BLOCK3D: EFFICIENT TEXT-TO-3D GENERATION VIA BLOCK-WISE DIFFUSION

Anonymous authors
Paper under double-blind review

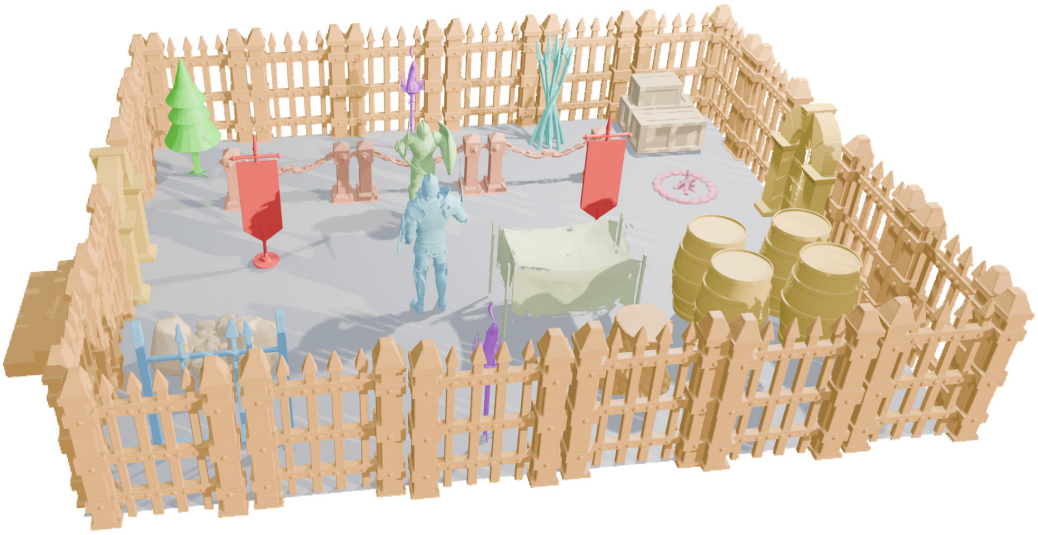

Figure 1: **Generated 3D assets.** Block3D meshes are manually assembled into an arena and rendered in Blender Blender Online Community (2018); prompts and isolated assets are documented in Appendix Section D.1.

ABSTRACT

While text-to-3D generation has advanced rapidly, achieving high geometric fidelity at low inference cost remains challenging. Existing text-to-3D methods either decode discrete shape tokens autoregressively or iteratively refine global 3D representations with diffusion or flow models. However, autoregressive decoding is sequential and cannot revise errors, whereas diffusion and flow-matching models repeatedly process the full representation, making high-quality generation increasingly expensive. In this paper, we propose Block3D, a block-wise diffusion framework that partitions the discrete shape-token sequence into contiguous blocks, generates the blocks autoregressively, and jointly denoises all tokens within the current block. To alleviate error accumulation, we introduce confidence-guided intra-block correction, which revises low-confidence tokens before each block is finalized. On a held-out set from TRELLIS-500K, Block3D reduces mean end-to-end generation time from 25.71 seconds to 4.99 seconds, achieving a $5.15\times$ speedup over the fine-tuned autoregressive baseline without sacrificing geometric fidelity.

1 INTRODUCTION

High-quality 3D assets are increasingly required in game development, film production, virtual and augmented reality, robotics, and embodied AI, creating a growing demand for systems that can generate 3D content from natural-language descriptions. Recent methods have substantially improved the fidelity and diversity of generated assets by learning effective 3D representations Xiang et al. (2025); Zhao et al. (2025); Yang et al. (2025); Foundation AI Team et al. (2025); Ye et al. (2025).

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.

Despite this progress, generating high-fidelity geometry typically requires autoregressive methods to sequentially decode long shape-token sequences or diffusion methods to repeatedly refine the complete latent representation over multiple denoising steps. As the representation becomes more detailed, both choices increase inference cost. Consequently, existing methods still struggle to achieve high geometric quality and low generation latency at the same time. Practical deployment further demands interactive latency, as highlighted by recent real-time 3D scene generation systems Ni et al. (2025).

In this paper, we propose Block3D, a block-wise autoregressive diffusion framework for efficient text-to-3D generation. Its central idea is to shift the causal dependency of autoregressive generation from individual shape tokens to contiguous latent blocks. Blocks are generated causally from left to right, while all tokens within the current block are jointly denoised under bidirectional attention. To preserve geometric fidelity when predicting all tokens within a block in parallel, our confidence-guided correction mechanism not only fills masked positions but also revises low-confidence tokens before the current block is finalized. Consequently, Block3D substantially reduces end-to-end generation latency while alleviating error accumulation within each block. Experiments on 100 held-out objects show that Block3D achieves the best geometric metrics among the evaluated methods while reducing the mean end-to-end generation time from 25.71 seconds to 4.99 seconds compared with the separately fine-tuned Cube Foundation AI Team et al. (2025) baseline, achieving a $5.15\times$ speedup.

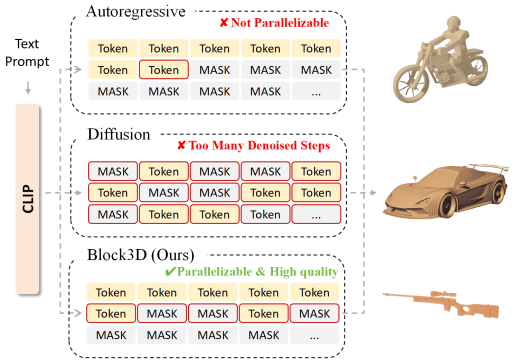

Figure 2: **Comparison of 3D token generation schedules.** (a) Token-wise generation predicts one shape token at a time and cannot revise emitted tokens. (b) Full-sequence denoising updates all positions in parallel but repeatedly processes the complete representation. (c) Block3D generates blocks from left to right and denoises all positions in the current block in parallel; a filled token can still be corrected before its block is finalized.

Recent work has pursued different efficiency strategies within diffusion/flow and autoregressive generation. Diffusion and flow models avoid token-wise decoding by refining multiple latent variables in parallel. TRELLIS, Hunyuan3D 2.0, and Pandora3D reduce the state being refined through compressed 3D latents, while TSSR denoises the complete mesh-token sequence and PartDiffuser limits each denoising stage to a semantic part Xiang et al. (2025); Zhao et al. (2025); Yang et al. (2025); Song et al. (2025); Yang et al. (2026b). These designs reduce the representation size or the active denoising region, but detailed geometry still requires repeated refinement; full-sequence denoising revisits every token, whereas part-wise denoising additionally relies on point-cloud conditioning and semantic part segmentation. Autoregressive generators instead reduce sequential cost by organizing geometry into adaptive, hierarchical, or local units. Adaptive octree tokenization allocates tokens according to geometric complexity, PointNSP and OctGPT generate geometry progressively across spatial scales, TreeMeshGPT follows triangle adjacency, and MeshMosaic generates and assembles local mesh patches Deng et al. (2025); Meng et al. (2025); Wei et al. (2025); Lionar et al. (2025); Xu et al. (2025). Although these representations shorten the causal sequence or its individual dependencies, fine geometry still requires additional tokens, hierarchy levels, branches, or patches to be generated sequentially. Consequently, high-fidelity generation continues to involve repeated latent refinement in diffusion-based methods or substantial sequential decoding in autoregressive methods, making low-latency generation difficult to achieve.

In summary, our main contributions are as follows:

- We introduce Block3D, which shifts the causal dependency in Cube Foundation AI Team et al. (2025) from individual shape tokens to latent blocks and performs parallel denoising within each block, substantially reducing sequential generation latency.
- We introduce confidence-guided intra-block correction, which combines mask-to-token recovery with token-to-token editing Bie et al. (2026) to alleviate error accumulation before the current block is finalized.
- Experiments on 100 held-out objects show that Block3D achieves the best geometric metrics among the evaluated methods and reduces mean end-to-end generation time from 25.71

seconds to 4.99 seconds relative to the fine-tuned Cube Foundation AI Team et al. (2025) baseline.

2 Related Work

2.1 Diffusion-Based 3D Generation

Recent feed-forward 3DGS work further studies scalable scene representations through compact latent compression Wang et al. (2025); these image-conditioned systems are complementary to our text-conditioned mesh generator. Diffusion and flow matching support several distinct text-to-3D paradigms. DreamFusion, Magic3D, and MOC-3D optimize a 3D representation with supervision derived from pretrained 2D diffusion models Poole et al. (2023); Lin et al. (2023); Fan et al. (2026). Shap-E and Rodin instead diffuse learned 3D representations, while TRELLIS performs flow matching over structured 3D latents Jun & Nichol (2023); Wang et al. (2023); Xiang et al. (2025). Hunyuan3D 2.0, Pandora3D, and CraftsMan3D use staged or native 3D generation pipelines for detailed assets Zhao et al. (2025); Yang et al. (2025); Li et al. (2025a). LGM directly predicts 3D Gaussians from multi-view images, whereas DreamCS and VLM3D study learned or vision-language reward signals Tang et al. (2024); Zou et al. (2025); Bai et al. (2025). These systems should therefore not be grouped under a single VAE-plus-latent-diffusion formulation.

Recent work reduces optimization cost or improves view consistency through iterative reconstruction, flow distillation, direct trajectory design, and multi-view memory Chen et al. (2025a); Yan et al. (2025); Li et al. (2025b); Zhou et al. (2025); Yang et al. (2026a); Zhou et al. (2026). These approaches update a global, implicit, or multi-view representation. Block3D instead retains a fixed discrete 3D representation and shortens the sequential horizon of its learned prior without changing the tokenizer or mesh decoder.

2.2 AR-Based 3D Generation

Another line of work formulates 3D generation as sequence prediction. MeshGPT autoregressively generates quantized triangle sequences, whereas Cube represents a shape by a compact, fixed-length sequence of VQ codes Siddiqui et al. (2024); Foundation AI Team et al. (2025). MeshAnything, MeshAnything V2, MeshRipple, and HiFi-Mesh improve mesh tokenization or shorten local autoregressive dependence Chen et al. (2025b;c); Lin et al. (2025); Li et al. (2026). LLaMA-Mesh serializes vertices and faces as text tokens for a language model Wang et al. (2024). ShapeLLM targets language-grounded 3D understanding, while ShapeLLM-Omni extends multimodal language modeling to both understanding and generation Qi et al. (2024); Ye et al. (2025). VAR-3D introduces a view-aware 3D tokenizer, and AR3D-R1 studies reinforcement learning for autoregressive text-to-3D generation Han et al. (2026); Tang et al. (2025).

Token-wise autoregression incurs one sequential decision per code and makes every emitted code irreversible. Later predictions are consequently conditioned on generated rather than ground-truth prefixes, the standard exposure-bias setting Bengio et al. (2015); Ranzato et al. (2016). Block3D does not remove generated-prefix exposure bias: completed blocks remain fixed. It provides only a bounded opportunity to revise codes within the active block before that block becomes part of the prefix.

2.3 Block-Wise and Editable Decoding

Discrete denoising models replace continuous Gaussian noise with categorical transitions, while masked generators iteratively reveal token subsets with bidirectional context Austin et al. (2021); Chang et al. (2022). In 3D generation, PartDiffuser uses semantic mesh parts as causal groups and denoises tokens within each part, whereas TSSR performs global discrete mesh-token generation followed by remasking-based refinement Yang et al. (2026b); Song et al. (2025). Block3D differs from both by operating on part-free, fixed-length Cube Foundation AI Team et al. (2025) codes and freezing every completed prefix block.

Block Diffusion establishes autoregressive factorization across blocks, bidirectional denoising within the active block, clean/noisy training attention, and prefix caching Arriola et al. (2025). Subsequent

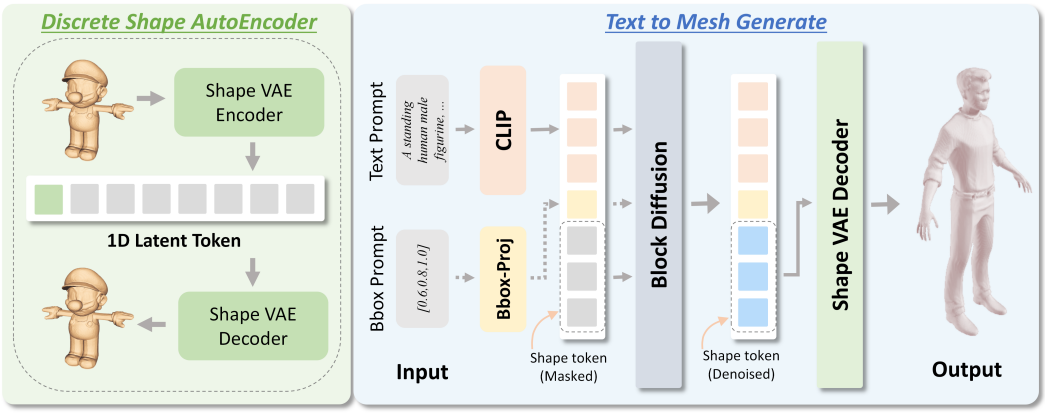

Figure 3: **Block3D overview.** Text and optional bounding-box conditions guide left-to-right block generation. Completed blocks form a frozen causal prefix, while M2T and T2T Bie et al. (2026) edit only the active block before the frozen decoder maps the completed codes to a mesh.

systems study efficient block-diffusion language decoding, vision-language tokens, sparse attention, and vision-language-action generation Wu et al. (2025); Cheng et al. (2025); Xi et al. (2026); Wang et al. (2026); Lee et al. (2026). LLaDA2.1 introduces M2T/T2T editing and mixed mask/random-token supervision with multi-turn-forward augmentation Bie et al. (2026). Block3D adopts these established mechanisms but specifies the sample-level mixture, shape-code corruption, residual objective, confidence-gated updates, and deterministic quota for fixed-length 3D codes. The resulting T2T operation is restricted to the active block and executed within the fixed decoding horizon; Appendix Section A.4 gives the complete update procedure.

3 Method

3.1 Overview

Figure 3 summarizes the text-to-mesh pipeline. Block3D uses the frozen Cube VQ autoencoder Foundation AI Team et al. (2025), which represents each mesh as $N = 1024$ discrete codes from a codebook of size $V = 16384$ and decodes a completed sequence into a mesh. Given a prompt y, the frozen CLIP ViT-L/14 encoder Radford et al. (2021) produces 77 token-level features; although an optional projected bounding-box token is supported, all reported experiments use text alone. The resulting condition C drives a Cube-initialized generator that partitions the N shape positions into K contiguous blocks. Following Block Diffusion Arriola et al. (2025), each active block starts fully masked, attends to C and the committed prefix, and is denoised in parallel. M2T and T2T updates adapted from LLaDA2.1 Bie et al. (2026) fill masked positions and revise filled positions before commitment; each committed block is then frozen, cached, and generated from left to right. After all K blocks are committed, the frozen VQ decoder maps $\hat{x}$ to the output mesh $\hat{S}$.

Block3D has three components: **(1) Conditioned Block-Causal Denoising**, which partitions the fixed-length sequence into computational blocks, defines clean/corrupted training visibility, and restricts bidirectional attention to the active block; **(2) Edit-Aware Training**, which combines masked and substituted shape codes with one model-based rollout and residual supervision; and **(3) Bounded Confidence-Guided Decoding**, which uses conditional confidence for M2T filling and T2T replacement Bie et al. (2026) together with a deterministic reveal quota that removes every mask within at most T iterations. The first component transfers block-causal generation to fixed-length 3D codes, while the latter two train and complete editable states under a fixed inference horizon. Since committed prefix blocks are never reopened, all revisions remain intra-block.

3.2 Block-Causal Shape-Code Denoising

The conditional prior retains the DualStream RoFormer generator of Cube Foundation AI Team et al. (2025). We write its shape-code logits as $\ell = F_\theta(C, z; A, p)$, where z is the visible shape state, A is its attention mask, and p gives the logical position indices. Probabilities and generated codes use

only the first V output entries. The auxiliary mask $[M]$ reuses the inherited padding-token identifier for input embedding lookup, but it is excluded from output normalization and is never passed to the frozen shape decoder.

Let the block size B and denoising horizon T be positive integers. We form $K = \lceil N/B \rceil$ contiguous blocks. Block k contains the positions $I_k = \{(k-1)B + 1, \ldots, \min(kB, N)\}$ for $k = 1, \ldots, K$, so only the final block can contain fewer than B codes. Writing $x^{(k)} = x_{I_k}$, $\hat{x}^{(k)} = \hat{x}_{I_k}$, and $\hat{x}^{(<k)} = (\hat{x}^{(1)}, \ldots, \hat{x}^{(k-1)})$, the block-causal dependency inherited from Block Diffusion Arriola et al. (2025) is realized as

$$\hat{x}^{(k)} = \mathcal{G}_{\theta,T}\Big(\hat{x}^{(<k)}, C\Big), \qquad k = 1, \ldots, K, \tag{1}$$

where $\mathcal{G}_{\theta,T}$ is the deterministic transition in the bounded editable block sampler; the other fixed decoding hyperparameters are suppressed from its notation. Equation (3) trains the token denoiser used inside this transition; we do not interpret $\mathcal{G}_{\theta,T}$ as an exact block likelihood.

At inference for block k, the generator receives $[\hat{x}^{(<k)}; z_k]$ on the shape side. Prefix queries use token-causal visibility. Every active-block query attends to the complete prefix and bidirectionally to all positions in z_k; future blocks are absent.

Adapting the vectorized clean-and-corrupted construction of Block Diffusion Arriola et al. (2025) to Cube Foundation AI Team et al. (2025), training evaluates all block contexts in one forward pass over the concatenated sequence $[x; \tilde{x}]$. A query in the clean copy can attend only to clean tokens in its own block and all preceding blocks; it cannot attend to a later clean block or any token in the corrupted copy. A query in the corrupted copy can attend to the clean tokens of all preceding blocks and bidirectionally to the corrupted tokens within its own block. It cannot access the clean target tokens of the current block, clean tokens from future blocks, or corrupted tokens from any other block. Every shape query additionally attends to all condition tokens, whereas condition queries attend only to condition tokens. This visibility rule allows the corrupted states of all blocks to be trained in parallel without exposing the clean target of the block being reconstructed.

The two copies share logical positions, $p(x_i) = p(\tilde{x}_i) = i$. At inference, prefix queries use token-causal visibility and active-block queries remain bidirectional. Training thus uses block-bidirectional teacher-forced prefixes, whereas inference uses token-causal generated prefixes. Caching reduces repeated computation Arriola et al. (2025), but does not remove this mismatch or exposure bias. Appendix Section A.2 gives the complete masks and position-ID construction.

3.3 Edit-Aware Corruption and Training

LLaDA2.1 Bie et al. (2026) motivates supervision on both masked and substituted states but does not determine the shape-code corruption used here. For each training sample and block, we draw $\tau_k \sim \mathcal{U}(t_{\min}, t_{\max})$ and independently sample $e_i \sim \mathrm{Bernoulli}(\tau_k)$ for $i \in I_k$. A single sample-level variable $a \sim \mathrm{Bernoulli}(\rho)$ selects the initial corruption stream: $a = 0$ is M2T and $a = 1$ is T2T. We set

$$\tilde{x}_i^{(0)} = \begin{cases} [M], & e_i = 1,\ a = 0, \\ u_i, & e_i = 1,\ a = 1, \\ x_i, & e_i = 0, \end{cases} \tag{2}$$

where u_i is uniform over the $V - 1$ shape codes different from x_i. We use $(t_{\min}, t_{\max}) = (0.45, 0.95)$ and $\rho = 0.5$. The variables τ_k control corruption only and are not supplied to F_θ as time embeddings. Uniform wrong codes diversify substituted-state supervision, while the model-based rollout below introduces model-generated candidates before the residual loss. Inference starts each active block from the all-mask state.

Classifier-free condition dropout Ho & Salimans (2022) is sampled once per batch element before either model evaluation. Let $\bar{C}_n = C_n$ for a retained condition and $\bar{C}_n = C_n^-$ for a dropped condition, where C_n^- contains the empty-text encoding and, when enabled, a zero bounding box. The same realized condition $\bar{C}_n$ is reused by both evaluations below.

Inspired by the multi-turn-forward augmentation of LLaDA2.1 Bie et al. (2026), we instantiate one model-based rollout ($R = 1$) before computing the loss. For every block, the rollout initializes $z_k^{(0)} = \tilde{x}_{I_k}^{(0)}$. A no-gradient forward pass on $[x; \tilde{x}^{(0)}]$ supplies unguided logits ($g = 0$) under $\bar{C}_n$, after

which the update sets in Equations (7)–(8) are applied independently within every corrupted block using (η_M, η_T) and the first-step quota q_0. Under the default $B = 64$ and $T = 4$, $q_0 = 16$. The simultaneous updates define $z_k^{(1)}$, and the rollout output is assembled by setting $\tilde{x}_{I_k}^{(R)} = z_k^{(1)}$ for every block. This state may contain masks, correct codes, and model-produced incorrect codes even when its initial state used only one corruption stream.

We then perform a separate gradient-carrying forward pass on $[x; \tilde{x}^{(R)}]$ and retain only corrupted-copy logits. For batch index n, let $\pi_{\theta,n,i}^{(R)}$ be their softmax over the first V entries. Under the training visibility rule above, this distribution is conditioned on $\bar{C}_n$, the clean blocks preceding the block that contains position i, and the current corrupted block. Over all valid shape positions in the batch, define $\mathcal{D} = \{(n,i) : \tilde{x}_{n,i}^{(R)} \neq x_{n,i}\}$. The objective is

$$\mathcal{L} = -\frac{1}{|\mathcal{D}| + \epsilon} \sum_{(n,i)\in\mathcal{D}} \log \pi_{\theta,n,i}^{(R)}(x_{n,i}). \tag{3}$$

We set $\epsilon = 10^{-8}$ and reduce over all residual tokens in the batch. Samples without residuals add no term; an empty $\mathcal{D}$ yields zero loss. The one-step rollout adds a generation-dependent state, while preceding blocks remain teacher-forced and cross-block exposure bias remains. Appendix Section A.3 specifies corruption, rollout ordering, residual reduction, and empty-set handling.

3.4 Bounded Confidence-Guided Decoding

Let the guidance coefficient satisfy $g \geq 0$, and let $\eta_M, \eta_T \in [0,1]$ be the M2T and T2T confidence thresholds Bie et al. (2026). For block k, let $n_k = |I_k|$ and initialize $z_{k,i}^{(0)} = [M]$ for every $i \in I_k$. We suppress k in iteration-level logits and update sets. At iteration $s \in \{0, \ldots, T-1\}$, the conditional branch C^+ encodes the prompt, while C^- encodes the empty prompt and, when enabled, a zero bounding box. The branches produce ℓ_s^+ and ℓ_s^-. Classifier-free guidance (CFG) Ho & Salimans (2022) gives

$$\ell_s^g = (1+\gamma_s)\ell_s^+ - \gamma_s \ell_s^-, \qquad \gamma_s = g\frac{T-s}{T}. \tag{4}$$

Thus the guidance coefficient decreases from g to g/T, rather than to zero; when $g = 0$, the unconditional branch is omitted and $\ell_s^g = \ell_s^+$. For the deterministic sampler used in our experiments, the candidate and its acceptance confidence are

$$\begin{aligned} \hat{z}_{s,i} &= \arg\max_{0\leq v<V} \ell_{s,i,v}^g, \\ \alpha_{s,i} &= \mathrm{softmax}(\ell_{s,i,0:V-1}^+)[\hat{z}_{s,i}]. \end{aligned} \tag{5}$$

The guided logits propose a shape code, whereas the conditional logits determine whether it is accepted; CFG therefore does not directly inflate the threshold score. Ties in the argmax are resolved by selecting the smallest shape-code index.

We assign each iteration a deterministic minimum reveal quota

$$q_s = \left\lfloor \frac{n_k}{T} \right\rfloor + \mathbf{1}[s < n_k \bmod T], \qquad \sum_{s=0}^{T-1} q_s = n_k. \tag{6}$$

Let $\mathcal{M}_s = \{i \in I_k : z_{k,i}^{(s)} = [M]\}$. Following the editable update in LLaDA2.1 Bie et al. (2026), M2T and T2T select

$$\begin{aligned} \mathcal{U}_s^{\mathrm{M2T}} &= \{i \in \mathcal{M}_s : \alpha_{s,i} > \eta_M\} \\ &\quad \cup \operatorname*{TopK}_{i\in\mathcal{M}_s}(\alpha_{s,i}, \min(q_s, |\mathcal{M}_s|)), \end{aligned} \tag{7}$$

$$\mathcal{U}_s^{\mathrm{T2T}} = \{i \in I_k \setminus \mathcal{M}_s : \hat{z}_{s,i} \neq z_{k,i}^{(s)}, \alpha_{s,i} > \eta_T\}. \tag{8}$$

Here, $\mathrm{TopK}_{i\in\mathcal{M}_s}(\alpha_{s,i}, r)$ returns the r masked positions with the largest confidence values and returns the empty set when $r = 0$. Confidence ties are resolved in ascending order of the global position index i. This defines short final blocks and settings with $n_k < T$. Appendix Section A.4 gives the complete sampler, including tie handling and short-block execution. The active state is updated simultaneously:

$$z_{k,i}^{(s+1)} = \begin{cases} \hat{z}_{s,i}, & i \in \mathcal{U}_s^{\mathrm{M2T}} \cup \mathcal{U}_s^{\mathrm{T2T}}, \\ z_{k,i}^{(s)}, & \text{otherwise.} \end{cases} \tag{9}$$

High-confidence M2T updates can make the number of revealed positions exceed q_s, so q_s is a lower bound on progress rather than an upper update budget. T2T does not inspect the confidence of the old code: it replaces a filled code only when a different new candidate exceeds η_T.

If $m_s = |\mathcal{M}_s|$, the TopK fallback reveals at least $\min(q_s, m_s)$ masked positions, so Equation (7) implies $m_{s+1} \leq \max(m_s - q_s, 0)$. Applying this bound across iterations and using $\sum_{s=0}^{T-1} q_s = n_k$ gives $m_T = 0$. The quota therefore completes every active block within T iterations while confidence-gated T2T editing Bie et al. (2026) remains available at every iteration before commitment.

We retain the 23-layer DualStream RoFormer from Cube Foundation AI Team et al. (2025) (12 heads, width 1536), bypass its inherited single-stream layer on this path, and initialize the first V token embeddings from a projection of the frozen VQ codebook. We fine-tune only this generator. Inference uses $B = 64$, $T = 4$, $(\eta_M, \eta_T) = (0.95, 0.9)$ for M2T/T2T editing Bie et al. (2026), and guidance coefficient $g = 3.0$ under Equation (4), giving $K = 16$.

With CFG Ho & Salimans (2022), conditional and unconditional branches are concatenated in the batch. Decoding makes K cache calls and at most KT active-block calls, equivalent to at most $2K(T + 1)$ logical branch evaluations when $g > 0$; the default uses at most 80 batched or 160 logical calls. Following Block Diffusion Arriola et al. (2025), the condition and prefix cache are reused within the T updates and rebuilt after commitment. Thus intra-block revision has a fixed horizon, while completed blocks cannot be corrected. Appendix Section A.5 gives the cache construction and model-call complexity derivation.

4 Experiments

4.1 Experimental Settings

Datasets. We fine-tune the text-to-shape generator on 300K objects sampled from TRELLIS-500K Xiang et al. (2025), a large-scale collection of 3D assets with paired text descriptions. To construct the evaluation set, we first randomly select 100 objects from TRELLIS-500K using seed 42 and exclude these exact objects from the fine-tuning candidate pool. Each shape is converted into discrete shape tokens using the frozen Cube Foundation AI Team et al. (2025) tokenizer, and its paired text serves as the generation condition. Each evaluation object retains its paired text prompt, and every method generates one shape per prompt. Appendix Section B.1 records the split, identifiers, prompts, and exclusion rule; Appendix Section B.2 specifies preprocessing.

Baselines and Evaluation Metrics. We compare Block3D with ShapeLLM-Omni Ye et al. (2025), TRELLIS-text Xiang et al. (2025), AR3D-R1 Tang et al. (2025), and Cube Foundation AI Team et al. (2025). Cube provides the controlled comparison because both models use the same released checkpoint, frozen geometry components, training subset, and optimization budget; the remaining methods use their official inference settings. Appendix Section C.4 lists releases and complete baseline settings.

Geometry evaluation samples 8,192 surface points and normals per mesh pair and reports Chamfer-L_1 distance, normal consistency, and F-score at 1% of the target bounding-box diagonal Fan et al. (2017); Mescheder et al. (2019). Eight-view CLIPScore measures text-shape alignment with CLIP ViT-L/14 Hessel et al. (2021); Radford et al. (2021). Latency statistics are measured on one NVIDIA A100 and include condition encoding, shape-code generation, and mesh decoding, but exclude model loading and disk I/O. Protocol details appear in Appendix Section C.1 (geometry), C.2 (CLIPScore), C.3 (latency), and C.5 (invalid outputs and statistical scope).

Implementation Details. We initialize Block3D from the released Cube Foundation AI Team et al. (2025) checkpoint and freeze its shape tokenizer, CLIP ViT-L/14 Radford et al. (2021) text encoder, and shape decoder. Only the text-to-shape generator is fine-tuned. We train for 35K steps with AdamW Loshchilov & Hutter (2019), a learning rate of 1×10^{-4}, bfloat16 precision, and a global batch size of 40 on four NVIDIA A100 80GB GPUs. We use a 100-step linear warm-up, zero weight decay, a gradient clipping norm of 1.0, and random seed 42. Appendix Section B.3 gives the complete training and checkpoint configuration.

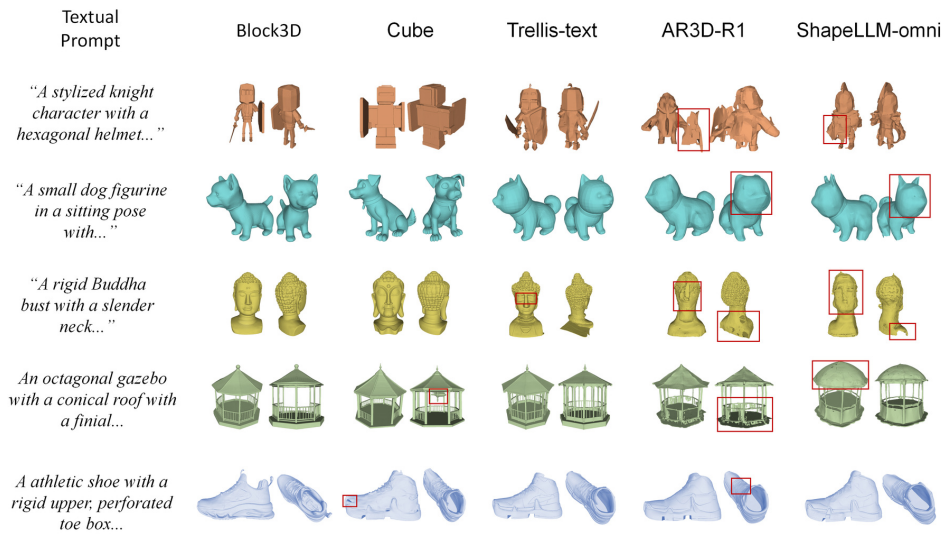

Figure 4: **Qualitative Comparison of Text-to-3D Generation.** Compared to existing paradigms, Block3D produces coherent front and back views in these examples, while several baselines exhibit missing or distorted geometry. Complete prompts and the selection protocol are documented in Appendix Section D.2.

Table 1: Geometry and text-shape alignment on the 100-object TRELLIS-500K Xiang et al. (2025) evaluation set. F@1 denotes an F-score threshold of 1% Fan et al. (2017); Mescheder et al. (2019). Geometry metrics are rounded to three decimals and CLIPScore Hessel et al. (2021) to two decimals.

Method	CD-L1↓	NC↑	F@1%↑	CLIP↑
ShapeLLM-Omni	0.229	0.490	0.089	19.08
TRELLIS-text	0.222	0.496	0.090	20.41
AR3D-R1	0.145	0.583	0.162	21.92
Cube	0.094	0.632	0.219	**23.87**
Block3D	**0.078**	**0.668**	**0.309**	23.24

During training, the block corruption level is sampled uniformly from $[0.45, 0.95]$, the T2T branch Bie et al. (2026) is selected with probability 0.5, and one no-gradient model-based filling rollout is performed. Classifier-free condition dropout Ho & Salimans (2022) is applied with probability 0.1. Unless otherwise stated, inference uses a block size of 64, four denoising steps per block, M2T and T2T confidence thresholds of 0.95 and 0.9, and guidance coefficient $g = 3.0$ in Equation (4). With $T = 4$, γ_s decreases linearly from 3.0 to 0.75 across the four iterations. We use deterministic argmax decoding without top-p sampling. Parameter selection is in Appendix Section B.4; code and materials are indexed in Appendix Sections E.1–E.3.

4.2 Comparison with 3D Generation Methods

Geometry Metrics Comparison. Table 1 compares paired geometric fidelity and text-shape alignment. Under the shared evaluation protocol, Block3D attains the strongest CD-L1, NC, and F@1% scores Fan et al. (2017); Mescheder et al. (2019) among the evaluated methods, while its CLIPScore Hessel et al. (2021) remains competitive. The controlled comparison with Cube Foundation AI Team et al. (2025) shows that block-wise denoising improves paired geometric fidelity in addition to accelerating generation.

Efficiency Metrics Comparison. Table 2 shows that Block3D is the fastest evaluated method across all reported latency statistics. Its mean end-to-end generation time is 4.99 seconds, corresponding to a $5.15\times$ speedup over the controlled Cube Foundation AI Team et al. (2025) baseline. The consistent improvements in median and P90 latency show that this advantage holds across the 100 TRELLIS-500K Xiang et al. (2025) prompts rather than arising from a small number of easy cases.

Table 2: End-to-end generation time in seconds over the 100 TRELLIS-500K Xiang et al. (2025) prompts, measured on one NVIDIA A100 80GB GPU.

Method	Mean↓	Median↓	P90↓	Std.↓
TRELLIS-text	11.65	9.75	20.06	5.59
ShapeLLM-Omni	36.89	33.69	40.64	8.48
AR3D-R1	79.80	82.48	94.19	17.34
Cube	25.71	25.43	26.68	0.80
Block3D	**4.99**	**4.96**	**5.60**	**0.43**

Table 3: Ablations of block size and denoising steps. Time is reported in seconds. Every configuration is trained independently.

(a) Block size, $T = 4$

B	Time↓	CD-L1↓	NC↑	F@1%↑
$B = 32$	12.98	**0.076**	0.667	0.296
$B = 64$	4.99	0.078	**0.668**	**0.309**
$B = 96$	3.62	0.085	0.660	0.279
$B = 128$	2.90	0.090	0.647	0.258
$B = 256$	**2.15**	0.186	0.578	0.103

(b) Denoising steps, $B = 64$

T	Time↓	CD-L1↓	NC↑	F@1%↑
$T = 4$	**4.99**	**0.078**	0.668	**0.309**
$T = 8$	7.33	0.079	0.665	0.300
$T = 12$	9.37	0.084	0.668	0.284
$T = 20$	13.66	**0.078**	**0.676**	0.303

4.3 Ablation Study

We study block size, the number of denoising steps, and token-to-token editing Bie et al. (2026). Every configuration in Table 3, as well as both variants in Table 4, is trained independently for the same number of optimization steps and evaluated on the same 100-object TRELLIS-500K Xiang et al. (2025) evaluation set.

Table 4: Effect of token-to-token editing Bie et al. (2026). Geometry metrics are rounded to three decimals and CLIPScore Hessel et al. (2021) to two decimals.

Model	CD-L1↓	NC↑	F@1%↑	CLIP↑
Block3D (M2T)	0.081	0.666	0.287	22.74
Block3D (M2T+T2T)	**0.078**	**0.668**	**0.309**	**23.24**

Effect of Block Size. Table 3 shows that larger blocks reduce the number of left-to-right stages but make parallel denoising harder. Moving from $B = 64$ to 96 lowers latency from 4.99 to 3.62 seconds but reduces F@1% from 0.309 to 0.279; at $B = 256$, F@1% falls to 0.103. We therefore use $B = 64$ as the quality–latency balance.

Effect of Denoising Steps. Table 3 shows that additional steps increase latency without monotonic quality gains. At $T = 20$, latency rises to 13.66 seconds while CD-L1 remains 0.078 and F@1% is 0.303, compared with 4.99 seconds and 0.309 at $T = 4$. We therefore use four steps.

Effect of Token-to-Token Editing. Table 4 compares the complete model with an independently trained M2T-only variant Bie et al. (2026). Adding T2T editing reduces CD-L1 by 4.7%, raises F@1% from 0.287 to 0.309, and raises CLIPScore from 22.74 to 23.24. The simultaneous gains support token correction for revising errors that mask-only recovery cannot address.

5 Conclusion

Block3D replaces the token-wise prior of Cube Foundation AI Team et al. (2025) with block-causal denoising, generating blocks from left to right while jointly denoising and correcting the active shape codes. Among the evaluated methods, it improves paired geometry and reduces Cube's mean generation time from 25.71 to 4.99 seconds while retaining competitive text-shape alignment. Completed blocks remain fixed; future work will study cross-block refinement and broader shape representations and datasets.

AI USE STATEMENT

Large language models were used solely as writing assistants to improve grammar, clarity, readability, and sentence structure. They were not used for research ideation, hypothesis formulation, methodological design, experimental design, data analysis, result interpretation, or generation of scientific conclusions. All AI-assisted text was carefully reviewed and edited by the authors. The authors take full responsibility for the final content of this paper.

REPRODUCIBILITY STATEMENT

The main paper presents the Block3D architecture, training objective, decoding procedure, and primary experimental settings. Appendix A provides the complete attention masks, corruption and rollout construction, residual objective, bounded sampler, and cache-complexity analysis. Appendix B specifies the data split, preprocessing, optimization, and checkpoint protocol. Appendix C defines the geometry, CLIPScore, latency, baseline, and statistical evaluation procedures, while Appendix D lists the qualitative prompts and scene-assembly protocol. Appendix E summarizes the implementation structure, configuration settings, execution procedures, and evaluation records. Together, the main paper and supplementary material specify the data, training, inference, and evaluation protocols used to obtain the reported results.

REFERENCES

Marianne Arriola, Aaron Gokaslan, Justin T. Chiu, Zhihan Yang, Zhixuan Qi, Jiaqi Han, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Block diffusion: Interpolating between autoregressive and diffusion language models. In *Int. Conf. Learn. Represent.*, 2025. URL https://arxiv.org/abs/2503.09573.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. *arXiv preprint arXiv:2107.03006*, 2021.

Weimin Bai, Yubo Li, Weijian Luo, Wenzheng Chen, and He Sun. Vision-language models as differentiable semantic and spatial rewards for text-to-3d generation. *arXiv preprint arXiv:2509.15772*, 2025.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Adv. Neural Inform. Process. Syst.*, pp. 1171–1179, 2015.

Tiwei Bie, Maosong Cao, Xiang Cao, Bingsen Chen, Fuyuan Chen, Kun Chen, Lun Du, Daozhuo Feng, Haibo Feng, Mingliang Gong, Zhuocheng Gong, Yanmei Gu, Jian Guan, Kaiyuan Guan, Hongliang He, Zenan Huang, Juyong Jiang, Zhonghui Jiang, Zhenzhong Lan, Chengxi Li, Jianguo Li, Zehuan Li, Huabin Liu, Lin Liu, Guoshan Lu, Yuan Lu, Yuxin Ma, Xingyu Mou, Zhenxuan Pan, Kaida Qiu, Yuji Ren, Jianfeng Tan, Yiding Tian, Zian Wang, Lanning Wei, Tao Wu, Yipeng Xing, Wentao Ye, Liangyu Zha, Tianze Zhang, Xiaolu Zhang, Junbo Zhao, Da Zheng, Hao Zhong, Wanli Zhong, Jun Zhou, Junlin Zhou, Liwang Zhu, Muzhi Zhu, and Yihong Zhuang. LLaDA2.1: Speeding up text diffusion via token editing. *arXiv preprint arXiv:2602.08676*, 2026.

Blender Online Community. Blender: A 3d modelling and rendering package, 2018. URL https://www.blender.org.

Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T. Freeman. MaskGIT: Masked generative image transformer. *arXiv preprint arXiv:2202.04200*, 2022.

Luxi Chen, Zhengyi Wang, Zihan Zhou, Tingting Gao, Hang Su, Jun Zhu, and Chongxuan Li. MicroDreamer: Efficient 3d generation in $\sim$20 seconds by score-based iterative reconstruction. *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 1–12, 2025a. doi: 10.1109/TPAMI.2025.3600494.

Yiwen Chen, Tong He, Di Huang, Weicai Ye, Sijin Chen, Jiaxiang Tang, Xin Chen, Zhongang Cai, Lei Yang, Gang Yu, Guosheng Lin, and Chi Zhang. MeshAnything: Artist-created mesh generation with autoregressive transformers. In *Int. Conf. Learn. Represent.*, 2025b. URL https://arxiv.org/abs/2406.10163.

Yiwen Chen, Yikai Wang, Yihao Luo, Zhengyi Wang, Zilong Chen, Jun Zhu, Chi Zhang, and Guosheng Lin. MeshAnything V2: Artist-created mesh generation with adjacent mesh tokenization. In *Int. Conf. Comput. Vis.*, pp. 13922–13931, 2025c. doi: 10.1109/ICCV51701.2025.01292.

Shuang Cheng, Yuhua Jiang, Zineng Zhou, Dawei Liu, Wang Tao, Linfeng Zhang, Biqing Qi, and Bowen Zhou. SDAR-VL: Stable and efficient block-wise diffusion for vision-language understanding. *arXiv preprint arXiv:2512.14068*, 2025.

Kangle Deng, Hsueh-Ti Derek Liu, Yiheng Zhu, Xiaoxia Sun, Chong Shang, Kiran S. Bhat, Deva Ramanan, Jun-Yan Zhu, Maneesh Agrawala, and Tinghui Zhou. Efficient autoregressive shape generation via octree-based adaptive tokenization. In *Int. Conf. Comput. Vis.*, pp. 11685–11696, 2025. doi: 10.1109/ICCV51701.2025.01087.

Chenyang Fan, Junshi Cheng, Wen Yang, Zihong Li, Wenfeng Zhang, Wei Hu, Yi Zhang, and Pan Zeng. MOC-3D: Manifold-order consistency for text-to-3d generation. In *Int. Conf. Multimed. Retr.*, 2026. doi: 10.1145/3805622.3810761. URL https://arxiv.org/abs/2605.01743.

Haoqiang Fan, Hao Su, and Leonidas J. Guibas. A point set generation network for 3d object reconstruction from a single image. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pp. 2463–2471, 2017. doi: 10.1109/CVPR.2017.264.

Foundation AI Team, Kiran Bhat, Nishchaie Khanna, Karun Channa, Tinghui Zhou, Yiheng Zhu, Xiaoxia Sun, Charles Shang, Anirudh Sudarshan, et al. Cube: A Roblox view of 3d intelligence. *arXiv preprint arXiv:2503.15475*, 2025.

Zongcheng Han, Dongyan Cao, Haoran Sun, and Yu Hong. VAR-3D: View-aware auto-regressive model for text-to-3d generation via a 3d tokenizer. *arXiv preprint arXiv:2602.13818*, 2026.

Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. CLIPScore: A reference-free evaluation metric for image captioning. In *Conf. Empir. Methods Nat. Lang. Process.*, pp. 7514–7528. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.emnlp-main.595.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Heewoo Jun and Alex Nichol. Shap-E: Generating conditional 3d implicit functions. *arXiv preprint arXiv:2305.02463*, 2023.

Sung-Wook Lee, Xuhui Kang, and Yen-Ling Kuo. TBD-VLA: Temporal block diffusion vision language action model. *arXiv preprint arXiv:2606.07895*, 2026.

Weiyu Li, Jiarui Liu, Hongyu Yan, Rui Chen, Yixun Liang, Xuelin Chen, Ping Tan, and Xiaoxiao Long. CraftsMan3D: High-fidelity mesh generation with 3d native diffusion and interactive geometry refiner. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pp. 5307–5317, 2025a. doi: 10.1109/CVPR52734.2025.00500.

Yanfeng Li, Tao Tan, Qinquan Gao, Zhiwen Cao, Xiaohong Liu, and Yue Sun. HiFi-Mesh: High-fidelity efficient 3d mesh generation via compact autoregressive dependence. In *AAAI Conf. Artif. Intell.*, pp. 6566–6574, 2026. doi: 10.1609/aaai.v40i8.37586.

Ziying Li, Xuequan Lu, Xinkui Zhao, Guanjie Cheng, Shuiguang Deng, and Jianwei Yin. Walking the Schrödinger bridge: A direct trajectory for text-to-3d generation. *arXiv preprint arXiv:2511.05609*, 2025b.

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 *IEEE Conf. Comput. Vis. Pattern Recog.*, pp. 300–309, 2023. doi: 10.1109/CVPR52729.2023.00037.

Junkai Lin, Hang Long, Huipeng Guo, Jielei Zhang, Jiayi Yang, Tianle Guo, Yang Yang, Jianwen Li, Wenxiao Zhang, Matthias Nießner, and Wei Yang. MeshRipple: Structured autoregressive generation of artist-meshes. *arXiv preprint arXiv:2512.07514*, 2025.

Stefan Lionar, Jiabin Liang, and Gim Hee Lee. TreeMeshGPT: Artistic mesh generation with autoregressive tree sequencing. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pp. 26608–26617, 2025. doi: 10.1109/CVPR52734.2025.02478.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *Int. Conf. Learn. Represent.*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Ziqiao Meng, Qichao Wang, Zhiyang Dou, Zixing Song, Zhipeng Zhou, Irwin King, and Peilin Zhao. PointNSP: Autoregressive 3d point cloud generation with next-scale level-of-detail prediction. *arXiv preprint arXiv:2503.08594*, 2025.

Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pp. 4460–4470, 2019. doi: 10.1109/CVPR.2019.00459.

Chaojun Ni, Xiaofeng Wang, Zheng Zhu, Weijie Wang, Haoyun Li, Guosheng Zhao, Jie Li, Wenkang Qin, Guan Huang, and Wenjun Mei. Wonderturbo: Generating interactive 3d world in 0.72 seconds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025. URL https://arxiv.org/abs/2504.02261.

Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. DreamFusion: Text-to-3d using 2d diffusion. In *Int. Conf. Learn. Represent.*, 2023. URL https://arxiv.org/abs/2209.14988.

Zekun Qi, Runpei Dong, Shaochen Zhang, Haoran Geng, Chunrui Han, Zheng Ge, Li Yi, and Kaisheng Ma. ShapeLLM: Universal 3d object understanding for embodied interaction. In *Eur. Conf. Comput. Vis.*, pp. 214–238, 2024. doi: 10.1007/978-3-031-72775-7_13.

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. In *Int. Conf. Mach. Learn.*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. In *Int. Conf. Learn. Represent.*, 2016.

Yawar Siddiqui, Antonio Alliegro, Alexey Artemov, Tatiana Tommasi, Daniele Sirigatti, Vladislav Rosov, Angela Dai, and Matthias Nießner. MeshGPT: Generating triangle meshes with decoder-only transformers. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pp. 19615–19625, 2024. doi: 10.1109/CVPR52733.2024.01855.

Kaiyu Song, Hanjiang Lai, Yaqing Zhang, Chuangjian Cai, Yan Pan, Kun Yue, and Jian Yin. Topology sculptor, shape refiner: Discrete diffusion model for high-fidelity 3d meshes generation. *arXiv preprint arXiv:2510.21264*, 2025.

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 *Eur. Conf. Comput. Vis.*, pp. 1–18, 2024. doi: 10.1007/978-3-031-73235-5_1.

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. *arXiv preprint arXiv:2512.10949*, 2025.

Ruiheng Wang, Shuanghao Bai, Haoran Zhang, Badong Chen, and Xiangyu Xu. BlockVLA: Accelerating autoregressive VLA via block diffusion finetuning. *arXiv preprint arXiv:2605.13382*, 2026.

Tengfei Wang, Bo Zhang, Ting Zhang, Shuyang Gu, Jianmin Bao, Tadas Baltrusaitis, Jingjing Shen, Dong Chen, Fang Wen, Qifeng Chen, and Baining Guo. RODIN: A generative model for sculpting 3d digital avatars using diffusion. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pp. 4563–4573, 2023. doi: 10.1109/CVPR52729.2023.00443.

Weijie Wang, Donny Y. Chen, Zeyu Zhang, Duochao Shi, Akide Liu, and Bohan Zhuang. Zpressor: Bottleneck-aware compression for scalable feed-forward 3dgs. In *Advances in Neural Information Processing Systems*, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/a48a6c9a03e87d6426b3f9bd18bbb86b-Abstract-Conference.html.

Zhengyi Wang, Jonathan Lorraine, Yikai Wang, Hang Su, Jun Zhu, Sanja Fidler, and Xiaohui Zeng. LLaMA-Mesh: Unifying 3d mesh generation with language models. *arXiv preprint arXiv:2411.09595*, 2024.

Si-Tong Wei, Rui-Huan Wang, Chuan-Zhi Zhou, Baoquan Chen, and Peng-Shuai Wang. OctGPT: Octree-based multiscale autoregressive models for 3d shape generation. In *ACM SIGGRAPH Conf. Comput. Graph. Interact. Tech.*, pp. 1–11, 2025. doi: 10.1145/3721238.3730601.

Chengyue Wu, Hao Zhang, Shuchen Xue, Shizhe Diao, Yonggan Fu, Zhijian Liu, Pavlo Molchanov, Ping Luo, Song Han, and Enze Xie. Fast-dLLM v2: Efficient block-diffusion LLM. *arXiv preprint arXiv:2509.26328*, 2025.

Haocheng Xi, Harman Singh, Yuezhou Hu, Coleman Hooper, Rishabh Tiwari, Aditya Tomar, Minjae Lee, Wonjun Kang, Michael Mahoney, Chenfeng Xu, Kurt Keutzer, and Amir Gholami. LoSA: Locality aware sparse attention for block-wise diffusion language models. *arXiv preprint arXiv:2604.12056*, 2026.

Jianfeng Xiang, Zelong Lv, Sicheng Xu, Yu Deng, Ruicheng Wang, Bowen Zhang, Dong Chen, Xin Tong, and Jiaolong Yang. Structured 3d latents for scalable and versatile 3d generation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pp. 21469–21480, 2025. doi: 10.1109/CVPR52734.2025.02000.

Rui Xu, Tianyang Xue, Qiujie Dong, Le Wan, Zhe Zhu, Peng Li, Zhiyang Dou, Cheng Lin, Shiqing Xin, Yuan Liu, Wenping Wang, and Taku Komura. MeshMosaic: Scaling artist mesh generation via local-to-global assembly. *arXiv preprint arXiv:2509.19995*, 2025.

Runjie Yan, Yinbo Chen, and Xiaolong Wang. Consistent flow distillation for text-to-3d generation. *arXiv preprint arXiv:2501.05445*, 2025.

Feng Yang, Wenliang Qian, Wangmeng Zuo, and Hui Li. Bridging geometry-coherent text-to-3d generation with multi-view diffusion priors and gaussian splatting. *Neural Networks*, 197:108511, 2026a. doi: 10.1016/j.neunet.2025.108511.

Jiayu Yang, Taizhang Shang, Weixuan Sun, Xibin Song, Ziang Cheng, Senbo Wang, Shenzhou Chen, Weizhe Liu, Hongdong Li, and Pan Ji. Pandora3D: A comprehensive framework for high-quality 3d shape and texture generation. *arXiv preprint arXiv:2502.14247*, 2025.

Yichen Yang, Hong Li, Haodong Zhu, Linin Yang, Guojun Lei, Sheng Xu, and Baochang Zhang. PartDiffuser: Part-wise 3d mesh generation via discrete diffusion. *arXiv preprint arXiv:2511.18801*, 2026b.

Junliang Ye, Zhengyi Wang, Ruowen Zhao, Shenghao Xie, and Jun Zhu. ShapeLLM-Omni: A native multimodal LLM for 3d generation and understanding. *arXiv preprint arXiv:2506.01853*, 2025.

Zibo Zhao, Zeqiang Lai, Qingxiang Lin, Yunfei Zhao, Haolin Liu, Shuhui Yang, Yifei Feng, et al. Hunyuan3D 2.0: Scaling diffusion models for high resolution textured 3d assets generation. *arXiv preprint arXiv:2501.12202*, 2025.

Kaichen Zhou, Zeyang Bai, Xinhai Chang, Mengyu Wang, Paul Liang, and Fangneng Zhan. Stream3D: Sequential multi-view 3d generation via evidential memory. *arXiv preprint arXiv:2605.21472*, 2026.

Yuan Zhou, Shilong Jin, Litao Hua, Wanjun Lv, Haoran Duan, and Jungong Han. Cons-Dreamer: Advancing multi-view consistency for zero-shot text-to-3d generation. *arXiv preprint arXiv:2504.02316*, 2025.

Xiandong Zou, Ruihao Xia, Hongsong Wang, and Pan Zhou. DreamCS: Geometry-aware text-to-3d generation with unpaired 3d reward supervision. *arXiv preprint arXiv:2506.09814*, 2025.