BEYOND GLOBAL LATENTS: CHUNK-BASED SPARSE GRID VAE FOR SCALABLE 3D MODELING

Anonymous authors
Paper under double-blind review

ABSTRACT

Sparse voxel grids preserve the spatial structure needed for detailed 3D reconstruction, but their memory still grows rapidly with resolution as active surface cells increase. We introduce *ChunkVAE*, a sparse grid variational autoencoder organized around local chunks rather than a global latent volume. Local learned operators permit independently chosen encoder and decoder partitions and allow inference chunk sizes to differ from training. Two complementary data operators make this flexibility practical: *Balanced Binary Object Partitioning* distributes active cells while limiting replicated overlap, while *S-Curve weighted stitching* attenuates unreliable boundary features when assembling a global latent or reconstruction. Across three object benchmarks, *ChunkVAE* is competitive with or better than strong baselines from 512^3 to 1536^3; smaller chunks lower peak allocated memory and shorten per-chunk compute, enabling faster parallel inference. Stable stitched latents and improved image-to-3D metrics indicate that local compression can scale geometry while retaining the global interface required downstream.

1 INTRODUCTION

High-fidelity 3D generation relies on an autoencoder that preserves geometry before a generative model learns its latent distribution (Zhang et al., 2023; Zhao et al., 2023; Li et al., 2024; Zhang et al., 2024; Hunyuan3D Team et al., 2025; Lai et al., 2025; Xiang et al., 2024; 2025). Errors introduced by this compression stage cannot be recovered downstream. Improving the spatial resolution and capacity of 3D variational autoencoders is therefore a direct route to sharper generated geometry.

Existing 3D VAEs commonly use global vector sets or sparse voxel grids. Vector set models compress a surface into global tokens and decode a continuous field (Zhang et al., 2023; Chen et al., 2025; Li et al., 2025b). Their compact latent is attractive for generation, but local detail must be conveyed through a limited global token set. Sparse grid models retain explicit spatial support and have produced strong high-resolution reconstructions (Ren et al., 2024; Xiang et al., 2024; He et al., 2025; Wu et al., 2025b; Xiang et al., 2025). Sparsity removes empty space, but it does not remove the resolution dependence of active surface cells. Increasing the grid resolution consequently raises activation memory and restricts both model size and usable spatial extent.

Chunking appears to offer a simple solution, yet it changes the representation learned by the VAE. Uniform spatial partitions can contain very different numbers of active cells, causing load imbalance; severely under-populated chunks may be eliminated by decoder pruning, leaving irrecoverable fractures after upsampling. Load balance alone, however, does not resolve boundary inconsistency: padding restores the missing context around each chunk but replicates active cells, while direct averaging transfers unstable edge features into visible seams. More importantly, methods that rely on global attention cannot freely change partition boundaries because a chunk no longer receives the same global context. Prior sparse grid systems therefore treat a global, unchunked encoder as necessary for preserving latent integrity, applying chunking primarily in the decoder or tying it to a fixed partition (Ren et al., 2024; He et al., 2025; Wu et al., 2025b).

Our starting point is a representation principle: a spatial latent can be assembled consistently from independently processed regions when both the learned operators and the assembly rule are local. Based on this principle, we develop ChunkVAE, a sparse grid VAE using sparse convolutions, windowed attention, and pointwise pruning, with neither global attention nor positional encoding. Its encoder and decoder partitions are independent, and the inference chunk budget need not match

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.

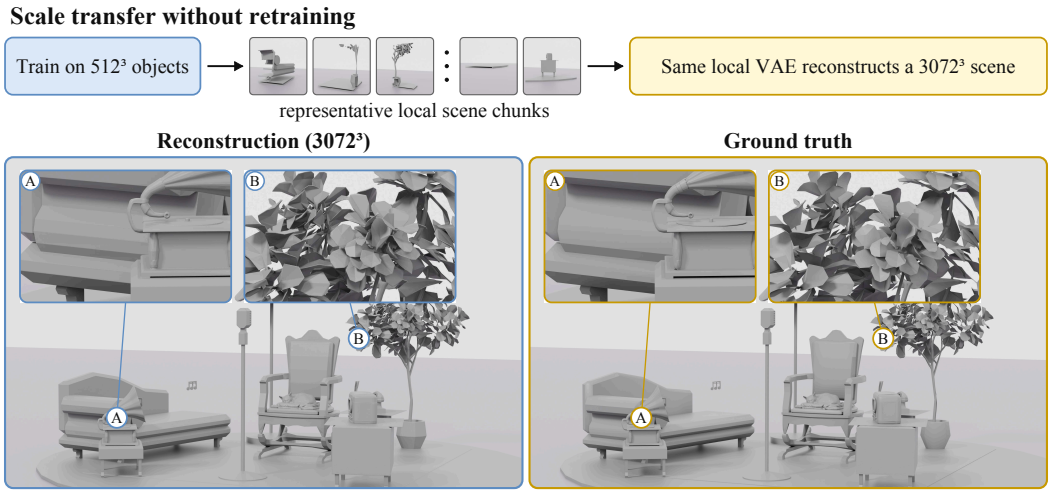


Figure 1: Resolution-flexible inference. Trained on chunks from 512^3 objects, the same local VAE reconstructs a 3072^3 scene without retraining. ***Best viewed with zoom-in.***

the training budget. This separates three quantities that are otherwise coupled: global resolution, per-device memory, and training chunk size. Figure 1 illustrates the resulting resolution flexibility at inference.

Two deterministic operators support this learned representation. Balanced Binary Object Partitioning (BBOP) recursively sorts active cells along their longest axis and divides them into equal-rank halves. It improves load balance without the large overlap induced by clustering. S-Curve weighted stitching assigns low confidence to the exterior of padded regions and smoothly increases confidence toward each chunk core. The same operator assembles encoder features and decoder outputs.

Our contributions are:

- We formulate chunk-wise sparse grid compression around local learned operators, enabling independent encoder and decoder partitions and different chunk budgets at training and inference.
- We introduce BBOP and S-Curve stitching to address load imbalance, overlap, and boundary reliability. Their effects are measured using balance, replication, and reconstruction.
- We demonstrate state-of-the-art overall object reconstruction from 512^3 to 1536^3, alongside leading image-to-3D generation metrics at 1536^3, and quantify memory, runtime, and resolution trade-offs.

2 Related Work

Global 3D latents. VecSet VAEs encode point samples into a fixed collection of latent vectors and decode occupancy or signed distance values at query points (Zhang et al., 2023; Zhao et al., 2023; Li et al., 2024; Zhang et al., 2024; Li et al., 2025b; Chen et al., 2025). Scaling tokens, surface sampling, and query resolution improves detail, while the encoder and decoder still bridge discrete samples and a continuous field. VoxSet methods inject spatial queries to retain more locality (Lai et al., 2025; Jia et al., 2025). These representations offer a compact global interface for diffusion or flow models, but increasing local detail raises the number of global tokens.

Sparse grid autoencoders. Dense voxel VAEs are limited by volumetric memory (Cheng et al., 2023; Li et al., 2023). XCube (Ren et al., 2024) and TRELLIS (Xiang et al., 2024) use sparse grids to concentrate computation near surfaces. SparseFlex (He et al., 2025), Direct3D-S2 (Wu et al., 2025b), Sparc3D (Li et al., 2025c), and TRELLIS.2 (Xiang et al., 2025) improve decoding, geometric representations, or high-resolution generation. XCube, SparseFlex, and Direct3D-S2 process spatial

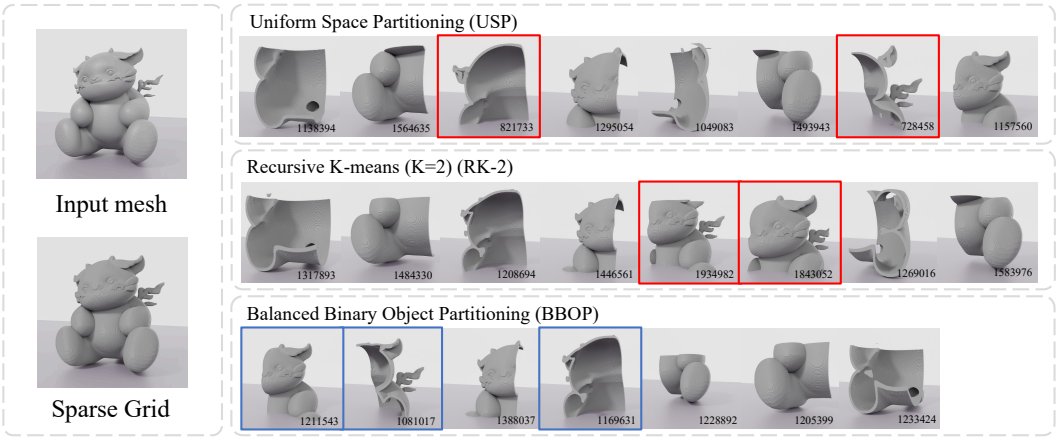


Figure 2: Partition strategies. Uniform Space Partitioning (USP) ignores the active cell distribution. Recursive K-means with $K = 2$ (RK-2) balances counts but produces large overlapping boxes. BBOP balances active cells while retaining compact spatial support. Numbers indicate chunk sizes.

chunks, but use uniform partitions and retain a global encoder. In contrast, our question is whether the latent itself can be encoded, assembled, repartitioned, and decoded locally without losing its downstream utility.

Latent 3D generation. Modern 3D generation moves expensive denoising or flow matching into autoencoder latents (Peebles & Xie, 2023; Ma et al., 2024; Black Forest Labs et al., 2025). The latent may be a triplane (Gupta et al., 2023; Lan et al., 2024), vector set (Li et al., 2025a; Hunyuan3D Team et al., 2025), or sparse grid (Xiang et al., 2024; Wu et al., 2025a). ChunkVAE preserves a stitched global latent for compatibility with these models while making the VAE computation local. The downstream generative model remains global in our current system.

3 Method

3.1 Local Compression of a Global Sparse Grid

Let a sparse grid at resolution R be

$$\mathcal{X} = \{(\mathbf{f}_j, \mathbf{c}_j)\}_{j=1}^{L}, \qquad \mathbf{f}_j \in \mathbb{R}^C, \quad \mathbf{c}_j \in \{0, \dots, R-1\}^3, \tag{1}$$

where $\mathbf{f}_j$ is a scaled signed distance value and $\mathbf{c}_j$ is its coordinate. A partition operator produces overlapping chunks $\mathcal{P}(\mathcal{X}) = \{\mathcal{X}_i\}_{i=1}^{N_c}$ whose active cell count is bounded by a budget B. An encoder E, a decoder D, and a stitching operator $\mathcal{S}$ yield

$$\mathbf{Z} = \mathcal{S}\big(\{E(\mathcal{X}_i)\}_{i=1}^{N_c}\big), \qquad \hat{\mathcal{X}} = \mathcal{S}\big(\{D(\tilde{\mathbf{Z}}_k)\}_{k=1}^{N_d}\big), \tag{2}$$

where $\{\tilde{\mathbf{Z}}_k\}$ is a new partition of the stitched latent. Different encoder and decoder partitions are valid because all learned operators have bounded receptive fields and use global sparse coordinates; convolution coordinates and Swin window assignments never reset at chunk origins. Only padded exteriors truncate context, whose uncertainty is handled by $\mathcal{S}$.

3.2 Balanced Partitioning and Weighted Stitching

Balanced Binary Object Partitioning. Figure 2 compares the three strategies. USP divides the bounding volume uniformly, so its memory balance depends on shape geometry. RK-2 instead clusters active cells, but spatial bounding boxes around the clusters overlap substantially. BBOP preserves spatial coherence and balances active cells. For a set of coordinates, it selects the longest bounding box axis, sorts cells along that axis with lexicographic tie-breaking, splits the ordered cells into two equal-rank halves, and recurses until each padded leaf contains at most B cells. Each leaf box is padded to provide network context. Rank splitting guarantees two nonempty, strictly smaller children, while longest-axis splitting keeps boxes compact. Algorithm 1 is provided in Appendix B.

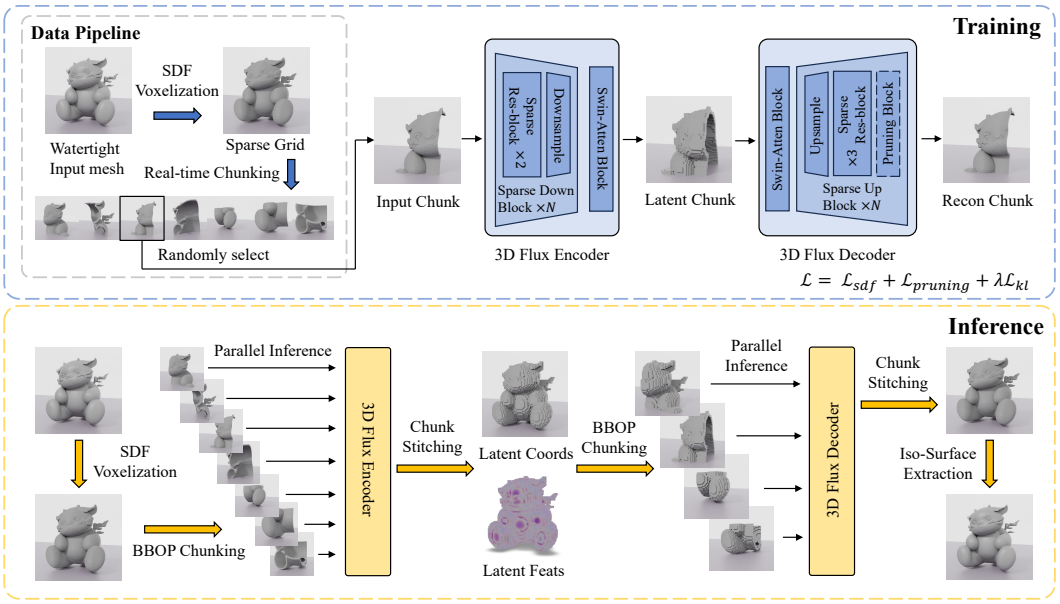


Figure 3: ChunkVAE uses local operators during training and inference. Training reconstructs sampled chunks. At inference, encoder chunks are stitched into a global latent, repartitioned independently, decoded, and stitched into the output grid.

S-Curve weighted stitching. Features near the exterior of a padded chunk have less context than features near its core because part of their receptive field falls outside the region processed by the network. Their predictions are therefore less reliable, even though the overlap remains necessary for connecting adjacent chunks. Uniform averaging ignores this spatial variation and gives boundary and interior predictions equal influence, allowing unstable edge features to propagate into the stitched result. We instead use the distance from the padded exterior to the chunk core as a deterministic proxy for prediction reliability. Let $[s_p, e_p]$ be a padded interval and $[s_c, e_c]$ its core. We use the smooth transition

$$S(t;p) = \begin{cases} \frac{1}{2}(2t)^p, & 0 \le t < \frac{1}{2}, \\ 1 - \frac{1}{2}\big(2(1-t)\big)^p, & \frac{1}{2} \le t \le 1, \end{cases} \tag{3}$$

and define the one-dimensional reliability

$$w(c) = \begin{cases} W_{\min} + \Delta W S\left(\frac{c-s_p}{s_c-s_p}; p\right), & s_p \le c < s_c, \\ W_{\max}, & s_c \le c < e_c, \\ W_{\max} - \Delta W S\left(\frac{c-e_c}{e_p-e_c}; p\right), & e_c \le c \le e_p. \end{cases} \tag{4}$$

We set $\Delta W = W_{\max} - W_{\min}$, $p = 2$, $W_{\min} = 0.01$, and $W_{\max} = 0.99$. The 3D weight is $W(\mathbf{c}) = \prod_{d\in\{x,y,z\}} w(c_d)$, and duplicate features are combined by a scatter-weighted average. The separable product further suppresses coordinates near padded edges and corners, where context is missing along multiple axes. Normalization by the accumulated weights preserves feature scale as the number and arrangement of overlapping chunks vary. This mitigates seams without aligned partitions or learned blending. Algorithm 2 appears in Appendix B; Figure 6 compares padded unique selection, full and restricted means, and S-Curve weighting with $p \in \{1, 2\}$.

3.3 ChunkVAE Architecture

Figure 3 summarizes training and inference. ChunkVAE adapts the FLUX.1 image VAE (Black Forest Labs et al., 2025) to the sparse framework of TRELLIS (Xiang et al., 2024), using three sparse-convolution downsampling and upsampling stages with pointwise supervised pruning. A single Swin bottleneck with window size 8 replaces global attention. These local operators use no positional encoding; sparse coordinates and window assignments retain global indices across chunks.

Training samples one BBOP chunk and reconstructs the same spatial region. The objective is

$$\mathcal{L} = \mathcal{L}_{\text{sdf}} + \sum_{i=1}^{3} \mathcal{L}^{i}_{\text{prune}} + \lambda \mathcal{L}_{\text{KL}}, \qquad \lambda = 10^{-6}. \tag{5}$$

At inference, the encoder and decoder follow Equation 2. Larger decoder chunks reduce stitched boundaries, while smaller chunks reduce peak memory. Section 4.5 evaluates reconstruction and latent consistency under changed budgets and partitions.

4 Experiments

4.1 Setup and Evaluation Protocol

Data and training. We convert meshes to watertight surfaces using flood filling (Li et al., 2025c) followed by least-squares optimization of geometric details, then voxelize a narrow SDF band extending two cells on each side of the surface. Each resolution uses watertight voxelization at that same resolution. Training uses about 500K assets curated from Objaverse and Objaverse-XL (Deitke et al., 2023b;a). We first train at 512^3, then fine-tune checkpoints at 1024^3 and 1536^3. The main model uses 32 GPUs, a total batch size of 64, AdamW with initial learning rate 10^{-4} and cosine decay, FP16 mixed precision, and 200K iterations. It reaches $8\times$ spatial compression in about 3.5 days. Architecture details appear in Appendix A.

Benchmarks and metrics. We evaluate on Toys4K (Stojanov et al., 2021), ABO (Collins et al., 2022), and 95 complex High-quality Models (HQM) that do not overlap with training. Metrics such as Chamfer Distance (CD), Absolute Normal Consistency (ANC), and F-score at 0.001 are computed using point-to-surface distances following LATTICE (Lai et al., 2025). All methods are evaluated against the same watertight target surfaces. VecSet and VoxSet baselines use 4096 latent tokens and 512^3 marching cubes. Sparse grid inputs and extracted meshes use the reported evaluation resolution.

4.2 Reconstruction and Resolution Scaling

Table 1: Reconstruction across resolutions. Metrics are CD ($\times 10^4$), ANC ($\times 10^2$), and F-score ($\times 10^2$). At 512^3, the 175M and 212M models are trained with chunk budgets below 8M and 1M cells, respectively. † *denotes our re-implementation.*

Resolution	Method	Toys4K			ABO			HQM		
		CD↓	ANC↑	F↑	CD↓	ANC↑	F↑	CD↓	ANC↑	F↑
512^3	Dora (Chen et al., 2025)	7.52	99.07	94.69	6.54	99.56	95.61	13.47	96.91	82.36
	Hunyuan3D-2.1 (Hunyuan3D Team et al., 2025)	8.02	98.96	92.08	6.01	99.45	96.22	14.20	96.49	79.38
	SparseFlex (He et al., 2025)	2.49	99.46	99.16	2.34	99.70	99.59	4.48	98.30	96.82
	Direct3D-S2 (Wu et al., 2025b)	5.83	98.98	97.22	4.74	99.47	98.74	9.60	96.79	88.90
	LATTICE† (Lai et al., 2025)	5.33	99.06	95.72	4.32	99.54	96.77	14.28	95.12	82.37
	TRELLIS.2 (Xiang et al., 2025)	1.45	99.45	99.61	1.04	99.72	99.89	3.43	98.24	98.32
	ChunkVAE (175M)	1.60	99.69	99.64	1.21	99.88	99.94	3.00	98.89	98.59
	ChunkVAE (212M)	**1.21**	**99.77**	**99.78**	**1.03**	**99.91**	**99.96**	**2.44**	**99.14**	**99.18**
1024^3	SparseFlex (He et al., 2025)	1.28	99.64	99.68	1.28	99.80	99.88	2.38	98.59	98.75
	Direct3D-S2 (Wu et al., 2025b)	3.38	99.37	99.47	3.19	99.63	99.76	4.32	97.83	97.79
	TRELLIS.2 (Xiang et al., 2025)	0.47	99.70	99.98	**0.35**	99.84	99.99	1.20	98.81	**99.89**
	ChunkVAE (212M)	**0.45**	**99.89**	**99.98**	0.67	**99.95**	**99.99**	**0.90**	**99.45**	99.88
1536^3	TRELLIS.2 (Xiang et al., 2025)	0.40	99.78	99.99	**0.27**	99.86	99.99	0.67	99.04	99.97
	ChunkVAE (212M)	**0.36**	**99.92**	**99.99**	0.39	**99.97**	**99.99**	**0.53**	**99.58**	**99.97**

Table 1 shows that ChunkVAE delivers state-of-the-art overall reconstruction quality from 512^3 to 1536^3. At 512^3, the 212M model reduces CD relative to TRELLIS.2 by 16.6% on Toys4K and 28.9% on HQM, while obtaining the best ANC and F-score on all three benchmarks. Each resolution is evaluated against separately generated, resolution-matched ground truth, and baseline results use the checkpoints released by the corresponding methods. Figure 4 further shows sharper, more continuous thin structures, including cello strings, front-grille lattices, horse manes, and tree crowns, while avoiding the holes visible in TRELLIS.2 reconstructions.

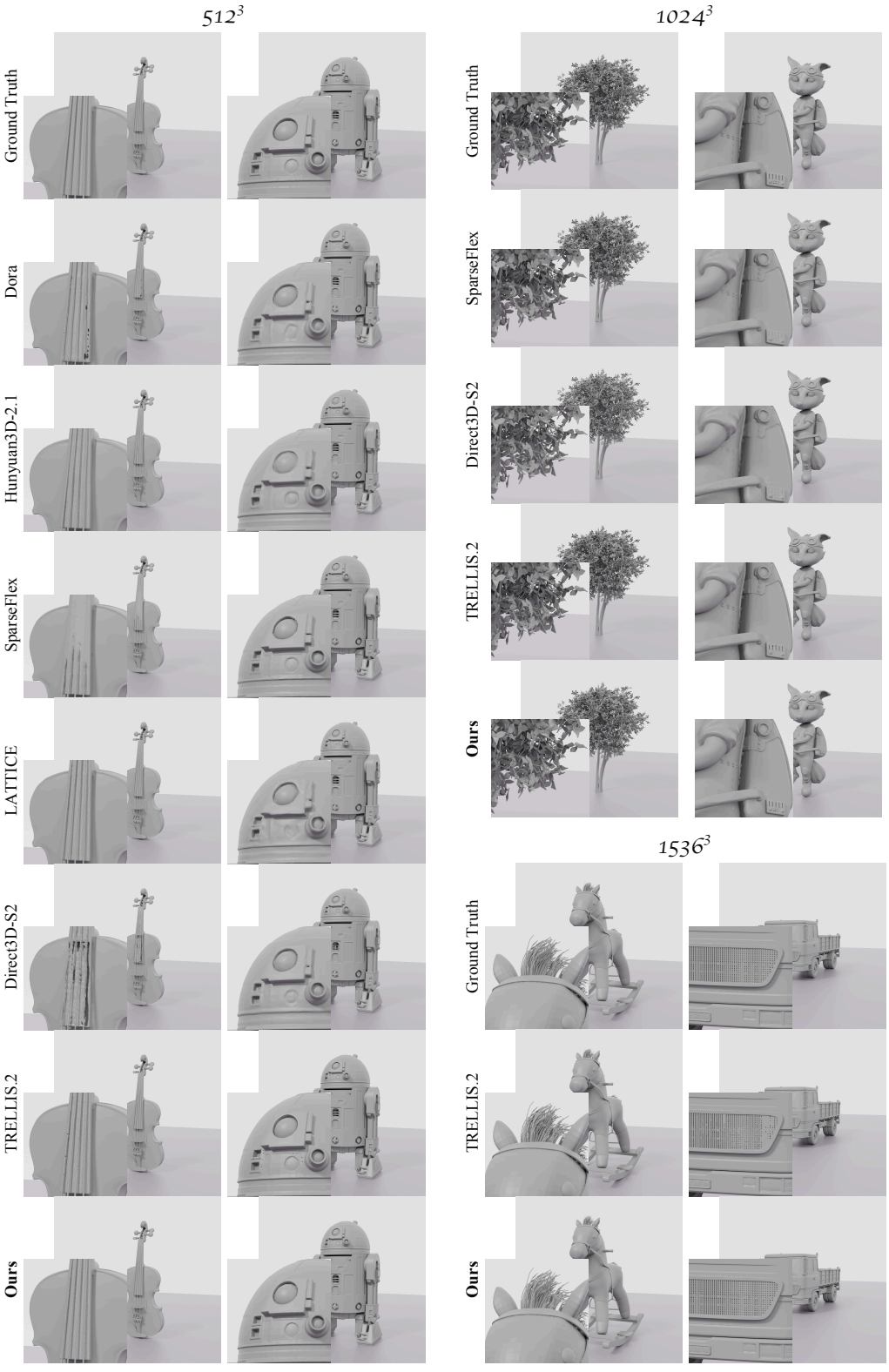


Figure 4: Qualitative Toys4K reconstruction at 512^3, 1024^3, and 1536^3. *Best viewed with zoom-in.*

Table 2: Average 512^3 inference on HQM objects using one GPU. Parentheses give mean time per chunk, an idealized parallel compute time when enough workers are available; scheduling and communication are excluded. Memory is peak allocated memory.

Metric	SparseFlex	Direct3D-S2	TRELLIS.2	ChunkVAE cell budget			
				<1M	<2M	<4M	No chunking
Time (s)↓	8.43	22.07	14.90	10.46 (**2.34**)	7.61 (3.29)	6.19 (4.67)	**5.71**
Memory (MB)↓	7267.60	21226.11	**2668.47**	**4569.84**	6394.03	10244.87	16219.47

Table 3: Image-to-3D evaluation following the LATTICE protocol.

Method	ULIP-T↑	ULIP-I↑	Uni3D-T↑	Uni3D-I↑
CraftsMan (Li et al., 2024)	0.074	0.129	0.237	0.298
Hunyuan3D-2.1 (Hunyuan3D Team et al., 2025)	0.077	<u>0.130</u>	0.251	0.315
Direct3D-S2 (Wu et al., 2025b)	0.074	0.122	0.247	0.314
LATTICE (Lai et al., 2025)	<u>0.078</u>	<u>0.130</u>	<u>0.254</u>	0.315
TRELLIS.2 (Xiang et al., 2025)	0.077	0.124	0.245	<u>0.317</u>
Ours (1536^3)	**0.079**	**0.134**	**0.255**	**0.318**

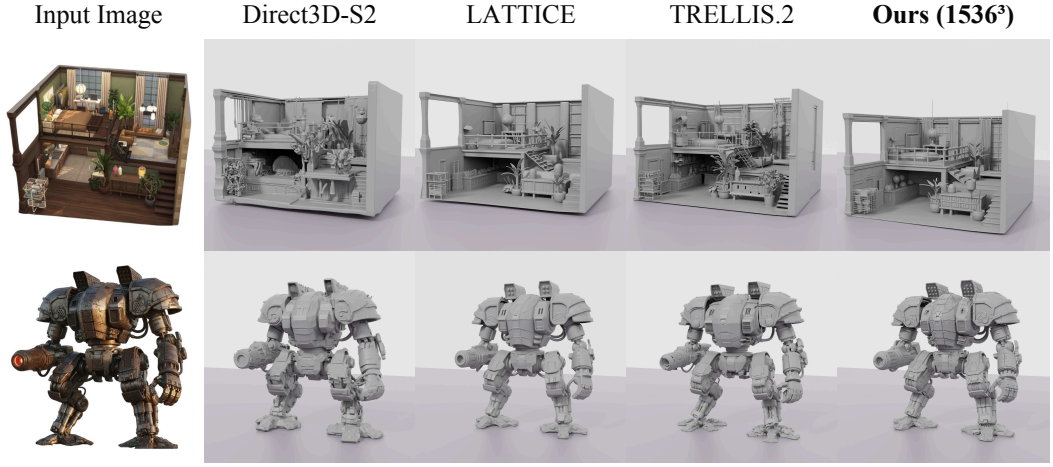


Figure 5: Image-to-3D comparison with in-the-wild images. *Best viewed with zoom-in.*

4.3 Efficiency

Table 2 compares inference time and peak allocated memory at 512^3. ChunkVAE is faster than Direct3D-S2 and TRELLIS.2 under every tested budget: it takes 5.71 s without chunking and 10.46 s even below 1M cells, versus 22.07 s and 14.90 s for the two baselines. The <2M setting also improves over SparseFlex in both latency (7.61 versus 8.43 s) and memory (6.39 versus 7.27 GB). Within ChunkVAE, reducing the budget from no chunking to <1M lowers memory by 71.8%, from 16.22 to 4.57 GB, while independent chunks reduce idealized parallel compute to 2.34 s per chunk. TRELLIS.2 uses fewer cells because its o-voxel represents only a surface layer; our SDF grid extends two cells inside and outside the surface to maintain watertight continuity and avoid the hole artifacts visible in the TRELLIS.2 reconstructions in Figure 4, naturally requiring more memory. Overall, ChunkVAE provides the strongest measured balance of inference speed and adjustable memory, with <2M serving as a practical default.

4.4 Image-to-3D Generation

We use the two-stage flow-matching pipeline of TRELLIS (Xiang et al., 2024), replacing stage two's projected DINOv2 features with ChunkVAE SDF latents; sampling uses 25 steps and classifier-free guidance 7.5. Table 3 and Figure 5 provide quantitative and qualitative comparisons, respectively.

Table 4: Partitioning on Toys4K, ABO, and HQM. Each resolution group reports CSCV, CRF, and Success Ratio. CRF is bounded below by one, so lower is better.

Dataset	Method	512^3			1024^3			1536^3		
		CSCV↓	CRF↓	Succ. Ratio↑	CSCV↓	CRF↓	Succ. Ratio↑	CSCV↓	CRF↓	Succ. Ratio↑
Toys4K	USP	0.220	2.058	1.000	0.644	1.967	0.974	0.684	1.951	0.927
	RK-2	0.082	1.344	1.000	0.194	1.815	1.000	0.204	1.954	0.998
	BBOP	**0.031**	**1.191**	**1.000**	**0.103**	**1.426**	**1.000**	**0.127**	**1.499**	**1.000**
ABO	USP	0.260	1.648	1.000	0.642	1.865	0.990	0.603	1.904	0.949
	RK-2	0.112	1.364	1.000	0.214	1.725	1.000	0.223	1.802	0.990
	BBOP	**0.053**	**1.276**	**1.000**	**0.137**	**1.510**	**1.000**	**0.156**	**1.558**	**1.000**
HQM	USP	0.318	1.950	1.000	0.884	1.887	0.989	0.876	2.303	0.800
	RK-2	0.137	1.797	1.000	0.230	2.437	1.000	0.226	2.584	0.989
	BBOP	**0.052**	**1.315**	**1.000**	**0.137**	**1.625**	**1.000**	**0.150**	**1.702**	**1.000**

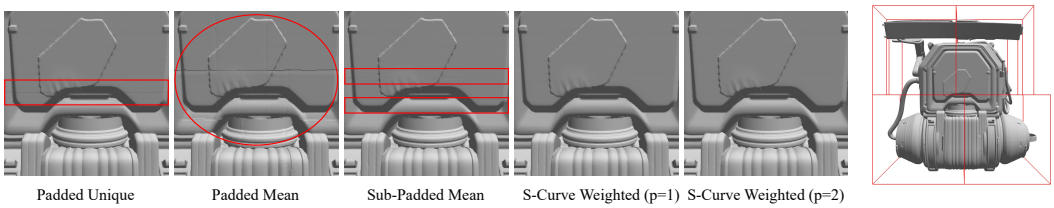


Figure 6: Stitching ablation under a fixed partition and trained VAE. Red boxes identify boundary artifacts. S-Curve weighting reduces the contribution of features near padded exteriors.

Under the LATTICE protocol, ChunkVAE achieves state-of-the-art scores, improving over the strongest baseline by $+0.004$ on ULIP-I and $+0.001$ on Uni3D-I. Qualitatively, our generated shapes contain richer and more complete details, including steps and bookshelves in the indoor scene and rivets and articulated joints on the robot, while aligning more faithfully with the input images.

4.5 What Makes Chunking Consistent?

Partition quality. For chunk sizes s_i with mean μ, Chunk Size Coefficient of Variation is $\mathrm{CSCV} = \mu^{-1}\sqrt{N_c^{-1}\sum_i (s_i-\mu)^2}$. Chunk Replication Factor is $\mathrm{CRF} = \sum_i |c_i| / |\cup_i c_i|$. Lower CSCV indicates better load balance and CRF close to one indicates less overlap. Success Ratio is the fraction of test shapes producing a complete reconstruction. Table 4 shows that BBOP reconstructs every shape at every resolution on all three datasets. The distinction is largest on HQM at 1536^3: USP succeeds on only 0.800 of the shapes, while RK-2 reaches 0.989 and BBOP reaches 1.000. At that resolution BBOP also reduces CSCV to 0.150, compared with 0.876 for USP, and avoids the high replication of RK-2 (CRF 1.702 versus 2.584). Thus, BBOP achieves the best trade-off between balanced chunk sizes and limited overlap while preserving complete reconstructions.

Boundary reliability. Partition quality and stitching answer different questions: Table 4 measures workload and overlap, whereas Figure 6 fixes the partition and VAE and changes only the merge rule. Padded unique selection, full averaging, and restricted averaging retain boundary artifacts. Uniform averaging gives boundary and interior predictions equal weight, allowing unreliable boundary features to contaminate the fused result. Restricting the averaging region removes these features but introduces a hard cutoff where the set of contributing chunks changes, potentially creating new seams. S-Curve weighting instead retains the overlap and varies each contribution smoothly with available context; $p = 2$ suppresses exterior features more strongly than the displayed $p = 1$ variant.

Chunk and partition decoupling. Table 5 evaluates decoupling across training budget, inference budget, and encoder–decoder partition choice. In the left panel, changing the training budget from $<$1M to $<$100K at any fixed inference budget gives $|\Delta\mathrm{CD}| \leq 0.08$, showing that inference

chunks need not match those used for training. Increasing the inference budget from $<$1M to $<$8M improves CD from 2.99/3.04 to 2.42/2.34 for the two training budgets because fewer boundaries are stitched. In the right panel, switching the encoder or decoder to global processing gives only $\Delta\text{CD} = +0.14/+0.19$, $\Delta\text{ANC} = 0.00/-0.03$, and $\Delta F = -0.07/-0.22$ relative to chunk-wise processing. Together, these small differences show that training and inference budgets, as well as encoder and decoder partitions, can be chosen independently without retraining.

Table 5: Decoupling on HQM. Left: CD across training and inference budgets. Right: global encoder or decoder processing (dash) against the $<$1M encoder/$<$2K decoder default.

Train budget	Inference budget (CD)			
	$<$1M	$<$2M	$<$4M	$<$8M
$<$1M	2.99	2.70	2.59	2.42
$<$100K	3.04	2.71	2.57	2.34

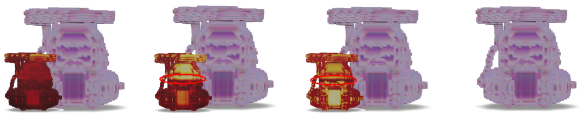
Encoder budget	Decoder budget	CD	ANC	F
$<$1M	$<$2K	2.99	99.04	98.89
–	$<$2K	3.13	99.04	98.82
$<$1M	–	3.18	99.01	98.67

Stitched latent integrity. Figure 7 provides a visual comparison: the global PCA structure remains stable across encoder budgets, while the difference insets localize residuals near chunk boundaries (highlighted by red circles), demonstrating the need for our proposed chunk stitching. Visual similarity alone, however, does not verify spatio-semantic integrity. We therefore adapt a PointNet classifier (Qi et al., 2017) to accept 16-channel latent features and train it on global ModelNet40 (Wu et al., 2015) latents extracted with unchunked VAE processing, achieving 99.52% training accuracy and 76.61% test accuracy. We then freeze the classifier and evaluate stitched latents without adaptation. Relative to no chunking, Table 6 reports $\Delta\text{Accuracy}$ values of -0.97, -0.36, and -0.24 points for budgets below 1M, 2M, and 4M cells, respectively. Since the classifier never observes stitched latents during training, the nearly unchanged accuracy confirms that stitching preserves latent feature integrity for downstream tasks.

Table 6: Frozen ModelNet40 accuracy ($\times 100$); Δ from no chunking.

Encoder processing	Accuracy	Δ
Chunk budget $<$1M	75.64	-0.97
Chunk budget $<$2M	76.25	-0.36
Chunk budget $<$4M	76.37	-0.24
No chunking	76.61	0.00

Figure 7: Latent PCA (purple) across encoder budgets; insets show differences to global encoding.



5 Scope, Limitations, and Conclusion

ChunkVAE makes sparse grid VAE computation local while preserving a stitched global latent. Local learned operators, BBOP, and S-Curve stitching decouple training and inference budgets as well as encoder and decoder partitions. Experiments demonstrate state-of-the-art overall object reconstruction from 512^3 to 1536^3 and a strong balance between inference speed and adjustable memory. The ablations validate robust chunking, stitching, and latent integrity across partition choices. This preserved latent representation, in turn, supports state-of-the-art image-to-3D generation.

Despite these results, the current formulation has two main limitations. *(1) Watertight Constraint.* To reduce geometric artifacts in generated outputs, our current implementation uses watertight SDF geometry, which excludes open surfaces and some internal structures. Extending the local representation to non-watertight geometry while retaining reliable boundary assembly is an important direction, for example by applying ChunkVAE to the o-voxel representation of TRELLIS.2 (Xiang et al., 2025). *(2) Global DiT Training Bottleneck.* Although VAE encoding and decoding are chunk-wise, the downstream DiT still processes the stitched global latent, whose token count grows with global resolution. Fully scalable generation therefore requires extending the same locality principle to chunk-wise generative training and inference.

AI USE STATEMENT

Generative AI tools were used only to edit the text for readability and organize existing related-work material. They were not used to generate research ideas or hypotheses, develop theoretical or mathematical claims or proofs, design the methodology or experiments, generate or process data, interpret results, translate text, create figures or other artifacts, or implement code. The authors reviewed all AI-assisted edits and take full responsibility for the final content of the paper.

REPRODUCIBILITY STATEMENT

Section 4, particularly the Setup and Evaluation Protocol, provides the data, training, benchmark, and evaluation-metric details needed to reproduce our experiments. Appendix A specifies the ChunkVAE architecture, and Appendix B provides detailed algorithmic specifications for both chunk partitioning and chunk stitching. Together, these details are sufficient to reproduce the work.

REFERENCES

Black Forest Labs, Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, Sumith Kulal, Kyle Lacey, Yam Levi, Cheng Li, Dominik Lorenz, Jonas Müller, Dustin Podell, Robin Rombach, Harry Saini, Axel Sauer, and Luke Smith. FLUX.1 Kontext: Flow matching for in-context image generation and editing in latent space. *arXiv preprint arXiv:2506.15742*, 2025.

Rui Chen, Jianfeng Zhang, Yixun Liang, Guan Luo, Weiyu Li, Jiarui Liu, Xiu Li, Xiaoxiao Long, Jiashi Feng, and Ping Tan. Dora: Sampling and benchmarking for 3D shape variational auto-encoders. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 16251–16261, 2025.

Yen-Chi Cheng, Hsin-Ying Lee, Sergey Tulyakov, Alexander G Schwing, and Liang-Yan Gui. SDFusion: Multimodal 3D shape completion, reconstruction, and generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4456–4465, 2023.

Jasmine Collins, Shubham Goel, Kenan Deng, Achleshwar Luthra, Leon Xu, Erhan Gundogdu, Xi Zhang, Tomas F Yago Vicente, Thomas Dideriksen, Himanshu Arora, et al. ABO: Dataset and benchmarks for real-world 3D object understanding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 21126–21136, 2022.

Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, et al. Objaverse-XL: A universe of 10M+ 3D objects. *Advances in Neural Information Processing Systems*, 36:35799–35813, 2023a.

Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3D objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13142–13153, 2023b.

Anchit Gupta, Wenhan Xiong, Yixin Nie, Ian Jones, and Barlas Oğuz. 3DGen: Triplane latent diffusion for textured mesh generation. *arXiv preprint arXiv:2303.05371*, 2023.

Xianglong He, Zi-Xin Zou, Chia-Hao Chen, Yuan-Chen Guo, Ding Liang, Chun Yuan, Wanli Ouyang, Yan-Pei Cao, and Yangguang Li. SparseFlex: High-resolution and arbitrary-topology 3D shape modeling. *arXiv preprint arXiv:2503.21732*, 2025.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (GELUs). *arXiv preprint arXiv:1606.08415*, 2016.

Hunyuan3D Team, Shuhui Yang, Mingxin Yang, Yifei Feng, Xin Huang, Sheng Zhang, Zebin He, Di Luo, Haolin Liu, Yunfei Zhao, et al. Hunyuan3D 2.1: From images to high-fidelity 3D assets with production-ready PBR material. *arXiv preprint arXiv:2506.15442*, 2025.

Tanghui Jia, Dongyu Yan, Dehao Hao, Yang Li, Kaiyi Zhang, Xianyi He, Lanjiong Li, Yuhan Wang, Jinnan Chen, Lutao Jiang, et al. UltraShape 1.0: High-fidelity 3D shape generation via scalable geometric refinement. *arXiv preprint arXiv:2512.21185*, 2025.

Zeqiang Lai, Yunfei Zhao, Zibo Zhao, Haolin Liu, Qingxiang Lin, Jingwei Huang, Chunchao Guo, and Xiangyu Yue. LATTICE: Democratize high-fidelity 3D generation at scale. *arXiv preprint arXiv:2512.03052*, 2025.

Yushi Lan, Fangzhou Hong, Shuai Yang, Shangchen Zhou, Xuyi Meng, Bo Dai, Xingang Pan, and Chen Change Loy. LN3Diff: Scalable latent neural fields diffusion for speedy 3D generation. In *European Conference on Computer Vision*, pp. 112–130. Springer, 2024.

Muheng Li, Yueqi Duan, Jie Zhou, and Jiwen Lu. Diffusion-SDF: Text-to-shape via voxelized diffusion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12642–12651, 2023.

Weiyu Li, Jiarui Liu, Rui Chen, Yixun Liang, Xuelin Chen, Ping Tan, and Xiaoxiao Long. CraftsMan: High-fidelity mesh generation with 3D native generation and interactive geometry refiner. *arXiv preprint arXiv:2405.14979*, 2024.

Weiyu Li, Xuanyang Zhang, Zheng Sun, Di Qi, Hao Li, Wei Cheng, Weiwei Cai, Shihao Wu, Jiarui Liu, Zihao Wang, et al. Step1X-3D: Towards high-fidelity and controllable generation of textured 3D assets. *arXiv preprint arXiv:2505.07747*, 2025a.

Yangguang Li, Zi-Xin Zou, Zexiang Liu, Dehu Wang, Yuan Liang, Zhipeng Yu, Xingchao Liu, Yuan-Chen Guo, Ding Liang, Wanli Ouyang, et al. TripoSG: High-fidelity 3D shape synthesis using large-scale rectified flow models. *arXiv preprint arXiv:2502.06608*, 2025b.

Zhihao Li, Yufei Wang, Heliang Zheng, Yihao Luo, and Bihan Wen. Sparc3D: Sparse representation and construction for high-resolution 3D shapes modeling. *arXiv preprint arXiv:2505.14521*, 2025c.

Nanye Ma, Mark Goldstein, Michael S Albergo, Nicholas M Boffi, Eric Vanden-Eijnden, and Saining Xie. SiT: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In *European Conference on Computer Vision*, pp. 23–40. Springer, 2024.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4195–4205, 2023.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. PointNet: Deep learning on point sets for 3D classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 652–660, 2017.

Xuanchi Ren, Jiahui Huang, Xiaohui Zeng, Ken Museth, Sanja Fidler, and Francis Williams. XCube: Large-scale 3D generative modeling using sparse voxel hierarchies. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4209–4219, 2024.

Stefan Stojanov, Anh Thai, and James M Rehg. Using shape to categorize: Low-shot learning with an explicit shape bias. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1798–1808, 2021.

Guanjun Wu, Jiemin Fang, Chen Yang, Sikuang Li, Taoran Yi, Jia Lu, Zanwei Zhou, Jiazhong Cen, Lingxi Xie, Xiaopeng Zhang, et al. UniLat3D: Geometry-appearance unified latents for single-stage 3D generation. *arXiv preprint arXiv:2509.25079*, 2025a.

Shuang Wu, Youtian Lin, Feihu Zhang, Yifei Zeng, Yikang Yang, Yajie Bao, Jiachen Qian, Siyu Zhu, Xun Cao, Philip Torr, et al. Direct3D-S2: Gigascale 3D generation made easy with spatial sparse attention. *arXiv preprint arXiv:2505.17412*, 2025b.

Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3D ShapeNets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1912–1920, 2015.

Jianfeng Xiang, Zelong Lv, Sicheng Xu, Yu Deng, Ruicheng Wang, Bowen Zhang, Dong Chen, Xin Tong, and Jiaolong Yang. Structured 3D latents for scalable and versatile 3D generation. *arXiv preprint arXiv:2412.01506*, 2024.

Jianfeng Xiang, Xiaoxue Chen, Sicheng Xu, Ruicheng Wang, Zelong Lv, Yu Deng, Hongyuan Zhu, Yue Dong, Hao Zhao, Nicholas Jing Yuan, et al. Native and compact structured latents for 3D generation. *arXiv preprint arXiv:2512.14692*, 2025.

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

Longwen Zhang, Ziyu Wang, Qixuan Zhang, Qiwei Qiu, Anqi Pang, Haoran Jiang, Wei Yang, Lan Xu, and Jingyi Yu. CLAY: A controllable large-scale generative model for creating high-quality 3D assets. *ACM Transactions on Graphics (TOG)*, 43(4):1–20, 2024.

Zibo Zhao, Wen Liu, Xin Chen, Xianfang Zeng, Rui Wang, Pei Cheng, Bin Fu, Tao Chen, Gang Yu, and Shenghua Gao. Michelangelo: Conditional 3D shape generation based on shape-image-text aligned latent representation. In *Advances in Neural Information Processing Systems*, volume 36, pp. 73969–73982, 2023.

A Implementation Details

ChunkVAE uses latent dimension 16. Encoder channels are [128, 256, 512, 512], and decoder channels are [512, 512, 256, 128]. The bottleneck Swin block has eight heads of dimension 64 and uses no positional encoding. Each pointwise pruning head is a GELU-activated MLP (Hendrycks & Gimpel, 2016); teacher forcing supervises its loss using the corresponding intermediate encoder grid. Encoder and decoder padding are 32 and 4 cells, respectively.

Raw meshes are first repaired with heuristic hole filling and then converted into watertight surfaces before SDF computation. Flood filling identifies exterior empty cells, and least-squares optimization restores geometric details on the watertight surface. The surface field is then evaluated in a narrow band around the resulting boundary. Empirically, extending the band by two cells on each side prevents holes in reconstructed meshes without introducing excessive memory overhead.

B Partitioning and Stitching Algorithms

Algorithm 1 Balanced Binary Object Partitioning

Require: sparse grid X, coordinates C, padding P, cell budget B

$B_{\text{pad}} \leftarrow \text{Pad}(\text{BBox}(C), P)$
$X_c \leftarrow \{x \in X : x \in B_{\text{pad}}\}$
if $|X_c| > B$ **then**
  $a \leftarrow \arg\max_d(\max C_d - \min C_d)$
  $(c^{(1)}, \ldots, c^{(n)}) \leftarrow \text{Sort}(C, a)$ ▷ lexicographic tie-breaking
  $h \leftarrow \lfloor n/2 \rfloor$; $C_L \leftarrow \{c^{(1)}, \ldots, c^{(h)}\}$; $C_R \leftarrow \{c^{(h+1)}, \ldots, c^{(n)}\}$
  return $\text{BBOP}(X, C_L, P, B) \cup \text{BBOP}(X, C_R, P, B)$
else
  return $\{X_c\}$
end if

Algorithm 2 S-Curve Weighted Stitching

Require: chunks $\{(C_i, F_i, B_i^{\text{core}}, B_i^{\text{pad}})\}$

initialize global buffers F_{sum} and W_{sum}
for each chunk i **do**
  retain coordinates $\hat{C}_i$ inside B_i^{pad}
  compute $W_i = \prod_{d \in \{x,y,z\}} w(\hat{C}_{i,d})$ using Equation 4
  scatter add $F_i W_i$ to F_{sum} and W_i to W_{sum}
end for
return $F_{\text{sum}}/W_{\text{sum}}$ at occupied global coordinates

Algorithmic behavior. BBOP separates recursive split coordinates from padded execution boxes. Equal-rank splits preserve coverage and balance active cells, while padding supplies context. Changing B alters partition depth and boundary count, not the input grid or learned parameters. Stitching normalizes accumulated reliability to preserve scale and suppress padded-face predictions.

C Extended Qualitative Results

Reconstruction. Figure 8 localizes bicycle spokes and figurine ribbon detail at 512^3, a horse mane and mechanical appendage at 1024^3, and truck panels and mech parts at 1536^3. Silhouettes remain stable; differences concentrate in thin, repeated, or high-curvature geometry. Our results remain close to the ground truth, consistent with the saturated F-scores.

Image-conditioned generation. Figure 9 spans diverse in-the-wild inputs. Compared with the baselines, our results align more closely with the input images, preserve finer details such as machine logos, sculptural details, and clothing accessories on cartoon characters, and produce more complete geometry without breakage. In contrast, TRELLIS.2 exhibits frequent fractures and holes.

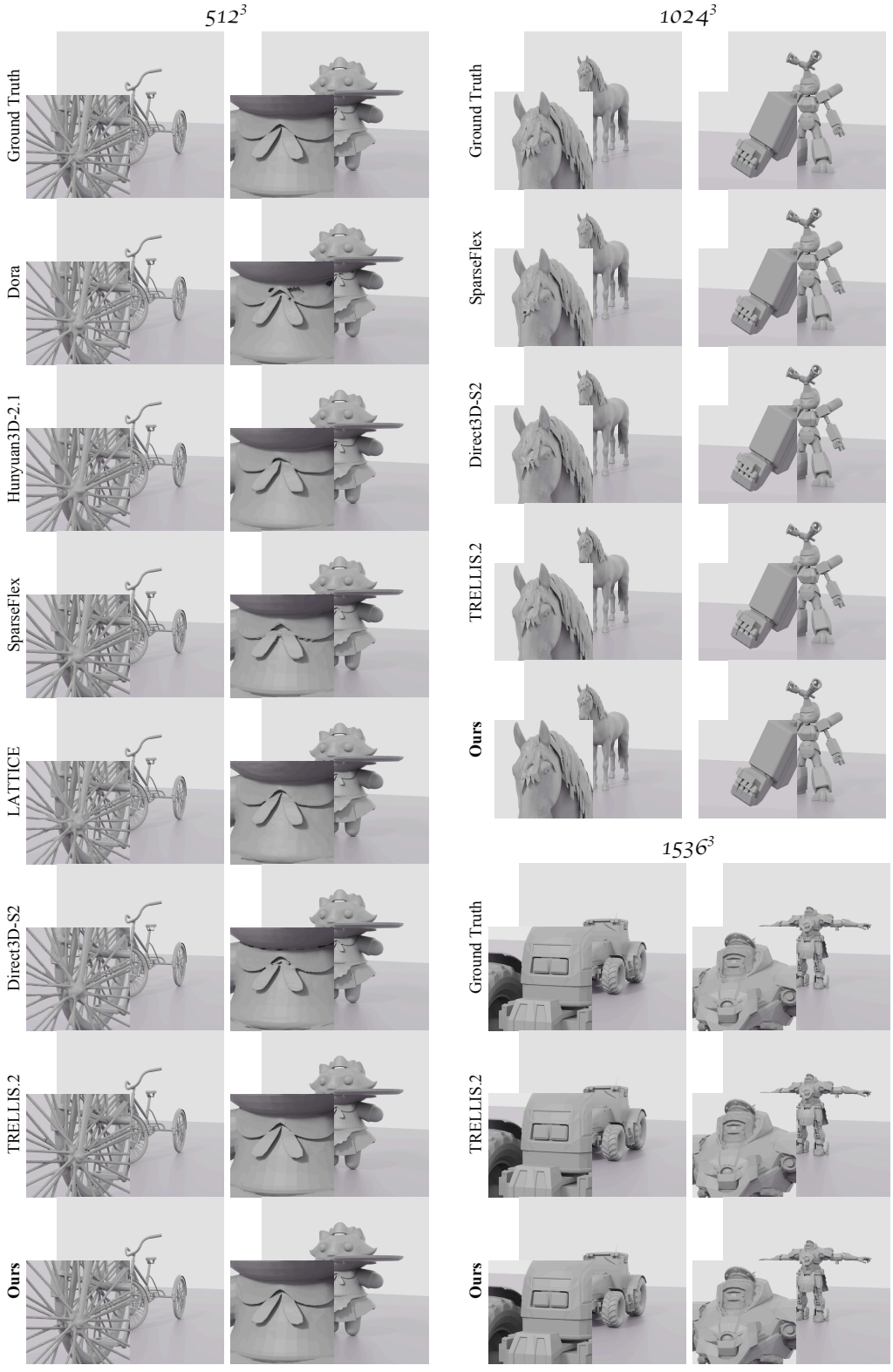


Figure 8: Additional qualitative VAE reconstruction on Toys4K at 512^3, 1024^3, and 1536^3. *Best viewed with zoom-in.*



Figure 9: Additional image-to-3D generation comparisons. *Best viewed with zoom-in.*