GaussianGPT: Towards Autoregressive 3D Gaussian Scene Generation

Nicolas von Lützow, Barbara Rössle, Katharina Schmid, and Matthias Nießner

Technical University of Munich, Germany

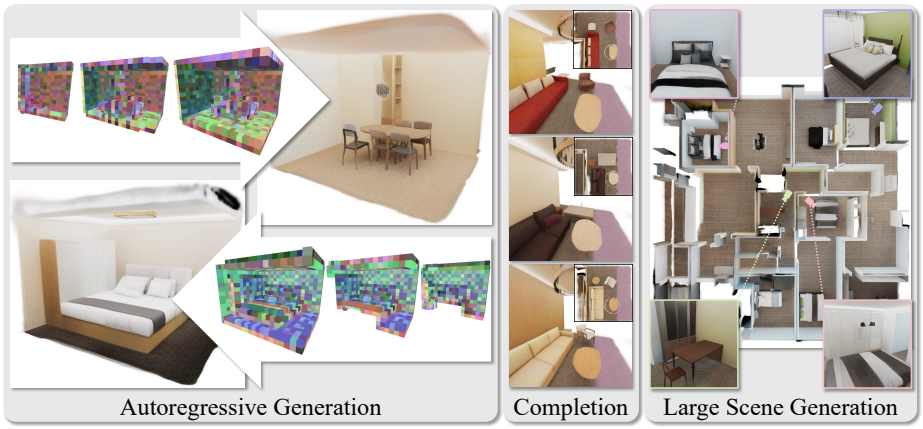

Fig. 1: GaussianGPT is a purely autoregressive approach for 3D Gaussian scene generation. Our approach enables unconditional scene generation, scene completion, and large-scale scene synthesis using only a single model.

Abstract. Most recent advances in 3D generative modeling rely on diffusion or flow-matching formulations. We instead explore a fully autoregressive alternative and introduce **GaussianGPT**, a transformer-based model that directly generates 3D Gaussians via next-token prediction, thus facilitating full 3D scene generation. We first compress Gaussian primitives into a discrete latent grid using a sparse 3D convolutional autoencoder with vector quantization. The resulting tokens are serialized and modeled using a causal transformer with 3D rotary positional embedding, enabling sequential generation of spatial structure and appearance. Unlike diffusion-based methods that refine scenes holistically, our formulation constructs scenes step-by-step, naturally supporting completion, outpainting, controllable sampling via temperature, and flexible generation horizons. This formulation leverages the compositional inductive biases and scalability of autoregressive modeling while operating on explicit representations compatible with modern neural rendering pipelines, positioning autoregressive transformers as a complementary paradigm for controllable and context-aware 3D generation.

Keywords: Autoregressive Generation · 3D Gaussian Splatting · Scene Synthesis

1 Introduction

Recent advances in deep neural networks have enabled high-quality synthesis across images, video, audio, and language. Extending these capabilities to structured 3D scene generation is a key step toward immersive virtual environments, embodied AI, simulation, and content creation. While current 3D generative approaches often rely on holistic denoising, they lack the flexibility required for the way environments are actually constructed: incrementally. In practice, 3D scenes are rarely static; they are progressively extended, completed, and edited. This creates a critical need for models that treat 3D space as a structured sequence, enabling step-by-step construction. Accordingly, we frame 3D generation as an autoregressive process in which the scene is iteratively built by predicting new elements conditioned on the existing spatial context.

Despite rapid progress, generating structured 3D scenes remains difficult due to the inherent high dimensionality and lack of a natural sequence in 3D data. Unlike 2D images, 3D environments require representations that jointly capture geometry, appearance, and semantics while maintaining multi-view consistency. Indoor scenes exhibit both strong regularities and high variability, demanding a balance between global structural coherence (e.g., room layout) and local compositional flexibility, including varied object placement, geometry, and appearance. Furthermore, the lack of a canonical ordering makes it difficult to apply successful autoregressive paradigms from other domains, as models need to serialize the 3D scene into a sequence while preserving spatial dependencies.

Recent state-of-the-art approaches [31, 49, 65, 66] predominantly rely on diffusion or flow-based models operating over neural fields, Gaussian primitives, or latent scene representations. While these methods achieve impressive visual fidelity, they typically formulate generation as a global denoising or refinement process, which can make incremental editing and structured completion less natural. In contrast, autoregressive transformers have demonstrated strong capabilities for structured sequence modeling and controllable generation across language and vision domains, yet remain comparatively underexplored for structured 3D scene synthesis.

In this work, we introduce **GaussianGPT**, a novel method for autoregressive indoor scene generation and completion that formulates 3D synthesis as sequential prediction over structured scene primitives. We represent scenes as sequences of discrete tokens derived from vector-quantized 3D Gaussian primitives, enabling transformer-based models to incrementally generate, extend, and edit scenes through next-token prediction. This formulation combines explicit 3D representations with the compositional inductive biases of autoregressive transformers, providing a complementary alternative to diffusion-based paradigms.

To sum up, our contributions are:

(i) a structured tokenization of Gaussian-based scene representations that enables autoregressive modeling of complex indoor environments
(ii) an autoregressive transformer framework for incremental 3D Gaussian scene generation, completion, and outpainting.

2 Related Works

3D Representations for Generation. A critical design choice in 3D generative modeling is the underlying 3D representation. Early generative approaches predominantly modeled shapes using discretized or implicit geometric formulations, including voxel-based occupancy grids, signed distance functions, and point clouds [1, 13, 44, 52, 62, 64, 67]. Meshes constitute another explicit surface representation and have recently been explored for 3D generative modeling [53,65,66]. The introduction of Neural Radiance Fields (NeRF) [40] enabled high-fidelity 3D generative modeling through continuous volumetric representations [41,42,47,59]. However, NeRF-based representations require dense volumetric sampling, resulting in high training and inference costs that limit scalability. 3D Gaussian Splatting [29] has emerged as an efficient and expressive alternative for 3D generative modelling, combining high visual quality with real-time rendering [11, 31, 66]. However, unconstrained sets of 3D Gaussians lack structural regularity, which makes them difficult to model. To address this, structured Gaussian parameterizations aligned to spatial grids have been proposed to introduce regularity while preserving rendering efficiency [49]. Furthermore, rather than generating the structured 3D Gaussians directly, we follow the paradigm of latent-space generation [16,50], which has been widely adopted in 3D generative modeling to improve stability, scalability, and sample quality.

Autoregressive 3D Generation. Early generative approaches for 3D modeling were largely based on variational autoencoders and generative adversarial networks [1, 62, 64, 67]. More recently, advances in 3D generative modeling have been driven predominantly by diffusion and flow-matching paradigms, which model complex data distributions through iterative denoising or continuous-time transformations and achieve strong synthesis quality across representations [13, 21, 35, 47, 52]. In parallel, autoregressive modeling, popularized by Transformer-based sequence architectures such as GPT-style models [4, 48, 60], has achieved remarkable success in language, vision, and multimodal generation by factorizing joint distributions into a sequence of conditional predictions. This formulation enables likelihood-based training, progressive generation, and flexible conditioning. Despite its success in tokenized domains, autoregressive modeling remains comparatively underexplored for 3D generation. Existing works primarily focus on sequential modeling of discretized geometric representations, including mesh-based generation [9,18,53], hierarchical voxel or octree tokenizations [24], and structured geometric token modeling [8, 17, 69]. In contrast, we explore autoregressive generation over structured Gaussian primitives, enabling scalable modeling of both objects and full scenes while avoiding the iterative sampling procedures characteristic of diffusion-based methods.

From Objects to Scenes. On object level, 3D generative modeling has seen remarkable progress in recent years [12, 31, 58, 65, 66]. In contrast, full 3D scene generation remains comparatively underexplored. Scenes introduce additional

challenges, including large spatial scale, long-range spatial dependencies, multi-object compositionality, and the need for coherent layout and physical plausibility. Existing approaches address these challenges in different ways. Some decompose scene generation into structured layout prediction followed by object retrieval or object-level synthesis [46, 57, 61]. Others leverage strong priors from pretrained image or video generative models to guide 3D optimization or reconstruction [23, 51], or rely on large-scale foundation models to provide semantic or structural supervision [14, 32, 33, 71]. A further line of work formulates scene generation as feed-forward reconstruction using depth or multi-view cues combined with Gaussian splatting, focusing on reconstruction rather than unconditional generative modeling [7,10,25,34,56,70]. While these strategies demonstrate promising results, they often depend on external priors, multi-stage pipelines, or optimization-heavy procedures. In contrast, we aim to learn a unified autoregressive generative model that directly models 3D scenes without relying on pretrained 2D diffusion or video priors, enabling scalable scene synthesis within a single probabilistic framework.

3 Methodology

Our goal is to synthesize 3D Gaussian scenes using unconditional autoregressive generation with a GPT-style transformer. Since GPT models operate on discrete token sequences and have limited contextual scope, we first require a compact and discrete representation of 3D scenes. To this end, we leverage a sparse 3D convolutional autoencoder that maps Gaussian scenes to a discrete latent grid representation and faithfully reconstructs them (Sec. 3.1). Given this representation, we serialize the latent grids into token sequences and train a causal transformer to model the joint distribution of grid occupancy and features. An overview of the full pipeline can be seen in Fig. 2. We detail the autoregressive formulation, architectural design, inference procedure, and the incorporation of 3D spatial priors in Sec. 3.2.

3.1 Scene Compression via Sparse 3D Latent Encoding

To enable token-based autoregressive modeling, we first transform a continuous 3D Gaussian scene into a compact discrete latent grid representation. Our compression stage consists of (i) projecting Gaussian primitives into a sparse 3D feature grid, (ii) encoding this grid using a sparse 3D convolutional autoencoder, and (iii) discretizing the latent representation via vector quantization. The resulting quantized latent grid forms the basis for subsequent autoregressive modeling.

Sparse 3D Feature Grid. Following 3D Gaussian Splatting [29], our input is a scene represented by a set of Gaussian primitives. Each primitive is defined by a set of continuous attributes: position, opacity, size, rotation, and color. We define a grid in world coordinates and assign Gaussians to the corresponding voxels

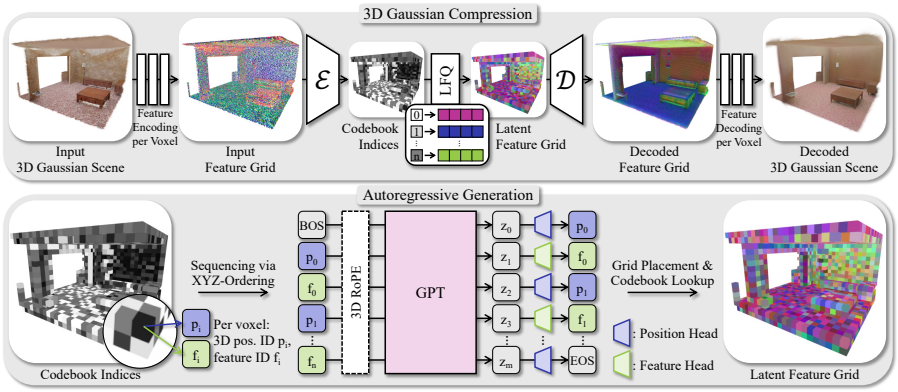

Fig. 2: Overview of **GaussianGPT**. A 3D Gaussian scene is subsampled into a sparse voxel grid and encoded into per-voxel features. A sparse 3D CNN compresses the grid into discrete codebook indices. The resulting latent grid is serialized via xyz ordering and represented as interleaved position tokens p_i and feature tokens f_i. A causal transformer with 3D RoPE predicts alternating position and feature tokens, which are mapped back to voxel locations and decoded through LFQ to reconstruct the 3D Gaussian scene.

based on their positions. We replace absolute positions with relative offsets from the voxel centers and randomly subsample, when multiple Gaussians are present in a voxel, to obtain our input 3D Gaussian scene. For each voxel, we use a set of lightweight encoding heads to encode each Gaussian feature, then concatenate the results into a unified vector to obtain a sparse input feature grid. Symmetrically, after passing through the 3D CNN, a set of per-feature decoding heads converts the predicted per-voxel feature vectors back into individual Gaussian attributes.

Sparse 3D CNN Encoder-Decoder. The input feature grid is processed by a sparse 3D convolutional encoder $\mathcal{E}$ and decoder $\mathcal{D}$ following L3DG [49]. The encoder progressively downsamples the grid to a compact latent representation, while the decoder reconstructs voxel-level features. The convolutional design preserves spatial locality and translation equivariance, yielding structured latent features well suited for subsequent generative modeling.

Vector Quantization. In contrast to the L3DG implementation, we adopt lookup-free quantization (LFQ) [68], which has been shown to improve codebook utilization and quality. Following the concept of LFQ, the output of our encoder $\mathbf{z}$ directly corresponds to the codebook indices by discretizing to 0 or 1 based on the sign.

Training. The network is trained on randomly selected images by using a combination of re-rendering, occupancy, and codebook losses. For the re-rendering

losses, $\mathcal{L}_{RGB}$ is an L_1 color loss and $\mathcal{L}_{perc}$ is a VGG19-based [54] perceptual loss; both are aggregated over a set of sampled images and poses. Occupancy predictions in the decoder upsampling layers are guided by a binary cross-entropy loss, $\mathcal{L}_{occ}$, following L3DG. Lastly, $\mathcal{L}_{LFQ}$ aims to increase the codebook usage by increasing entropy, as introduced in [68]. We pass the result through an offset softplus function to obtain positive loss values.

$$\mathcal{L} = \underbrace{\lambda_{RGB}\mathcal{L}_{RGB} + \lambda_{perc}\mathcal{L}_{perc}}_{\text{re-rendering}} + \underbrace{\lambda_{occ}\mathcal{L}_{occ}}_{\text{occupancy}} + \underbrace{\lambda_{LFQ}\text{softplus}\left(\mathcal{L}_{LFQ} + 5\right)}_{\text{codebook entropy}}. \quad (1)$$

3.2 Autoregressive Modeling of Latent 3D Grids

Given the quantized latent grid, our goal is to model the joint distribution of its occupancy and features using a sequence-based autoregressive model. Our first step is therefore to serialize the 3D grid into a 1D token sequence, after which we can train a causal transformer to model the underlying patterns. We additionally provide priors to the model by separating position and feature vocabularies and injecting 3D positional embeddings.

3D Grid Serialization. We linearize the 3D structure into a 1D sequence using a simple and fixed xyz traversal order. Since z is the least significant dimension, an intuitive understanding of this is that we iterate a scene-height column at every (x, y) position before jumping to the next one. While this pattern does not preserve 3D locality in the 1D sequence, its advantages lie in its simple, interpretable ordering (see Sec. 4.5).

Given the ordering and relative position indices of the voxels, we construct our sequence by interleaving position and feature tokens. Since the possible sequence grows cubically with scene size, we limit the required context size by operating our GPT model on chunks rather than entire scenes. Each voxel is assigned a position index relative to the current grid chunk rather than its absolute position, enabling the model to operate locally and generalize across positions, scenes, and layouts.

Vocabulary Design. Although our sequences are processed by a shared transformer backbone, we employ separate vocabularies for position and feature tokens. Following the regular alternating pattern of position and feature tokens, we alternate between a distinct position and feature head. The task of the position head is to predict the next occupied voxel index, and the feature head predicts the feature at the preceding position. This explicit separation in vocabulary and predictions decouples geometric structure from appearance modeling and prevents competition between spatial and semantic features over shared indices. Incidentally, this design also allows us to freely control the chunk size and the corresponding number of position indices, irrespective of the feature codebook size.

3D Rotary Positional Encoding. Transformers, as used in GPT-style models, process inputs as a 1D token sequence and require an explicit notion of position. In standard applications, positional information is injected along this 1D index, either via learned absolute embeddings or via relative encodings such as rotary positional embeddings (RoPE) [55]. However, in our setting, the token sequence is not an inherently one-dimensional signal: it is obtained by serializing a sparse 3D latent grid. If we were to apply standard 1D positional encoding directly to the serialized order, the model would primarily learn notions of sequence proximity, which can translate poorly to spatial proximity. In particular, voxels that are close in (x, y, z) may be far apart in the serialization, and vice versa.

To inject an explicit spatial inductive bias, we encode actual voxel coordinates inside the attention mechanism using 3D RoPE. This way, the attention score becomes a function of the relative spatial offset between the tokens rather than the sequence offset. The approach follows recent uses of coordinate-aware rotary embeddings in 3D generation [65], enabling the model to reason about the spatial locality independent of the serialization order.

Finally, our sequence alternates between position and feature tokens, both of which correspond to the same 3D location. In practice, we therefore extend the rotary embeddings with an additional 4th dimension that indicates the token type. This token-type rotary component helps the transformer to further disentangle geometry from appearance while retaining a single unified attention formulation over the mixed token stream.

Transformer Architecture. Our decoder-only causal transformer architecture follows the standard GPT formulation with stacked multi-head self-attention and feed-forward blocks. Our model is based on GPT-2 [48] and builds on nanochat [28] as its technical backbone. Compared to the vanilla GPT-2 pipeline, we additionally employ rotary positional embeddings (extended to 3D as described above), query–key normalization, per-layer residual scaling, and the Muon optimizer [19, 26, 27]. While part of the default configuration of the nanochat backbone, we do not use value embeddings or sliding-window attention [4, 72].

Training. The transformer is trained to model the distribution of serialized scene tokens using a standard autoregressive objective. Given a tokenized scene sequence $\mathbf{t} = (t_1, \ldots, t_T)$, the model predicts each token conditioned on all previous tokens:

$$\mathcal{L}_{CE} = -\sum_{i=1}^{T} \log p_\theta(t_i \mid t_{<i}). \tag{2}$$

Training is performed with teacher forcing, where we provide ground-truth tokens as input, and the model learns to predict the next token in the sequence. Since our representation alternates between position tokens and feature tokens, the cross-entropy loss is computed over the corresponding vocabulary at each step. Invalid vocabulary entries are masked such that the model only predicts position tokens at position steps and feature tokens at feature steps.

Scene Generation and Completion. At inference time, scenes are generated autoregressively by sampling tokens conditioned on previously generated context. Starting from a begin-of-sequence (BOS) token, the transformer alternates between predicting position tokens and feature tokens until an end-of-sequence (EOS) token is produced.

A key advantage of the autoregressive formulation is that scene completion and unconditional generation are handled by the same mechanism. Given a partial scene, we simply serialize the observed tokens and use them as a prefix prompt from which the model continues generation. This allows the transformer to naturally infer missing geometry and appearance while remaining fully consistent with the existing scene context. The same principle also enables large-scale scene synthesis beyond the fixed training chunk size. By repeatedly applying completions on a sliding window of previously generated tokens as context, we can continuously outpaint the scene.

Due to the compositional nature of autoregressive generation, we can further limit predictions based on the current sequence. Specifically, since each position token directly corresponds to a voxel index within the current chunk, we can mask already generated locations, ensuring that the sampled sequence always respects ordering constraints. Additionally, the sequential nature of our approach enables tree search. We use this when generating larger scenes by resampling columns without occupancy, thereby creating more occupied and connected scenes.

4 Experiments

We evaluate GaussianGPT on unconditional generation, scene completion, and large scene outpainting. First, we describe the datasets and experimental setup. We then present quantitative results on shape generation, scene generation, and scene completion, as well as qualitative results and large-scale outpainting examples. Lastly, we showcase larger scenes synthesized by our approach and ablate the effect of our design choice for sequence ordering.

4.1 Datasets

PhotoShape. For object-level experiments, we follow the setup of DiffRF [41] using PhotoShape [45]. The dataset contains 15,576 chairs normalized to a canonical grid, and associated with 200 rendered views each.

3D-FRONT. To obtain high-fidelity Gaussian scenes, we construct a dataset based on 3D-FRONT [15]. For each scene, we render multi-view images and initialize Gaussian primitives from depth maps. These Gaussians are optimized using Scaffold-GS [37] for $60k$ iterations. By aligning anchors with voxel centers and restricting each anchor to a single Gaussian, we obtain a high-fidelity Gaussian scene within the input domain of our autoencoder. After filtering scenes based on spatial extent, our resulting dataset contains 4,472 scenes. To increase

data diversity, we additionally apply rotational and reflection-based augmentations, effectively increasing the dataset size by a factor of eight.

Aria Synthetic Environments. For large-scale scene modeling, we additionally use Gaussian scenes derived from Aria Synthetic Environments (ASE) [2] optimized as part of SceneSplat++ [39]. The dataset contains 25,000 indoor scenes with diverse layouts, object arrangements, and appearances.

4.2 Experimental Setup

Autoencoder Configuration. The scene autoencoder operates on a base voxel size of 0.025 m and applies three downsampling stages, resulting in a latent grid with a voxel size of 20 cm. For objects, shapes are discretized on a 128^3 grid with two downsampling stages to match prior work [49]. We do not model view-dependent appearance for scenes, while objects use first-order spherical harmonics. Across all experiments, the codebook size is set to 4,096. For more information on the autoencoder hyperparameters, see Appendix F. For the re-rendering losses, we use $\lambda_{RGB} = 7.5$ and $\lambda_{perc} = 0.3$ with 12 images for scenes, and $\lambda_{RGB} = 12.5$ and $\lambda_{perc} = 0.1$ with 4 images for objects following [49]. $\lambda_{occ} = 1.0$ and $\lambda_{LFQ} = 0.1$ are the same across all trainings.

Transformer Configuration. For scene modeling, we use a GPT-2 medium-sized transformer with a context window of 16,384 tokens, which is sufficient to cover a fully occupied chunk. For object generation, we use a GPT-2 small-sized transformer with a context window of 8,192 tokens. All results are sampled with a temperature of 0.9 and Nucleus Sampling [22] with $p = 0.9$.

Training Strategy. Scene training is performed on spatial chunks with fixed vertical positions in order to align floor heights across scenes and datasets. During training, we randomly sample chunk locations subject to minimum occupancy thresholds. For ASE scenes, supervision for autoencoder training is obtained by comparing against rendered ground-truth Gaussians, whereas for 3D-FRONT scenes, we use the previously rendered images. We limit supervision to image pixels within the corresponding chunk, as defined by the GT geometry. For object experiments, both the autoencoder and transformer are trained on full shapes without chunking.

Optimization and Compute. The autoencoder is trained using Adam [30] with a learning rate of 10^{-4}, and the transformer uses a combination of AdamW [36] and Muon [26] with per-module learning rates [28]. Both models use cosine learning rate decay to 10% of the initial value. Autoencoder training uses 4 RTX A6000 GPUs, with an effective batch size of 8 for scenes and 24 for objects. Transformer training uses 4 GH200 GPUs with an effective batch size of 64. We train the autoencoder for around 4 days on scenes and 2 days on PhotoShape. GPT training takes around 1 day on scenes and 4.5 hours on PhotoShape. We additionally fine-tune both models on 3D-FRONT only, which takes around 1 day for the autoencoder and 10 hours for the GPT model.

Fig. 3: Qualitative results on unconditional chair generation. From left to right: DiffRF [41], L3DG [49], and GaussianGPT (ours). All models generate high-fidelity shapes; our method produces clean Gaussian allocations with consistent geometric structure.

Table 1: Quantitative evaluation of unconditional generation of 1000 shapes after training on the PhotoShape Chairs dataset [45]. MMD and KID scores are multiplied by 10^3.

Method	FID ↓	KID ↓	COV ↑	MMD ↓
π-GAN [6]	52.71	13.64	39.92	7.387
EG3D [5]	16.54	8.412	47.55	5.619
DiffRF [41]	15.95	7.935	58.93	4.416
L3DG [49]	8.49	3.147	63.80	**4.241**
Ours	**5.68**	**1.835**	**67.40**	4.278

4.3 Shape Synthesis

Following the evaluation protocol of L3DG [49], we evaluate GaussianGPT on unconditional chair generation using PhotoShape [45], Rendering quality is measured using Frechet Inception Distance (FID) and Kernel Inception Distance (KID) on 128×128 images [3,20,43]. For geometric evaluation, we compute Coverage (COV) and Minimum Matching Distance (MMD) using Chamfer Distance on $2k$ surface points of meshes extracted from generated Gaussians, following [1]. COV measures sample diversity, while MMD reflects geometric fidelity.

Quantitative results are reported in Tab. 1. Our autoregressive model achieves the best FID, KID, and COV among all methods, while remaining competitive in MMD, indicating strong alignment with the target distribution in both appearance and geometry.

Qualitative results are shown in Fig. 3. Both L3DG and GaussianGPT produce high-quality chairs with plausible geometry and appearance. However, our model tends to avoid noisy outlier primitives, resulting in sharper structures and cleaner renderings while maintaining substantial variation in shape and style. Overall, these results demonstrate that autoregressive modeling of vector-quantized Gaussian representations can achieve state-of-the-art performance in unconditional 3D shape synthesis.

4.4 Scene Synthesis

We evaluate unconditional scene generation and scene completion. For qualitative comparison, we contrast GaussianGPT with the original L3DG model [49], which is trained to generate full normalized indoor rooms. To enable a controlled quantitative comparison, we additionally train both GaussianGPT and L3DG on the same chunked 3D-FRONT data under matched spatial settings. The quantitative results below refer to this matched chunk-level setup. Qualitative results for the chunked L3DG model are provided in Appendix G.

Fig. 4: Qualitative results on unconditional scene generation. We compare the original full-room L3DG [49] against GaussianGPT trained only on 3D-FRONT.

Evaluation Protocol. We evaluate unconditional generation as well as completion from partial context using the metrics in Tab. 2. The given context is a

spatially contiguous subset containing the first 25% or 50% in the x dimension. Appearance quality is measured by FID and KID over rendered views sampled along horizontal orbits around each chunk. For geometry, we compute Chamfer distance-based COV and MMD from point sets sampled from the Gaussian centers. Compared to the shape-level evaluation, we raise the resolution to 256×256 and the number of points to $16k$. We additionally assess layout quality using FID and KID on top-down depth renderings. For this, we only keep the bottom 60% of Gaussians per-chunk to remove ceilings. Completion is evaluated using CLIP similarity between the orbit renderings and their GT equivalents. We use RePaint [38] to evaluate L3DG for completion.

Qualitative Results. As shown in Fig. 4, both methods produce coherent indoor layouts with plausible object placement and appearance. The causal formulation naturally conditions on arbitrary prefixes, enabling scene completion without architectural changes or specialized conditioning mechanisms, as can be seen in Fig. 6. We observe that generated completions are both semantically meaningful and diverse across samples, indicating strong modeling of the conditional scene distribution from partial inputs.

Table 2: Quantitative evaluation on scene chunks. We compare unconditional generation and completion with 25% and 50% context in terms of appearance, geometry, layout, and completion consistency after training on 3D-FRONT data. Generation is evaluated on 1000 generated and randomly selected GT chunks, while completion uses 200 GT chunks with 5 completions each. GT chunks are selected from held-out scenes.

	Appearance		Geometry		Layout		Completion
	FID ↓	KID ↓	COV ↑	MMD ↓	FID ↓	KID ↓	CLIP-Sim. ↑
Generation							
L3DG	100.98	0.095	**0.692**	**0.096**	93.84	0.092	–
Ours	**94.85**	**0.084**	0.548	0.118	**84.14**	**0.077**	–
Completion (25% context)							
L3DG	109.35	0.100	**0.925**	0.106	112.57	0.093	0.834
Ours	**100.06**	**0.084**	0.879	**0.089**	**95.60**	**0.068**	**0.841**
Completion (50% context)							
L3DG	106.54	0.096	**0.940**	0.084	111.71	0.092	0.843
Ours	**98.13**	**0.081**	**0.940**	**0.057**	**90.77**	**0.063**	**0.851**

Quantitative Results. Table 2 highlights a trade-off for unconditional generation. Consistent with the qualitative examples, L3DG produces more varied samples and better matches the ground-truth geometry distribution, while GaussianGPT produces cleaner geometry and stronger appearance and layout metrics. As the task shifts toward completion and more of the scene is observed, this trade-off increasingly favors our autoregressive formulation.

Fig. 5: 12 m × 12 m scene synthesis via autoregressive outpainting. GaussianGPT sequentially generates and appends latent grid columns, enabling scenes to grow beyond the training horizon.

Large Scene Generation. As shown in Fig. 5, our model can extend scenes beyond the fixed training horizon. We iteratively generate and append latent-grid columns using previously generated regions as context. To promote occupancy across larger scene generation, we leverage the step-by-step generative process to introduce a backtracking resampling strategy. If a column without any geometry is predicted, we move back to the last prior token and attempt sampling again up to 5 times. GaussianGPT produces coherent and stylistically consistent extended scenes over substantial spatial ranges, although quality gradually degrades with distance from the initial context, as analyzed in Appendix C. These results highlight the flexibility of autoregressive modeling for open-ended 3D scene synthesis, where scene size is not predetermined, and generation can naturally scale with context.

4.5 Autoregressive Modeling Ablations

We ablate key design choices of our autoregressive model on the combined ASE and 3D-FRONT training setup. Specifically, we study how the 3D latent grid is serialized into a token sequence and how spatial and feature information are represented within the transformer. For all variants, we keep the remaining architecture and training configuration fixed and report average training and validation cross-entropy over epochs 70–74.

Serialization Strategies. Table 3 compares different 3D-to-1D serializations. Besides our column-wise xyz traversal, we evaluate Z-order and Hilbert space-filling curves and their transposed variants following Point Transformer V3 [63]. Despite stronger locality guarantees, these orderings do not improve performance. The simple xyz traversal performs best overall, while transposed Z-order achieves nearly identical validation performance. This suggests that 3D RoPE already provides sufficient spatial information, reducing the importance of locality in the serialized sequence.

Core Model Components. Table 4 evaluates separate vocabularies for position and feature tokens, as well as 3D RoPE. A shared vocabulary increases validation cross-entropy, while learned positional embeddings or 1D RoPE degrade

Fig. 6: Qualitative results on scene chunk completion. Given one quarter of a validation chunk as context, GaussianGPT produces plausible and diverse completions.

both training and validation performance. Together, these results show that separating spatial and feature prediction and explicitly encoding 3D coordinates provide useful inductive biases for autoregressive scene modeling.

Table 3: Ablation of 3D-to-1D serialization strategies.

Ordering	Train CE ↓	Val CE ↓
Z-order	2.203	2.432
Trans. Z-order	2.204	2.423
Hilbert	2.347	2.439
Trans. Hilbert	2.334	2.434
xyz (ours)	**2.151**	**2.421**

Table 4: Ablation of the core design decisions: separate vocabulary and 3D RoPE.

Setup	Train CE ↓	Val CE ↓
Ours	**2.151**	**2.421**
w/ Shared Vocab.	2.157	2.449
w/ Learned PE	2.210	2.461
w/ 1D RoPE	2.227	2.446

5 Conclusion

We introduced **GaussianGPT**, a fully autoregressive framework for 3D scene generation and completion operating directly on vector-quantized Gaussian representations. By combining sparse 3D latent compression with structured tokenization and 3D-aware transformer modeling, we show that sequential next-token prediction is a viable and complementary alternative to diffusion-based approaches for structured 3D synthesis. Compared to diffusion-based methods, our formulation enables several new capabilities. Scenes can be generated incrementally as a sequence of discrete spatial decisions, allowing explicit control over sampling, causal reasoning over partially generated geometry, and seamless support for completion and outpainting using the same model.

We demonstrate improved visual quality in 3D shape synthesis and scene-level generation and completion, with stronger perceptual and layout quality while retaining a trade-off in geometric diversity. More broadly, our work highlights the potential of treating 3D scenes as structured token sequences, opening new opportunities for controllable and compositional 3D generation. Future work includes extending the framework to real-world data, where uncertainty-aware modeling becomes essential, and autoregressive approaches are particularly well-suited. We also imagine other promising avenues for mitigating large-scene degradation, improving long-horizon stability, and extending the generation context beyond fixed spatial chunks and orderings.

Acknowledgements

Nicolas von Lützow is supported by the MDSI focus topic *Understanding Existing Structures in Building Planning* (USP), and Katharina Schmid is an MDSI doctoral fellow. This work was further supported by the ERC Consolidator Grant *Gen3D* (101171131), and by compute resources from the Jülich Supercomputing Centre under project *MeshFoundation*. We thank Angela Dai for the video voice-over.

References

1. Achlioptas, P., Diamanti, O., Mitliagkas, I., Guibas, L.: Learning Representations and Generative Models for 3D Point Clouds (Jun 2018). https://doi.org/10.48550/arXiv.1707.02392, http://arxiv.org/abs/1707.02392, arXiv:1707.02392 [cs]
2. Avetisyan, A., Xie, C., Howard-Jenkins, H., Yang, T.Y., Aroudj, S., Patra, S., Zhang, F., Frost, D., Holland, L., Orme, C., Engel, J., Miller, E., Newcombe, R., Balntas, V.: SceneScript: Reconstructing Scenes With An Autoregressive Structured Language Model (Mar 2024). https://doi.org/10.48550/arXiv.2403.13064, http://arxiv.org/abs/2403.13064, arXiv:2403.13064 [cs]
3. Bińkowski, M., Sutherland, D.J., Arbel, M., Gretton, A.: Demystifying MMD GANs (Jan 2021). https://doi.org/10.48550/arXiv.1801.01401, http://arxiv.org/abs/1801.01401, arXiv:1801.01401 [stat]
4. Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D.M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., Amodei, D.: Language Models are Few-Shot Learners (Jul 2020). https://doi.org/10.48550/arXiv.2005.14165, http://arxiv.org/abs/2005.14165, arXiv:2005.14165 [cs]
5. Chan, E.R., Lin, C.Z., Chan, M.A., Nagano, K., Pan, B., Mello, S.D., Gallo, O., Guibas, L., Tremblay, J., Khamis, S., Karras, T., Wetzstein, G.: Efficient Geometry-aware 3D Generative Adversarial Networks (Apr 2022). https://doi.org/10.48550/arXiv.2112.07945, http://arxiv.org/abs/2112.07945, arXiv:2112.07945 [cs]
6. Chan, E.R., Monteiro, M., Kellnhofer, P., Wu, J., Wetzstein, G.: pi-GAN: Periodic Implicit Generative Adversarial Networks for 3D-Aware Image Synthesis (Apr 2021). https://doi.org/10.48550/arXiv.2012.00926, http://arxiv.org/abs/2012.00926, arXiv:2012.00926 [cs]
7. Charatan, D., Li, S., Tagliasacchi, A., Sitzmann, V.: pixelSplat: 3D Gaussian Splats from Image Pairs for Scalable Generalizable 3D Reconstruction (Apr 2024). https://doi.org/10.48550/arXiv.2312.12337, http://arxiv.org/abs/2312.12337, arXiv:2312.12337 [cs]
8. Chen, J., Zhu, L., Hu, Z., Qian, S., Chen, Y., Wang, X., Lee, G.H.: MAR-3D: Progressive Masked Auto-regressor for High-Resolution 3D Generation (Apr 2025). https://doi.org/10.48550/arXiv.2503.20519, http://arxiv.org/abs/2503.20519, arXiv:2503.20519 [cs]

9. Chen, Y., He, T., Huang, D., Ye, W., Chen, S., Tang, J., Chen, X., Cai, Z., Yang, L., Yu, G., Lin, G., Zhang, C.: MeshAnything: Artist-Created Mesh Generation with Autoregressive Transformers (Oct 2024). https://doi.org/10.48550/arXiv.2406.10163, http://arxiv.org/abs/2406.10163, arXiv:2406.10163 [cs]
10. Chen, Y., Xu, H., Zheng, C., Zhuang, B., Pollefeys, M., Geiger, A., Cham, T.J., Cai, J.: MVSplat: Efficient 3D Gaussian Splatting from Sparse Multi-View Images. vol. 15079, pp. 370–386 (2025). https://doi.org/10.1007/978-3-031-72664-4_21, http://arxiv.org/abs/2403.14627, arXiv:2403.14627 [cs]
11. Chen, Y., Mihajlovic, M., Chen, X., Wang, Y., Prokudin, S., Tang, S.: SplatFormer: Point Transformer for Robust 3D Gaussian Splatting (Mar 2025). https://doi.org/10.48550/arXiv.2411.06390, http://arxiv.org/abs/2411.06390, arXiv:2411.06390 [cs]
12. Chen, Z., Wang, F., Wang, Y., Liu, H.: Text-to-3D using Gaussian Splatting (Apr 2024). https://doi.org/10.48550/arXiv.2309.16585, http://arxiv.org/abs/2309.16585, arXiv:2309.16585 [cs]
13. Chou, G., Bahat, Y., Heide, F.: Diffusion-SDF: Conditional Generative Modeling of Signed Distance Functions (Mar 2023). https://doi.org/10.48550/arXiv.2211.13757, http://arxiv.org/abs/2211.13757, arXiv:2211.13757 [cs]
14. Feng, W., Zhu, W., Fu, T.j., Jampani, V., Akula, A., He, X., Basu, S., Wang, X.E., Wang, W.Y.: LayoutGPT: Compositional Visual Planning and Generation with Large Language Models (Oct 2023). https://doi.org/10.48550/arXiv.2305.15393, http://arxiv.org/abs/2305.15393, arXiv:2305.15393 [cs]
15. Fu, H., Cai, B., Gao, L., Zhang, L., Li, J.W.C., Xun, Z., Sun, C., Jia, R., Zhao, B., Zhang, H.: 3D-FRONT: 3D Furnished Rooms with layOuts and semaNTics (May 2021). https://doi.org/10.48550/arXiv.2011.09127, http://arxiv.org/abs/2011.09127, arXiv:2011.09127 [cs]
16. Gao, Q., Georgiev, I., Wang, T.Y., Singh, K.K., Neumann, U., Yoon, J.S.: Can3Tok: Canonical 3D Tokenization and Latent Modeling of Scene-Level 3D Gaussians (Aug 2025). https://doi.org/10.48550/arXiv.2508.01464, http://arxiv.org/abs/2508.01464, arXiv:2508.01464 [cs]
17. Han, Z., Cao, D., Sun, H., Hong, Y.: VAR-3D: View-aware Auto-Regressive Model for Text-to-3D Generation via a 3D Tokenizer (Feb 2026). https://doi.org/10.48550/arXiv.2602.13818, http://arxiv.org/abs/2602.13818, arXiv:2602.13818 [cs]
18. Hao, Z., Romero, D.W., Lin, T.Y., Liu, M.Y.: Meshtron: High-Fidelity, Artist-Like 3D Mesh Generation at Scale (Dec 2024). https://doi.org/10.48550/arXiv.2412.09548, http://arxiv.org/abs/2412.09548, arXiv:2412.09548 [cs]
19. Henry, A., Dachapally, P.R., Pawar, S., Chen, Y.: Query-Key Normalization for Transformers (Oct 2020). https://doi.org/10.48550/arXiv.2010.04245, http://arxiv.org/abs/2010.04245, arXiv:2010.04245 [cs]
20. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium (Jan 2018). https://doi.org/10.48550/arXiv.1706.08500, http://arxiv.org/abs/1706.08500, arXiv:1706.08500 [cs]
21. Ho, J., Jain, A., Abbeel, P.: Denoising Diffusion Probabilistic Models (Dec 2020). https://doi.org/10.48550/arXiv.2006.11239, http://arxiv.org/abs/2006.11239, arXiv:2006.11239 [cs]
22. Holtzman, A., Buys, J., Du, L., Forbes, M., Choi, Y.: The Curious Case of Neural Text Degeneration (Feb 2020). https://doi.org/10.48550/arXiv.1904.09751, http://arxiv.org/abs/1904.09751, arXiv:1904.09751 [cs]

23. Höllein, L., Cao, A., Owens, A., Johnson, J., Nießner, M.: Text2Room: Extracting Textured 3D Meshes from 2D Text-to-Image Models (Sep 2023). https://doi.org/10.48550/arXiv.2303.11989, http://arxiv.org/abs/2303.11989, arXiv:2303.11989 [cs]
24. Ibing, M., Kobsik, G., Kobbelt, L.: Octree Transformer: Autoregressive 3D Shape Generation on Hierarchically Structured Sequences (Nov 2021). https://doi.org/10.48550/arXiv.2111.12480, http://arxiv.org/abs/2111.12480, arXiv:2111.12480 [cs]
25. Jiang, L., Mao, Y., Xu, L., Lu, T., Ren, K., Jin, Y., Xu, X., Yu, M., Pang, J., Zhao, F., Lin, D., Dai, B.: AnySplat: Feed-forward 3D Gaussian Splatting from Unconstrained Views (Sep 2025). https://doi.org/10.48550/arXiv.2505.23716, http://arxiv.org/abs/2505.23716, arXiv:2505.23716 [cs]
26. Jordan, K.: Muon: An optimizer for hidden layers in neural networks. https://kellerjordan.github.io/posts/muon/ (2024), accessed 2026-03-01
27. Jordan, K., Bernstein, J., Rappazzo, B., @fernbear.bsky.social, Vlado, B., Jiacheng, Y., Cesista, F., Koszarsky, B., @Grad62304977: modded-nanogpt: Speedrunning the nanogpt baseline (2024), https://github.com/KellerJordan/modded-nanogpt, accessed 2026-03-01
28. Karpathy, A.: nanochat: The best chatgpt that $100 can buy (2025), https://github.com/karpathy/nanochat, accessed 2026-03-01
29. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3D Gaussian Splatting for Real-Time Radiance Field Rendering (Aug 2023). https://doi.org/10.48550/arXiv.2308.04079, http://arxiv.org/abs/2308.04079, arXiv:2308.04079 [cs]
30. Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization (Jan 2017). https://doi.org/10.48550/arXiv.1412.6980, http://arxiv.org/abs/1412.6980, arXiv:1412.6980 [cs]
31. Lan, Y., Zhou, S., Lyu, Z., Hong, F., Yang, S., Dai, B., Pan, X., Loy, C.C.: GaussianAnything: Interactive Point Cloud Flow Matching For 3D Object Generation (Apr 2025). https://doi.org/10.48550/arXiv.2411.08033, http://arxiv.org/abs/2411.08033, arXiv:2411.08033 [cs]
32. Li, H., Tian, Y., Lan, K., Liao, Y., Wang, L., Hui, P., Zhou, P.Y.: DreamScene: 3D Gaussian-based End-to-end Text-to-3D Scene Generation (Jul 2025). https://doi.org/10.48550/arXiv.2507.13985, http://arxiv.org/abs/2507.13985, arXiv:2507.13985 [cs]
33. Li, S., Yang, C., Fang, J., Yi, T., Lu, J., Cen, J., Xie, L., Shen, W., Tian, Q.: WorldGrow: Generating Infinite 3D World (Oct 2025). https://doi.org/10.48550/arXiv.2510.21682, http://arxiv.org/abs/2510.21682, arXiv:2510.21682 [cs]
34. Lin, H., Chen, S., Liew, J., Chen, D.Y., Li, Z., Shi, G., Feng, J., Kang, B.: Depth Anything 3: Recovering the Visual Space from Any Views (Nov 2025). https://doi.org/10.48550/arXiv.2511.10647, http://arxiv.org/abs/2511.10647, arXiv:2511.10647 [cs]
35. Lipman, Y., Chen, R.T.Q., Ben-Hamu, H., Nickel, M., Le, M.: Flow Matching for Generative Modeling (Feb 2023). https://doi.org/10.48550/arXiv.2210.02747, http://arxiv.org/abs/2210.02747, arXiv:2210.02747 [cs]
36. Loshchilov, I., Hutter, F.: Decoupled Weight Decay Regularization (Jan 2019). https://doi.org/10.48550/arXiv.1711.05101, http://arxiv.org/abs/1711.05101, arXiv:1711.05101 [cs]
37. Lu, T., Yu, M., Xu, L., Xiangli, Y., Wang, L., Lin, D., Dai, B.: Scaffold-GS: Structured 3D Gaussians for View-Adaptive Rendering (Nov 2023). https://doi.org/10.48550/arXiv.2312.00109, http://arxiv.org/abs/2312.00109, arXiv:2312.00109 [cs]

38. Lugmayr, A., Danelljan, M., Romero, A., Yu, F., Timofte, R., Gool, L.V.: RePaint: Inpainting using Denoising Diffusion Probabilistic Models (Aug 2022). https://doi.org/10.48550/arXiv.2201.09865, http://arxiv.org/abs/2201.09865, arXiv:2201.09865 [cs.CV]
39. Ma, M., Ma, Q., Li, Y., Cheng, J., Yang, R., Ren, B., Popovic, N., Wei, M., Sebe, N., Gool, L.V., Gevers, T., Oswald, M.R., Paudel, D.P.: SceneSplat++: A Large Dataset and Comprehensive Benchmark for Language Gaussian Splatting (Jun 2025). https://doi.org/10.48550/arXiv.2506.08710, http://arxiv.org/abs/2506.08710, arXiv:2506.08710 [cs]
40. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis (Aug 2020). https://doi.org/10.48550/arXiv.2003.08934, http://arxiv.org/abs/2003.08934, arXiv:2003.08934 [cs]
41. Müller, N., Siddiqui, Y., Porzi, L., Bulò, S.R., Kontschieder, P., Nießner, M.: DiffRF: Rendering-Guided 3D Radiance Field Diffusion (Mar 2023). https://doi.org/10.48550/arXiv.2212.01206, http://arxiv.org/abs/2212.01206, arXiv:2212.01206 [cs]
42. Niemeyer, M., Geiger, A.: GIRAFFE: Representing Scenes as Compositional Generative Neural Feature Fields (Apr 2021). https://doi.org/10.48550/arXiv.2011.12100, http://arxiv.org/abs/2011.12100, arXiv:2011.12100 [cs]
43. Obukhov, A., Seitzer, M., Wu, P.W., Zhydenko, S., Kyl, J., Lin, E.Y.J.: High-fidelity performance metrics for generative models in pytorch (2020). https://doi.org/10.5281/zenodo.3786539, https://github.com/toshas/torch-fidelity, version: 0.4.0, DOI: 10.5281/zenodo.3786539
44. Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S.: DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation (Jan 2019). https://doi.org/10.48550/arXiv.1901.05103, http://arxiv.org/abs/1901.05103, arXiv:1901.05103 [cs]
45. Park, K., Rematas, K., Farhadi, A., Seitz, S.M.: PhotoShape: Photorealistic Materials for Large-Scale Shape Collections. ACM Transactions on Graphics **37**(6), 1–12 (Dec 2018). https://doi.org/10.1145/3272127.3275066, http://arxiv.org/abs/1809.09761, arXiv:1809.09761 [cs]
46. Paschalidou, D., Kar, A., Shugrina, M., Kreis, K., Geiger, A., Fidler, S.: ATISS: Autoregressive Transformers for Indoor Scene Synthesis (Oct 2021). https://doi.org/10.48550/arXiv.2110.03675, http://arxiv.org/abs/2110.03675, arXiv:2110.03675 [cs]
47. Poole, B., Jain, A., Barron, J.T., Mildenhall, B.: DreamFusion: Text-to-3D using 2D Diffusion (Sep 2022). https://doi.org/10.48550/arXiv.2209.14988, http://arxiv.org/abs/2209.14988, arXiv:2209.14988 [cs]
48. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language Models are Unsupervised Multitask Learners (2019)
49. Roessle, B., Müller, N., Porzi, L., Bulò, S.R., Kontschieder, P., Dai, A., Nießner, M.: L3DG: Latent 3D Gaussian Diffusion. In: SIGGRAPH Asia 2024 Conference Papers. pp. 1–11 (Dec 2024). https://doi.org/10.1145/3680528.3687699, http://arxiv.org/abs/2410.13530, arXiv:2410.13530 [cs]
50. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-Resolution Image Synthesis with Latent Diffusion Models (Apr 2022). https://doi.org/10.48550/arXiv.2112.10752, http://arxiv.org/abs/2112.10752, arXiv:2112.10752 [cs]

51. Schneider, M.A., Höllein, L., Nießner, M.: WorldExplorer: Towards Generating Fully Navigable 3D Scenes (Sep 2025). https://doi.org/10.1145/3757377.3763946, http://arxiv.org/abs/2506.01799, arXiv:2506.01799 [cs]
52. Shim, J., Kang, C., Joo, K.: Diffusion-Based Signed Distance Fields for 3D Shape Generation. pp. 20887–20897 (2023), https://openaccess.thecvf.com/content/CVPR2023/html/Shim_Diffusion-Based_Signed_Distance_Fields_for_3D_Shape_Generation_CVPR_2023_paper.html
53. Siddiqui, Y., Alliegro, A., Artemov, A., Tommasi, T., Sirigatti, D., Rosov, V., Dai, A., Nießner, M.: MeshGPT: Generating Triangle Meshes with Decoder-Only Transformers (Nov 2023). https://doi.org/10.48550/arXiv.2311.15475, http://arxiv.org/abs/2311.15475, arXiv:2311.15475 [cs]
54. Simonyan, K., Zisserman, A.: Very Deep Convolutional Networks for Large-Scale Image Recognition (Apr 2015). https://doi.org/10.48550/arXiv.1409.1556, http://arxiv.org/abs/1409.1556, arXiv:1409.1556 [cs]
55. Su, J., Lu, Y., Pan, S., Murtadha, A., Wen, B., Liu, Y.: RoFormer: Enhanced Transformer with Rotary Position Embedding (Nov 2023). https://doi.org/10.48550/arXiv.2104.09864, http://arxiv.org/abs/2104.09864, arXiv:2104.09864 [cs]
56. Szymanowicz, S., Insafutdinov, E., Zheng, C., Campbell, D., Henriques, J.F., Rupprecht, C., Vedaldi, A.: Flash3D: Feed-Forward Generalisable 3D Scene Reconstruction from a Single Image (Jun 2025). https://doi.org/10.48550/arXiv.2406.04343, http://arxiv.org/abs/2406.04343, arXiv:2406.04343 [cs]
57. Tang, J., Nie, Y., Markhasin, L., Dai, A., Thies, J., Nießner, M.: DiffuScene: Denoising Diffusion Models for Generative Indoor Scene Synthesis (Mar 2024). https://doi.org/10.48550/arXiv.2303.14207, http://arxiv.org/abs/2303.14207, arXiv:2303.14207 [cs]
58. Tang, J., Ren, J., Zhou, H., Liu, Z., Zeng, G.: DreamGaussian: Generative Gaussian Splatting for Efficient 3D Content Creation (Mar 2024). https://doi.org/10.48550/arXiv.2309.16653, http://arxiv.org/abs/2309.16653, arXiv:2309.16653 [cs]
59. Tang, Z., Gu, S., Wang, C., Zhang, T., Bao, J., Chen, D., Guo, B.: VolumeDiffusion: Flexible Text-to-3D Generation with Efficient Volumetric Encoder (Aug 2024). https://doi.org/10.48550/arXiv.2312.11459, http://arxiv.org/abs/2312.11459, arXiv:2312.11459 [cs]
60. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention Is All You Need (Aug 2023). https://doi.org/10.48550/arXiv.1706.03762, http://arxiv.org/abs/1706.03762, arXiv:1706.03762 [cs]
61. Wang, X., Yeshwanth, C., Nießner, M.: SceneFormer: Indoor Scene Generation with Transformers (Apr 2021). https://doi.org/10.48550/arXiv.2012.09793, http://arxiv.org/abs/2012.09793, arXiv:2012.09793 [cs]
62. Wu, J., Zhang, C., Xue, T., Freeman, W.T., Tenenbaum, J.B.: Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling (Jan 2017). https://doi.org/10.48550/arXiv.1610.07584, http://arxiv.org/abs/1610.07584, arXiv:1610.07584 [cs]
63. Wu, X., Jiang, L., Wang, P.S., Liu, Z., Liu, X., Qiao, Y., Ouyang, W., He, T., Zhao, H.: Point Transformer V3: Simpler, Faster, Stronger (Mar 2024). https://doi.org/10.48550/arXiv.2312.10035, http://arxiv.org/abs/2312.10035, arXiv:2312.10035 [cs]

64. Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J.: 3D ShapeNets: A Deep Representation for Volumetric Shapes (Apr 2015). https://doi.org/10.48550/arXiv.1406.5670, http://arxiv.org/abs/1406.5670, arXiv:1406.5670 [cs]
65. Xiang, J., Chen, X., Xu, S., Wang, R., Lv, Z., Deng, Y., Zhu, H., Dong, Y., Zhao, H., Yuan, N.J., Yang, J.: Native and Compact Structured Latents for 3D Generation (Dec 2025). https://doi.org/10.48550/arXiv.2512.14692, http://arxiv.org/abs/2512.14692, arXiv:2512.14692 [cs]
66. Xiang, J., Lv, Z., Xu, S., Deng, Y., Wang, R., Zhang, B., Chen, D., Tong, X., Yang, J.: Structured 3D Latents for Scalable and Versatile 3D Generation (May 2025). https://doi.org/10.48550/arXiv.2412.01506, http://arxiv.org/abs/2412.01506, arXiv:2412.01506 [cs]
67. Yang, G., Huang, X., Hao, Z., Liu, M.Y., Belongie, S., Hariharan, B.: PointFlow: 3D Point Cloud Generation with Continuous Normalizing Flows (Sep 2019). https://doi.org/10.48550/arXiv.1906.12320, http://arxiv.org/abs/1906.12320, arXiv:1906.12320 [cs]
68. Yu, L., Lezama, J., Gundavarapu, N.B., Versari, L., Sohn, K., Minnen, D., Cheng, Y., Birodkar, V., Gupta, A., Gu, X., Hauptmann, A.G., Gong, B., Yang, M.H., Essa, I., Ross, D.A., Jiang, L.: Language Model Beats Diffusion – Tokenizer is Key to Visual Generation (Mar 2024). https://doi.org/10.48550/arXiv.2310.05737, http://arxiv.org/abs/2310.05737, arXiv:2310.05737 [cs]
69. Zhang, J., Xiong, F., Xu, M.: G3PT: Unleash the power of Autoregressive Modeling in 3D Generation via Cross-scale Querying Transformer (Sep 2024). https://doi.org/10.48550/arXiv.2409.06322, http://arxiv.org/abs/2409.06322, arXiv:2409.06322 [cs]
70. Zhang, K., Bi, S., Tan, H., Xiangli, Y., Zhao, N., Sunkavalli, K., Xu, Z.: GS-LRM: Large Reconstruction Model for 3D Gaussian Splatting (Apr 2024). https://doi.org/10.48550/arXiv.2404.19702, http://arxiv.org/abs/2404.19702, arXiv:2404.19702 [cs]
71. Zhou, S., Fan, Z., Xu, D., Chang, H., Chari, P., Bharadwaj, T., You, S., Wang, Z., Kadambi, A.: DreamScene360: Unconstrained Text-to-3D Scene Generation with Panoramic Gaussian Splatting (Jul 2024). https://doi.org/10.48550/arXiv.2404.06903, http://arxiv.org/abs/2404.06903, arXiv:2404.06903 [cs]
72. Zhou, Z., Wu, T., Jiang, Z., Obeid, F., Lan, Z.: Value Residual Learning (Jun 2025). https://doi.org/10.48550/arXiv.2410.17897, http://arxiv.org/abs/2410.17897, arXiv:2410.17897 [cs]