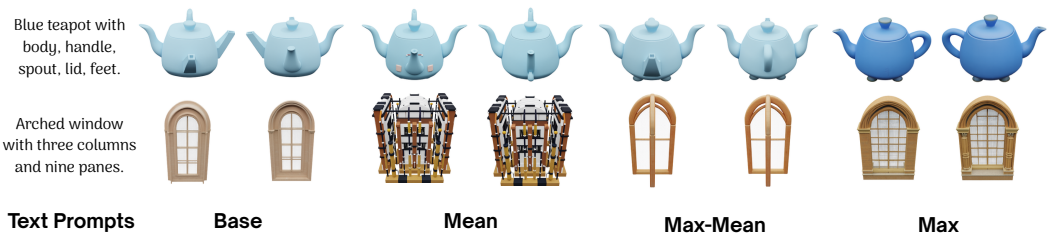

Figure 9: **Qualitative comparison of view aggregation strategies for text-to-3D post-training without CFG.** Max aggregation mitigates the Janus artifacts induced by mean aggregation.

As shown in Table 14 and Fig. 9, mean aggregation substantially degrades performance across all metrics, even relative to the base model. Alternating Max-Mean improves upon the base model but remains consistently inferior to max aggregation, indicating that applying mean aggregation in only half of the epochs still degrades generation quality despite alternating with max aggregation. Max aggregation achieves the best performance across all metrics, supporting our hypothesis that focusing on the most informative view provides a more effective training signal. We therefore adopt max aggregation by default.

F.4 3D Representations

We examine how post-training affects different 3D representations in generators equipped with multiple representation-specific decoders, such as TRELLIS (Xiang et al., 2025b) and TIGON (Cen et al., 2026). As shown in Fig. 10, a generator post-trained using the 3DGS decoder produces visually and semantically consistent outputs when evaluated using the mesh decoder. This suggests that post-training improves the shared latent representation rather than overfitting to the representation used for reward computation. Since the decoders remain frozen during post-training, our method

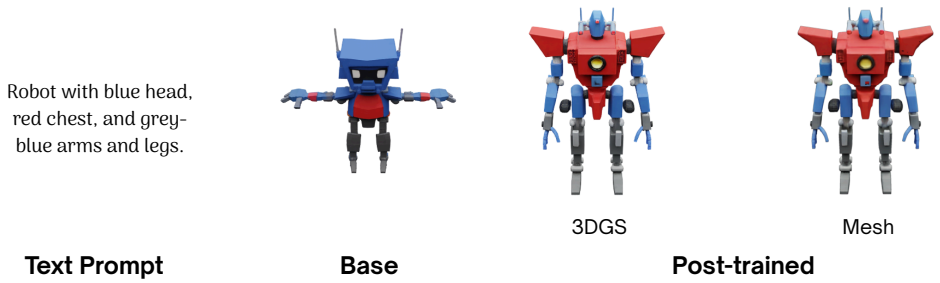

Figure 10: **Effect of post-training across 3D representations.** Models post-trained with the 3DGS decoder retain consistent improvements when evaluated with the mesh decoder.

does not depend on a specific 3D representation. We can therefore use the representation that enables the most efficient rendering and reward computation.

G Limitations and Future Work

Despite the strong results, our experiments use a relatively small post-training dataset due to computational constraints, and we have not yet evaluated scalability to larger prompt sets, such as 20K or 100K prompts. Multi-stage adaptation also introduces considerable computational and memory costs: with a group size of n at each stage, a L-stage rollout tree produces n^L terminal assets, making post-training expensive for RL algorithms that require large group sizes and for generators with three or more stages. Moreover, keeping multiple stages and rollouts in memory may require additional memory management or parameter offloading, potentially increasing training time. Future work could explore larger datasets and more efficient rollout and stage-training strategies, such as selective branching. Finally, our reward design does not explicitly assess material quality for Physically Based Rendering (PBR), such as reflectance, roughness, and lighting-dependent appearance, which we leave for future exploration.

H More Qualitative Results

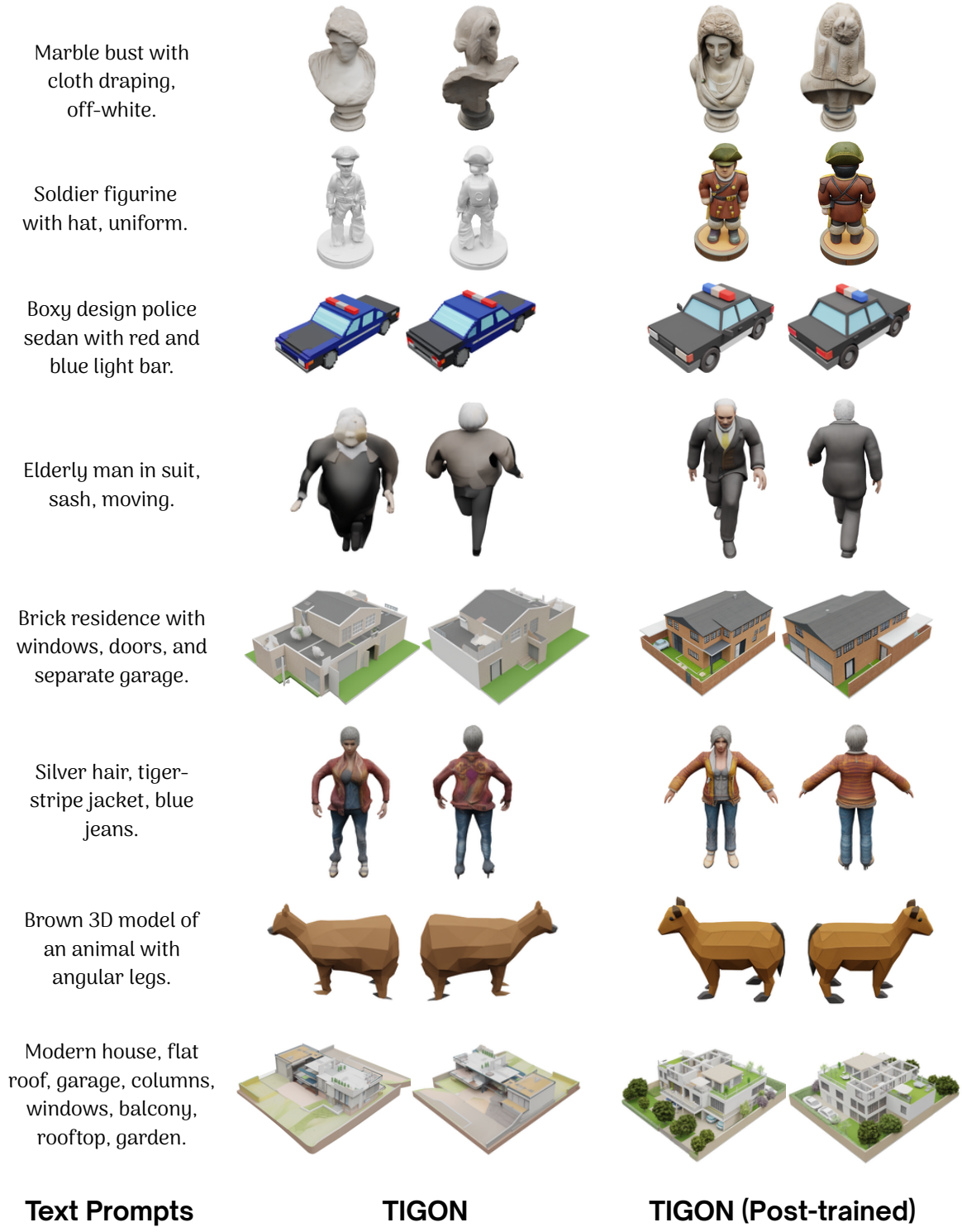

Figure 11: Qualitative results of post-trained text-conditioned TIGON without CFG.

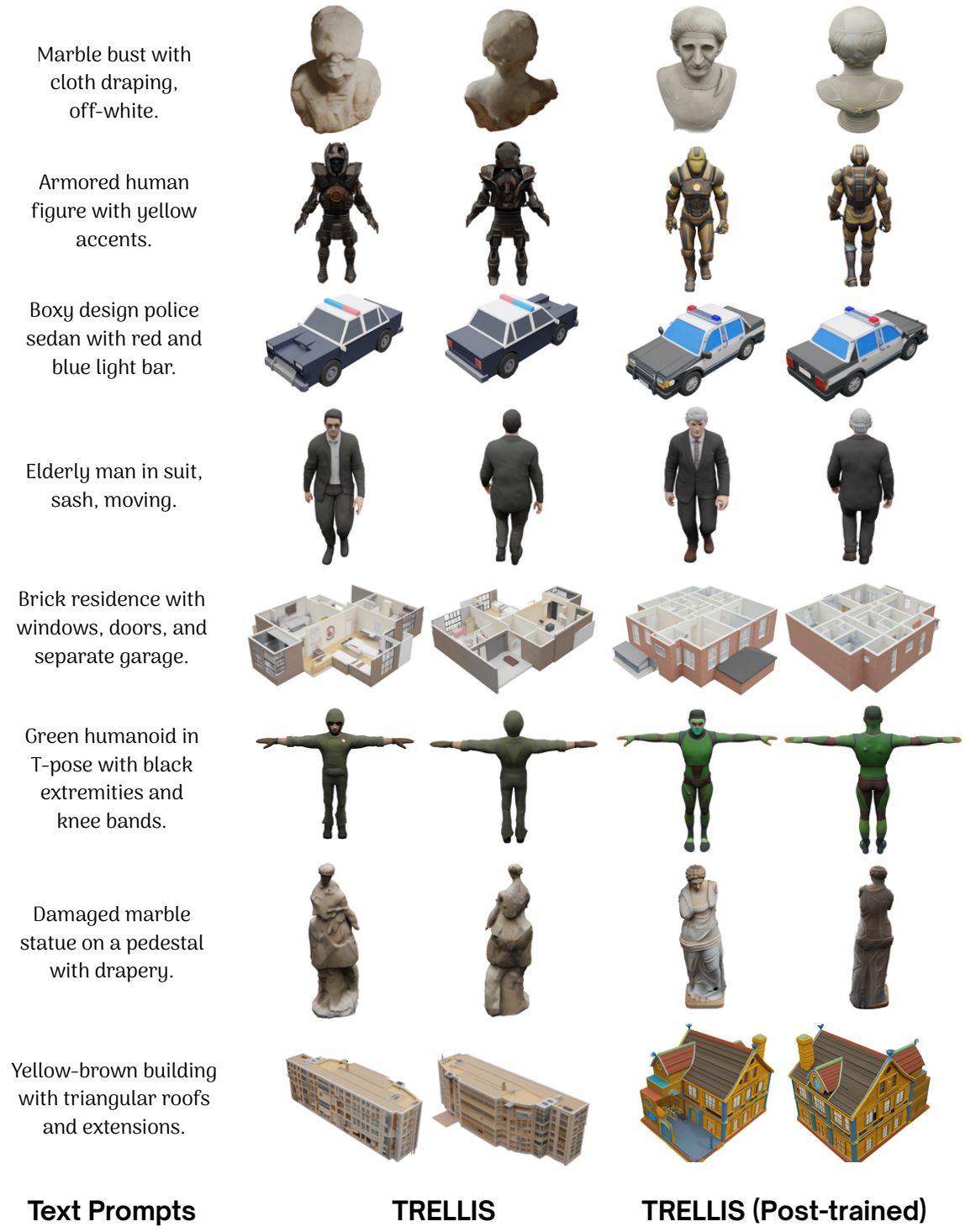

Figure 12: Qualitative results of post-trained text-conditioned TRELLIS without CFG.

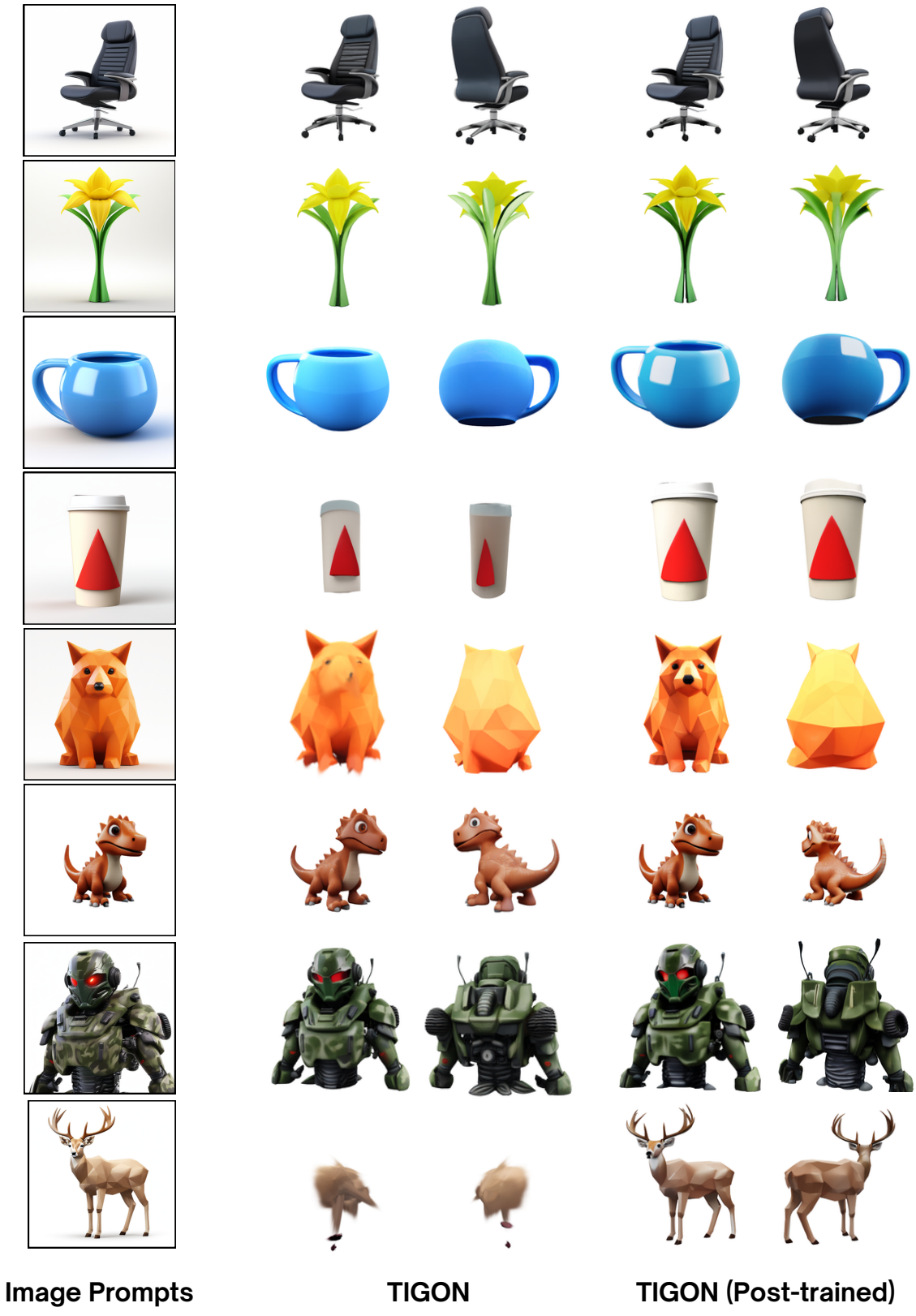

Figure 13: Qualitative results of post-trained image-conditioned TIGON without CFG.



Figure 14: Qualitative results of post-trained image-conditioned TripoSplat without CFG.

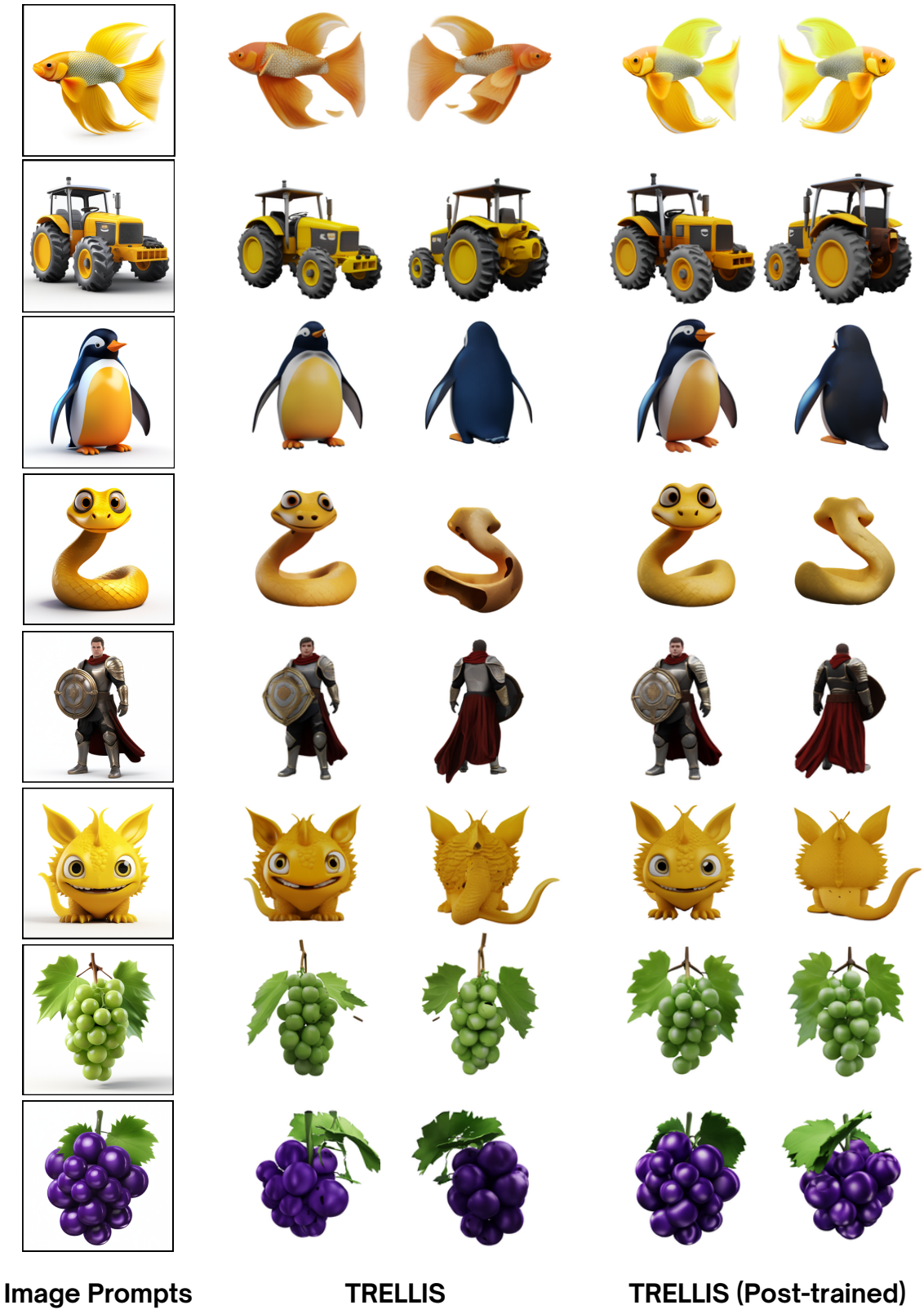

Figure 15: Qualitative results of post-trained image-conditioned TRELLIS without CFG.