

Repurposing 3D Generative Model for Autoregressive Layout Generation

Supplementary Material

6. Implementation Details

6.1. Base 3D Generative Model

To equip our system with a strong and expressive 3D prior, we begin by training a base 3D generative model. Our design follows the state-of-the-art structured 3D generative framework TRELLIS [6], particularly its structure-level generation stage, which predicts sparse voxel occupancies to capture the spatial organization of objects and to model physically and semantically plausible scene layouts. Concretely, we reuse the structured variational autoencoder of TRELLIS as the backbone, providing a compact and expressive representation of 3D structures. To further enhance the model’s ability to interpret complex semantic layouts, we adopt Qwen2.5-VL-7B-Instruct [1] as the text encoder, ensuring rich cross-modal grounding. The design of our flow transformer builds upon the architecture of Qwen-Image [5] and integrates the Multimodal Diffusion Transformer (MMDiT) [2], thereby enabling unified modeling of text and 3D representations within a single Transformer framework. Within each block of the MMDiT, we incorporate a novel positional encoding mechanism, Multimodal Scalable RoPE (MSRoPE), designed to provide consistent and scale-robust positional representations across both modalities. This formulation enables effective multimodal fusion while preserving stable positional semantics for text and scalable spatial modeling for 3D latent representations.

The detailed experimental settings are as follows. We represent the 3D scene using a 64^3 voxel grid, which is used across all training stages as well as during inference. For training, we adopt classifier-free guidance (CFG) [3] with a drop rate of 0.1 and use the AdamW optimizer [4] with a learning rate of 1×10^{-4} . The model is trained for 400K steps on 16 A100 GPUs (80GB) with a batch size of 16 per GPU. At inference time, the CFG strength is set to 3 and 50 sampling steps are used.

6.2. Teacher Model

With a strong 3D prior established, the next stage focuses on applying it to layout generation. To preserve the spatial knowledge already acquired by the base model, we minimize modifications to the original architecture. Specifically, as illustrated in Sec. 3.3, the teacher model builds upon the base 3D generative model by jointly taking the current scene state and the target object as input. To enable the model to distinguish between the scene and objects while facilitating faster convergence, we further introduce an identity-aware positional embedding. Together, these

designs allow the teacher model to achieve comprehensive modeling of the geometric relationships between the scene and objects. The experimental setup is largely consistent with that of the first stage, with the main differences being a reduced training length of 100K steps and a learning rate of 5×10^{-5} .

6.3. Post-Training via Dual-Guidance Self-Rollout

To mitigate the exposure bias inherent in autoregressive generation, we employ a dual-guidance self-rollout strategy, as summarized in Algorithm 1. This stage distills a pre-trained, few-step student generator following the methodology detailed in the main paper Sec. 3.4. Below, we specify the network components and hyperparameters used in this process.

The distillation framework comprises four distinct models. The Student Model G_θ is an efficient, few-step 3D layout diffusion model, initialized via distillation from the Autoregressive Teacher Model trained in Sec. 6.2. The Holistic Teacher $p_{\mathcal{T}_S}$ provides the final supervision signal $\mathcal{L}_{holistic}$ and is implemented using the frozen Base 3D Generative Model described in Sec. 6.1. The Step-Wise Teacher $p_{\mathcal{T}_P}$ provides intermediate corrective signals \mathcal{L}_{step} utilizing the frozen Autoregressive Teacher Model detailed in Sec. 6.2. Finally, to implement the Distribution Matching Distillation (DMD) loss, we employ a trainable Critic Model f_ψ . This critic is initialized with the same architecture and weights as the Base 3D Generative Model, i.e., the Holistic Teacher, and is trained to approximate the score function of the student’s generated data distribution.

Training Hyperparameters. We perform post-training with a batch size of 1 given the sequential, memory-intensive nature of the self-rollout process. The Student Model G_θ is optimized using AdamW with a learning rate of 2×10^{-6} , $(\beta_1, \beta_2) = (0.0, 0.999)$, and weight decay of 0.01. The Critic Network f_ψ is optimized separately using AdamW with a learning rate of 5×10^{-7} , $(\beta_1, \beta_2) = (0.0, 0.999)$, and weight decay of 0.01. To stabilize score estimation, we use a Generator/Critic update ratio of 1:5 (i.e., five critic updates per student update). For teacher score computation $s_{\mathcal{T}}$, Classifier-Free Guidance (CFG) is applied with a scale of 3.0.

Loss Function Formulation. Our dual-guidance objective $\mathcal{L}_{dual} = \mathcal{L}_{holistic} + \mathcal{L}_{step}$ is formulated using Distribution Matching Distillation [7]. This objective minimizes the reverse Kullback-Leibler divergence by leveraging the

Algorithm 1 Dual-Guidance Self-Rollout Distillation

Require: Denoise timesteps $\{t_1, \dots, t_T\}$, number of objects N

Require: Student generator G_θ , step-wise teacher $p_{\mathcal{T}_P}$, holistic teacher $p_{\mathcal{T}_S}$

Require: Initial state S_0 , object sequence $\{O_i\}_{i=1}^N$, text prompt c

```
1: loop
2:    $S_{\text{ctx}} \leftarrow S_0, S_{\text{outputs}} \leftarrow []$ 
3:   Sample  $s \sim \text{Uniform}(1, \dots, T)$ 
4:   for  $i = 1, \dots, N$  do
5:      $\mathcal{C}_i \leftarrow (S_{\text{ctx}}, O_i, c)$ 
6:     Initialize  $z_{t_T} \sim \mathcal{N}(0, I)$ 
7:     for  $j = T, \dots, s$  do
8:       if  $j == s$  then
9:         Enable gradient computation
10:         $\hat{S}_0 \leftarrow G_\theta(z_{t_j}; t_j, \mathcal{C}_i)$ 
11:         $S_{\text{outputs}}.append(\hat{S}_0)$ 
12:         $S_{\text{ctx}} \leftarrow \hat{S}_0.detach()$ 
13:      else
14:        Disable gradient computation
15:         $\hat{S}_0 \leftarrow G_\theta(z_{t_j}; t_j, \mathcal{C}_i)$ 
16:        Sample  $\epsilon \sim \mathcal{N}(0, I)$ 
17:         $z_{t_{j-1}} \leftarrow \Psi(\hat{S}_0, \epsilon, t_{j-1})$ 
18:      end if
19:    end for
20:  end for

21:   $\mathcal{L}_{\text{step}} \leftarrow 0, S_{\text{ctx}} \leftarrow S_0$ 
22:  for  $i = 1, \dots, N$  do
23:     $\hat{S}_i \leftarrow S_{\text{outputs}}[i]$ 
24:     $\mathcal{C}_i \leftarrow (S_{\text{ctx}}, O_i, c)$ 
25:     $\mathcal{L}_{\text{step}} \leftarrow \mathcal{L}_{\text{step}} + \mathcal{L}_{\text{DMD}}(\hat{S}_i; p_{\mathcal{T}_P}, \mathcal{C}_i)$ 
26:     $S_{\text{ctx}} \leftarrow \hat{S}_i.detach()$ 
27:  end for
28:   $\hat{S}_N \leftarrow S_{\text{outputs}}.last()$ 
29:   $\mathcal{L}_{\text{holistic}} \leftarrow \mathcal{L}_{\text{DMD}}(\hat{S}_N; p_{\mathcal{T}_S}, c)$ 
30:   $\mathcal{L}_{\text{dual}} \leftarrow \mathcal{L}_{\text{step}} + \mathcal{L}_{\text{holistic}}$ 
31:  Update  $\theta$  via  $\nabla_\theta \mathcal{L}_{\text{dual}}$ 
32: end loop
```

score difference between the student (approximated by the critic f_ψ) and the teacher. The gradient for the student G_θ is derived as follows:

$$\nabla_\theta \mathcal{L}_{\text{dual}} \approx \mathbb{E}_{x_t, t} [(s_{\mathcal{T}}(x_t, t) - s_\psi(x_t, t)) \nabla_\theta x_0] \quad (1)$$

where $x_0 = G_\theta(x_t, t, \mathcal{C}_i)$ denotes the clean layout predicted by the student. Here, $s_{\mathcal{T}}$ represents the score function of the fixed teacher (either $p_{\mathcal{T}_S}$ for $\mathcal{L}_{\text{holistic}}$ or $p_{\mathcal{T}_P}$ for $\mathcal{L}_{\text{step}}$), and s_ψ is the score estimated by the critic. The critic is concurrently trained to approximate the student’s score using a

standard denoising objective.

7. Additional Results

7.1. Scalability to More Objects

We evaluate on 8–10 objects for fair comparison against existing baselines. Thanks to self-rollout distillation, LaviGen inherently supports a “train short, test long” paradigm and can handle scenes with more than 20 objects, as shown in Fig. 8.



Figure 8. Qualitative results for long-sequence generation with more than 20 objects.

7.2. Generalizability Across Backbones

Our framework is not tied to a specific 3D generative backbone. To validate this, we apply LaviGen to TRELIS [6] using its original CLIP text encoder, without our additionally trained Qwen encoder. As shown in Fig. 9, the model maintains high physical plausibility and semantic coherence, confirming that our recipe successfully transfers across different base architectures without relying on large-scale training infrastructure.

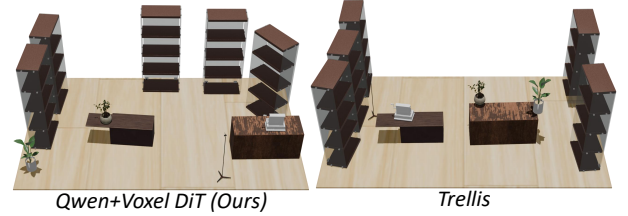


Figure 9. Generalization across different 3D generation backbones.

7.3. Generation Diversity

LaviGen naturally supports diverse outputs via stochastic sampling. Given the same input instruction, the model generates varied yet plausible layouts, as illustrated in Fig. 10.



Figure 10. Diverse layout results generated from the same input instruction.

8. Limitations and Future Work.

Although *LaviGen* gains strong geometric distribution modeling capability from operating in the native 3D space,

several limitations remain. First, due to constraints in model capacity and computational resources, we adopt a 64^3 3D grid resolution. While generally adequate for most objects, this resolution becomes insufficient for small instances, leading to mismatches in subsequent spatial coordinate computations. To address this issue, our future work will explore more efficient computation strategies for higher-resolution voxel grids and investigate denser 3D representations capable of supporting higher spatial resolutions and capturing finer spatial details. Additionally, as shown in Tab. 1, the semantic consistency of the generated layouts remains suboptimal. We attribute this primarily to the scarcity of high-quality annotations, particularly for layouts with complex spatial configurations or object arrangements. In future work, we will enhance our annotation pipeline to collect and process additional high-quality labeled data, and explore more advanced text-conditioning mechanisms to further improve the robustness and semantic reliability of *LaviGen*.

References

- [1] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibao Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report. In *arXiv*, 2025. [1](#)
- [2] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024. [1](#)
- [3] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *arXiv*, 2022. [1](#)
- [4] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *arXiv*, 2019. [1](#)
- [5] Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, Yuxiang Chen, Zecheng Tang, Zekai Zhang, Zhengyi Wang, An Yang, Bowen Yu, Chen Cheng, Dayiheng Liu, Deqing Li, Hang Zhang, Hao Meng, Hu Wei, Jingyuan Ni, Kai Chen, Kuan Cao, Liang Peng, Lin Qu, Minggang Wu, Peng Wang, Shuting Yu, Tingkun Wen, Wensen Feng, Xiaoxiao Xu, Yi Wang, Yichang Zhang, Yongqiang Zhu, Yujia Wu, Yuxuan Cai, and Zenan Liu. Qwen-image technical report. In *arXiv*, 2025. [1](#)
- [6] Jianfeng Xiang, Zelong Lv, Sicheng Xu, Yu Deng, Ruicheng Wang, Bowen Zhang, Dong Chen, Xin Tong, and Jiaolong Yang. Structured 3d latents for scalable and versatile 3d generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2025. [1](#), [2](#)
- [7] Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *CVPR*, 2024. [1](#)